A New Cache Invalidation and Searching Policy for Mobile Ad Hoc Networks

Yungho Leu, Jen-Jou Hung and Ming-Ben Lin
National Taiwan University of Science and Technology
Department of Information Management
43, Keelung Road, Section 4, Taipei
Taiwan
http://www.cs.ntust.edu.tw/tw/academic/yhl/

Abstract: Although cache invalidation in cellular wireless networks has been well studied, it is not adequately addressed in mobile ad hoc networks (MANET). The existing cache invalidation policies designed for cellular wireless networks are not well-suited for a MANET which features in multi-hop message relay, frequent link disconnection and power-constrained operation. In this paper, we proposed a New Cache Invalidation and Searching (NCIS) policy for MANETs. Unlike ACOD [2] which uses broadcast for validation and searching for a valid copy of a cached data item, NCIS uses unicast to validate local copy and to determine a maximal search range. If the local copy is valid, it is used to answer the query; otherwise, the mobile client finds a valid copy by searching within a limited search range. By accessing the nearest valid copy of a required data item and reducing the broadcast overhead, NCIS offers a better performance than that of ACOD in terms of energy consumption and response time. Several experiments are conducted to study the performance of ACOD and NCIS.

Key–Words: Mobile Ad Hoc Networks, Data Caching, Cache Invalidation and Searching

1 Introduction

Without pre-existing infrastructure, a mobile ad hoc network (MANET) can be easily deployed for the applications such as battlefield and disaster recovery. It can also be used as a wireless extension for the users to access the Internet in an area without network infrastructure [1]. MANETs features in multi-hop message relay, frequent link disconnection and power-constrained operation, which complicate the design of network protocols and application development. While a tremendous amount of research has been done on routing protocols [10] for MANETs, little work has been done on providing data services in MANETs. Recently, we have witnessed a growing interest on providing data caching to promote data services in MANETs. In [3][4], Cao et al. proposed *cache data* and *cache path* policies for MANETs. *Cache data* policy determines how to cache the copies of a data item in a MANET, while *cache path* policy dictates how to cache the path to a node that hosts a copy of a data item to help to reduce search efforts for the data item. In [5], Du and Gupta proposed a cooperative caching service which aims to download a copy of a cached data item efficiently and to cooperatively manage the cache space among the mobile nodes.

In [2] Lim et al. proposed two paradigms for data caching in MANETs–push and pull. In the push paradigm, a database server pushes, through broadcasting, the updates information (in an invalidation report) to mobile clients. A mobile clients then uses invalidation reports to maintain the consistency of its cached data items. In the pull paradigm, a mobile client validates its cached data items on demand. When a mobile client needs a data item, it will search for a valid copy from the network including itself and the database server, and then downloads the copy from the discovered node. Two policies for the push paradigm are proposed–MTS and MTS+UIR. They are all adapted for MANETs from the IR approach proposed by Barbara and Imielinski [6]. We argue that they are difficult to implement due to the unpredictable network delay of IR messages in a MANET. Unlike in a one-hop network such as a cellular wireless network for which the IR approach was initially proposed, a mobile node is not synchronized with its database server in a MANET due to the multi-hop communication pattern of a MANET. The information carried in an IR can be outdated when it arrives the mobile node. That is, many updates may have occurred after the server sending an IR and before the IR arriving the mobile client. In other words, the state of the database is not correctly reflected in the IRs, which fails the IR approach. In contrast, the pull paradigm can be effectively implemented. However,

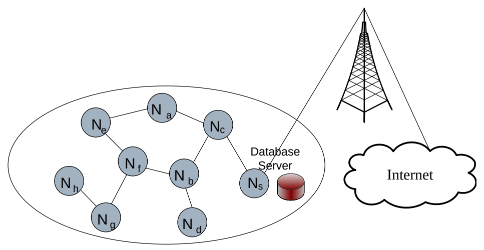

Figure 1: System Model

the proposed ACOD policy is not efficient due to its excessive use of broadcasts in searching for a valid copy of a data item. In this paper, we propose a new cache invalidation and searching policy, abbreviated as NCIS. In NCIS, a mobile node validates its local copy of a data item and determines a searching range through unicast routing. It only searches with a minimal broadcast range for a valid copy when it has no local copy of the required data item or its local copy is not valid. By accessing the nearest valid copy and reducing the broadcast overhead, NCIS offers a better performance than that of ACOD in terms of energy consumption and response time. The rest of this paper is organized as follows. Section 2 presents the background of this paper, including the system model for data service in MANETs and a brief review of the ACOD policy. In Section 3, we present the NCIS policy. In Section 4, we present the performance simulation results for ACOD and NCIS. Finally, in Section 5, we conclude this paper.

