Evaluation of Peer-to-Peer Network Content Discovery Techniques over Mobile Ad Hoc Networks

Leonardo B. Oliveira   Isabela G. Siqueira   Daniel F. Macedo
Antonio A. F. Loureiro   Hao Chi Wong   José M. Nogueira

Federal University of Minas Gerais, Computer Science Department
Belo Horizonte, MG, Brazil

E-mails: {leob,isabela,damacedo,hcwong,jmarcos,loureiro}@dcc.ufmg.br

Abstract

Both Mobile Ad Hoc Networks (MANETs) and Peer-to-Peer (P2P) Networks are decentralized and self-organizing networks with dynamic topology and responsible for routing queries in a distributed environment. Because MANETs are composed of resource-constrained devices susceptible to faults, whereas P2P networks are fault-tolerant, P2P networks are the ideal data sharing system for MANETs. In this paper we conducted an evaluation of the two approaches for P2P content discovery running over a MANET. The first, based on unstructured P2P networks, relies on controlled flooding, while the second, based on structured P2P networks, uses distributed indexing to optimize searches. We use simulations to evaluate the effect of network size, mobility, channel error rates, network workload, and application dynamics in the performance of P2P protocols over MANETs. Results show that unstructured protocols are the most resilient, although at a higher energy and delay costs. Structured protocols, conversely, consume less energy and are more appropriate for MANETs where topology is mostly static.

1 Introduction

Mobile Ad Hoc Networks (MANETs) enabled new applications e.g. rescue team communication in disaster situations and exchange of information in battle fields [1, 17], where it is not possible to rely on previous infrastructure.

Peer-to-Peer (P2P) networks emerged as a solution for data sharing and processing in distributed environments [1, 10, 16], and are nowadays widely adopted on the Internet.

Only recently, the synergy between these MANETs and P2P networks was recognized [1, 4–7, 10, 11, 14]. Both are decentralized and self-organizing networks, have dynamic topology, and are responsible for routing queries in a distributed environment. In addition, their nodes have equivalent functionalities and capabilities, being able to send and reply to requests originated from one another – as peers.

We go further and argue that MANETs and P2P networks are not only similar, but also complementary. Because nodes in mobile ad hoc networks usually have low computing capacity and, therefore, are unable to play the role of servers all the time, – or even supply many clients simultaneously – a P2P application appears to be a powerful tool to spread information on this scenario. In other words, since a P2P network does not possess a unique service provider at a certain time, the assignment of distributed network tasks among nodes prevents them to become overloaded. In addition, some applications enabled by MANETs (e.g. rescue team communication in disaster situations and exchange of information in battle fields) will have each instance working in cooperation with the others (i.e., sending and replying to queries like peers), e.g., a rescue team participant might require information about nearest neighbor location. Although a central server could be responsible for storing information, this approach would not only be more expensive (this would require more hops and constant location updates), but be also less resilient – a single point of failure is not desirable in rescue team situations and servers would be target of attacks in battle fields.

We contribute to the development of efficient information sharing in MANETs by evaluating existent Internet content discovery techniques in these new distributed environments. Broadly speaking, there are two classes of content discovery techniques for P2P networks [9]: unstructured and structured. In the former (e.g., Freenet, Napster, and Gnutella), data can be stored in any node in the network, hence nodes must flood the network with queries to locate the desired information. In the latter (e.g., Chord, CAN, PASTRY), content discovery is optimized by the creation of a Distributed Hash Table (DHT) which determines a direct path to the desired information. We used a network simulator to instantiate a Gnutella-like protocol and a Chord-like [15] protocol – as the main representatives of unstructured and structured P2P flavors, respectively – and evaluated their performance under different simulated scenarios. The chosen scenarios investigate the impact of different parameters on the performance of both protocols.

Results show that protocols which make use of redundant lookup messages to locate content (i.e., unstructured protocols) are, in general, more efficient for MANETs, although more costly than structured protocols. Structured protocols, on the other hand, are more suitable for static environments and also more energy-efficient.

The rest of this paper is organized as follows. Section 2 presents the simulation environment and the analysis method used in this work, whereas Section 3 shows the simulation results. Section 4 discusses the related work. Finally, Section 5 draws the conclusions.

