

**AUTOMATED REALLOCATOR OF REPLICAS OVER MOBILE AD HOC
NETWORKS**

**By
Inas Ismail Abuqaddom**

**Supervisor
Dr. Azzam T. Sleit, Assoc. Prof.**

**Co-Supervisor
Dr. Wesam A. Almobaideen, Assoc. Prof.**

**This Thesis was Submitted in Partial Fulfillment of the Requirements for the
Master's Degree of Science in Computer Science**

**Faculty of Graduate Studies
The University of Jordan**

May, 2009

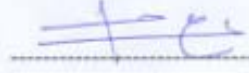
COMMITTEE DECISION

This Thesis/Dissertation (Automated Reallocator of Replicas Over Mobile Ad hoc Networks) was Successfully Defended and Approved on ---- 12/5/2009 ----

Examination Committee

Signature

Dr. Azzam Sleit (Supervisor)
Assoc. Prof. of Imaging DataBases



Dr. Wesam Almobaideen (Co-Supervisor)
Assoc. Prof. of Wireless Networks,
Mobile agents Technology



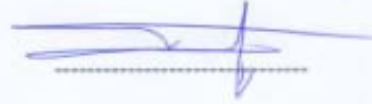
Dr. Imad Salah (Member)
Assoc. of Complex Systems & Networks



Dr. Mousa AL-Akhras (Member)
Assist. Prof. of Artificial Intelligence,
Artificial Neural Networks & Communications



Dr. Hamed Al-bdour (Member)
Assist. Prof. of Wireless Networks
(Mu'tah University)



تعتد كلية الدراسات العليا
هذه النسخة من الرسالة
التوقيع.....التاريخ: ١٢/٥/٢٠٠٩



DEDICATION

*To Islam,
the greatest religion that gives us the motive to work and achieve*

*To my parents, my sisters, my brothers and my uncle Yousef Dannoun
for their support, patience, and encouragement through my life*

To all my friends,

I dedicate this work.

ACKNOWLEDGEMENT

All the praise is due to Allah, the Lord of all creatures, for the warmhearted gracious support in every moment which gives me the perseverance and the ability to continue. May the prayers of blessing of Allah be upon Prophet Muhammad and upon his family and all his companions.

My gratitude and respect go to my supervisors Dr. Azzam T. Sleit and Dr. Wesam A. Al-Mobaideen, who spent a lot of their time guiding me to improve and complete this work. I am thankful for their constructive criticism, invaluable advice and patience in helping me through these years.

Inas I. Abuqaddom
Amman 2009

TABLE OF CONTENTS

Subject	Page
Committee Decision	ii
Dedication	iii
Acknowledgement	iv
List of Contents.....	v
List of Tables	vii
List of Figures	viii
List of Abbreviations	ix
List of Appendices	xii
ABSTRACT.....	xiii
1. Introduction	1
1.1 Mobile Ad Hoc Networks.....	1
1.2 Database.....	2
1.3 Replication Issues.....	3
1.4 Problem Formulation.....	4
1.5 Thesis Contribution.....	4
1.6 Thesis Outline.....	5
2. Replication Issues.....	6
2.1 Consistency Management.....	6
2.1.1 Old Replica Invalidation.....	7
2.1.1.1 UB Method.....	7
2.1.1.2 CR Method.....	8
2.1.2 Updated and Dissemination.....	9
2.1.2.1 DU Method.....	10
2.1.2.2 DC Method.....	10
2.2 Location Management.....	11
2.2.1 AL Method.....	11
2.2.2 GM Method.....	12
2.3 Replica Relocation.....	13
3. Related Works.....	14
3.1 Static Access Frequency (SAF).....	14
3.2 Dynamic Access Frequency and Neighborhood (DAFN).....	14
3.3 Dynamic Connectivity based Grouping (DCG).....	15
3.4 Extended Methods.....	15
3.5 Dynamic Replica Allocation Scheme (DRAM).....	16
3.6 Collaborative Allocation and Deallocation of Replicas with Efficiency (CADRE).....	17
3.7 Data Replication Technique for Real-Time Mobile Ad-hoc	

Network Databases (DREAM).....	17
4. Proposed Approach.....	19
4.1 ARROM Details.....	19
4.2 Cost Model	22
4.3 ARROM Example.....	23
4.3.1 Read Operation Example.....	26
4.3.2 Write Operation Example.....	28
4.3.3 Reallocation Process Example.....	30
5. Results and Discussion	33
5.1 System Specifications.....	33
5.2 Simulator's Overview.....	33
5.3 Simulation Environment	34
5.4 Performance Metrics.....	37
5.5 Results and Analysis.....	37
5.5.1 Effects of Reallocation Threshold on ARROM.....	38
5.5.2 Effects of Requested Data Object Size on ARROM.....	41
5.5.3 Effects of Read and Write Ratio on ARROM.....	43
5.5.4 Effects of Radio Transition Power on ARROM.....	46
5.5.5 Average Improvement Ratio of ARROM.....	48
6. Conclusion and Future Works.....	51
6.1 Conclusion.....	51
6.2 Future Works.....	52
REFERENCES.....	53
APPENDIX	56
ABSTRACT IN ARABIC.....	58

LIST OF TABLES

NUMBER	TABLE CAPTION	PAGE
1	DCT of D_k in M_6	25
2	List5 - the RHs list of direction 5	25
3	List7 - the RHs list of direction 7	25
4	List8 - the RHs list of direction 8	26
5	List20 - the RHs list of direction 20	26
6	Updated DCT of D_k in M_6	27
7	Updated list20	28
8	Updated DCT of D_k in M_6	29
9	Updated list5	30
10	DCT of D_k in M_1	32
11	System specifications	33
12	Simulation Parameters	35
13	The improvement ratios of ARROM with different parameters	49
14	The GloMoSim models currently available at each of the major layers (Nuevo, 2004)	56

LIST OF FIGURES

NUMBER	FIGURE CAPTION	PAGE
1	ARROM example.	24
2	ARROM example after reallocation process.	32
3	The ART of a request with different reallocation threshold values.	38
4	The average reallocation frequencies for different reallocation thresholds.	39
5	The ANT with different reallocation threshold values.	40
6	The ART of a request with different sizes of requested data object.	42
7	The ANT with different sizes of requested data object.	43
8	The ART of a request with different Read/Write ratios.	44
9	The ANT with different Read/Write ratios.	45
10	The ART of a request with different values of radio transition power.	46
11	The average reallocation frequencies for different values of radio transition power.	47
12	The ANT with different values of radio transition power.	48

LIST OF ABBREVIATIONS OR SYMBOLS

AL	Access Log
ANT	Average Network Throughput
AODV	Ad hoc On-demand Distance Vector
ARROM	Automated Reallocation of Replicas Over MANET
ART	Average Response Time
CADRE	Collaborative Allocation and Deallocation of Replicas with Efficiency
c_c	Cost of Control message
c_d	Cost of Data object message
CR	Connection Rebroadcast
DAFN	Dynamic Access Frequency and Neighborhood
DBMS	DataBase Management System
DC	Dissemination on Connection
DCG	Dynamic Connectivity based Grouping
DC/GG	Dissemination on Connection /Group-to-Group
DC/OO	Dissemination on Connection /One-to-One
DCT	Direction Cost Table
DRAM	Dynamic Replica Allocation Scheme
DREAM	Data Replication Technique for Real-time Mobile ad-hoc Network

	Databases
DU	Dissemination on Update
E-DAFN+	Extended DAFN
E-DCG+	Extended DCG
E-SAF+	Extended SAF
GloMoSim	Global Mobile information system Simulation library
GM	Group Management
GW	Gateway
IR	Improvement Ratio
LM	Location Management
MANET	Mobile Ad hoc Network
MH	Mobile Host
MRV	Most Resent Value
MRVP	Most Resent Value in Partition
OCV	Overwrite Current Value
OD	OutDated transaction
OSer	Original- Server
PARSEC	Parallel Simulation Environment for Complex Systems
PDA	Personal Digital Assistant
RH	Requesting Host

RPGM	Reference Point Group Mobility
RSer	Replica-Server
RWR	Read/Write Ratio
SAF	Static Access Frequency
SL	Server List
UB	Update Broadcast
UCV	Use Current Value

LIST OF APPENDICES

NUMBER	APPENDIX CAPTION	PAGE
A	About GloMoSim	56

AUTOMATED REALLOCATOR OF REPLICAS OVER MOBILE AD HOC NETWORKS

By
Inas Ismail Abuqaddom

Supervisor
Dr. Azzam T. Sleit, Assoc. Prof.

Co- Supervisor
Dr. Wesam A. Almobaideen, Assoc. Prof.

ABSTRACT

Mobile ad hoc networks (MANETs) have no static infrastructure; hence the network topology frequently changes. Moreover, the functionalities of mobile hosts (MHs) are both end-systems and routers, which impose workload on MHs. This workload affects them because of their limited battery power, limited bandwidth and poor resources. Data replication is a mechanism that is used to face these challenges. It improves data availability, but with increased cost of storage space and communication overhead.

Replication issues are classified into three categories. First one is replica relocation, which implies when, where and how replicas are allocated. Optimal replica allocation cannot be determined in MANET due to nodes mobility. Second category is consistency management to keep replicas consistent as possible as an update is performed on a replica. Third category is location management, i.e. Requesting Host (RH) must know the location of requested data object or its replicas. Otherwise, RH broadcasts the request over the entire network, which causes a heavy traffic. Every replication system provides method for each of these categories. All replication methods in MANET perform most of their works periodically, due to nodes mobility; this period is called relocation period. To reduce communication cost of accessing replicas, we must reallocate them frequently due to nodes mobility and request pattern. In other words, we have to use shorter relocation period, but this means loaded network due to redoing a lot of tasks very often without needing to redo them. On the other hand, longer relocation period means that replicas are reallocated few times regardless of the MHs' request pattern.

This thesis presents a new replication system called Automated Reallocation of Replicas Over MANET (ARROM), which attempts to get the best replica allocation during its lifetime in terms of communication cost. ARROM uses relocation period of appropriate length and reallocates existed replicas separately during relocation period in order to reduce the communication cost of both accessing and reallocating them, which improves the usage of MHs' poor resources due to decreasing the hops count of each request. We evaluated ARROM using GloMoSim simulator. Simulation results show that ARROM decreases the average response time compared with Extended Dynamic Connectivity based Grouping (E-DCG+) algorithm by 39.76%. In terms of average network throughput, ARROM increases it by 1.86% compared with E-DCG+.

1. INTRODUCTION:

This chapter gives an introduction about each topic that is related to this thesis, as such, mobile ad hoc networks are introduced in section 1.1, database is introduced in section 1.2 and replication issues are reviewed in section 1.3. Additionally, the problem formulation is explained in section 1.4 and thesis contribution is described in section 1.5. Finally, thesis outline is shown in section 1.6.

1.1 Mobile Ad Hoc Networks:

Recently, wireless communications have become an important technology in reality, as well as in science. Mobile ad hoc network (MANET) is a wireless network of laptops, mobile phones and personal digital assistants (PDAs) without static infrastructure, which provides high mobility for MANET's nodes (mobile hosts). Hence the network topology frequently changes (Fife and Gruenwald, 2003). Ad hoc is a Latin word, which means "for this or for this only". In other words it means "formed or used for specific or immediate problems or needs". Thus, MANET can be formed or used for specific or immediate problems or needs, such as a disaster recovery or military operation as it does not require infrastructure (Bakht, 2004) and (Viswacheda, et al., 2007). Moreover, the functionalities of mobile hosts (MHs) are both end-systems and routers, which impose workload on MHs (Shi and Haas, 2007). This workload affects MHs because of their limited battery power, limited bandwidth and poor resources (Fife and Gruenwald, 2003).

Many mechanisms are used to face these challenges; one of them is a data replication. It improves data accessibility (number of successful requests/number of successful and failed

requests), but with increased cost of storage space and communication overhead. In addition, the wired network protocols such as ADRW in (Wujuan and Veeravalli, 2003) and E-ADRW in (Sleit, et al., 2007) can be adapted in MANET due to its critical characteristics (Shi and Haas, 2007).

1.2 Database:

A computer database relies on specialized software to organize the storage of data. This software is known as a DataBase Management System (DBMS). It is categorized according to the database model which DBMS supports (Elmasri and Navathe, 2007). DBMSs consist of data distribution and application distribution among several nodes in network. Types of application distribution are distribution of either DBMS itself or programs that run on DBMS. On the other hand, data distribution has two basic alternatives; replication and fragmentation. Replicas are a replicated data of DBMS on different nodes mainly to reduce response time for read operations. Data is replicated either fully, i.e. the database is stored at each node, or partially, i.e. partition of the database is stored at some nodes but not all. The problem appears with write operations, because updates must be propagated to all replicas to keep them consistent. Fragment is equivalent to replica, but they are different in the number of data copies. Fragment has only one data copy for each data object, whereas replica has two or more data copies for each data object. Moreover, fragment is always for a partition of the database which is operated as a separated database (Özsu, et al., 1999). However, in this thesis, we deal with a database, which could be a file.