2 Background

2.1 System Model

The system model for data application in a MANET has been discussed by several researchers[2][5]. The system model we use in this paper is a summary of their models. Fig. 1 shows the system model which is summarized in the following:

- **Mobile Nodes:** There are n mobile nodes in the system. We use N= $\{N_a, N_b, \cdots, N_n\}$ to denote the set of mobile nodes. A mobile node, or simply node, has a cache in its local storage, and may issues queries for data items from the database server. We assume that each query requests only one data item. The mobile nodes are free to move in a designated area.

- **Database server:** One of the mobile nodes assumes the role of a database server. The database server, or simply a server, contains a database and is able to communicate with the Internet through an Access Point, a satellite or a WiMAX base station. A mobile nodes can communicate with the database server, directly if the server is within its radio range, or indirectly through multi-hop relay by other mobile nodes. We assume that there are m data items in the database. Data items are numbered from 1 to m. We use D= $\{D_1, D_2, \cdots, D_m\}$ to denote the set of data items in the database.

- **Data Updates:** The database server may repeatedly updates the data items in the database. To help a mobile node to keep track of the version of a cached copy of a data item, each copy of a data item is associated with a timestamp. The timestamp stores the time when the copy is created in the server.

2.2 Review of ACOD

Since this work is motivated by ACOD cache invalidation and searching scheme, we briefly review ACOD in this section. In ACOD[2], in answering a query, a mobile node searches, in the mobile nodes that lie in between itself and the server, for copies of its required data item. The search will be performed by broadcasting. When a search reaches the server, the server checks the validity of the most recent copy in the intermediate nodes on the search path. If the most recent copy on the search path is valid, the server reports an ACK with a valid bit to the node with the most recent copy; otherwise, the server sends the current copy (in the server) to the node with the most recent copy to refresh its local copy. Afterwards, the node with the most recent copy sends an ACK to the requester, and the requester accesses the required data item from the node with the most recent copy. There are maybe more than one ACK received by the requester. The requester will access the copy from the path whose ACK arrives first. Fig. 2 and 3 show a scenario to illustrate ACOD. In Fig.2, node N_p issues a query on data item k, which exists in its local cache. N_p prepares a request packet containing (item=k, Ts=10, Hp=4), meaning that N_p has a local copy of k with timestamp 10, and N_p is 4 hops away from the server. The packet is broadcasted toward the server. Each node on a broadcast path will attach its node id in the packet if it has a local copy of the same data item and the timestamp of its local copy is newer than the timestamp in the packet. After attaching the node id, the timestamp in the packet is changed to the timestamp of the newer copy. In Fig.2, node N_e and N_b will attach their node ids in their respective broadcast packets.

When the packet from node N_b reaches the server, as shown in Fig3. The server compares the timestamp

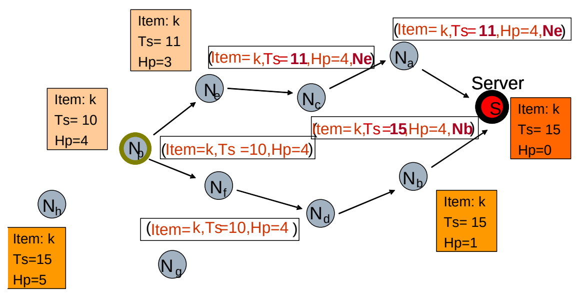

Figure 2: Data invalidation in ACOD

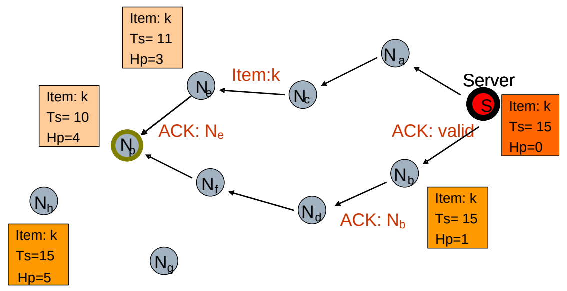

Figure 3: Accessing a valid copy in ACOD

in the packet with the timestamp of the server's local copy, and finds that N_b owns a valid copy; the server then replies with an ACK with a valid bit saying that the copy on this path is valid. On the other hand, when the packet from N_e reaches the server, the server replies with an the new value of item k to refresh the local copy in N_e; node N_e then sends an ACK to N_p . Depending on which ACK first arrives N_p, N_p will access a valid copy either from node N_b or from N_e.