2 Network Characterization

Due to the lack of actual production data, we derived our simulation model from a hypothetical search and rescue (e.g., in forests, deserts and battle fields) application. In our application, the ad hoc network is composed of Wi-Fi devices (handhelds or PDAs) where data gathered from the field are made available to a P2P network. We envision that in this scenario every member of the team will perform searches and share data with others in the P2P network.

The P2P applications run on top of the UDP protocol, since TCP does not perform well in this type of environment. We chose AODV [13] for routing as it presented the best performance under a P2P application in most common MANET scenarios [10]. Nodes are configured with typical PDA network parameters (11 Mbps IEEE 802.11b with 50 m of range). The interface queue (IFQ) length is set to 30 packets and the energy consumption is 230 mW for reception and 330 mW for transmission [3]. Radio propagation follows the two-ray-ground model. Nodes are equipped with sufficient energy to allow the application to run during the entire simulation – since we focus on efficiency rather than resilience. Each node has a 10% probability of searching or locally storing any given file.

The simulator does not provide P2P protocols and we ourselves implemented the structured and unstructured protocols. Our implementation followed the specification in Chord paper [15] and the Gnutella protocol specification v0.4. For a fair comparison, we chose not to use any optimization that could improve performance over ad hoc networks. Below we briefly describe their implementation.

Chord: We implemented Chord's complete set of functionalities, including the protocols necessary for building and maintaining the distributed indexes. We also implemented file insertion and deletion in the network, using protocols similar to the ones used for search. Regarding Chord's simulation parameters, the *finger table* is updated every 5s and *stabilize* runs every 10 s. PING messages (used for topology control) are sent every 10 s. Packet size is fixed in 64 bytes.

Gnutella: In Gnutella, we handle the problem of propagating queries indefinitely, by creating a time-to-live (TTL) field embedded in every query message. This field is decremented at each hop. Also, every node maintains a message cache. Messages arriving with zeroed TTL value or with an entry in the cache are discarded. To each peer about to join the network, a logical neighborhood composed of a fixed number of peers is assigned. The neighbors are picked at random among the pool of peers online. This assignment is done offline, similar to a central server that functions as the P2P network entry point (this is usual in most Internet Gnutella clients). Finally, Gnutella peers periodically send PING messages to their neighbors and wait for an answer (the PONG message), in order to check if their neighbors are still online. When no answer is received, the neighbor is substituted in the neighbors list by a new neighbor randomly chosen from the set of nodes online at the moment. Regarding Gnutella's simulation parameters, we assume that each node has a maximum number of 4 neighbors (see the Appendix for more details) and a message cache of 100 application messages. The TTL for queries is set to 4 and the PING messages are sent every 10 s. As for Chord, packet size is fixed in 64 bytes.

We investigate the impact of different parameters on performance. To this end we chose a default scenario that we consider as being the closest to the target application conditions we envision. We assume a network of 50 nodes scattered in a 200 m × 200 m grid area. Nodes move accordingly to the random way-point mobility model (since it is frequently used for individual movements [2]) with a pause time of 0.1 s and an average speed uniformly selected from 0 to 1.0 m/s. At any given point of the simulation, 50% of the nodes are always online, while the remaining nodes join the network at some point and leave after a time interval. Join and leave times are chosen following an uniform distribution. Each node provides 5 different files, thus there are 250 different files in the network. In the default scenario we do not consider losses due to channel error.

Each simulation was run 33 times, with different seeds for the random number generator, on ns-2.26 (*Network Simulator*). Results are presented with a 95% confidence interval. We focus our analysis in four metrics: *hit rate* (the

0-7695-2342-0/05 $20.00 © 2005 **IEEE**

fraction of the queries successfully resolved in the P2P network), *response time* (delay perceived by a user requesting some content, including the time for transmitting the query to the network, locating the desired content in the network, and returning a response back to the user), *energy per hit* (energy consumed per one percent of the total hits), *number of messages sent* (total number of messages sent during simulation time).

3 Simulation Results

To evaluate both protocols we analyze the impact of the following factors on performance: network load, network size, channel error rate, mobility, and application dynamics.

Although we have a default scenario, various real world applications could have different network parameters other than the ones we envisioned. Given that different parameter values might affect the performance significantly, we choose to evaluate their impact independently. In the following, we describe each parameter and present the results obtained, including their analysis.

