A Game Theoretic Approach to Energy Efficient Cooperative Cache Maintenance in MANETs

Mark Kai Ho Yeung and Yu-Kwong Kwok
Department of Electrical and Electronic Engineering
The University of Hong Kong, Pokfulam Road, Hong Kong

Abstract—**There have been an increasingly large number of mobile handsets equipped with dual or multiple network interfaces. The server interface (e.g., GPRS, EDGE, UMTS) is responsible for communicating with the network operator, while the peer interfaces (e.g., Bluetooth, IEEE 802.11) are used to connect with other computing devices. However, they are usually used separately. In this paper, we investigate the use of both network interfaces to support energy efficient data applications among mobile clients. Specifically, we proposed a fully distributed protocol for mobile handsets to form cooperative groups to maintain cache consistency with minimal communication with the network operator.**

Our proposed protocol takes advantage of the low power consumption and high data rate of the peer interface. The aim is to reduce the use of the server interface, which is typically slower and involves higher power consumption. Furthermore, we also consider the presence of selfish clients. It is shown that groups formed by the proposed protocol constitutes a pure Nash Equilibrium. This suggests that our protocol is robust even in the presence of selfish clients. Simulation results confirm that, given the same energy resource, mobile clients running the proposed protocol complete more queries, experience longer lifetime and achieve smaller query latency.

I. Introduction

With the wide-spread deployment of third generation (3G) cellular systems, the volume of data traffic delivered in the cellular environment has increased significantly [10]. On the other hand, different personal area network (PAN) standards such as IEEE 802.11 and Bluetooth are capable of providing high speed connections among mobile devices. Although there have been an increasingly large number of mobile handsets equipped with dual or multiple network interfaces, these two interfaces usually worked independently. It would be better to take advantage of the high data rates provided by PAN while utilizing the cellular network for universal coverage.

Energy conservation has become a fundamental issue in wireless networks [7], [9], [13]. To support energy efficient data applications, many emerging data applications utilize data caching [3], [11], [12]. Mobile clients cache frequently used data objects locally to reduce the amount of energy consumed in querying the server. Data caching is also capable of reducing query latency but requires mobile clients to maintain cache consistency. Since the cellular networks and PANs have significant differences in terms of complexity, data rate, transmission range, etc., their power consumption characteristics have dramatic differences [14], [15].

This research was supported by a grant from the Research Grants Council of the HKSAR under project number HKU 7157/04E. Corresponding Author: Yu-Kwong Kwok (Email: *ykwok@hku.hk*)

Based on the above observations, we propose utilizing both network interfaces to support data applications between mobile handsets and network operator. Specifically, we proposed a fully distributed, energy-efficient protocol such that mobile handsets form groups for cooperative cache maintenance. In our proposed protocol, members belonging to the same group cooperatively retrieve and cache data objects, which are originated from the network operator. Our protocol utilizes the low power and high data rate of the peer interface (PAN) while reducing the use of the server interface (cellular network), which is typically slower and has higher power consumption. We also consider the presence of selfish clients. It is shown that groups formed by the proposed cooperative cache maintenance protocol constitutes a pure Nash Equilibrium. This suggests that the protocol is robust even in the presence of selfish clients. Simulation results confirm that, given the same energy resource, our protocol is capable of increasing the number of completed queries; extending the lifetime; while reducing the query latency.

The rest of the paper is organized as follows. In Sections II-A and II-B, we provide the motivations and describe the details of our cooperative cache maintenance scheme, respectively. We analyze our proposed protocol in Section II-C. The distributed protocol is described in Section II-D. Simulation results are presented and discussed in Section III. Finally, we draw some concluding remarks in Section IV.