1.3 Replication Issues:

Replication issues are classified into three categories. First one is replica relocation, which implies when, where and how replicas are allocated, i.e. every thing related to allocation process. Optimal replica allocation cannot be determined in MANET due to nodes mobility. Second category is consistency management to keep replicas as consistent as possible as an update is performed on a replica even if there is a partitioning in the network. Third category is location management, i.e. Requesting Host (RH) must know the location of requested data object or its replicas. Otherwise, RH will broadcast the request over the entire network, which causes a heavy traffic (Hara, 2005). Every replication system provides method to handle each of these issues.

Several researchers (Hayashi, et al., 2005, a), (Hara, et al., 2003), (Hara, 2003), (Hayashi, et al., 2005, b) and (Hara, 2005) show that all replication methods in MANET perform most of their works periodically, due to nodes' mobility; this period is called relocation period. Some common tasks are performed at every relocation time in any replication system as the following:

1. Determining groups of nodes according to the mechanism, which the replication system uses.
2. Specifying the number of replicas for each data object depending on Read Write Ratio (RWR), that is explained in section 3.4.
3. Allocating all data objects and their replicas.
4. Discarding replica allocation information and the logs of past data accesses to rebuild them during and for the next period. This information is used in location

management methods, such as Access Log (AL) and Group Management (GM) methods.

1.4 Problem Formulation:

Reducing the communication cost of accessing replicas improves the usage of MHs' poor resources. The best way to reduce the communication cost of accessing replicas is by reallocating replicas frequently due to nodes mobility and request pattern. In other words, we have to use shorter relocation period, but this means loaded network due to redoing the above tasks many times without needing to redo them very often. On the other hand, longer relocation period means that replicas are relocated few times regardless of the MHs' request pattern. To solve this problem, we proposed to use appropriate long relocation period and to reallocate existing replicas separately during relocation period. Thus, we get higher performance due to reallocating replicas frequently with longer relocation period.

1.5 Thesis Contribution:

In this thesis, we propose Automated Reallocation of Replicas Over Mobile ad hoc networks (ARROM) algorithm. It reallocates existed replicas in order to reduce the communication cost of both accessing and reallocating them, which improves the usage of MHs' poor resources. ARROM reallocates replicas frequently during relocation period to get higher performance.

1.6 Thesis Outline:

Rest of this thesis is organized as follows: Chapter 2 discusses replication issues. Chapter 3 describes previous works, which are classified as replica relocation methods. Details of our proposed approach (ARROM) are explained in chapter 4. Chapter 5 presents our simulation settings, performance metrics and results. Finally we give conclusions and future works in chapter 6.

2. REPLICATION ISSUES:

Replication issues are categorized into three categories; consistency management, location management and replica relocation. In this chapter, we explain these categories in more details as follows; consistency management is introduced in section 2.1, location management is reviewed in section 2.2 and replica relocation is reviewed in section 2.3.

2.1 Consistency Management:

It is a mechanism to keep replicas as consistent as possible when update is performed on a replica. Propagation system is a system used to send the updated data object(s) for all MHs that hold their replicas to keep them consistent. Guy, et al., (1998) show that propagation system has three alternatives; client-server, peer-to-peer and hybrid systems. In client-server, propagation is the responsibility of the MH holding the original data object. Thus, propagation is slow, because it makes a bottleneck at the MH holding the original data object. Additionally, any failure on that MH affects propagation. In peer-to-peer systems, propagation is faster, but this system is costly because it is designed to make all nodes tightly connected. Hybrid systems have the advantages of client-server and peer-to-peer systems. In hybrid system, propagation is done by a MH that holds either the original data object or one of its replicas, with other nodes serving as clients.

There are three approaches for propagation. In the first approach the MH, that holds the original data object, updates its replicas at inconstant intervals. However, there is a probability to access an invalid replica before it is updated. To solve this problem, the next two approaches are introduced. One approach assigns a lifetime to each replica at its

allocation time. When the lifetime expires, the holder of this replica discards it automatically. The second approach broadcasts an invalidation report when the original data object is updated. The approach that uses lifetime does not cause any extra overhead, because there is no message exchange among MHs. However, selecting appropriate lifetime for each data object is very difficult, because it depends on the data updates which usually occur randomly. Thus, the approach, that using invalidation report, is more effective than updating replicas at inconstant intervals (Hara, 2005).

Consistency management methods use a time stamp for each data object to distinguish between an old data object and its updated version. Time stamp of a data object is the latest update time for that data object, assuming that MHs have global clock synchronization. Each MH stores the time stamps of all data objects in the entire network into an information table; this table is called a time stamp table. Replica invalidation report methods are categorized into old replica invalidation and updated data dissemination (Hara, 2005), (Hayashi, et al., 2006) and (Hara, 2006).

2.1.1 Old Replica Invalidation:

Methods of this category reduce the number of accesses to old replicas. Such methods are; Update Broadcast (UB) method and Connection Rebroadcast (CR) method.

2.1.1.1 UB Method:

In this method, the MH holding the original data object initiates the invalidation report and broadcasts it to its connected MHs every time that the original data object is updated. The

invalidation report includes data identifier and its time stamp. When a MH receives an invalidation report, the MH compares the time stamp in the received invalidation report with the time stamp of the corresponding data object in the MH's time stamp table. If the time stamp in the invalidation report is more recent, the MH updates the time stamp in its own time stamp table to the received value. Additionally, the MH forwards the received invalidation report to its neighboring MHs. At the same time, if the MH has a replica of that data object, then it discards this replica, keeping the free memory space of the discarded replica to allocate the updated data object again. The new replica is created in this MH, whenever, it accesses the original or replica of the deleted data object with time stamp greater than or equals to time stamp in the time stamp table of the MH.

When a MH has received the same invalidation report more than once, the receiving MH discards this report without being forwarded to others. The receiving MH decides that the received invalidation report has been received before, when the time stamp in it is smaller than or equal to the time stamp in the time stamp table of the receiving MH.

In UB method, invalidation reports are broadcasted every time a data object is updated. Hence, traffic in this method due to broadcasting invalidation reports is little (Hara, 2006), (Hayashi, et al., 2006) and (Hara, 2005).

2.1.1.2 CR Method:

This method is similar to UB method, with an addition to go with mobility of MHs. CR method broadcasts invalidation reports every time an original data object is updated and

every time a new connection is established between two MHs (M_i and M_j). The MH with larger suffix (assume $j > i$) sends its own time stamp table to the MH with smaller suffix (M_i). M_i broadcasts the invalidation reports to all data objects whose time stamps in time stamp table of M_i are smaller than their corresponding time stamps in time stamp table of M_j . This broadcast is limited to M_i 's connected MHs except for M_j . After that M_i sends invalidation reports to M_j . These reports are about data objects, whose time stamps are greater than their corresponding time stamps in time stamp table of M_j . Again the same scenario, M_j broadcasts the invalidation reports, which are received from M_i , to all M_j 's connected MHs except for M_i . Comparing with UB method, CR method generates higher traffic which depends on the number of network topology changes, because invalidation reports are rebroadcasted every network topology change (Hara, 2006), (Hayashi, et al., 2006) and (Hara, 2005).

2.1.2 Updated and Dissemination:

UB and CR methods discard old replicas without getting new replicas of the updated data object. Hence, UB and CR methods can not improve data accessibility, but they can reduce the number of accesses to old replicas. Updated and dissemination methods similar to UB and CR methods with an additional step called dissemination to refresh old replicas every time their original data objects are updated. Thus, updated and dissemination methods improve data accessibility as well as they reduce the number of accesses to old replicas. Updated and dissemination methods are Dissemination on Update (DU) method and Dissemination on Connection (DC) method.

2.1.2.1 DU Method:

DU works exactly like UB method with an addition to refresh old replicas. When a MH discards its own replica, it requests the updated data object to get a new replica of the updated data object. Hence, DU generates more traffic than UB.

2.1.2.2 DC Method:

DC works exactly like CR method with an addition that each of the newly connected MHs disseminates the updated data objects. This dissemination increases network traffic; therefore, DC method has two types which are different in the traffic size. They are DC/One-to-One (DC/OO) and DC/Group-to-Group (DC/GG).

DC/OO Method:

In this method, two newly connected MHs disseminate updated data objects with each other without flooding the dissemination to their connected MHs. Therefore, MHs that are connected to the two MHs can not refresh their old replicas. However, data accessibility and traffic in DC/OO are higher than those in DU.

DC/GG Method:

In this method, the groups of the two newly connected MHs disseminate updated data objects. Thus, MHs that are connected to the two MHs can refresh their old replicas. Data accessibility and traffic in DC/GG are higher than those in DC/OO, due to flooding dissemination (Hara, 2006), (Hayashi, et al., 2006) and (Hara, 2005).

2.2 Location Management:

Location management methods are techniques to find the locations of data objects and their replicas. These locations are changed frequently due to nodes mobility and replicas relocation. There are two location management methods; AL and GM methods. They essentially depend on replica allocation information at every relocation time and the logs of past data accesses.

2.2.1 AL Method:

In this method, each MH has an Access Log (AL) table, which includes the following information:

1. Data identifier.
2. MHs' identifiers that hold the data object or its replicas.

Assume a MH (M_i) wants to access a data object (D_k). M_i searches its own AL table to get the list of MHs' identifiers that have D_k . M_i sends a request to the first MH in the list. If this request fails, M_i sends another request to the next MH in the list. In other words, M_i continues sending requests according to the descending order of MHs in the list of D_k until a request succeeds. If no request succeeds, M_i broadcasts the request over the entire network.

Assume that M_i accesses D_k on M_j , then M_j is inserted at the top of the list of D_k in AL table of M_i . If M_j is already in the list, it is deleted from the old location and is inserted at the top location. The size of each list in AL table does not exceed a certain value L . Whenever, the list reaches L , we delete the last MH from the list to insert the new one.

However, at every relocation time, each MH deletes its own AL table and rebuilds it again (Hara, 2005).

2.2.2 GM Method:

Group Management (GM) method works like AL method with an enhancement to get the benefits of grouping or clustering. As we will see in the next chapter, many studies proved that MHs can be grouped according to their mobility behavior. Group's members move freely in all directions without leaving their group. For example, in Jordan University, IT students use wireless connection in IT school, sciences schools and engineering schools but they rarely use wireless connection in other schools. GM method uses groups as a second level on its technique of location management.

In GM method, each group has a set of gateway nodes, which connect to MH(s) belonging to other groups. Each MH holds the following:

1. Gateway (GW) information about gateway nodes of its group. GW information includes identifiers of gateway nodes and their priorities. Each MH calculates gateway priority according to hops count between the MH and the gateway node. Such that, shorter hop count has higher priority.
2. Location Management (LM) table, which is similar to AL table in AL method.

Assume a MH (M_i) wants to access a data object (D_k). M_i sends a request according to the descending order of MHs in LM table of M_i . If all requests fail, M_i sends request to gateway node according to the descending order of gateway nodes' priorities in the GW information of M_i and it keeps sending until one request succeeds. The gateway node

forwards the received request to its neighboring MH(s), that belonging to other group(s). MH, that receives the forwarded request from the gateway node, uses its own LM table to query about D_k in its group. If the request fails, this MH sends the request to its gateway nodes in its group. If no request succeeds, M_i broadcasts the request over the entire network (Hara, 2005).

2.3 Replica Relocation:

As mentioned in the previous chapter, replica relocation implies when, where and how replicas are allocated. Additionally, allocation process needs to be re-executed periodically, due to nodes mobility. This thesis introduces a new method in replica relocation category; hence its methods are explained separately in chapter three.

3. RELATED WORKS:

This chapter describes previous works, which are classified as replica relocation methods.

3.1 Static Access Frequency (SAF):

In SAF method, each MH allocates replicas according to descending order of its own access frequencies and the limitation of its own memory space. Each MH does not know the list of data objects that are allocated in any other MH even if they are neighbors. Thus, SAF provides low accessibility when many MHs have similar access frequencies, because each replica serves only the MH that holds this replica. Moreover, SAF provides low overhead and traffic, because of no relocation after allocation process. In addition, SAF continues execution until no memory space is available, which produces a large number of replicas. SAF has no consistency management, because MH can not perform write request on any replicas (Hara, 2003), (Hara, 2005) and (Hara and Madria, 2006).

3.2 Dynamic Access Frequency and Neighborhood (DAFN):

DAFN works as SAF with a mechanism to eliminate duplicated replicas among neighboring MHs. The MH whose duplicated replica has lower access frequency than other duplicated replicas, this replica will be replaced by a replica of other data object. Thus, DAFN runs each relocation period, and provides higher accessibility. As a result of elimination duplicated replicas among neighboring MHs which are connected just by one or two hops, DAFN generates overload and traffic higher than SAF. In addition, DAFN continues execution until no memory space is available, which produces a large number of

replicas. DAFN has no consistency management, because MH can not perform write request on any replicas (Hara, 2003), (Hara, 2005) and (Hara and Madria, 2006).

3.3 Dynamic Connectivity based Grouping (DCG):

DCG is the same as DAFN but with larger group size. DCG uses broadcast messages to group MHs into biconnected components, which is discussed in section 4.1. That is if any MH disappears or any link is broken, the group will not be divided into separated smaller groups, this provides groups with high stability. Data objects are selected to be replicated according to the summation of access frequencies of all MHs in the group for the same data object. The data object with the highest summation will be replicated and be allocated in a MH whose access frequency of this data object is the highest among all MHs in the group. DCG provides larger group with sharing replicas, which means higher accessibility but with more overhead and traffic than DAFN. In addition, DCG continues execution until no memory space is available, which produces a large number of replicas. DCG has no consistency management, because MH can not perform write request on any replicas (Hara, 2003), (Hara, 2005) and (Hara and Madria, 2006).