Network Load: We analyze the effect of network load over the performance of the two P2P protocols. We varied the number of distinct files on the network proportionally to the number of queries.

Fig. 1 shows that Gnutella presented the highest hit rate (between 60% and 70%, approximately, against 10% to 20% from Chord). This discrepancy is due to message redundancy in Gnutella, in which peers forward the received query to all neighbors. Chord, conversely, relies on just one copy of a query, being more susceptible to a message drop. However, Gnutella's good result comes at a higher cost in terms of response time and traffic overhead (Figs. 2 and 3). To be specific, Gnutella incurred from 200% to 1570% and from 111% to 851% overhead in response time and traffic, respectively, as compared to Chord.

Gnutella incurred the highest energy consumption (Fig. 4), but its consumption per hit was lower than Chord's. This is due to Chord's high overhead, which has to periodically update information concerning current network state.

Finally, although Chord presented lower hit rates (10%, approximately), it scales gracefully and suffers less impact on load variation.

Network Size: We varied the number of nodes by changing the grid size, at the same time maintaining a fixed network density (1.0 node/m^2). The number of queries per peer was also fixed.

For networks under 100 nodes, Gnutella achieved the highest hit rate, namely 370%, 580%, and again 370% for 25, 50 and 75 nodes, respectively (Fig. 5). When considering response time and traffic overhead, however, Chord outperformed Gnutella: the former was 77% faster and incur from 25% to 70% less traffic overhead (Fig. 6 and Fig. 7, respectively).

It is worth noting that both protocols presented similar energy consumption per hit (Fig. 8), both were very sensitive to variations in network size and had a performance peak at medium-sized networks (from 20 to 50 nodes).

Channel Error: Due to its dependency on link reliability, Chord presented hit rates below 10% (Fig. 9). Gnutella, on the other hand, could perform well in environments with low and medium channel error rates, i.e., 0.05 and 0.01%. However, Gnutella also suffered with high channel error. This occurs because PING, PONG and result messages are not redundant.

Results for response time, traffic overhead, and energy per hit are shown in Figs. 10, 11, and 12, respectively. Gnutella's response time grew almost exponentially (600%) with channel error variation. Chord was more stable, with a maximum variation of 12% (Fig. 10). It also sends approximately 72% less messages than Gnutella (Fig. 11).

Again, the energy consumed per hit by Gnutella is lower than Chord's (approximately, 8 times lower when the error rate was 20%). The same metric for Chord showed an undesirable growth. As less queries are completed, the overhead for maintaining a Chord ring dominates energy consumption and leads to an increase in energy consumed per hit. This is supported by Fig. 13, which shows that Chord still consumes less energy per node than Gnutella, even for scenarios where Chord spends more energy per hit.

Node Mobility: We studied mobility by varying the nodes average speed from 0, 0.25, 0.5, 1, 2, 4 up to 8 m/s. Under low mobility, Gnutella transmitted more messages (Fig. 16) and, as a consequence, response times (Fig. 15) increased. From Fig. 15, it seems that some mobility is beneficial to Gnutella, since less packets were correctly delivered and thus less traffic was imposed to the network. However, under higher mobility, both protocols suffered an increase in response time and a decrease in hit rate. Unlike other sets of simulation, Chord was the less stable protocol and presented high variation in many metrics. As speed was increased, Chord's hit rate (Fig. 14) decreased from 50% to 5% at 2 m/s, and achieved values close to zero when mobility is increased. Gnutella was less affected, having hit rates above 60% for the whole set of simulations.

Fig. 18 shows the average energy consumption per node during the entire simulation. Curiously, energy consumption decreased as mobility increased. Concerning energy per hit, Gnutella maintained its stability, whereas Chord had its consumption increased (Fig. 17).