While ACOD is an initial work on cache invalidation and searching in MANETs, its has the following disadvantages:

1. A requesting node has to broadcast a request packet even though its local copy is valid. Broadcast may induce broadcast storm and broadcast is in itself costly.

2. The node from which the requesting node accesses a valid copy depends on the receiving order of ACKs. The requesting node is not guaranteed to access a valid copy its the nearest neighbor. For example, in Fig. 3, node N_h has a valid copy and is nearest to N_p, but N_p accesses a valid copy from N_b if its ACK arrives first.

3. The server has to validate a copy for each different path. There might be many paths to validate. This will cause a serious overhead on the server. The server may quickly run out of power.

3 The NCIS policy

To remedy the disadvantages of ACOD, we propose a new cache invalidation and searching policy called NCIS. The main idea behind NCIS is simple. In NCIS a requesting node first validates its local copy by sending a request packet through unicast routing (e.g., AODV) to the server. The server replies with the current timestamp of the request data item along the reverse path of the arriving request packet. Each node on the reverse path attaches its id and a hop count in the response packet if it has a valid copy for the required data item. Upon receiving the response packet from the server, the requesting node validate its local copy; if the local copy is valid, the requesting node uses it to answer the query; otherwise, it broadcasts a request packet to its neighbors with a minimal broadcast range determined by the response packet. The policy consists of four steps which are detailed in the following.

1. Sending a request packet: The requesting node sends a request packet containing the requested data item to the server.

2. Replying with a response packet: Upon receiving a request packet, the server replies within a response packet the current timestamp of the requested data item and a hop count which is equal to 0 through the reverse path. Each node on the path checks its local copy; if it has a valid copy, it resets the hop count to 0, otherwise, it adds 1 to the hop count. After updating the hop count, it transmits the packet to the next node along the path.

3. Broadcasting a search packet: When the requesting node receives the response packet from the server, it adds the hop count by 1. If it has a valid local copy, the node uses it to answer the query; otherwise, the node broadcasts a search packet with the requested data item and its current timestamp and the hop count. The hop count serves as TTL (Time-To-Live) which limits the range of the broadcast. That is, each node that receives a search packet will subtract 1 from the hop count. Once the hop count reaches 0, the search packet is dropped. Each node receives a search packet searches it local cache for a valid copy. It it has one, then, it sends an ACK to the requesting node; otherwise, it rebroadcasts the search packet.

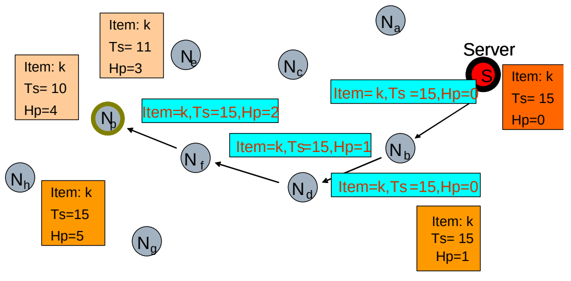

Figure 4: Replying with a response packet in NCIS

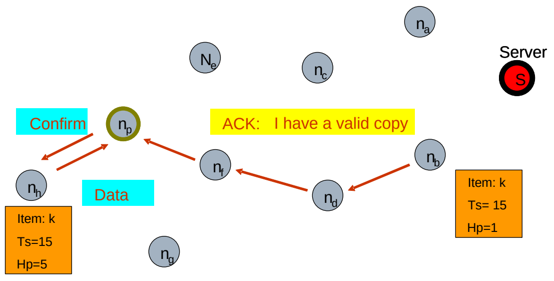

Figure 5: Accessing a valid copy in NCIS

4. Accessing a valid copy: The requesting node will only act on the first received ACK by accessing a valid copy from the node whose ACK receives first.

Fig. 4 shows the step of replying a request packet. In Fig. 4, server receives a request packet from node N_p for data item k. The server then sends a response packet containing the current timestamp of data item k, which is 15, and an initial hop count Hp=0 to node N_b. Since N_b has a valid copy of data item k, it resets the timestamp in the response packet to 0, and send the packet to the next node on the path, which is N_d. Since N_d has no valid copy, it adds 1 to hop count Hp and sends it to the next node. When node N_p finally receives the response packet, it will adds one to Hp which equals to 3. Node N_p uses the received timestamp, which is equal to 15, to check the validity of its local copy. It turns out that the local copy is invalid since its timestamp is less than 15. It then broadcasts a search packet with hop count value (TTL) equal to 3. Node N_h receives the search packet, as shown in FIg 5, and replies an ACK to N_p, N_p then sends a CONFIRM packet to node N_h to access the valid copy. NCIS has the following two advantages:

- A requesting node validates its local copy before searching a valid copy from its neighbor nodes. As such, the number of broadcasts is reduced.
- When searching is required, the searching range is minimized.