3.4 Extended Methods:

SAF, DAFN and DCG methods are missing write operations, which means any update operation over a data object makes all its replicas invalid. To solve this problem an extended E-SAF+, E-DAFN+ and E-DCG+ methods are introduced. These methods work the same as SAF, DAFN and DCG respectively, but rather than using access frequency they use Read/Write Ratio (RWR). It is the ratio between the probability of read operations that

are performed on data object by a MH at a unit of time, and the probability of write operations that are performed on that data object by the MH that has the original data object. Higher RWR indicates more read operations than write ones. Thus, we can replicate the data object. Lower RWR indicates the opposite, i.e. no more replication on this data object. To keep consistency among replicas, extended methods use a lazy update protocol, which is not executed very often on MHs (Hara, 2003), (Hara, 2005) and (Hara and Madria, 2006).

Additional to what mentioned in sections 3.3 and 3.4, E-DCG+ algorithm uses DC/GG method, which is described in section 2.1.2.2, as a consistency management method and uses GM method, which is described in section 2.2.2, as a location management method. Furthermore, E-DCG+ is a hybrid system, as such, users can perform read and write operations over either the original data object or its replicas and keep them consistent (Hara, 2003), (Hara, et al., 2004), (Hara, 2005) and (Hara and Madria, 2006).

3.5 Dynamic Replica Allocation Scheme (DRAM):

DRAM introduced by (Huang, 2003). DRAM groups MHs according to their mobility behavior. The mobility behavior of mobile users is usually regular and has some mobility patterns. DRAM uses Reference Point Group Mobility (RPGM) model to determine the mobility behavior, and uses a decentralized clustering algorithm to cluster MHs with similar mobility behavior into one group. Then data objects are replicated in each mobility group at the MH that has the highest access frequency and enough memory space. As a result, DRAM provides higher data accessibility than E-DCG+ method. Moreover, the broadcasting in each mobility group is limited to its MHs only, which reduces network

traffic. DRAM continues execution until no memory space is available, which produces a large number of replicas.

3.6 Collaborative Allocation and Deallocation of Replicas with Efficiency (CADRE):

Mondal, et al., (2006) introduced CADRE method, which works on groups like E-DCG+. CADRE adds two new properties to prevent thrashing condition and to achieve fairness in replication. To explain thrashing assume M is a MH, that has a replica in rarely uses, but M's neighbors use this replica a lot. If M decides to remove this replica, this decision leads M's neighbors to allocate a new copy of this replica. As a result, there is a wasted overload due to deallocation and reallocation for the same replica in the same group. CADRE assigns sigma variable for each data object to indicate its importance. Thus, data objects are replicated when their sigmas or access frequencies are high, and that is the fairness in replication.

3.7 Data Replication Technique for Real-Time Mobile Ad-hoc Network Databases (DREAM):

In (Pabmanabhan and Gruenwald, 2006, a) and (Pabmanabhan and Dr.Gruenwald, 2006, b), DREAM was introduced. It is differentiated from other techniques by using weight concept of access frequency. Weight is affected by the type of data and transaction, which are classified in many specific classifications. In general, it classifies data type into read-write data and transactions into read transactions and write transactions. Furthermore, read transactions are classified into Most Recent Value (MRV) which needs the most recent value across all replicas in all partitions, Outdated transaction (OD) which can be executed

successfully even with stall data and Most Recent Value in Partition (MRVP) which needs the most recent value across all replicas in partitions. Additionally, DREAM classifies write transactions into Insert/Delete transactions, Use Current Value (UCV) and Overwrite Current Value (OCV).

The best method among extended methods, DREAM , DRAM and CADRE methods depends on the criteria that you interested in, such that according to traffic size E-DCG+ is the best. Thus, to study the performance of our idea we built our proposed algorithm based on the most famous algorithm, which is E-DCG+.

4. PROPOSED APPROACH:

Most replica relocation methods relocate replicas in order to improve data availability in MANET. Our proposed approach (ARROM) relocates replicas to reduce communication cost of accessing these replicas, which improves the usage of MHs' poor resources, in addition to improve data availability in MANET. ARROM has the assumption that MHs have global clock synchronization.

4.1 ARROM Details:

We divide the execution of ARROM into two stages, whereas most replica relocation methods have one stage. Each stage has certain aims, and they are re-executed periodically. We call the period of the first stage the relocation period and the period of the second stage the reallocation period, which must be smaller than the relocation one. First stage is the common stage among most replica relocation methods. We built our first stage exactly as E-DCG+ method, which includes the following tasks:

1. Grouping MHs into biconnected components.
2. Specifying replicas count for each data object according to its RWR.
3. Allocating all data objects and their replicas according to the summation of access frequencies on the data object from all MHs in the group.
4. Discarding replica allocation information and the logs of past data accesses to rebuild them during and for the next period. This information is used in GM method.

ARROM uses broadcast messages to group MHs into biconnected components. Each MH broadcasts its identifier and its own access frequencies on each data object. After completing broadcasts, every MH knows its connected MHs by forwarding broadcast messages, such that each receiver MH adds its identifier to the forwarded message. Then each set of connected MHs executes an algorithm to find biconnected components on the MH having the lowest identifier. This algorithm finds the largest set of connected MHs that can forward a message among them before it comes back to the initial sender. We deal with each biconnected component as a group, i.e. if any MH disappears or any link is broken, the group will not be divided into separated smaller groups due to looping path(s). This provides groups with high stability. If a MH belongs to more than one biconnected components, it belongs to only one group in which the corresponding biconnected component is first found while executing the algorithm. After creating groups, the summation of access frequencies of all MHs in the group to each data object is calculated. These calculations could be done by any MH in the group, but for standardization, we assumed these calculations are done by the MH having the lowest identifier in the group. Replicas are allocated in the group according to the descending order of the summation of access frequencies on data objects. Each replica is allocated a MH whose access frequency to the data object is the highest among MHs that have free memory space to create it.

Second stage is just for reallocating existing replicas, which are created and allocated in the first stage. We proposed to reallocate existing replicas in order to reduce the communication cost of accessing them in each group. ARROM executes this stage on the MH which has a replica of data object and which has the original data object. We call the former MH as Replica-Server (RSer) and the later MH as Original-Server (OSer). We

introduced direction concept, which is determined by grouping MHs according to the last node in the path between a requesting MH and the server (either RSer or OSer). ARROM finds the ratio between the total access costs of read and write operations on a data object from one direction, and the total access costs of read and write operations on that data object from all directions. See equation (1).

$$\text{Ratio}_D = \frac{\text{Access costs to a data object from one direction}}{\text{Access costs to a data object from all directions}}, \quad (1)$$

Ratio is calculated for each direction. The new location for a data object is the direction that has the highest ratio. For further control we use threshold to keep the ratio high enough. Then ARROM calculates the ratio for each Requesting Host (RH) in the selected direction. In other words, ARROM finds the ratio between the total access costs of read and write operations on a data object from one RH in the selected direction, and the total access costs of read and write operations on that data object from all RHs in the selected direction. See equation (2).

$$\text{Ratio}_{MH} = \frac{\text{Access costs to a data object from one RH}}{\text{Access costs to a data object from all RH in the selected direction}}, \quad (2)$$

The data object is allocated in the RH that has enough memory space and has the highest ratio among all RHs in the selected direction.

Each ARROM server (either OSer or RSer) has a Direction Cost Table (DCT) for each data object. DCT has a record for each direction. This record includes the following:

1. Direction identifier, which equals to the identifier of the last MH in the path between RH and the server.
2. Summation of access costs of the data object from all RHs in the direction.

3. Summation of access frequencies of the data object from all RHs in the direction.
4. List of all RHs in the direction. This list includes RH identifier, RH's own summation of access costs on the data object, RH's own access frequencies on the data object and RH's free memory space.

We use the summation of access frequencies, because if two directions or RHs have the same ratio, ARROM compares their access frequencies and decides reallocation according to their access frequencies instead of ratio.

ARROM is a hybrid system, as such, users can perform read and write operations over either the original data object or its replicas and keep them consistent. However, ARROM uses DC/GG method for consistency management and GM method for location management. Additionally, each group limits broadcasting to its MHs only.

4.2 Cost Model:

In reallocation condition, we use cost parameter, which is the hops count between RH and server. Our proposed cost model consists of two cases:

1. Case A for read operations:

$$\begin{cases} 0 & \text{if } RH \in SL, & (3) \\ c_d + c_c & \text{if } RH \notin SL, & (4) \end{cases}$$

2. Case B for write operations:

$$\begin{cases} \left(\sum_{u \in SL} c_d \right) & \text{if } RH \in SL, & (5) \\ \left(\sum_{u \in SL} c_d \right) + (c_d + c_c) & \text{if } RH \notin SL, & (6) \end{cases}$$

We group servers both RSer and OSer in a server list (SL). If Requesting Host (RH) is a member of SL, then the cost of read operation is zero as shown in equation (3), whereas cost of write operation equals to the propagation cost, which means to send the updated data object for each member of SL to keep replicas on all servers consistent as show in equation (5). There is an additional cost for each RH that is not a member of SL. It is the cost of getting the data object from the nearest server. This additional cost is divided into two steps. The first one is the cost of sending a control message (c_c) from RH to the nearest server. The second step is the cost of sending the data object (c_d) from the nearest server to the RH. This additional cost appears in equations (4) and (6).

4.3 ARROM Example:

Figure 1 shows a MANET of 40 MHs where three of them act as ARROM servers of a data object D_k , whose size is 50 MB. Servers are M_6 , M_{12} and M_{23} , such that M_{12} is the OSer of D_k , whereas M_6 and M_{23} are the RSers of D_k . According to Figure 1, M_6 has four directions as it has four neighbors, because it can receive messages from four nodes. They are M_{20} , M_5 , M_8 and M_7 . M_{12} has six directions, which are M_{13} , M_{11} , M_9 , M_{37} , M_4 and M_{15} . Finally, M_{23} has three directions, which are M_{10} , M_{39} and M_0 .

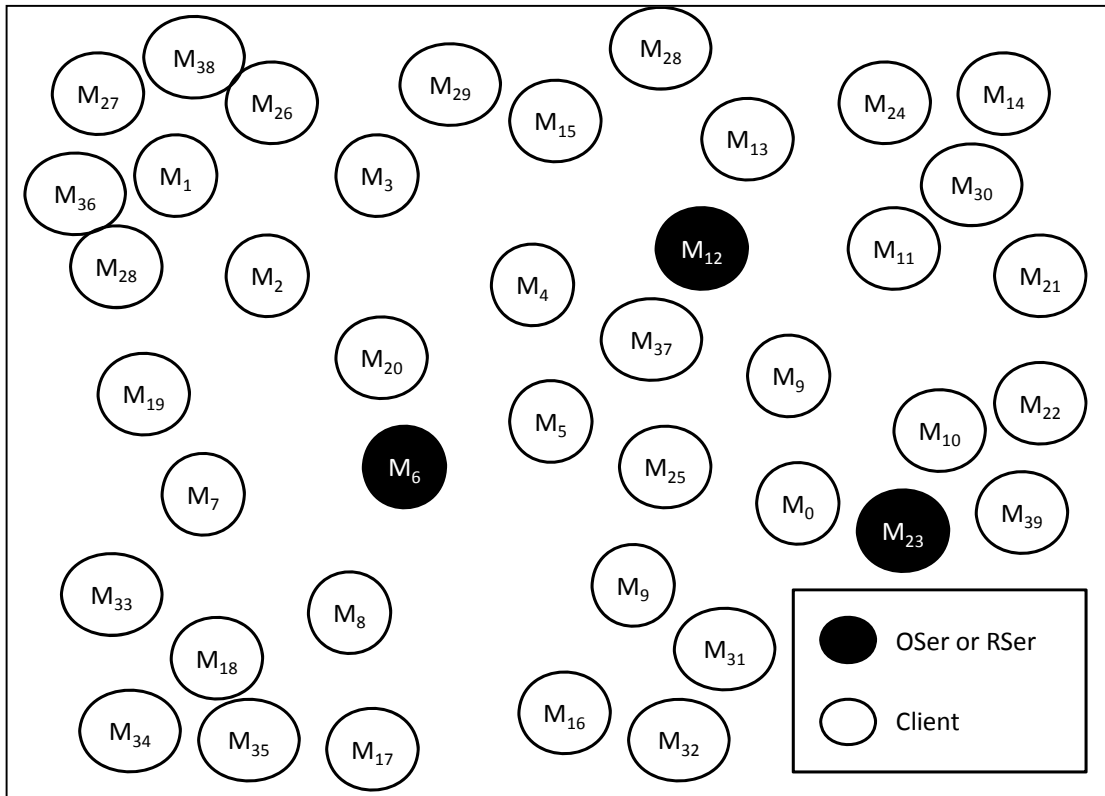


Figure 1: ARROM example.

