Optimal Path Selection for File Downloading in P2P Overlay Networks on MANETs

Afzal Mawji and Hossam Hassanein
Telecommunications Research Lab, School of Computing, Queen's University
Kingston, Ontario, Canada, K7L 3N6
{mawji, hossam}@cs.queensu.ca

***Abstract*—P2P networks are extremely popular on the Internet, with their uses including file sharing, gaming, IP telephony, and much more. Users are increasingly moving toward wireless networks and expect to continue using P2P applications in these new environments. Meanwhile, MANET usage is expected to grow as wireless mesh and 4G networks become more prevalent. In P2P file sharing networks, peer clients must choose which servers to download from, and in a MANET must also select the paths to those servers. It is often possible to download from multiple servers simultaneously. This paper formulates and solves the problem of optimally selecting which servers and paths to choose, such that the cost or download time is minimized.**

I. Introduction

Peer–to–peer (P2P) networks are extremely popular among users on the Internet. They are used for many different applications including file sharing, IP telephony, gaming, instant messaging, etc. P2P networks are an alternative to the common client/server computing paradigm, and they are usually implemented in the form of overlay networks. These overlays consist of upper–layer connections between nodes, or peers, that are independent of the underlay or substrate network, abstract peers' view of the connections that make up the network. The amalgamation of the logical connections that peers form with one another constitutes the overlay network. The peers self–organize and handle "churn", the constant connections and disconnections of peers as they join and leave the network. This requires P2P networks to be resilient. On the Internet, peers exist at the "edge" of the network, and many P2P networks in use today scale to millions of simultaneous users.

Mobile ad hoc networks (MANETs) are multi–hop, variable topology networks which do not require any pre–existing infrastructure. Nodes instead form a self–organizing network, and node act as both clients and servers. Devices are heterogeneous and resource–constrained. This results in a large variation in link and node capabilities. As a result, these autonomous and infrastructureless networks pose particularly challenging design problems, especially with regard to lifetime and scalability.

Users want to run P2P applications regardless of the type of network they are using, including combined infrastructure–infrastructureless networks such as a wireless mesh or 4G networks. Therefore, we must consider how to effectively operate P2P overlays in infrastructureless networks such as cooperative MANETs. Even if the user is connected to a partly–infrastructured network, avoiding the base station and instead communicating with nearby nodes may lower delay and energy consumption due to transmissions traveling a shorter distance and requiring less power. If there are multiple infrastructured networks in an area, it may be possible for users of these different networks to join together to form large, cooperative MANET. Avoiding communication with the base station may also avoid charges from the network operator.

This research is supported in part by the Natural Science and Engineering Research Council of Canada (NSERC) and by a grant from Qatar Telecom (QTel).

It is a natural evolution for MANETs and P2P networks to be combined so a P2P overlay runs on a cooperative MANET, since both network types share many similarities. We use the term P2P–MANET to refer to such networks. Both network types are decentralized, both dynamically organize the nodes, both deal with frequent topology changes, whether due to node mobility or churn, both must be resilient to node failure, and both perform the routing function.

Despite these similarities, it is not desirable to adopt existing P2P overlay techniques and use them in MANETs because, though similar, both networks have their differences. P2P networks are often very large–scale networks, with millions of simultaneous users and they are designed as overlays for deployment on the "edge" of the Internet. They are not designed for node mobility. On the other hand, MANETs consist of much fewer nodes, which are severely resource–constrained in comparison, and the links between nodes often have higher delay. Energy consumption is a significant concern and nodes are usually geographically nearby one another.

Figure 1 illustrates a possible peer–to–peer overlay running on top of a MANET. The light circles represent nodes participating in the overlay network, while the dark circles are not part of the overlay. The solid lines show how the overlay network is connected, with potentially multiple hops of the underlying MANET providing the direct overlay connections. The dashed lines represent the MANET links.