II. Proposed Cooperative Cache Maintenance

A. *Motivations*

Recently, there have been an increasingly large number of handsets equipped with dual or multiple network interfaces. Often, the server interface (e.g., GPRS, EDGE, UMTS, etc.) is responsible for data communication with the network operator while the peer interface (e.g., Bluetooth, IEEE 802.11, etc.) provides short range connectivity with other computing devices. Both interfaces complement with each other to provide ubiquitous data access for mobile users. More importantly, the interfaces have drastic differences in terms of data rates, power consumption characteristics, transmission range, connection availability, etc. For example, the IEEE 802.11 standards can provide data rates up to 54Mbps and beyond. Although EDGE and other 3G technologies are promised to achieve data rates up to 2Mbps with stationary terminals, the typical rates are still hundreds of Kbps. Slow data rate means the server interface should remain in the transmit or receive state longer. Further, the transmission range of the cellular network is much longer than personal area networks. Both factors make communication via the server interface more expensive in terms of energy. Another drawback for the client is to bear the service fee charged by the network operator. These factors make the server interface less attractive for data applications. However, its major strength is the ubiquitous connection availability. Once subscribed to the service, mobile users can access to the Internet at anytime anywhere. Therefore, we propose a novel scheme that make data applications take advantage of both interfaces. Specifically, the idea is to leverage (1) the ubiquitous access of the server interface (cellular network); and (2) the high data rates and high energy efficiency of the peer interface (personal area network).

B. Descriptions

We assume the following system model: a number of mobile clients, denoted by $\{N\}$, are interested in a set of data objects $\{D\}$ kept at a central server. Each client is equipped with two interfaces: (1) server interface; and (2) peer interface, which are used to communicate with the server and other peer devices, respectively. Clients generate read-only queries to the set of data objects. It is up to the clients to decide whether to cache data objects locally for future queries or not. Each data object is associated with a time-to-live (TTL) field such that clients can only use the data object to serve queries within the time specified. Upon expiry of the data object, clients perform cache maintenance: either drop the cache or prefetch the updated data object from the server.

Given a fixed battery budget, the challenges are (1) extending clients' lifetime; (2) increasing number of completed queries; while at the same time, (3) reducing query latency. Our proposed cache maintenance scheme differs from existing ones [3], [11], [12] in that clients make use of the peer interface to form cooperative groups. Further, we assume that clients are selfish such that they join a group only if it is beneficial to themselves. In other words, each client maximizes its own utility and is not bound by the protocol for any specific tasks. Our protocol defines four different roles in a group: (1) cacher; (2) replicator; (3) requester; and (4) free-rider.

Within a cooperative group, a requester uses the server interface to send query requests on behalf of the cacher. Upon receiving the query request from the *requester*, the server replies the *cacher*. As such, cacher obtains the data object without sending any query requests. The role of cacher is to broadcast the received data object to all replicators via the peer interface. Upon receiving the data object from the cacher, each replicator broadcasts the data objects to the requester and free-riders via the peer interface. Therefore, the caches of all the group members are now updated with only one query request sent to the server. The communication among group members take advantage of the energy-efficient and high data rate of the peer interface, the costly operations associated with the server interface are reduced to a minimum. Algorithm (1) summarizes the four roles in a cooperative group.

Algorithm 1 Four roles in a cooperative group

```
1: **cacher**:
2: Upon receiving data object $j$ from the server,
3: cache data object $j$;
4: broadcast data object $j$ to replicators;
5:
6: **replicator**:
7: Upon receiving data object $j$ from the cacher,
8: cache data object $j$;
9: broadcast data object $j$ to the requester and free-riders;
10:
11: **requester** and **free-rider**:
12: Upon receiving data object $j$ from replicator,
13: cache data object $j$;
14:
15: **requester**:
16: Upon expiry of data object $j$,
17: query data object $j$ on behalf of the cacher;
```

C. Analysis

In this section, we provide justifications of the proposed cooperative cache maintenance protocol and quantify the energy cost of the four different roles of a group.

We begin our discussion with the basic cache maintenance operation. To complete a query using only the server interface, a mobile client first sends a query request (uplink) and then receives the data object (downlink). The total energy cost for querying data object j, $C_j^s(\text{query})$, is given by:

$$C_j^s(\text{query}) = S_x \frac{P_{TX}^s}{R_{UL}^s} + S_j \frac{P_{RX}^s}{R_{DL}^s} \quad (1)$$

where S_x and S_j represent the size of a query request and data object j; P_{TX}^s and P_{RX}^s are the transmit and receive power of the server interface; and R_{UL}^s and R_{DL}^s are the uplink and downlink data rates of the server interface, respectively.

Next, we quantify the energy cost of the four roles defined in Section II-B. In general, the total energy cost to complete a query of data object j is the sum of the energy dissipated in the server interface and the peer interface. For a cacher, it expends energy in downloading the data object via the server interface and broadcast the data object in the peer interface. Therefore, its energy cost is given by:

$$C_j(\text{cacher}) = S_j \frac{P_{RX}^s}{R_{DL}^s} + S_j \frac{P_{TX}^p}{R^p} \quad (2)$$

where P_{TX}^p and R^p represents the transmit power and data rate[1] of the peer interface, respectively.