Initially, all nodes have at least 50 MB as free memory space, according to the size of D_k . Each RSer or OSer has a DCT table for D_k . Every DCT table has a set of pointers that pointing to RHs lists of each direction. Table 1 is the DCT table of D_k in M_6 , which is a RSer of D_k , whereas tables from Table 2 to Table 5 are the RHs lists of directions that exist in Table 1. Notice that list5 is an empty list, because there is no RH from direction 5, see Table 2.

Table 1: DCT of D_k in M_6

Identifier	Summation of access costs	Summation of access frequencies	Pointer to RHs list
5	0	0	* list5
7	28	7	* list7
8	18	4	* list8
20	50	5	* list20

Table 2: List5 - the RHs list of direction 5

Identifier	Summation of access costs	Summation of access frequencies	Free memory space
0	0	0	NULL

Table 3: List7 - the RHs list of direction 7

Identifier	Summation of access costs	Summation of access frequencies	Free memory space
7	6	3	200MB
19	4	1	10MB
28	6	1	0
33	4	1	22MB
36	8	1	0

Table 4: List8 - the RHs list of direction 8

Identifier	Summation of access costs	Summation of access frequencies	Free memory space
17	4	1	0
18	4	1	55MB
34	6	1	250MB
35	4	1	300MB

Table 5: List20 - the RHs list of direction 20

Identifier	Summation of access costs	Summation of access frequencies	Free memory space
1	16	1	200MB
2	13	1	1000MB
26	6	1	155MB
27	8	1	10MB
36	7	1	23MB

4.3.1 Read Operation Example:

Assume that M_1 is performing a read operation on data object D_k and M_1 does not have D_k . Thus, M_1 runs GM method to get the location of D_k . Assume that M_6 is responding to M_1 's request. M_1 can send requests to M_6 using multi paths. One of these paths is from M_1 , M_2 , M_{20} to M_6 . Assume M_1 is using this path in this request. When M_6 receives the request, it fills the information about this request in its own DCT of D_k . The updated tables are shown in Table 6 and Table 7. M_6 receives this request from M_{20} hence data of this request is inserted to the direction whose identifier is 20. According to our cost model the cost of this

request is three hops, which is added to the summation of access costs of the direction 20, as well as, to the summation of access costs of M_1 in list20. M_6 increases access frequencies of direction 20 and access frequencies of M_1 each of them by one. At the same time, M_6 updates list20 to include data of M_1 , which includes its identifier, its own summation of access costs, its own access frequencies and its own free memory space. ARROM gets the free memory space of RH from its request, such that each request includes the free memory space of the RH. M_6 overwrites the stored free memory space of M_1 by the new value, which is received from the request. Finally, M_6 sends D_k to M_1 . No need for fragmentation, because D_k is a small object. However, M_6 has multi paths to send D_k . Assume that M_6 sent D_k to M_{20} , M_2 to M_1 . Thus, the cost of sending D_k is also three hops. Finally, M_6 adds three to the summation of access costs of direction 20, and to the summation of access costs of M_1 as shown in Table 6 and Table 7.

Table 6: Updated DCT of D_k in M_6

Identifier	Summation of access costs	Summation of access frequencies	Pointer to RHs list
5	0	0	* list5
7	28	7	* list7
8	18	4	* list8
20	56	6	* list20

Table 7: Updated list20

Identifier	Summation of access costs	Summation of access frequencies	Free memory space
1	22	2	200MB
2	13	1	1000MB
26	6	1	155MB
27	8	1	10MB
36	7	1	23MB

4.3.2 Write Operation Example:

Assume that M_5 is performing a write operation on data object D_k and M_5 does not have D_k . Thus, M_5 run GM method to get the location of D_k . Assume that M_6 is responding to M_5 request. M_5 can send requests to M_6 using multi paths. One of these paths is from M_5 to M_6 . Assume M_5 is using this path in this request. When M_6 receives the request, it fills the information about this request in its own DCT of D_k . The updated tables are shown in Table 8 and Table 9. M_6 receives this request from M_5 hence data of this request is inserted to the direction whose identifier is 5. According to our cost model the cost of this request is one hop, which is added to the summation of access costs of the direction 5 and of M_5 in list5. Moreover, M_6 increases access frequencies of direction 5 and access frequencies of M_5 by one. At the same time, M_6 updates list5 to include data of M_5 . After that, M_6 sends D_k directly to M_5 i.e. the cost of sending D_k is also one hop. Then M_6 adds one to the summation of access costs of direction 5, and to the summation of access costs of M_5 .

When M_5 has finished modifying D_k , it sends the modified D_k back to M_6 . Then M_6 updates its stored D_k and propagates the modified D_k to M_{12} and M_{23} . Assume that M_6 is using this path M_6, M_5, M_{37} to M_{12} and the path M_6, M_5, M_{25}, M_0 to M_{23} in propagation. Thus, the cost of sending the modified D_k to M_{12} and M_{23} is three hops and four hops respectively. Finally, M_6 adds the propagation cost (three hops and four hops) to the summation of access costs of the direction 5 and to the summation of access costs of M_5 as shown in Table 8 and Table 9. We summarize the summation of access costs of direction 5 that is shown in Table 8 as follows:

1. Cost of sending the request from M_5 to M_6 (1 hop).
2. Cost of sending D_k from M_6 to M_5 (1 hop).
3. Cost of sending the updated version of D_k from M_5 to M_6 (1 hop).
4. Cost of sending the updated version of D_k from M_6 to M_{12} (3 hops).
5. Cost of sending the updated version of D_k from M_6 to M_{23} (4 hops).

Table 8: Updated DCT of D_k in M_6

Identifier	Summation of access costs	Summation of access frequencies	Pointer to RHs list
5	10	1	* list5
7	28	7	* list7
8	18	4	* list8
20	56	6	* list20

Table 9: Updated list5

Identifier	Summation of access costs	Summation of access frequencies	Free memory space
5	10	1	200MB

4.3.3 Reallocation Process Example:

Assume that reallocation period and reallocation threshold are 20 seconds each. M_6 calculates the ratio of each direction at reallocation time. The calculations according to Table 8 and using equation (1) as follows:

$$\text{Ratio}_D(5) = \frac{10}{10 + 28 + 18 + 56} = \frac{10}{112}$$

$$\text{Ratio}_D(7) = \frac{28}{112}$$

$$\text{Ratio}_D(8) = \frac{18}{112}$$

$$\text{Ratio}_D(20) = \frac{56}{112}$$

Direction 20 has the highest ratio. At the same time, the summation of access costs of direction 20 is greater than our pre-defined threshold ($56 > 20$). Thus, D_k is reallocated in direction 20. To specify the new server of D_k , M_6 calculates the ratio of each RH in direction 20. According to Table 7 there are five RHs, which are M_1 , M_2 , M_{26} , M_{27} and M_{36} . Their ratios as follows:

$$\text{Ratio}_{MH}(M_1) = \frac{22}{22 + 13 + 6 + 8 + 7} = \frac{22}{56}$$

$$\text{Ratio}_{MH}(M_2) = \frac{13}{56}$$

$$\text{Ratio}_{MH}(M_{26}) = \frac{6}{56}$$

$$\text{Ratio}_{\text{MH}}(M_{27}) = \frac{8}{56}$$

$$\text{Ratio}_{\text{MH}}(M_{36}) = \frac{7}{56}$$

M_1 had the highest ratio and enough free memory space for D_k ($200\text{MB} > 50\text{MB}$). Thus, M_1 becomes the new RSer of D_k , because the type of new server is the same as the type of the old server. In other words, if M_6 is RSer, M_1 becomes RSer and if M_6 is an OSer, M_1 becomes OSer. As we mentioned previously M_6 was RSer of D_k . Additionally, M_6 announces this change as follows:

1. It sends a message to M_1 to announce it about its new role.
2. It sends the D_k with its last modifications to M_1 .
3. It broadcasts a message to all MHs in the group that M_1 became the RSer of D_k and M_6 is not RSer of D_k anymore.
4. It discards D_k , and its DCT with all its followed lists.

M_1 starts its new role by creating DCT table of D_k with six directions, see Table 10 and Figure 2.

Table 10: DCT of D_k in M_1

Identifier	Summation of access costs	Summation of access frequencies	Pointer to RHs list
2	0	0	* list2
26	0	0	* list26
27	0	0	* list27
28	0	0	* list28
36	0	0	* list36
38	0	0	* list38

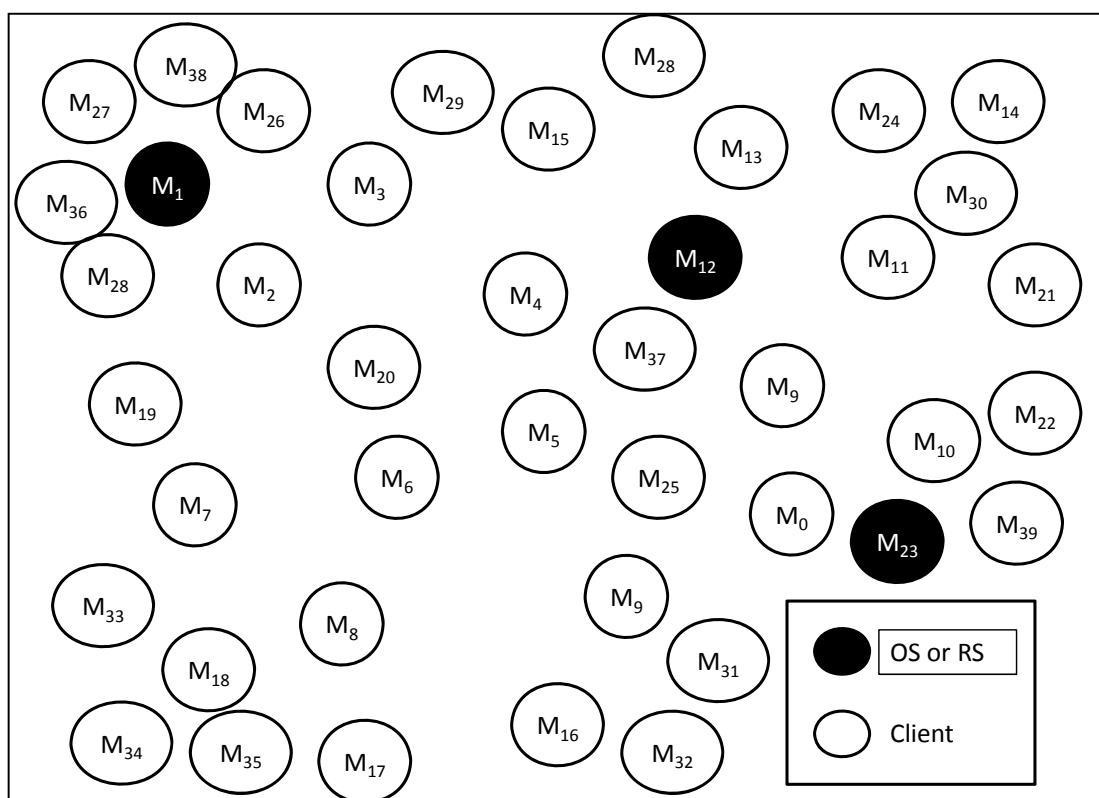


Figure 2: ARROM example after reallocation process.

5. RESULTS AND DISCUSSION

We evaluated ARROM using GloMoSim simulator, whose overview is presented in section 5.2. Our simulation settings are presented in section 5.1 and 5.3. Additionally, section 5.4 explains our performance metrics. The simulation results and their discussion are presented in section 5.5.

5.1 System Specifications:

The experiments were evaluated using HP PC. Table 11 presents the system specifications.

Table 11: System specifications

Item	Value
Processor	Intel Pentium 4 CPU 3.20GHz
System Model	HP Compaq
Memory (RAM)	512MB
OSer Name	Microsoft Windows XP Professional Version 2002 service Pack 3

5.2 Simulator's Overview:

We used Global Mobile Information System Simulator (GloMoSim) to implement and evaluate ARROM. GloMoSim is a library-based sequential and parallel simulator for wireless networks. It is designed as a set of library modules; each of them simulates a specific wireless communication protocol in the protocol stack. This simulator was built based on the Parallel Simulation Environment for Complex Systems (PARSEC) simulation language (Bagrodia et al., 1998). Additionally, GloMoSim has a layered approach similar

to the OSI model. Furthermore, GloMoSim provides different protocols for the Physical layer (PHY), Data link layer (MAC), Network layer (Routing), Transport layer and Application layer (Bajaj et al., 1999).

5.3 Simulation Environment:

The simulation experiments contained 37 clients, one OSer and two RSers according to the maximum size of traffic file (Shi and Haas 2007), which is selected according to the GloMoSim possibilities. These 40 MHs were distributed randomly for generalization using uniform distribution model over a terrain of 2000×2000 meters. This terrain is sufficient to disseminate 40 MHs without changing the default radio power, which generates a coverage area with 391 meters as radius. We used random waypoint model for mobility, because GloMoSim provides only this model for random mobility. In waypoint model a node randomly selects a destination and moves along the direction of that destination in a speed uniformly chosen between the minimum speed that we selected and the maximum speed. These speeds specified as MOBILITY-WP-MIN-SPEED and MOBILITY-WP-MAX-SPEED parameters respectively. When the node reaches its destination, it stays there for pause time, which is specified in MOBILITY-WP-PAUSE parameter (Zeng, et al., 1998), (Jorge, 2004) and (Takai, et al., 1999). We selected the speed between 0 meter/second and 10 meters/second with 30 seconds pause period according to our experiments. We chose Ad hoc On-demand Distance Vector (AODV) protocol for routing, because it is one of the most famous protocols for flat ad hoc networks (Cheng, et al., 2006). Moreover, to get more accurate results, each experiment was repeated 30 times with thirty different SEEDs (1 to 30). The SEED represents a random number used to initialize

various random numbers in the simulation such as the delay time before broadcasting packets (Nuevo, 2004). The simulation parameters are summarized in Table 12. The table presents the general parameters used in all experiments. Most of the general parameters are chosen as their default values in IEEE standards and GloMoSim possibilities, except that the parameters whose effects on ARROM are studied.