In this paper, we consider the problem of how a client peer selects not only which servers to download files from, but also which intermediate nodes to choose that constitute a path to those servers. We assume that files are split into blocks and that both servers and intermediate nodes will indicate a cost in terms of delay and price. The delay indicates the waiting time before a block will be transmitted, and the price indicates the compensation the node expects to receive for each block in the

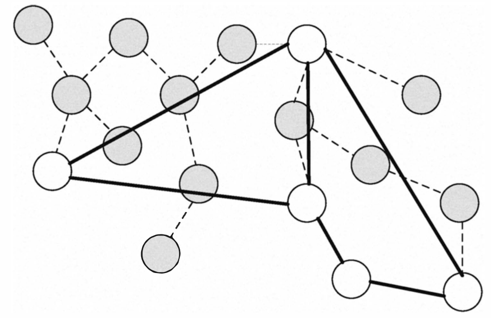

Fig. 1. An example of a peer–to–peer overlay running on a MANET

form of virtual credits. The client may then minimize either the download time, subject to a maximum cost, or the price, subject to a maximum download time. The resulting optimal path selection indicates how many blocks to download from each server, and also which path to choose to reach the server.

The rest of the paper is organized as follows. Section II provides some background on server selection. Section III discusses the optimal path selection linear program in detail. In Section IV we examine the results of the linear program. Section V provides some conclusions.

II. Background

Selecting the optimum server peer for a file download is examined by Adler *et al.* [1], [2]. The pricing mechanism and how client nodes determine what peers have what files and what they are charging is beyond the scope of the problem considered by the authors. It is assumed that the servers have somehow indicated to the client the cost of downloading the file from them as well as their available uplink bandwidth. The cost may be either real money or some form of virtual currency.

Given a set of servers, their upload bandwidth, and their prices, the client must select how much to download from each server in order to minimize the download time, while also not exceeding a set budget. The optimum solution is given in the form of a linear program. The goal of the program is to minimize the maximum download time from each server. Over all servers chosen, the slowest one, i.e., the one with the maximum upload time, determines the total download time, so this is the time the program must minimize. The constraints are that the cost must be less than the budget, the bytes downloaded over all servers must obtain the complete file, and the bytes per server must be greater than zero.

Next, the authors examine a similar problem as it applies to real–time streaming. The problem is now to select peers such that the total streaming cost is minimized while ensuring that at any point in time, all bytes before that point in the stream have already been downloaded. This problem has the added challenge of scheduling the parts of the file correctly. They propose two approaches, time segmentation and rate segmentation. In time segmentation, the video is split up by time and each server streams only the segment it is responsible for, while the client buffers. The problem with this approach is that the total amount of data being sent to the client may exceed its downlink bandwidth. In rate segmentation, each server sends a portion of the frame and the total amount of data uploaded to the client is the playback rate. The authors then propose a linear program for rate segmentation, which attempts to minimize the cost of the rates across all servers such that the bit rate is greater than the playback rate. This ensures that clients are able to obtain the frames as they are needed for playback.

Han and Xia [3] study techniques for server selection in an overlay network structured as a hypercube. The client selects k server candidates, where k is the degree of parallelism, to satisfy a cost–minimizing criterion that reflects either network resource usage or interference among the connections from the servers to the client. The interference is minimized by selecting those servers whose position within the hypercube does not cause traffic collisions on the path between the server and the client. This can be determined based on the node IDs, which are indicative of how they are connected within the hypercube. The network resource usage is defined by the number of flows over a link, with the goal of avoiding those that are most used. The minimum interference algorithm runs in $O(n \log n)$ time and the resource usage algorithm runs in $O(nm)$ time, where n is the number of possible servers, and 2^m is the size of the network.

In a P2P–MANET, it is important to consider the costs of all intermediate nodes on the path to the server as well as the server itself. This allows clients to download via multiple paths. Also, many P2P–MANETs are not structured overlays due to the reduced resilience. Hora *et al.* [4] found that in MANETs, unstructured protocols are more resilient than structured ones at the cost of higher energy and delay. It is important, therefore, for the server selection algorithm to support unstructured networks. The path selection algorithm presented in this paper satisfies both of these goals.

III. P2P–MANET Path Selection

After a user has queried the P2P overlay for the data it is searching for, it is presented with a set of servers which have the file it is interested in. The MANET may provide multiple paths to each of these servers, resulting in a set of paths with different costs that the user must choose from. Furthermore, modern P2P file sharing services allow users to obtain files from multiple servers simultaneously by dividing the file into blocks, or chunks.

The problem is to select which servers to download from, and also paths to the servers, such that costs are minimized. We formulate a path selection algorithm that employs a mixed integer linear program to produce the optimal set of paths for clients to select.

Given a set of paths terminating with servers that are willing to upload a file, as well as the total cost and estimated download time along a path to the server, it can be determined how much to download from each path in order to minimize the download time, subject to a maximum cost. This allows users to download from multiple servers simultaneously, resulting in faster downloads. Conversely, the user may elect to obtain the file at minimum cost, subject to a maximum download time.