As described in Section II-B, a replicator receives the data object from the cacher and broadcasts the data object to its free-riders and/or requester, both using the peer interface. Thus, its energy cost is given by:

$$C_j(\text{replicator}) = S_j \frac{P_{RX}^p}{R^p} + S_j \frac{P_{TX}^p}{R^p} \quad (3)$$

[1]It is assumed that the peer interface offers symmetric data rate. However, the results can be extended to the asymmetric case.

where P^s_{RX} is the receive power of the peer interface.

To a free-rider, the only energy cost is to receive the data object from the replicator:

$$C_j(\text{free-rider}) = S_j \frac{P^p_{RX}}{R^p} \tag{4}$$

Upon cache expiry, the requester is also responsible for sending a query request on behalf of the cacher. Therefore, the energy cost is:

$$C_j(\text{requester}) = S_x \frac{P^s_{TX}}{R^s_{UL}} + S_j \frac{P^p_{RX}}{R^p} \tag{5}$$

It becomes clear that if the following inequalities hold for some clients, they would be willing to form a group for cooperative caching of data object j:

$$C_j(\text{cacher}) < C_j(\text{query}) \tag{6}$$
$$C_j(\text{replicator}) < C_j(\text{query}) \tag{7}$$
$$C_j(\text{free-rider}) < C_j(\text{query}) \tag{8}$$
$$C_j(\text{requester}) < C_j(\text{query}) \tag{9}$$

As discussed in Section II-A, the server interface offers much lower data rates and much larger coverage than the peer interface. Both factors increase the power consumption of the server interface. It is possible for the four inequalities above to hold among a set of clients (see Section III-B). The remaining question is to assign an appropriate role for each client, which is addressed in the following section.

D. Protocol

Algorithm (2) depicts a distributed role assignment protocol for cooperative cache maintenance. The protocol consists of six message types, namely: (1) replicator request; (2) replicator reply; (2) replicator ack; (3) free-rider request; (4) free-rider reply; (5) free-rider ack. In words, a cacher first finds its replicators from the neighbor set. Each replicator then searches for free-riders from its neighbors. Then, replicators report the free-riders to the cacher. From the set of free-riders, the cacher selects one to become a requester. The details of the distributed protocol are shown in Algorithm (2).

First, it is not difficult to see that the roles assigned by Algorithm (2) satisfy the corresponding relations, as indicated in Equations (6)–(9). This suggests that the energy cost of each group member is reduced. However, one may argue that selfish clients may deviate from the role assigned by the algorithm to improve their own performance. We justify that selfish clients would follow the cooperative cache maintenance protocol with the aid of game theory. Game theory [5] is a rigorous mathematical tool to analyze the presence of selfish clients. In fact, it has been widely used to study different aspects in computer networking [2], [4], [6]. We justify our claim using Nash Equilibrium, the most popular solution concept in game theory. To begin with, we define the following wireless cooperative cache maintenance (WCC) game:

Denote $M = \{1, \ldots, m\}$ as the set of players (*clients*). Each player chooses a strategy (*its role in a group*) to optimize its own utility (*in terms of its energy cost*). The strategy space of player i is given by,

Algorithm 2 Distributed group formulation algorithm

```
1: if $C_j$(cacher) > $C_j$(query) then
2:   role := not-a-member;
3:   return;
4: end if
5: Broadcast a "replicator request" message to the neighbors;
6: for each neighbor that receives the "replicator request" message do
7:   if $C_j$(replicator) > $C_j$(query) then
8:     ignore the message and return;
9:   end if
10:  Broadcast a "free-rider request" message to the neighbors;
11:  for each neighbor that receives the "free-rider request" message do
12:    if $C_j$(requester) > $C_j$(query) then
13:      ignore the message and return;
14:    end if
15:    Reply with a "free-rider reply" message;
16:  end for
17:  Upon receiving the "free-rider reply" messages,
18:  reply the cacher with a "replicator reply" message, which includes the list of its free-riders;
19: end for
20: Upon receiving the "replicator reply" messages,
21: (1) Pick a free-rider from the set of free-riders;
22: (2) Broadcast a "replicator ack" message that includes the ID of the chosen free-rider;
23: role := cacher;
24: for each neighbor that receives the "replicator ack" message do
25:   role := replicator;
26:   Broadcast a "free-rider ack" message, including the ID chosen by the cacher;
27:   for each neighbor that receives the "free-rider ack" message do
28:     if my ID == the ID chosen by the cacher then
29:       role := requester;
30:     else
31:       role := free-rider;
32:     end if
33:   end for
34: end for
```