Table 12: Simulation Parameters

Parameter	Value
Terrain	2000 x 2000 m
Node placement	UNIFORM
Node mobility	RANDOM-WAYPOINT
MOBILITY-WP-PAUSE	30 seconds
MOBILITY-WP-MIN-SPEED	0 meter/second
MOBILITY-WP-MAX-SPEED	10 meters/second
PROPAGATION-PATHLOSS	TWO-RAY
RADIO-TYPE	RADIO-ACCNOISE
RADIO-BANDWIDTH	2000000
MAC-PROTOCOL	IEEE 802.11
NETWORK-PROTOCOL	IP
Traffic Generator	FTP/Generic
SEEDs	1-30
ROUTING-PROTOCOL	AODV

Each traffic file has two important parameters; which nodes represent RHs, and how many requests are sent from each one. These parameters were chosen randomly using uniform distribution model to generate 40 different traffic files for each experiment to get more accurate results. Each traffic file has 30 requests, because GloMoSim can deal with only 120 requests per experiment. If the 30 requests were write requests, they will become 120 requests during execution due to propagation process. However, these requests were for one data object and they were generated using FTP/Generic protocol. These requests are served remotely or locally when RH is also RSer or OSer. To avoid changing the server, when it has some requests in progress, we studied the maximum response time among all requests that were used in our experiments. Then we made our reallocation period greater than this maximum response time, which was approximately 2 seconds. For more accuracy, we doubled the maximum response time that was extracted to become 5 seconds. Furthermore, we generated one request per second and selected 5 seconds as the minimum period between any two requests. Therefore, our reallocation period was 5 seconds to guarantee that the server has no requests in progress.

We used E-DCG+ approach for comparison, because that ARROM works exactly as E-DCG+ with the addition to reallocate replicas during relocation period. Thus, we can get the exact performance of reallocation protocol. We chose 10 minutes as a relocation period, because it is the average minimum relocation period in previous works. Thus, our reallocation protocol runs 120 times during each relocation period.

5.4 Performance Metrics:

Our performance metrics are Average Response Time (ART) of a request and Average Network Throughput (ANT). ARROM aims to reduce the communication cost of requests. The most important parameter in communication cost is the response time, because it indicates the consumption of MHs' resources. Each MH acts as both end system and router. Therefore, during response time, the request is either being forwarded among MHs or waiting in some MHs due to congestion. We selected throughput as another performance metric, because it indicates the number of sent bytes per unit of time. Therefore, improving throughput means increasing the usage of transmission media, which is one of the most important poor resources in MHs. Furthermore, we define the Improvement Ratio (IR) as:

$$IR = \frac{MAX\ value - MIN\ value}{MAX\ value}, \quad (7)$$

We used IR to show the improvement of ARROM comparing with E-DCG+. Equation (7) normalizes the differences between the two approaches to make values between zero and one.

5.5 Results and Analysis:

We studied ARROM performance using four parameters, which are reallocation threshold; data object size, read/write ratio and radio transition power. In addition, we classified the simulation results into categories according to these parameters as presented in sections 5.5.1 to 5.5.4. Finally, we calculated the average IR of ARROM in section 5.5.5.

5.5.1 Effects of Reallocation Threshold on ARROM:

We studied the effects of reallocation threshold on ARROM. Figure 3 shows the reallocation threshold effects on average response time. For all thresholds ARROM gives better average response time than E-DCG+. Additionally, average response time of E-DCG+ is constant for all threshold values, because E-DCG+ does not reallocate replicas during relocation period.

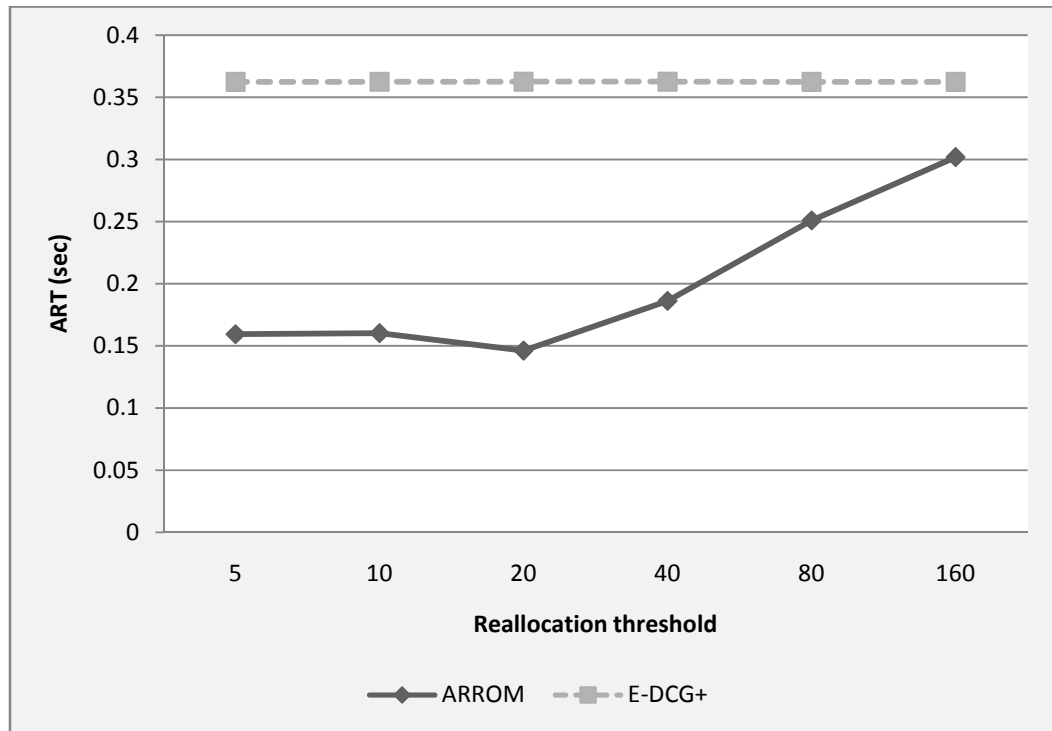


Figure 3: The ART of a request with different reallocation threshold values.

When reallocation threshold is large, ARROM becomes close to E-DCG+. This is because large reallocation thresholds reduce reallocation occurrences as Figure 4 shows. On the other hand, small reallocation thresholds reallocate the data object many times. As a result we get better average response time when the reallocation thresholds decrease. This result

is not true all the time, because there are limitations on decreasing average response time, such as the transition media speed and the processing time of each request. Figure 3 shows that the average response time of ARROM with the first several small thresholds are approximately constant, although their corresponding reallocation frequencies are different. Figure 3 and Figure 4 illustrate that with three reallocation thresholds, which are 5, 10 and 20.

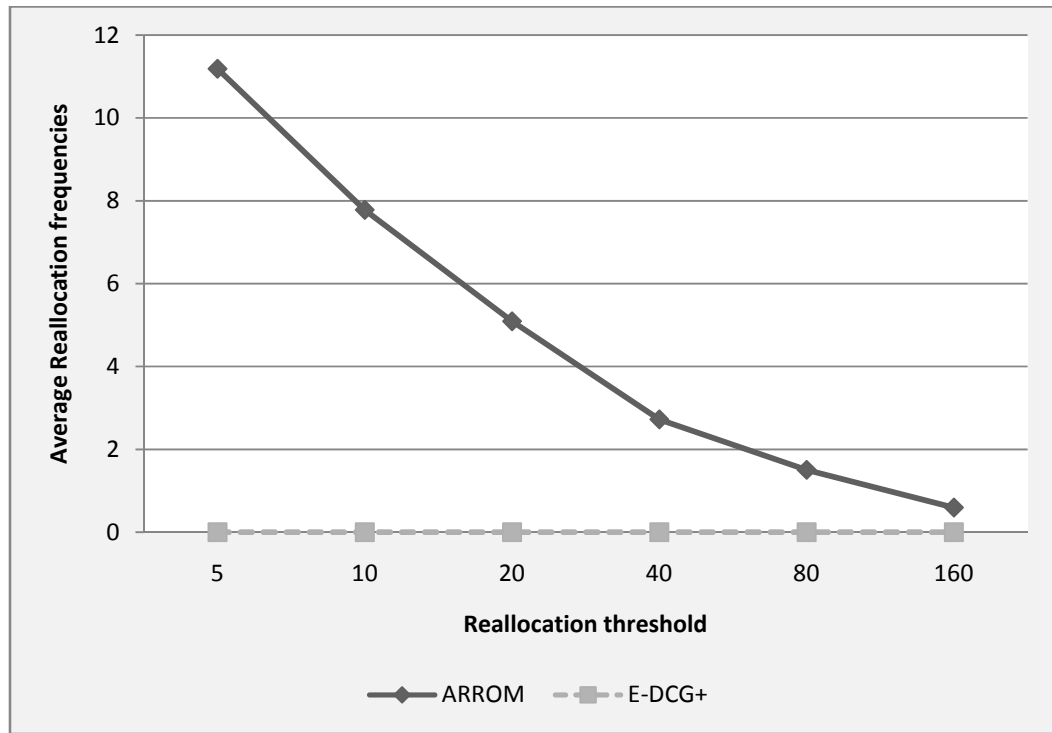


Figure 4: The average reallocation frequencies for different reallocation thresholds.

Figure 5 shows the reallocation threshold effects on average network throughput. For all thresholds ARROM outperforms E-DCG+ in the average network throughput. Additionally, E-DCG+ is constant for all thresholds, because E-DCG+ does not reallocate

replicas during relocation period. Notice that Figure 3 and Figure 5 have the same behavior, because it depends totally on reallocation occurrences. At the same time, there are limitations on increasing the average network throughput, such as the bandwidth of transition media. Therefore, the average network throughputs of ARROM with the first several small thresholds are approximately constant, although their corresponding average reallocation frequencies are different. Figure 4 and Figure 5 illustrate that with three reallocation thresholds, which are 5, 10 and 20.

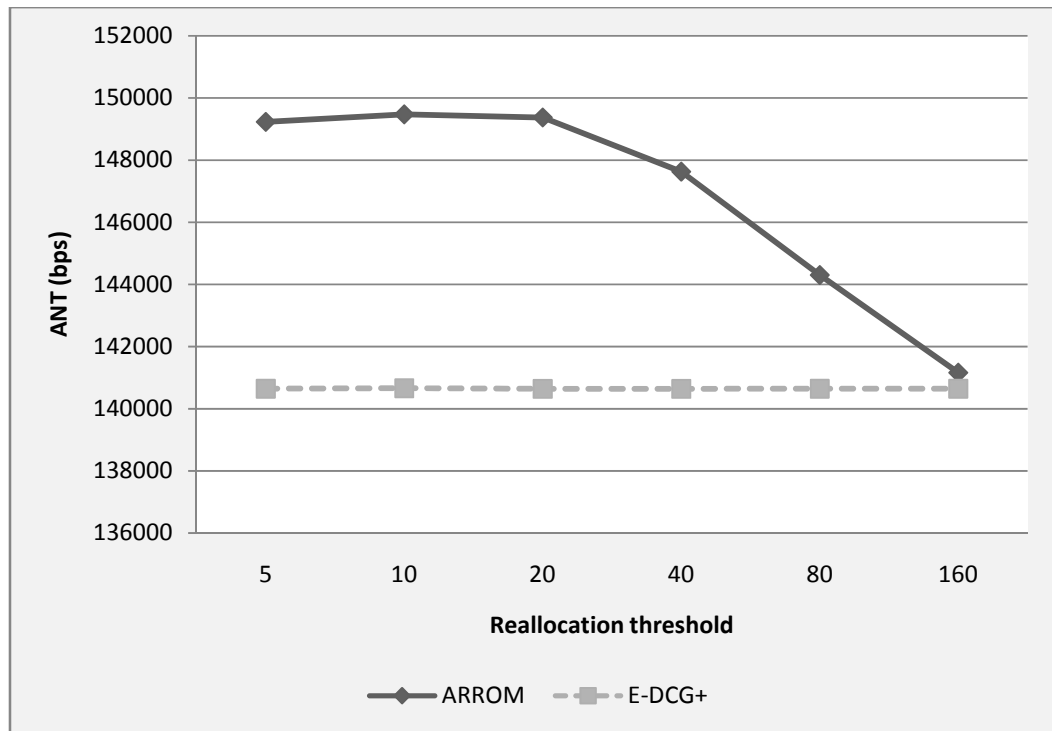


Figure 5: The ANT with different reallocation threshold values.