To illustrate this idea, Figure 2 shows client node A, which wants to download a file and has the choice of three

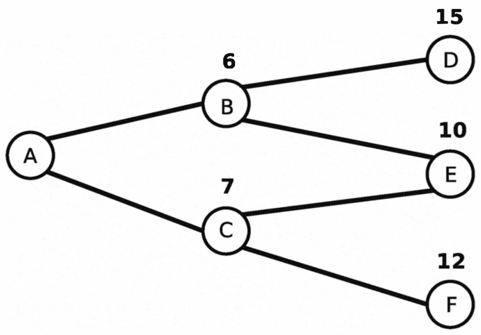

Fig. 2. A choice of download paths and servers

possible servers. For clarity, download time estimates have been omitted. Node A may download from server D via node B for a total cost of 21, from server E via B for a total cost of 16, from server E via C for a total cost of 17, or from server F via C for a total cost of 19. If A is concerned only with the cost, then choosing E via B is the cheapest. However, A might also wish to reduce the download time, and selecting multiple paths subject to a maximum cost and downloading simultaneously from them would accomplish this.

A. System Model

We assume that there are a total of m download requests, each of which corresponds to one file request using the P2P overlay. These requests cover many different files for many different clients. If a client wishes to download more than one file, than it will have more than one of the m requests. Each file is split into a set of blocks. We assume that the file associated with request i is split into B_i blocks, and each of these blocks is indistinguishable. In order to download the entire file, the client must acquire all B_i blocks. Clients set a maximum price they are willing to pay for each request, P_i, or a maximum time they must download the file within, T_i. The price may represent anything, such as hop distance, or the cost as determined in [5].

There are n paths which constitute all possible routes that connect the set of clients to the set of servers. Only one server is associated with each path, but there may be multiple paths per server. For a particular request i, and a path j, there is a cost per block $c_{i,j}$ and an estimated download time per block $d_{i,j}$.

The server associated with path j possesses b_j blocks of the file. If $b_j = B_i$ then this indicates that the server has the entire file available to share. If $0 < b_j < B_i$, the server has only part of the file, but can upload some blocks. The terminal node in the path is the server, which possesses the file to be downloaded. If the server associated with path j does not possess any blocks of the file corresponding to request i, it sets the cost and download time to infinity (i.e. $c_{i,j} = \infty$ and $d_{i,j} = \infty$) and sets b_j to zero.

We formulate the path selection problem as a mixed integer linear program (MILP) in order to determine an optimal solution such that the overall total download time is minimized and the cost of downloading the file does not exceed an amount specified by the client. The variables for the download times and costs can easily be interchanged to obtain the solution to the problem of minimizing the cost while not exceeding a maximum download time specified by the client. The linear program also ensures that no more blocks are requested from a server than it has available.

B. Optimal Path Selection

We now address the problem of a group of clients selecting from a set of download paths, each with an associated cost and download time. The clients must determine not only which of the paths to download from, but also how much to download from each path. The files are assumed to be split up into a set of fixed sized blocks, and the clients determine how many blocks, if any, to download from each path. We assume that because peers download files in blocks, once they have at least one block of a particular file, they are able to share it. Therefore, the servers that clients may choose from do not necessarily possess the entire file. For simplicity, each block is assumed to be equivalent, something that can be accomplished using network coding, as shown in [6]. It is also possible for a client to download blocks from the same server via multiple paths, using a multipath routing protocol such as DSR [7] or [8], or by forwarding at the overlay layer.

The problem to be solved is, given the set of n paths with known costs and times, determine for each request i, which of these n paths to select and how many blocks to download from each path. The goal may be either to minimize the overall download time subject to a budget constraint, or to minimize the cost subject to a time constraint. This problem is formulated as a mixed integer linear program. Let $x_{i,j}$ be an integer decision variable that indicates how many blocks the request i should download via path j. If $x_{i,j} = 0$, then no blocks of request i are to be downloaded from path j.