$R_i = \{\text{cacher, replicator, requester, free-rider, not-a-member}\}$. The strategy combination is denoted as, $r = (r_1, \ldots, r_m) \in R$, where $R = \times_{j \in M} R_j$ is the Cartesian product of the m players' strategy spaces. Furthermore, define $r_{-i} = (r_1, \ldots, r_{i-1}, r_{i+1}, \ldots, r_n) \in R_{-i}$, where $R_{-i} = \times_{j \in N \setminus \{i\}} R_j$, as the strategy combination of all players, except i. $C^i(s) \in S$ represents the utility of player i when the strategy combination is s. A strategy combination, r^* is said to achieve the state of *Nash equilibrium* when: $C^i(r^*) \leq C^i(r^*_{-i}, r_i) \forall r_i \in R_i, i \in M$ In general, a game may have multiple equilibria or even none at all.

Next, we would like to prove the existence of Nash Equilib-

TABLE I
SIMULATION PARAMETERS.

Parameter	Value
Number of data objects, $\lvert D\rvert$	500
Data object size, S_j	10KB–100KB
Control message size	1KB
Time-to-live, TTL_j	1,000s–10,000s
Client population size, $\lvert N\rvert$	50
Mean query arrival time, T^i	10s–100s
Energy available	1KJ–10KJ
Transmit power (server interface), P^s_{TX}	3.00W
Receive power (server interface), P^s_{RX}	1.25W
Transmit power (peer interface), P^p_{TX}	1.35W
Receive power (peer interface), P^p_{RX}	0.90W
Data rate (server interface), R^s_{UL} and R^s_{UL}	130Kbps
Data rate (peer interface), R^p	11Mbps
Simulation dimension	100m×100m
Transmission range (peer interface)	48m
Speed	5km/hr

rium of the above WCC game. It is also desirable for players to adopt deterministic strategies at equilibrium. If all players do so, it becomes a pure Nash Equilibrium. The details are formalized in Theorem (1):

Theorem 1: Any group formed by Algorithm (2) results in a pure Nash Equilibrium.

Proof: From Algorithm (2), we can see that Equations (6)–(9) are satisfied for cacher, free-rider, requester and replicator, respectively. Therefore, clients would not leave the group. If the cacher changes its role, its cache would eventually expire since no requester would query the data object on behalf of itself. If a replicator changes its role, it could not update the cache with the cacher's broadcast. Furthermore, a replicator could not switch to free-rider or requester because there is no other neighboring replicator. If such a replicator does exist, the protocol would have already assigned it with the appropriate role, see Algorithm (2) lines 10–16. Free-riders would not become a requester since $C_j(\text{free-rider}) < C_j(\text{requester})$. On the other hand, the requester could not change to a free-rider because it is the only one that sends the query request to the server. If that requester deviates, its cache would become invalid upon expiry.

We can conclude that each client chooses a deterministic role and has no incentive to deviate. Therefore, the group formed by Algorithm (2) constitutes a pure Nash Equilibrium.

III. Performance Evaluation

In this section, we evaluate our proposed energy efficient cooperative cache maintenance protocol using simulation. In particular, the performance metrics are: (1) number of completed queries; (2) lifetime, i.e., time taken to deplete the available energy; (3) query latency, i.e., time lapsed between the arrival of a new query and the complete retrieval of its reply.

A. Simulation Model and Parameters

Table I shows the major simulation parameter values. We assume that the server interfaces is EDGE while the peer interface is IEEE 8021.11b. The parameter values of the two interfaces (power consumptions, data rates) are taken from the data sheets [14], [15]. We assume an indoor simulation environment. As such, the transmission range of the peer interface is taken to be 48m [15]. Other parameter values are similar to those reported in [3], [8], [12].