The best threshold here is 20, because it provides minimum average response time and maximum average network throughput with minimum reallocation frequencies. However,

the best threshold reallocates replicas not very often, because each reallocation process includes the following:

1. Moving the data object to the new location.
2. Deleting the data object from the previous location.
3. Broadcasting the new location of the data object to all MHs in the group.

To get high performance, we must keep the reallocation occurrences of the data object in moderate frequencies. Figure 4 shows that threshold 20 gives moderate reallocation frequencies between maximum and minimum reallocation frequencies.

5.5.2 Effects of Requested Data Object Size on ARROM:

We studied the effects of requested data object size on ARROM. Figure 6 shows the average response time of ARROM and E-DCG+ approaches with different requested data object sizes. The difference between the two approaches increases as the data object size increases, because the number of sent bytes increases. In other words, the delay of sending a set of bytes increases as the number of sent bytes increases.

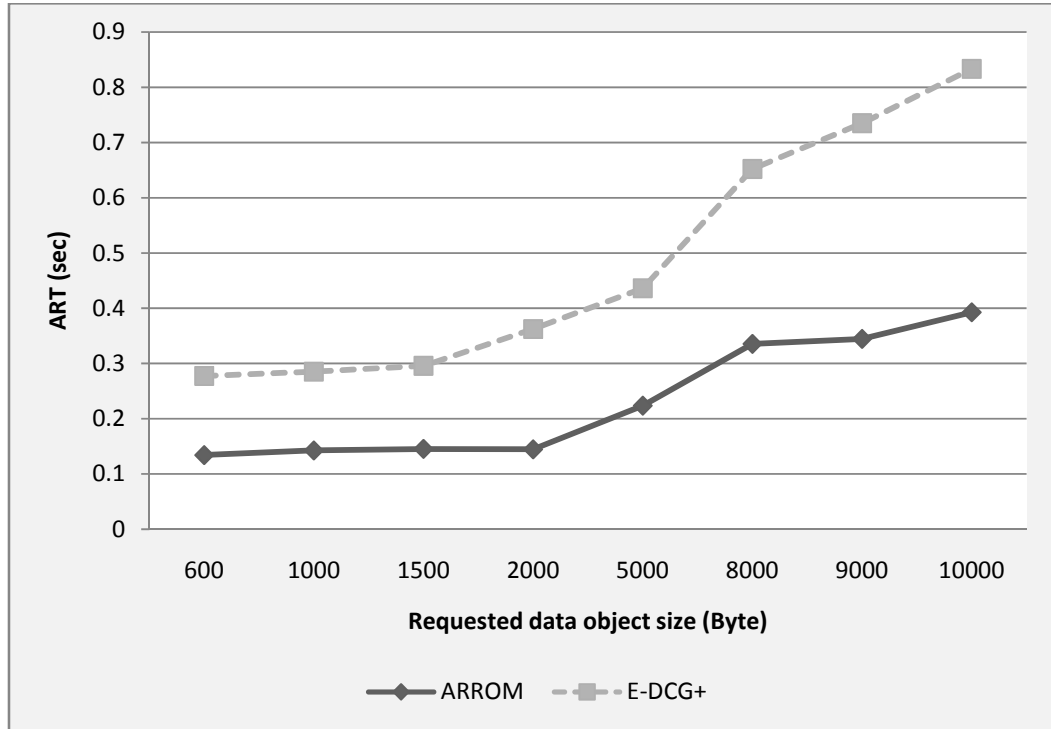


Figure 6: The ART of a request with different sizes of requested data object.

The same behavior is for average network throughput in Figure 7. Additionally, IR of ARROM for the average response time with different requested data object sizes equals to 51.99%. On the other hand, the IR of ARROM for the average network throughput with different requested data object sizes equals to 6.68%. As we mentioned in chapter two, each replica reallocation needs to broadcast the new location for all MHs in the group. This broadcasting reduces IR of average network throughput. In other words, increasing traffic in a network makes the network congested, which reduces the number of sent bytes per a unit of time (throughput).

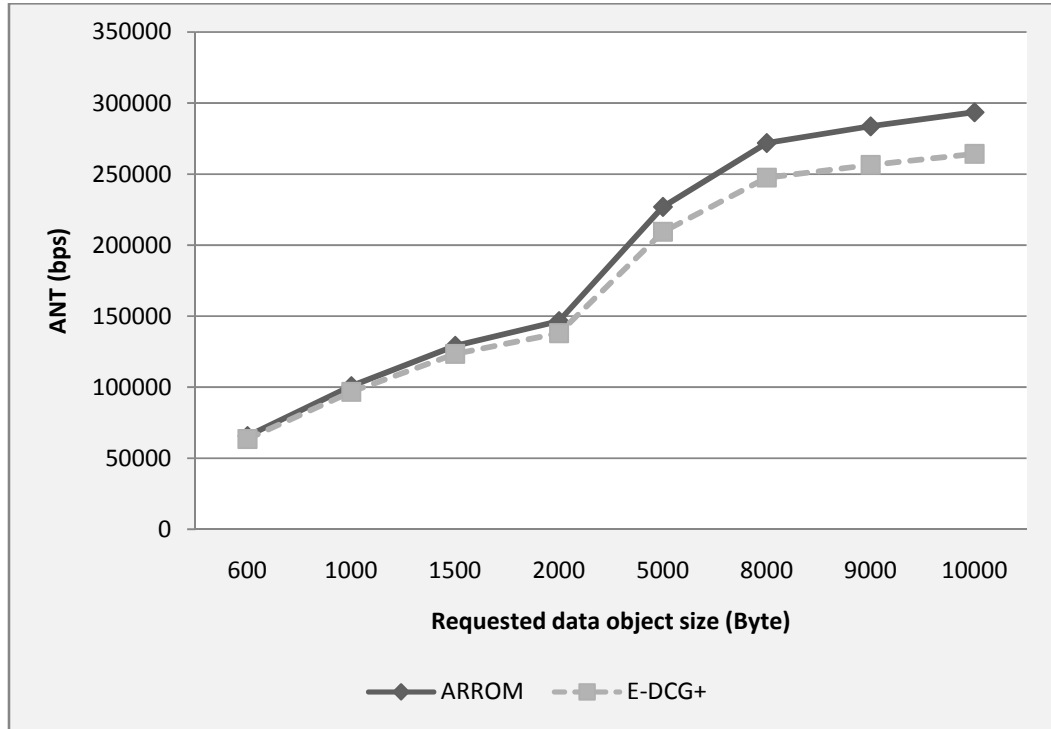


Figure 7: The ANT with different sizes of requested data object.

5.5.3 Effects of Read/Write Ratio on ARROM:

We studied the effects of read/write ratio on ARROM. Figure 8 shows the average response time of a request with different read ratios. The complement of each read ratio is write ratio, for example 50 read ratio has 50 write ratio at the same time. ARROM outperforms E-DCG+ in average response time with all read ratios. When all requests are write ones (*Read ratio* = 0), we get the maximum average response time difference between ARROM and E-DCG+. This difference decreases as the number of read requests increases. At the same time, ART of E-DCG+ and ART of ARROM decrease as the number of read requests increases. The reason behind this scale down is that write request has a number of propagation requests due to consistency management. Thus, when the number of write

requests increases, the network traffic increases i.e. the average response time becomes longer due to congestion.

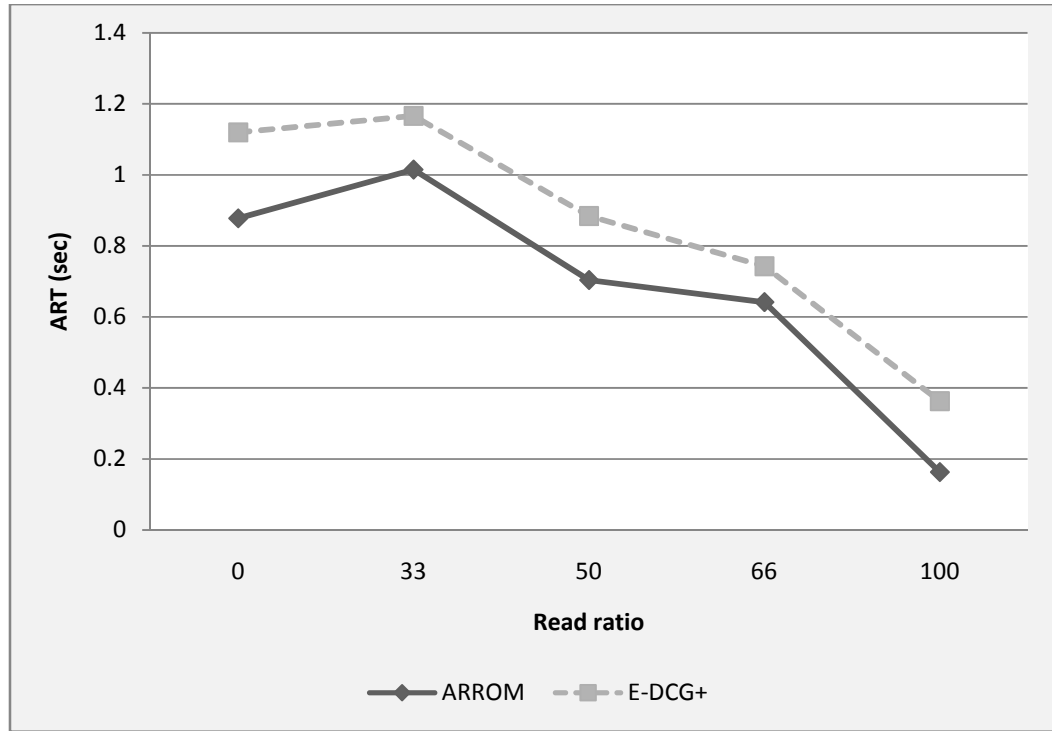


Figure 8: The ART of a request with different Read ratios.

Figure 9 shows the average network throughput for different read ratios. When all requests are read ones (*Read ratio* = 100), we get the best average network throughput and the best difference between ARROM and E-DCG+. At the same time, average network throughput of both E-DCG+ and ARROM decrease as the number of write requests increases due to propagation process. Notice that when we have any number of write requests (*Read ratios* = 0, 33, 50 and 66), E-DCG+ outperforms ARROM due to the reallocation condition, which ignores the cost between the new location and each server of the reallocated data object. We ignore this cost, because if it is included, then all replicas will

be close to each other, which disagree with the reason of creating them. In other words, data availability decreases in MANET, when replicas are close to each other. Additionally, most requests are read requests not write ones. However, ARROM may decide to reallocate a data object in insufficient location in terms of hops count between the new location and locations of other servers. This case increases the communication cost and leads us to another reallocation, which maybe insufficient too. This lopping affects average network throughput more than average response time, because our cost model depends on hops count which indicates relatively to response time more than network throughput.

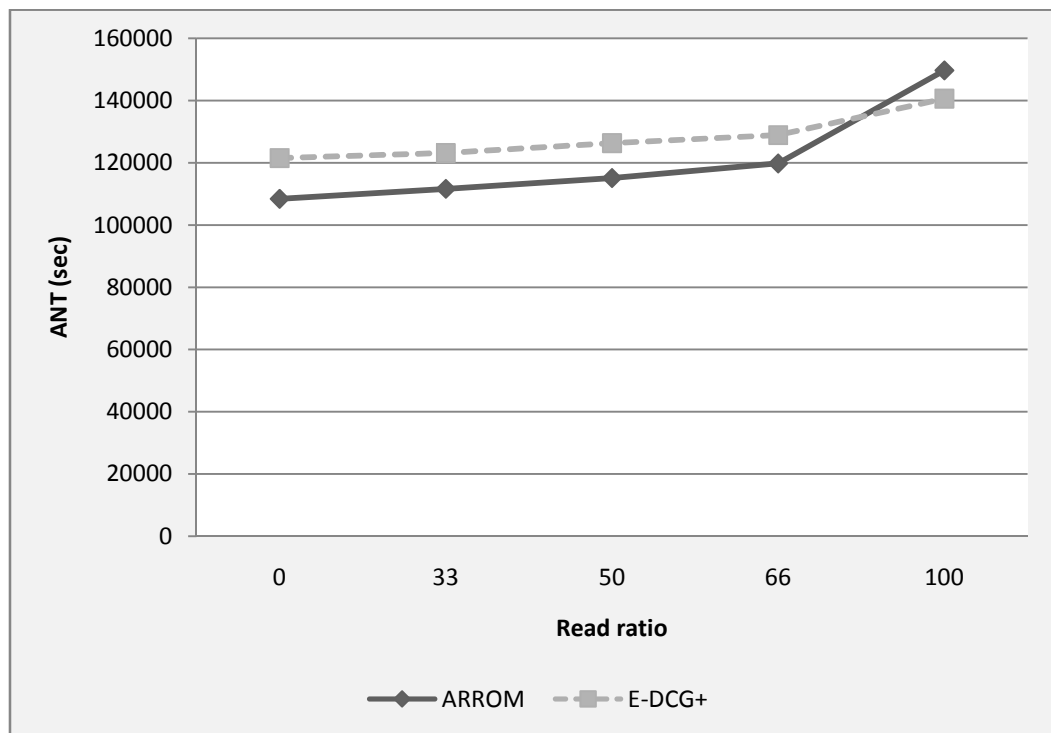


Figure 9: The ANT with different Read ratios.