We first provide the MILP to minimize the download time, which is the number of blocks requested multiplied by the download time per block for each path selected. The download time problem is formulated as follows:

$$\text{Minimize} \qquad \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} x_{i,j} d_{i,j} \tag{1}$$

s.t.

$$\sum_{j=0}^{n-1} x_{i,j} c_{i,j} \leq P_i \qquad \forall i \tag{2}$$

$$\sum_{j=0}^{n-1} x_{i,j} = B_i \qquad \forall i \tag{3}$$

$$0 \leq x_{i,j} \leq b_j \qquad \forall i,j \tag{4}$$

$$x_{i,j} \in \mathbb{Z} \tag{5}$$

The counterpart cost problem minimizes the download cost and is formulated as follows:

$$\text{Minimize} \qquad \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} x_{i,j} c_{i,j} \tag{6}$$

TABLE I
ENERGY CONSUMPTION CONSTANTS USED IN SIMULATIONS

m_{send}	1.89	$\mu W \cdot sec/byte$
b_{send}	246	$\mu W \cdot sec$
m_{recv}	0.494	$\mu W \cdot sec/byte$
b_{recv}	56.1	$\mu W \cdot sec$
$b_{sendctl}$	120	$\mu W \cdot sec$
$b_{recvctl}$	29.0	$\mu W \cdot sec$

s.t.

$$\sum_{j=0}^{n-1} x_{i,j} d_{i,j} \leq T_i \qquad \forall i \tag{7}$$

$$\sum_{j=0}^{n-1} x_{i,j} = B_i \qquad \forall i \tag{8}$$

$$0 \leq x_{i,j} \leq b_j \qquad \forall i,j \tag{9}$$

$$x_{i,j} \in \mathbb{Z} \tag{10}$$

These two programs differ only in what they are trying to minimize (Eqns. 1 and 6) and the first constraint (Eqns. 2 and 7). The former minimizes the download time and the constraint ensures that the maximum budget, P_i is not exceeded. The latter minimizes the cost of downloading and the constraint stipulates that the download time not exceed the maximum time, T_i. The next constraint (Eqns. 3 and 8) ensures that all blocks of the file are downloaded. The penultimate constraint (Eqns. 4 and 9) prevents non-negative block requests and also prevents downloading more blocks than are available on a given path. Finally, the last constraint (Eqns. 5 and 10) ensures that number of requested blocks is an integer.

IV. RESULTS

In this section we examine some results of the mixed integer linear program optimal path selection algorithm. We first use simulation to obtain random sets of file requests, servers, paths, costs, and download times. These values are then use as input for the linear program.

The network simulator *ns–2* 2.33 [9] is used to model a 100 node MANET. The network area is 1500 m $\times$ 1500 m and the transmission rate is 54 Mbps. The two ray ground radio propagation model along with an omnidirectional antenna are used by all nodes. The random waypoint mobility model is used, with all nodes evenly distributed in the simulation area. Nodal velocities are distributed according to a uniform distribution, with a minimum 1 m/s and a maximum 3 m/s, and a uniformly distributed pause time with mean 60 s.

The energy consumption model used is the linear model proposed by Feeney [10]. Each MAC layer operation takes a certain amount of power as defined by $cost = m \times size + b$ where m is the incremental cost of the operation, b is the fixed cost, and $size$ is the amount of data sent or received. The constants are obtained by physical measurements for a Lucent IEEE 802.11 WaveLAN PC Card from [10] and are summarized in Table I.

Files are placed randomly throughout the network with between 10% and 60% of nodes having at least one block of a given file. Block sizes range from 100 to 1000 blocks. An unstructured Gnutella–like overlay is used. As the simulation progresses, requests are made for random files by random nodes.

No actual downloads took place in the simulation, so to better simulate a P2P file–sharing network, and to obtain useful cost values, constant bit rate (CBR) traffic of 1000 byte packets sent 100 times per second was sent between members of the P2P overlay. The traffic started and stopped at random times, between random peers.

The resulting file requests, costs, and download times obtained from the simulations are used to create input files for the MILP solver, and the program is solved to produce the optimal values. The mixed integer linear programs are solved using lp_solve 5.5.0.10, a free linear programming solver that uses the revised simplex and branch–and–bound methods.

A. Cost Model

A brief discussion of the cost model used to obtain the results is presented here. For more details, please see [5].

The price charged by server s for a block of file i is determined by the Cobb–Douglas [11] utility function

$$P_i^s = o_i^{\gamma} \times a_i^{1-\gamma} \tag{11}$$

where o_i is the perceived popularity of file i, calculated as an exponential moving average (EMA) of the number of requests for i, a_i is the perceived availability of i, calculated as an EMA of the number of downloads of i per request, and γ is a parameter that indicates s's preference for popularity against availability.