We adopt the following query model: client i generates a stream of exponentially distributed queries with mean query arrival time, T^i, drawn uniformly between $T^i(\text{min})$ and $T^i(\text{max})$. This setting models different query rates among the set of clients. The smaller the values of query arrival time, the higher the query rates are. Furthermore, we model clients' non-uniform access pattern using the *Zipf-like* distribution with $\theta = 0.7$ [1]. In other words, client i queries the data object j with probability given by: $p_j(\theta) = \frac{1}{j^\theta \sum_{k=1}^{|D|} \frac{1}{k^\theta}}$, where $0 \leq \theta \leq 1$. The value of θ determines the "skewness" of the access pattern. $\theta = 1$ gives the strict *Zipf* distribution while $\theta = 0$ results in the uniform distribution. The size of data objects increases from 1KB to 100KB while the time-to-live values are uniformly distributed between 10s and 10,000s. The amount of energy available to each client is uniformly distributed between 1KJ and 10KJ. We adopt the random way point model and assume mobile clients are of pedestrian speed (5km/hr) only. It is also assumed that clients have unlimited storage such that they can cache as many data objects as they wish. Upon cache expiry, however, clients only prefetch data objects with index between 1 and 200 (those with higher query probabilities) and drop the other data objects (those with lower query probabilities).

B. Simulation Results

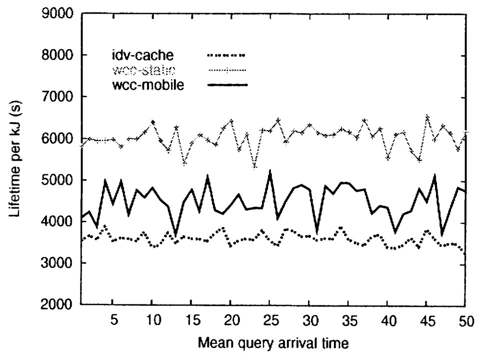

Fig. 1. Lifetime (s).

In Figures 1–3, "idv-cache" represents the results when clients cache data objects individually while "wcc-static" and "wcc-mobile" represents the results when clients run the proposed cooperative cache maintenance protocol in static and mobile settings, respectively.

Figure 1 shows the lifetime of individual clients in different schemes. Client's lifetime is defined as the time for a client to deplete its energy source. The values are normalized by the amount of available energy for fair comparisoon. First, we can observe that clients running the proposed protocol result in a longer lifetime, which is consistent with the analysis in Section II-C. Although clients running the distributed protocol

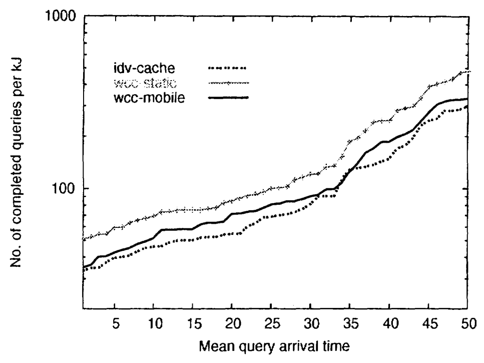

Fig. 2. Number of completed queries.

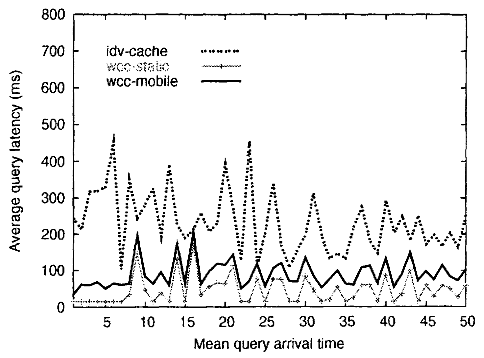

Fig. 3. Average query latency (ms).

dissipate energy in the group formation process, the benefits gained from being a group member outweigh the initial overheads. This is explained by the high data rate and low power requirement of the peer interface (IEEE 802.11), such that the energy expended in the control messages are kept at a minimum. The costly operations on the server interface are reduced. The energy conversed from joining a group makes the lifetime longer.

Figure 2 shows the performance results in terms of number of completed queries. This metric is also normalized by the amount of available energy for fair comparison. Similar to lifetime, both the static and dynamic cases indicate that clients running the proposed protocol (WCC game) complete more queries compared with the individual caching scheme. This follows from the results that clients conserve energy for being a group member. The energy conserved is available for more query operations. As such, clients complete more queries with the same amount of energy source.

Figure 3 shows the average query delay results. We can observe that the proposed protocol achieves smaller query latency compared with individual caching. This is explained by the reduced amount of traffic in the server interface. In individual caching, each client is required to query the server to prefetch data objects upon cache expiry. However, only the cacher and the requester communicate with the server in the proposed protocol. When there are cache miss occurs, clients would experience a lower query latency in the proposed scheme.