5.5.4 Effects of Radio Transition Power on ARROM:

Finally, we studied the effects of radio transition power on ARROM. We show the radio transition power in dBm unit. 10 dBm, 15 dBm, 20 dBm and 40 dBm are equivalent to 282.6 meters, 376.78 meters, 502.5 meters and 1589 meters respectively.

Figure 10 shows the average response time of a request with different values of radio transition power. As the radio transition power increases, hops count between client and server decreases, as shown in Figure 11. Decreasing hops count makes ARROM performance very close to E-DCG+ performance, because reallocation frequencies decrease. As a result, reallocation threshold must be decreased as the radio transition power increases to keep the ARROM performance high.

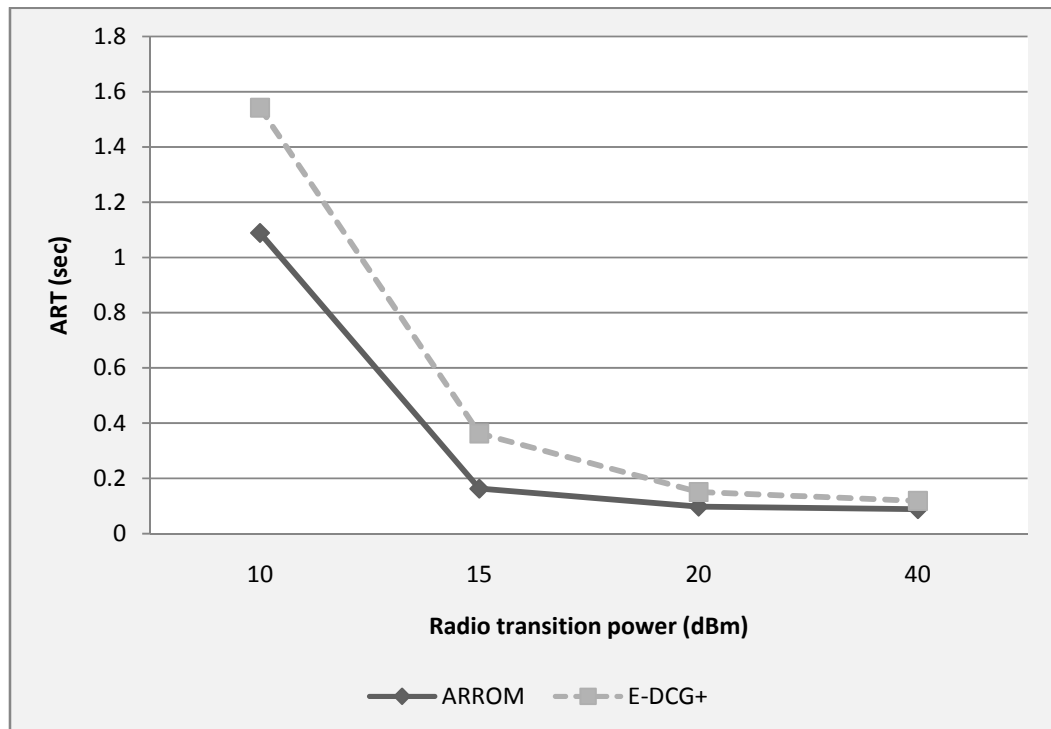


Figure 10: The ART of a request with different values of radio transition power.

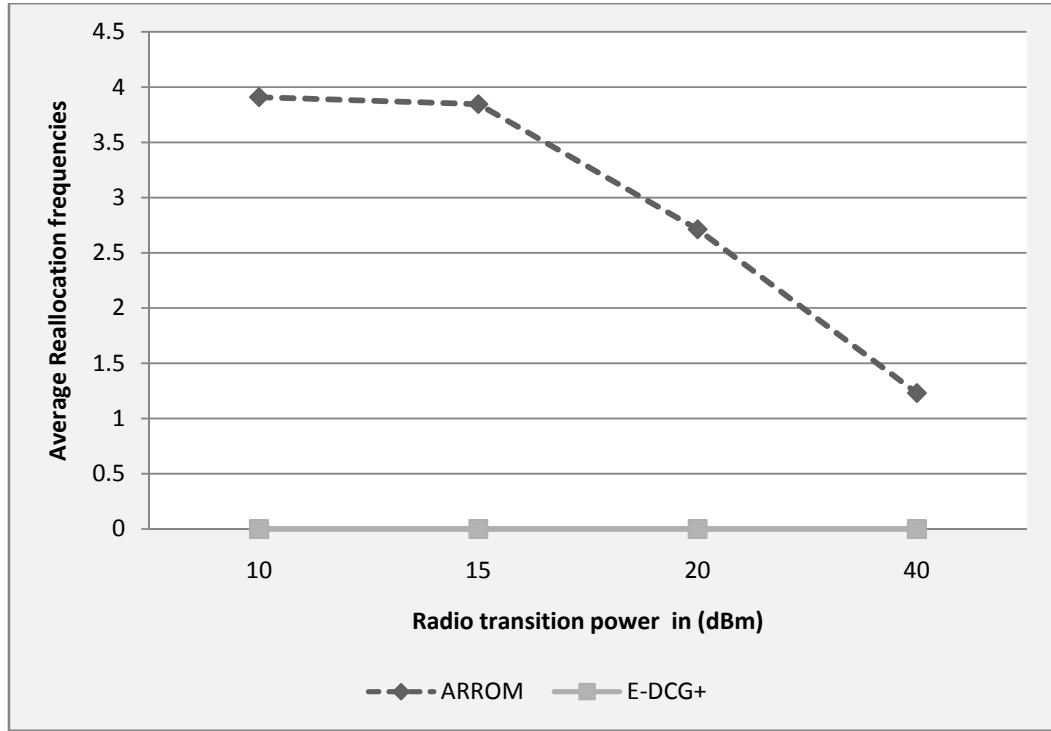


Figure 11: The average reallocation frequencies for different values of radio transition power.

Figure 12 shows that average network throughput increases as the radio transition power of nodes increases. This behavior is due to the reallocation frequencies as shown in Figure 11. As mentioned previously, each reallocation causes broadcast message, which increases the network traffic, therefore, the average network throughput is decreased. Notice that ARROM gives better average network throughput than E-DCG+ approach for all radio transition power values.

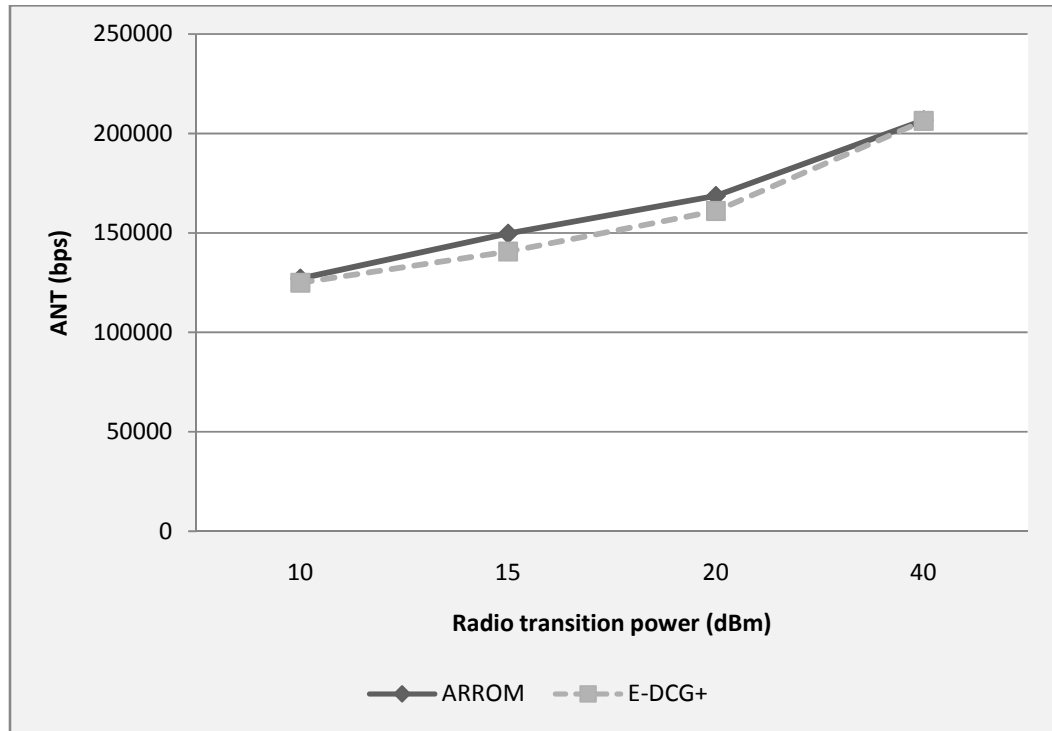


Figure 12: The ANT with different values of radio transition power.

5.5.5 Average Improvement Ratio of ARROM:

The results show that ARROM approach outperforms E-DCG+ approach in terms of average response time and average network throughput. Table 13 shows the improvement ratio of ARROM according to the results' categories and using equation (7).

Table 13: The improvement ratios of ARROM with different parameters

Results category	IR of average response time	IR of average network throughput
reallocation threshold	46.23%	4.41%
requested data object size	51.99%	6.68%
Read/Write ratio	24.72%	-6.76%
radio transition power	36.12%	3.09%
Average IR	39.76%	1.86%

For average response time in IR of ARROM, we assumed the ART of E-DCG+ as the max value and the ART of ARROM as the min value, because we decreased the average response time to improve it. The improvement of the average network throughput is by increasing it, therefore, the max value in IR of ARROM in average network throughput is the ANT of ARROM and the min value is the ANT of E-DCG+. In other words, ARROM generally decreases ART so the ART of ARROM is the min value and ARROM generally increases ANT so the ANT of ARROM is the max value. For example, the IR of ART for reallocation threshold category according to Figure 3 is as follows:

$$IR \text{ of threshold (5)} = \frac{0.362404383 - 0.159417378}{0.362404383} = 0.560111895$$

$$IR \text{ of threshold (10)} = \frac{0.362503794 - 0.160246861}{0.362503794} = 0.557944321$$

$$IR \text{ of threshold (20)} = \frac{0.362580744 - 0.146187203}{0.362580744} = 0.596814762$$

$$IR \text{ of threshold (40)} = \frac{0.362578805 - 0.186177618}{0.362578805} = 0.486518199$$

$$IR \text{ of threshold (80)} = \frac{0.362404383 - 0.250943126}{0.362404383} = 0.307560456$$

$$IR \text{ of threshold (160)} = \frac{0.362404383 - 0.301870636}{0.362404383} = 0.167033707$$

Average IR of threshold (5, 10, 20, 40, 80 and 160)

$$= \frac{0.560112 + 0.557944 + 0.596815 + 0.486518 + 0.307560 + 0.167034}{6}$$

$$= 0.462299 * 100\% = 46.23 \%$$

Other numbers are calculated in similar way. As mentioned in section 5.5.3, E-DCG+ outperforms ARROM in the average network throughput with different read/write ratios due to propagation process of each write request. Therefore, ANT of different Read/Write ratios is negative value. In general, ARROM outperforms E-DCG+ in both the average response time by 39.76% and the average network throughput by 1.86%.

6. CONCLUSION AND FUTRE WORKS:

This chapter gives the conclusion of studying ARROM in section 6.1 and introduces some future works that are related to this study in section 6.2.

6.1 Conclusion:

This thesis presents a new replication system called Automated Reallocation of Replicas Over MANET (ARROM), which attempts to get the best replica allocation during its lifetime in terms of communication cost. To achieve that, ARROM uses appropriate long relocation period and reallocates existing replicas separately during relocation period in order to reduce the communication cost of both accessing and reallocating replicas, which improves the usage of MHs' poor resources. We evaluated ARROM using GloMoSim simulator with four parameters, which are reallocation threshold, data object size, read/write ratio and radio transition power.

ARROM outperforms E-DCG+ in the average response time and in the average network throughput with different reallocation thresholds. In general, ARROM decreases the average response time by 46.23% and increases the network throughput by 4.41%. In terms of data object size, ARROM gives better results for the average response time by 51.99% and average network throughput by 6.68%. For read/write ratio parameter ARROM outperforms E-DCG+ in the average response time by 24.72%, but E-DCG+ outperforms ARROM in the average network throughput by 6.76%. This weak point is due to propagation process and reallocation condition. Finally, our proposed ARROM gives better

results for the average response time by 36.12% and the average network throughput by 3.09% using different radio transition power.

In general, simulation results show that ARROM decreases the average response time compared with E-DCG+ approach by 39.76%. Furthermore, ARROM increases the average network throughput by 1.86%. Additionally, we do not recommend using ARROM in an environment, whose requests are mostly write ones such as in registration systems. Especially, if increasing the average network throughput is more important than decreasing the average response time in such environment.

6.2 Future Works:

In the near future, we will study the effects of more parameters on ARROM such as the mobility level of MHs. In addition, we will insert the reallocation algorithm to other methods for example DRAM to get an enhancement method. At the same time, we want to improve the reallocation condition to include more performance metrics somehow, such as energy power of MHs and the bandwidth usage of MHs. In terms of security, we will use encryption and decryption techniques in transferring data objects to protect them from intruders.

We will improve ARROM by adding an algorithm that reallocates replicas even if one of their servers has a request in progress.