These two advantages render small response time and low energy consumption in answering a query. To further enhance the performance of NCIS, we also implement the cache path mechanism into NCIS. For a detail description of the cache path mechanism, please refer to [4].

4 Performance Evaluation

To compare the performance of NCIS and ACOD, we conducted several experiments using the ns-2.1b8-mcast [9].

4.1 Simulation Model

In the simulation, we model a area of 1500m $\times$ 1000m, in which 50 mobile nodes move freely with the speed of 2 m/s. Each mobile node has a cache of 500K bytes. One of the mobile nodes is designated as the database server which contains 1000 data items. The data items are numbered from 1 to 1000. Each data item is 10K bytes. The server updates its data items in the database repeatedly. Updates are modelled by update events. We assume that the inter-arrival time of update (T_u) events follows an exponential distribution with mean value ranging from 0.1 to 200 seconds. The default value of T_u is set to 80 seconds. When an update event occurs, 5 to 15 data items are updated. The ids of the updated data items are randomly selected from 1 to 1000. Queries are modelled by query events, and the query inter-arrival time (T_q)is exponentially distributed with the mean value equal to 80 seconds. Each query accesses only one data item and the id of the accessed data item ranges from 1 to 1000. Mobile clients access the data items with a biased pattern. The access pattern follows a zipf-like distribution in which the data item k is accessed with a probability P(k) which is determined by the following equation:

$$P(k) = \begin{cases} \frac{(\frac{1}{k})^\theta}{\sum_{j=1}^{1000}(\frac{1}{j})^\theta} & k = 1 \sim 1000 \\ 0 & otherwise, \end{cases}$$

where $0 \leq \theta \leq 1$. When $\theta = 0$, it follows a uniform distribution. When $\theta = 1$, the access pattern is very skew. Data item 1 receives the highest access probability, and data item 2 receives the second highest access probability, and so on. In this experiment, we set θ to 0.8, which implies a 80-20 rule [2] meaning

Table 1: Parameters in the Experiments

Parameters	Values
Simulated network dimensions	1500m $\times$ 1000m
Number of mobile nodes	50
Radio range	250m
Node moving speed	2m/s
Cache size of mobile node	500k Bytes
Query inter-arrival time, T_q	80 seconds
Number of data server	1
Number of data items	1000
Data item size	10k Bytes
Update inter-arrival time, T_u	80 (0.1-200)seconds
no. of data items/ update (N)	10(5-15)
Routing protocol	AODV
Degree of data skew (θ)	0.8

Table 2: Parameters of energy consumption

point to point, send	1.9 $\times$ packet-size+454 μ joule
point to point, receive	0.5 $\times$ packet-size+356 μ joule
broadcast, send	1.9 $\times$ packet-size+266 μ joule
broadcast, receive	0.5 $\times$ packet-size+56 μ joule

that the top 20 percent of the data items receives 80 percent of access probability. The parameters of the simulation are listed Table 1. To compare the energy consumption of NCIS and ACOD, we assume that the network adaptor Lucent WaveLan 2.4 GHz is used for the MAC layer and PHY layer of the simulation. The energy consumption parameters [8] of this network adaptor are listed in Table 2. The unit of the packet size is byte.

4.2 Simulation Results

In the following, we compare NCIS and ACOD in terms of the average hop-distance, query response time, server energy consumption and client energy consumption for a mobile client to access a valid copy of a queried data item. The definition of the comparison metrics are listed in the following:

- **Average hop-distance:** Number of hops from the node issuing a query on a data item to the mobile node providing a valid copy of the data item;
- **Query response time** The elapsed time since a mobile node poses a query till the time when the mobile node downloads a valid copy of the required data item;
- **Server energy consumption:** The total energy consumed by the server to answer a query; and
- **Client energy consumption:** The total energy consumed by all the mobile node to answer a query.