In addition to the price, s also calculates an estimated delay for serving the blocks. This delay can be determined in several ways such as based on historical information, or s can evaluate its outgoing queue size and bandwidth to calculate a delay value. We define the total nodal delay is a combination of three factors

$$d_s = \tau + T_x + D_e \tag{12}$$

where τ is the wait time in the server's file sharing upload queue including service time, T_x is the block transmission time, and D_e is the delay due to energy constraints. This equation takes into consideration many of the peer's resources, namely CPU utilization, queue length, bandwidth, and energy.

Since we have no historical information on which to determine arrival and service rates, without loss of generality, we assume constants for these values and thus use an M/M/1 queue. Therefore

$$\tau = \frac{1/\mu}{1-\rho} \tag{13}$$

where μ is the service rate of download requests and ρ is the utilization, equal to $\frac{\lambda}{\mu}$ where λ is the arrival rate of download requests. T_x is equal to the block size divided by the bandwidth.

The energy available at the server is an important consideration. If the node has little energy, it would be better off conserving it and not responding to requests. Therefore it indicates this preference by increasing the delay as energy decreases. D_e is defined as

$$D_e = \frac{k}{E + a} \tag{14}$$

where E is the energy available at the time of the request. We define $a = \frac{D_{max} \times E_{min}}{E_{max} - E_{min}}$ and $k = \frac{E_{max} \times D_{max} \times E_{min}}{E_{max} - E_{min}}$, where D_{max} is the maximum delay, and E_{max} and E_{min} are the maximum and minimum energy levels, respectively. These values are chosen because when there is very little energy available, delay is very high, but this decreases quickly as energy increases. When energy is at a maximum, the added delay is zero.

An intermediate node will determine its estimated delay using the function

$$d_i = T_x + D_e \tag{15}$$

Blocks are not placed on the intermediate peer's file sharing upload queue, and it is assumed that packet queues introduce negligible delay so the τ component discussed previously is unused here. T_x and D_e are calculated as before.

The intermediate node determines its price for forwarding the data based on its perception of the demand for its services. If it has been asked to forward a lot of traffic in the recent past, then the chances are higher than it will be selected again. Servers do not need to add this extra profit component as the price they charge already takes demand into account in the form of popularity and availability. The price that a node charges for forwarding traffic is the perceived demand calculated as an EMA of the amount of data forwarded,

$$p_i = \beta x_i + (1 - \beta) p_{i-1} \tag{16}$$

where x_i is the demand as of the ith packet, p_{i-1} is the historical price of the demand, and β is the smoothing constant.

The total cost and delay for a file request are the sum of the components of each, for each node along the path from client to server. The following values are used for the constants in the cost model: D_{max} = 10 sec, E_{max} = 100,000 J, E_{min} = 10,000 J, ρ = query rate/avg. block transfer time, $\gamma = 0.5$, $\beta = 0.375$, and the epoch for the EMA is five minutes.

The results obtained in all experiments have a 95% confidence level based on 10 independent runs. The confidence intervals are indicated in the figures below.

Figure 3 shows the total delay over all file downloads by all clients, for between 100 and 1000 file requests. As expected, the linear program which optimizes the delay has lower delay, while the linear program that optimizes the cost has higher delay, since this is not optimized in the program. The delay increases because as more requests are made and satisfied, the total delay must go up since more nodes are involved.

Figure 4 shows the total cost over all file downloads by all clients. Again, as expected, the linear program which optimizes the cost has the lower cost, while the linear program that optimizes the delay has the higher cost, since it is not optimized in the program. As previously, the total cost increases as more requests are made because of increased network usage and more requests constituting the total.