REFERENCES

- Bagrodia R., Meyer R., Takai M., Chen Y-A., Zeng X., Martin J. and Song H. Y. (1998), PARSEC: A Parallel Simulation Environment for Complex System, **Proceedings of IEEE** 0018-9162/98, October 98.
- Bajaj L., Takai M., Ahuja R., Tang K., Bagrodia R. and Gerla M. (1999), GloMoSim: A Scalable Network Simulation Environment, **Technical report for Computer Science Department CA 90095, University of California.**
- Bakht H. (2004), WIRELESS INFRASTRUCTURE-Understanding mobile ad hoc networks, <http://www.computingunplugged.com/issues/issues200406/00001301001.html>
- Cheng R-H., Wu T-K., Yu C. W., and Kuo C-H. (2006), An Altitude Based Dynamic Routing Scheme for Ad Hoc Networks, **Proceedings of the WASA 2006 and of the LNCS 4138**, pp. 609 – 619, 2006, Published in Springer-Verlag Berlin Heidelberg 2006.
- Elmasri R. and Navathe S. B. (2007), **Fundamentals of Database Systems**, (5th), United States of America: Addison Wesley.
- Fife L. D. and Gruenwald L. (2003), Research Issues for Data Communication in Mobile Ad-Hoc Network Database Systems, **Proceedings of the SIGMOD Record**, Vol. 32, No. 2, June 2003.
- Guy R., Reither P., Ratner D., Gunter M., Ma W. and Popek G. (1998), Rumor: Mobile Data access Through Optimistic Peer-to-Peer Replication, This work was supported by **the United States Defense Advanced Research Projects Agency** under contract number DABT63-94-C-0080 (1998).
- Hara T. (2003), Replica Allocation Methods in Ad Hoc Networks with Data Update, Netherlands: **Kluwer Academic Publisher**, Mobile Networks and Applications 8,343-354, 2003.
- Hara T., Loh Y-H. and Nishio S. (2003), Data Replication Methods Based on the Stability of Radio Links in Ad Hoc Networks, **Proceedings of the 14th International Workshop on Database and Expert Systems Applications (DEXA'03)** 1529-4188/03, 2003 IEEE.
- Hara T., Murakami N. and Nishio S. (2004), Replica Allocation for Correlated Data Items in Ad Hoc Sensor Networks, **Proceedings of the SIGMOD Record**, Vol. 33, No. 1, March 2004.
- Hara T. (2005), Data Replication Issues in Mobile Ad Hoc Networks, **Proceedings of the 16th International Workshop on Database and Expert Systems Applications (DEXA'05)**1529-4188/05 2005 IEEE.

Hara T. (2006), Data Replication and Update Management in Mobile Ad Hoc Networks (Invited Paper), **Proceedings of the 17th International Conference on Database and Expert Systems Applications (DEXA'06)**0-7695-2641-1/06 2006 IEEE.

Hara T. and Madria S. K. (2006), Data Replication for Improving Data Accessibility in Ad Hoc Networks, **Proceedings of the IEEE CS, CASS, ComSoc, IES, and SPS**, 1536-1233/06 2006 IEEE.

Hayashi H., Hara T. and Nishio S. (2005, a), A Replica Allocation Method Adapting to Topology Changes in Ad Hoc Networks, **Proceedings of DEXA 2005, and Proceedings of LNCS 3588**, pp. 868-878, 2005.

Hayashi H., Hara T. and Nishio S. (2005, b), Updated data dissemination methods for updating old replicas in ad hoc networks, **Proceedings of the Pers Ubiquit Comput** 9:273-283, published online: 15 January 2005.

Hayashi H., Hara T. and Nishio S. (2006), On Updated Data Dissemination Exploiting an Epidemic Model in Ad Hoc Networks, **Proceedings of the BioADIT 2006**,LNCS 3853, PP. 306-321, 2006.

Huang J-L., Chen M-S. and Peng W-C. (2003), Exploring Group Mobility for Replica Data Allocation in a Mobile Environment, **Proceedings of the CIKM'03**, November 3–8, 2003, Copyright 2003 ACM 1581137230/03/0011.

Mondal A., Madria S. K. and Kitsuregawa M. (2006), CADRE: A Collaborative replica allocation and deallocation approach for Mopile-P2P networks, **Proceedings of the 10th International Database Engineering and Applications Symposium (IDEAS'06)** 0-7695-2577-6/06 2006 IEEE.

Nuevo J. (2004), A Comprehensible GloMoSim Tutorial, Retrieved November 15, 2008 from www.sm.luth.se/csee/courses/smd/161_wireless/glomoman.pdf

Özsu M. T. and Valduriez P. (1999), **Principles of Distributed Database Systems**, (2nd) United States of America: Prentice Hall, Paperback: 666 pages, (January 29, 1999).

Pabmanabhan P. and Gruenwald L. (2006, a), DREAM: A Data Replication Technique for Real-Time Mobile Ad-hoc Network Databases*, **Proceedings of the 22nd International Conference on Data Engineering (ICDE'06)**8-7695-2570-9/06 2006 IEEE. And **Proceedings of the NFS** Grant No. IIS-0312746.

Pabmanabhan P. and Dr.Gruenwald L. (2006, b), MANAGING DATA REPLICATION IN MOBILE AD_HOC NETWORK DATABASES (Invited Paper)*, **Proceedings of the IEEE** (1-4244-0429-0/06), And **proceedings of the National Science Foundation (NSF)**, grant No. IIS-0312746

Shi K. and Haas Z. J. (2007), Quantitative Analysis of Partition Statistics and their Impact on Data Replication in MANETs, **Proceedings of the 6th International Symposium on Parallel and Distributed Computing (ISPDC'07)** [0-7695-2936-4/07 2007 IEEE.

Sleit A., Almobaideen W., AlAreqi S. and Yahya A. (2007), A Dynamic Object Fragmentation and Replication algorithm in Distributed Database Systems, **American Journal of Applied Sciences**, Vol. 4, No. 8, 613-618, 2007.

Takai M., Martin J., Meyer R., Park B. and Song H. Y. (1999), PARSEC User Manual, PARSEC was developed by members of **the UCLA Parallel Computing Laboratory**, written by Richard A. Meyer and Rajive Bagrodia, Revised in September 1999.

Viswacheda D. V., Arifianto M. S. and Barukang L. (2007), Architectural Infrastructural Issues of Mobile Ad Hoc Network Communications for Mobile Telemedicine System”, **Proceedings of the 4th International Conference: Sciences of Electronic, Technologies of Information and Telecommunications(SETIT)**. March 25-29, 2007 – TUNISIA

Wujuan L. and Veeravalli B. (2003), An Adaptive Object Allocation and Replication Algorithm in Distributed Databases, **Proceedings of the 23rd International Conference on Distributed Computing Systems Workshops (ICDCSW'03)** 0-7695-1921-0/03 2003 IEEE.

Zeng X., Bagrodia R. and Gerla M. (1998), GloMoSim: A Library for Parallel Simulation of Large-scale Wireless Networks, **Proceedings of the IEEE** 1087-4097/98.

APPENDIX A

1. About GloMoSim

GloMoSim is a scalable environment for wireless and wireline communication networks. It uses a parallel discrete-event simulation which provided by Parsec (Nuevo, 2004). "GloMoSim simulates networks with up to thousand nodes linked by a heterogeneous communications capability that includes multicast, asymmetric communications using direct satellite broadcasts, multi-hop wireless communications using ad-hoc networking, and traditional Internet protocols" (Nuevo, 2004). Table 14 lists the GloMoSim models which are available at each layer.

Table 14: The GloMoSim models currently available at each of the major layers (Nuevo, 2004)

Layer	Models
Physical (Radio Propagation)	Free space, Two-Ray
Data Link (MAC)	CSMA, MACA, TSMA, 802.11
Network (Routing)	Bellman-Ford, FSR, OSPF, DSR, WRP, LAR, AODV
Transport	TCP, UDP
Application	Telnet, FTP

2. GloMoSim Installation on Windows XP

Before installation, the following software must be installed correctly:

- Microsoft VC++ version 6.0 (Essential)
- JAVA JRE version 1.2 or higher (For VT)
- JAVA SDK version 1.2 or higher (For VT)

This is a step-by-step installation guide for GloMoSim on Microsoft Windows XP

1. Copy "glomosim" and "parsec" directories to the "c:\\" directory: C:\glomosim and C:\Parsec

2. Set *pcc* environmental variables (For Parsec)

1) My Computer -> Properties -> Advanced -> Environmental Variables

2) New "PCC_DIRECTORY", value = "C:\parsec"

3) Set *path* "C:\parsec\bin;C:\glomosim\bin;C:\Program Files\Microsoft Visual Studio\VC98\Bin;C:\Program Files\Microsoft Visual Studio\Common\MSDev98\Bin;C:\Program Files\Microsoft Visual Studio\Common\Tools;C:\Program Files\Microsoft Visual Studio\VC98\Include;"

3. Set VC6.0 environmental variables (for both user variables and system variables)

1) Set *include*:

"C:\Program Files\Microsoft Visual Studio\VC98\MFC\Include;C:\Program Files\Microsoft Visual Studio\VC98\Include"

2) Set *lib*:

"C:\Program Files\Microsoft Visual Studio\VC98\MFC\Lib;C:\Program Files\Microsoft Visual Studio\VC98\Lib"

4. Check *pcc* environment by "set pcc" in DOS prompt (cmd)

You should get: " PCC_DIRECTORY = C:\parsec"

5. GloMoSim Installation

1) Go to glomosim\main, do "makent.dat"

2) Go to glomosim\bin, find "glomosim.exe"

3) Test glomosim. Under DOS: "glomosim config.in"

6. GloMoSim is now ready to use.

إعادة ترتيب نسخ البيانات بشكل تلقائي في الشبكات المتنقلة العشوائية

إعداد

إيناس إسماعيل أبوقدوم

المشرف

الدكتور عزام طلال سليط

المشرف المشارك

الدكتور وسام عبدالرحمن المبيضين

ملخص

في الشبكات المتنقلة العشوائية (MANET) لا يوجد بنية تحتية ثابتة، لذا فإن شكل الشبكة يتغير باستمرار. كما أن وظائف الأجهزة اللاسلكية هي أطراف تتصل مع بعضها البعض وموجهات أيضا. هذه الوظائف تؤثر على الأجهزة اللاسلكية لأن طاقتها محدودة، وتستخدم سعة نقل محدودة ومواردها ضعيفة. نسخ البيانات هي تقنية استخدمت لمواجهة هذه التحديات، والتي تحسن تواجد البيانات لكن مع زيادة في استهلاك مكان التخزين وفي الاتصالات.

قضايا النسخ يمكن أن تصنف إلى ثلاثة فئات: أولا تغيير مكان النسخ، والذي يعني من ومتى وأين وكيف يتم تحديد أماكن هذه النسخ. الفئة الثانية هي إدارة التماثل لكي نحافظ على هذه النسخ متماثلة كلما حدث تعديل على أي منها. الفئة الثالثة هي إدارة المكان والتي تعني أن الجهاز الطالب يجب أن يعرف مكان البيانات المطلوبة أو أيًا من نسخها، وإلا سيقوم بنشر طلبه على كل الأجهزة اللاسلكية في الشبكة. كل نظام نسخ يوفر طرق لمعالجة هذه الفئات بحيث تنجز معظم أعمالها كل فترة زمنية محددة نتيجة لحركة الأجهزة اللاسلكية. هذه الفترة الزمنية تسمى فترة تغيير المكان. لتقليل تكلفة الإتصال للوصول إلى نسخ البيانات، يجب علينا تغيير أماكنهم بشكل متكرر، حسب حركة الأجهزة اللاسلكية ونمط الطلب. لذا يجب أن نجعل فترة تغيير المكان قصيرة لكن هذا يجعل الشبكة مكتنزة نتيجة لإعادة تنفيذ الكثير من الأعمال دون الحاجة لذلك. من ناحية أخرى، إن فترة تغيير المكان الطويلة تعني أن تغيير أماكن نسخ البيانات سيكون محدوداً ومتجاهلاً لنمط الطلب للأجهزة اللاسلكية.

هذه الأطروحة تقدم نظام نسخ جديد يسمى "إعادة ترتيب نسخ البيانات بشكل تلقائي في الشبكات المتنقلة العشوائية (ARROM)" والذي يحاول أن يضع نسخ البيانات في أفضل مكان لها طوال فترة حياتها وهذا وفقاً لتكلفة الإتصال. يستخدم ARROM فترة تغيير مكان طويلة مناسبة، ويعمل على تغيير أماكن نسخ البيانات الموجودة خلال هذه الفترة، وذلك لتقليل نفقة الوصول لهذه النسخ ونفقة تغيير أماكنها، بحيث نحسن إستغلال موارد الأجهزة اللاسلكية. قيمنا ARROM باستخدام برنامج المحاكاة (GloMoSim)، وقد أظهرت النتائج أن ARROM تحسن معدل زمن الإستجابة بنسبة %39.76 و معدل كمية البيانات المنقولة في وحدة الزمن (Throughput) بنسبة %1.86 بالمقارنة مع أشهر طرق النسخ الأخرى (E-DCG+)، التي تغير أماكن نسخ البيانات فقط عند فترة تغيير الأماكن.