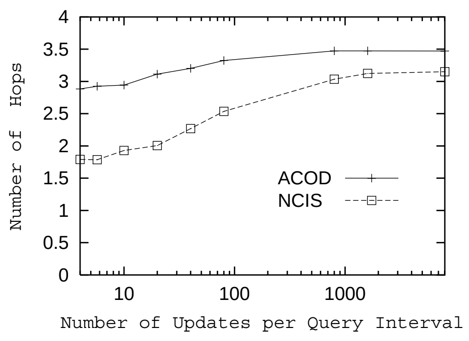

Figure 6: Average Number of Hops

4.2.1 Performance of NCIS and ACOD

Fig. 6 shows the comparison on hop-distance metric. The unit on X-axis is number of data items updated in between two queries. It is equivalent to $T_q/T_u \times N$, where T_q is the mean query inter-arrival time and T_u is the mean update inter-arrival time and N is the number of data items updated in an update event. In the paper, we call the value of $T_q/T_u \times N$ the update rate.

Fig. 6 shows that the average hop-distance increases as the update rate increases. This is because that when the update rate is high, the cached data copies easily becomes invalid and the mobile node has to access a valid copy of a data item from the server. It also shows that NCIS outperforms ACOD by about 33.3 percent in terms of average hop-distance when the update rate is less than 11. This is because that in NCIS a mobile node more often accesses a valid copy from a node that is closest to it, while in ACOD a mobile node may access a copy in a node that is far away from itself.

Fig. 7 shows the response time comparison of NCIS and ACOD. NCIS offers better response time than ACOD does. The reason is similar to that of the previous experiment. Since in NCIS a mobile node can accesses a valid copy that is nearby, it outperforms ACOD in response time when the update rate is low. For example, when the update rate is less then 11, NCIS reduces 33.3 percent of the response time that is required by ACOD.

Fig. 8 shows the server energy consumption. It

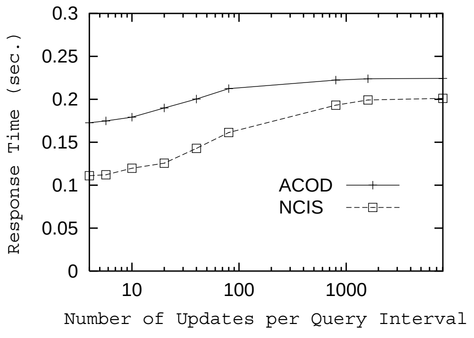

Figure 7: Query Response Time

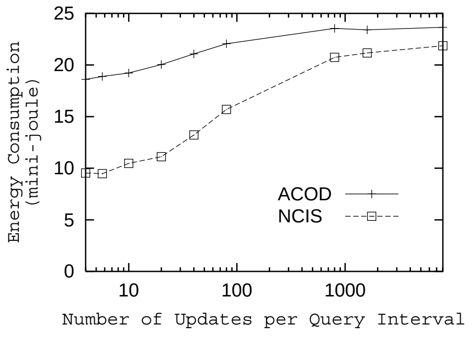

Figure 8: Energy Consumption of the server

shows that when update rate is high, both NCIS and ACOD suffer from high server energy consumption. This is because that the cached copies are mostly invalid with high data update rate, and most of the mobile nodes have to access a valid copy from the server. When the update rate is low, NCIS offers better server energy consumption since it can accesses a nearby copy, which avoid of asking the server to send the valid copy. In ACOD, a mobile node tends to access a valid copy from the server, and the server needs to send a copy of the new value of the requested data item to the node with the most recent copy in each path to refresh its local copy. This incurs sever overhead in energy consumption on the server. It shows that NCIS reduces almost 50 percent of the server energy that is required by ACOD for low update rate.

Fig. 9 shows the client energy consumption of NCIS and ACOD. It shows that NCIS saves about 50 percent of the client energy that is required by ACOD. This is because that the range of broadcast of NCIS is smaller than that of ACOD and NCIS does not perform broadcast operations when its local copy is valid. Since a broadcast requires many mobile nodes to relay messages, it dominates the energy consumption of the mobile nodes. Note that with high update rate the broadcast range of ACOD and NCIS are the same. However, in ACOD the server needs to refresh the most recent copy in each search path. This incurs significant overhead on the mobile clients.