IV. Concluding Remarks

Based on the observations that many mobile handsets are equipped with dual or multiple network interfaces, we proposed a novel energy efficient cooperative cache maintenance protocol for wireless networks. The protocol utilizes the low power consumption and high data rate of the peer interface (e.g., Bluetooth, IEEE 802.11) while reducing the use of the server interface (e.g., GPRS, EDGE, UMTS), which is typically slower and consumes higher power. Simulation results confirm that our protocol is capable of (1) increasing the number of completed queries; (2) extending clients' lifetime; while reducing average query latency, with the same amount of energy resource.

The protocol also takes the presence of selfish clients into considerations. Selfish clients concerns their own performance and may deviate from the protocol. We showed that groups formed by the cooperative cache maintenance protocol constitutes a pure Nash Equilibrium. This suggests that clients, whether selfish or not, would not deviate from the roles assigned by the protocol. Therefore, the proposed protocol is robust even in the presence of selfish clients.

References

[1] L. Breslau, P. Cao, L. Fan, G. Phillips and S. Shenker, "Web Caching and Zipf-like Distributions: Evidence and Implications," *Proc. INFOCOM 1999*, vol. 1, pp. 126–134, Mar. 1999.

[2] C. Buragohain, D. Agrawal and S. Suri, "A Game Theoretic Framework for Incentives in P2P Systems," *Proc. Peer-to-Peer Computing* vol. 3, pp. 48–56, Sept. 2003.

[3] G. Cao, "A Scalable Low-Latency Cache Invalidation Strategy for Mobile Environments," *IEEE Trans. Knowledge and Data Engineering*, vol. 15, no. 5, pp. 1–15, May 2003.

[4] B.-G. Chun, K. Chaudhuri, H. Wee, M. Barreno, C. H. Papadimitriou and J. Kubiatowicz, "Selfish Caching in Distributed Systems: A Game-Theoretic Analysis," *Proc. 23rd ACM Symp. Principles of Distributed Computing*, pp. 21–30, Jul. 2004.

[5] D. Fudenberg and J. Tirole, *Game Theory*, MIT Press, 1991.

[6] M. Goemans, L. E. Li, V. S. Mirrokni and M. Thottan, "Market Sharing Games Applied to Content Distribution in Ad-Hoc Networks," *Proc. 5th ACM Int'l Symp. Mobile Ad Hoc Networking and Computing*, pp. 55–66, May 2004.

[7] C. Gui, and P. Mohapatra, "Power Conservation and Quality of Surveillance in Target Tracking Sensor Networks," *Proc. 10th Int'l Conf. Mobile Computing and Networking*, pp. 129–143, Sept.–Oct. 2004.

[8] S. Lim, W.-C. Lee, G. Cao and C. R. Das, "A Novel Caching Scheme for Internet Based Mobile Ad Hoc Networks," *Proc. Computer Communications and Networks*, pp. 38–43, Oct. 2003.

[9] P. Nuggehalli, V. Srinivasan, C.-F. Chiasserini, "Energy-Efficient Caching Strategies in Ad Hoc Wireless Networks," *Proc. 4th ACM Int'l Symp. Mobile Ad Hoc Networking and Computing*, pp. 25–34, Jun. 2003.

[10] H. Schulzrinne, X. Wu, S. Sidiroglou and S. Berger, "Ubiquitous Computing in Home Networks," *IEEE Communications Magazine*, vol. 41, no. 11, pp. 128–135, Nov. 2003.

[11] Mark K. H. Yeung and Y.-K. Kwok, "Wireless Cache Invalidation Schemes with Link Adaptation and Downlink Traffic," *IEEE Transactions on Mobile Computing*, vol. 4, no. 1, pp. 68–83, Jan.-Feb. 2005.

[12] L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks," *Proc. INFOCOM 2004*, vol. 4, pp. 2537–2547, Mar. 2004.

[13] J. Zhu, C. Qiao, and X. Wang, "A Comprehensive Minimum Energy Routing Scheme for Wireless Ad Hoc Neworks," *Proc. INFOCOM 2004*, vol. 2, pp. 1437–1445, Mar. 2004.

[14] Sierra Wireless AirCard 775, *Sierra Wireless*.

[15] Cisco Aironet IEEE 802.11 a/b/g Wireless Cardbus Adapter, *Cisco Systems*.