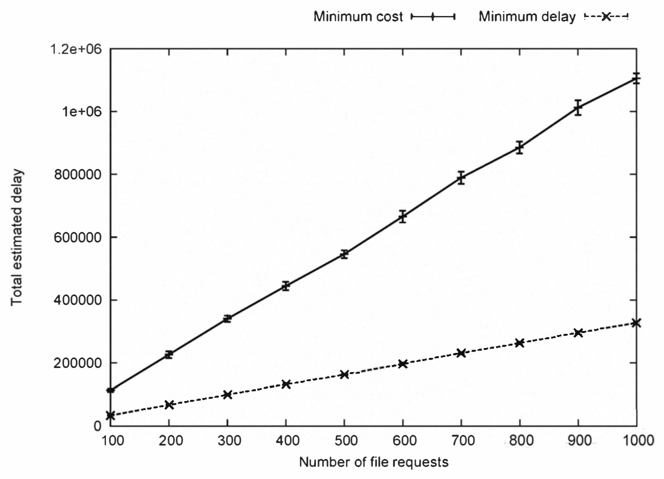

Fig. 3. Total delay

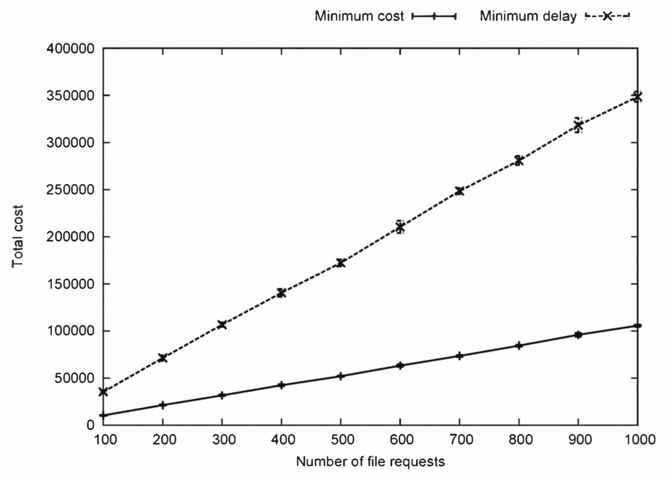

Fig. 4. Total cost

V. Conclusions

This paper addressed the problem of selecting the optimal servers and paths to those servers for a file sharing P2P overlay running on a MANET. The problem was formulated as a mixed integer linear program. In one version, the total cost of all file requests to be satisfied is minimized, subject to a maximum download time. In the other version, the total delay to download the files is minimized, subject to a maximum cost. Files are assumed to be split into blocks, so that users may download from multiple servers simultaneously. The program finds the optimal number of blocks to request from each server along each possible path, but clearly requires *a priori* knowledge of all client download requests, server locations, costs, and download times. In a dynamic P2P–MANET, this knowledge is typically not known beforehand, and so in the future we expect to propose a heuristic algorithm that attempts to match the performance of the optimal path selection algorithm as closely as possible. The linear program solution therefore will serve as a useful benchmark for a heuristic algorithm.

References

[1] M. Adler, R. Kumar, K. Ross, D. Rubenstein, D. Turner, and D. D. Yao, "Optimal peer selection in a free-market peer-resource economy," in *Second Workshop on Economics of Peer-to-Peer Systems*, June 2004.

[2] ——, “Optimal peer selection for p2p downloading and streaming,” in *IEEE INFOCOM*, 2005, pp. 1538–1549.

[3] S. C. Han and Y. Xia, “Constructing an optimal server set in structured peer-to-peer networks,” *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 1, pp. 170–178, January 2007.

[4] D. N. da Hora, D. F. Macedo, L. B. Oliveira, I. G. Siqueira, A. A. Loureiro, J. M. Nogueira, and G. Pujolle, “Enhancing peer-to-peer content discovery techniques over mobile ad hoc networks,” *Computer Communications*, vol. 32, pp. 1445–1459, August 2009.

[5] A. Mawji and H. Hassanein, “A utility–based incentive scheme for P2P file sharing in mobile ad hoc networks,” in *IEEE International Conference on Communications (ICC)*, 2008, pp. 2248–2252.

[6] ——, “Efficient multipoint P2P file sharing in MANETs,” in *IEEE Global Telecommunications Conference (GLOBECOM)*, 2008.

[7] D. Johnson, Y. Hu, and D. Maltz, “The dynamic source routing protocol (DSR) for mobile ad hoc networks for IPv4,” RFC 4728, February 2007.

[8] M. Tarique, K. E. Tepe, S. Adibi, and S. Erfani, “Survey of multipath routing protocols for mobile ad hoc networks,” *Journal of Network and Computer Applications*, vol. 32, pp. 1125–1143, November 2009.

[9] ns 2, “The network simulator,” http://www.isi.edu/nsnam/ns/.

[10] L. M. Feeney, “An energy–consumption model for performance analysis of routing protocols for mobile ad hoc networks,” *Mobile Networks and Applications*, vol. 6, no. 3, pp. 239–250, 2001.

[11] H. R. Varian, *Intermediate Microeconomics, a Modern Approach, 6th ed.* W. W. Norton & Company, 2003.