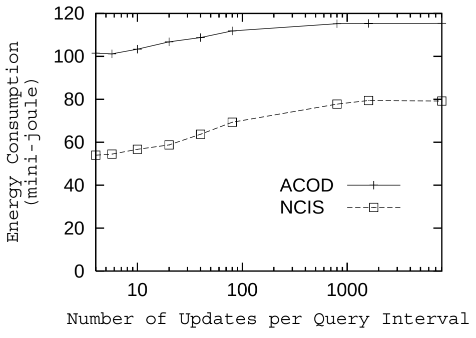

Figure 9: Energy consumption of all clients for a query

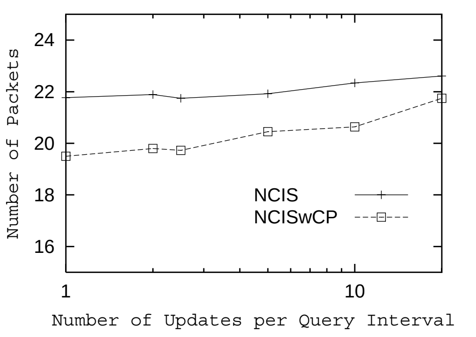

Figure 10: Number of packets per query

4.2.2 Performance of CachePath

The effectiveness of CachePath in NCIS is shown in Fig.10 and in Fig. 11. Fig. 10 shows the total number of packets that are transmitted in the network in processing a query. Note that the transmission of a message between two mobile nodes in one hop distance is counted as one packet. In Fig. 10, we use NCISwCP to denote NCIS with CachePath. Fig. 10 shows that CachePath is effective in reducing the number of packets in answer a query. When the update rate is

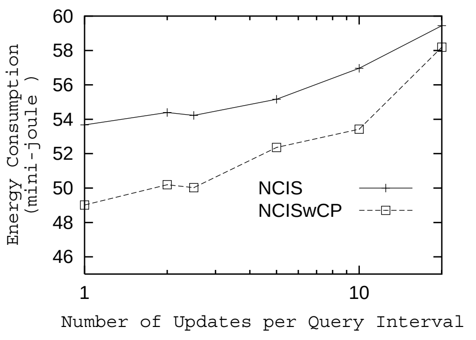

Figure 11: Energy consumption per query

low, CachePath helps in reducing the range of searching so as to reduce number of packets; however, when the update rate is high, most of the cached copies become invalid, and CachePath is not so effective. Fig. 11 shows that CachePath helps in reducing the client energy consumption by reducing broadcast packets.

5 Conclusion

Due to the limitations of a MANET, the existing cache invalidation schemes designed for a cellular wireless network are not well-suited for a MANET. In this paper we have presented a new cache invalidation and searching policy which features in minimal searching range and minimizing the number of broadcast messages. Experiments show that the proposed policy outperforms the existing policy in terms of query response time and energy consumption on both the server and the mobile clients.

Acknowledgements: The research was supported by the National Science Council of Taiwan (grant No. NSC 94-2213-E-011-064).

References:

[1] I. Chlamtac, M. Conti and J. J.-N Liu, Mobile ad hoc networking:imperatives and chanllenges, *Ad Hoc Networks*, 1, 2003, 13-64.

[2] S. Lim, W. Lee, G. Cao, and C. Das, Performance Comparison of Cache Invalidation Strategies for Internet-based Mobile Ad Hoc Networks, *IEEE International Conference on Mobile Ad-hoc and Sensor Systems(MASS)*, 2004, 104-113.

[3] L. Yin and G. Cao, Supporting cooperative caching in ad hoc networks, *IEEE Transactions on Mobile Computing*, 5(1), 2006, 77-89.

[4] G. Cao, L. Lin and C. Das, Cooperative cach-based data access in ad hoc networks, *IEEE Computer*, 37(2), 2004, 32-39.

[5] Y. Du, and S. K. S. Gupta, A cooperative caching service in MANETs, *Autonomous Systems and International Conference on Networking and Services*, ICAS/ICNS 2005.

[6] D. Barbara and T. Imielinski, Sleeper and Workaholics: Caching Stragegies in Mobile Environments, *Proceedings of ACM SIGMOD*, 1994, 1-12.

[7] C.E. Perkins and E.M. Royer, Ad-hoc on-demand distance vector routing, *Proc. IEEE WMCSA'99*, 1999, 90-100.

[8] L. M. Freeney and M. Nilsson, Investigating the Energy Consumption of a Wireless Networking Interface in an Ad Hoc Networking Environment, *Proc. INFOCOM 2001: Twentieth Annual Joint Conference of the IEEE Computer and Communications Society*, 2001, 1548-1557.

[9] ns Notes and Documentation, http://www.isi.edu/nanam/ns/, 2002.

[10] M. Abolhasan, T. Wysocki and E. Dutkiewicz, A review of routing protocols for mobile ad hoc networks, *Ad Hoc Networks*,2, 2004, 1-22.