Finally, note that the lower the mobility the smaller the number of messages dropped during the simulation

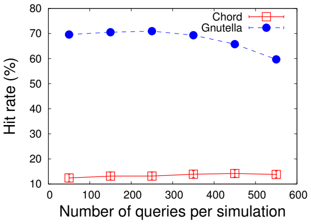


Figure 1. Scen. A: hit rate

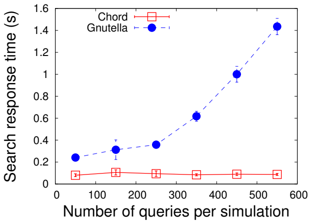


Figure 2. Scen. A: response time

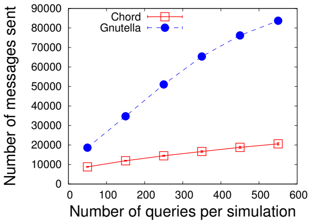


Figure 3. Scen. A: messages sent

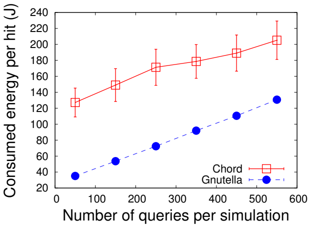


Figure 4. Scen. A: energy per hit

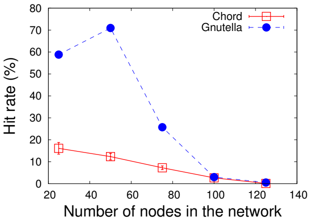


Figure 5. Scen. B: hit rate

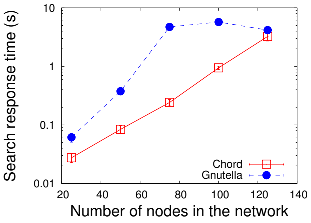


Figure 6. Scen. B: response time

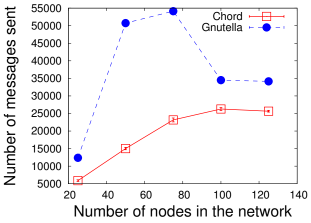


Figure 7. Scen. B: messages sent

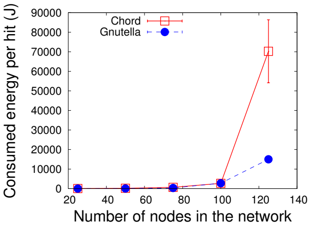


Figure 8. Scen. B: energy per hit

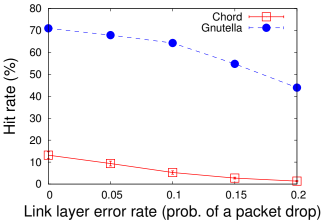


Figure 9. Scen. C: hit rate

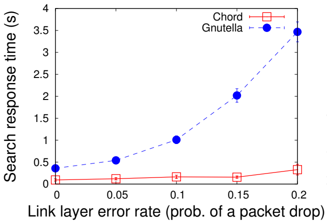


Figure 10. Scen. C: response time

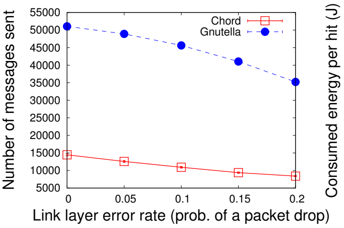


Figure 11. Scen. C: messages sent

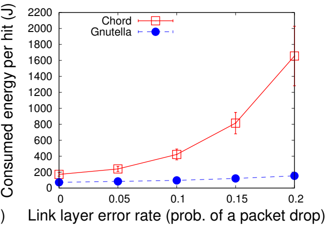


Figure 12. Scen. C: energy per hit

(Fig 19). These drops are caused by collision and neighbors that are out of reach. The increase in mobility results in more broken routes and collisions in the MAC layer. For speeds over 4 m/s, however, the number of dropped messages seems to stabilize in around 500 for Chord and 10.000 for Gnutella.

Application Dynamics: Next, we evaluate how the network dynamics, i.e., nodes joining and leaving the network, impacts the performance of both protocols. Gnutella has fast setup, as a node's task before joining the network is to find neighbors. Chord, in contrast, requires a node to carry out a lengthy set of operations in order to join and leave the network. We varied the percentage of nodes leaving the network from 50% to 0%. Initially, we varied the number of dynamic nodes, i.e, nodes that leave the network at some point in the simulation, in a mobility environment. In this situation, both protocols showed almost no performance variation. So, we repeated the simulations for a stationary network, achieving the results presented below.

As the number of dynamic peers was decreased, the availability of peers increased and the Gnutella network saturated with queries. This is apparent from the response time (Fig. 21) and the number of messages sent (Fig. 22), which are inversely proportional to the number of dynamic peers.

Chord did not perform well in highly dynamic topologies, but it was the best under 20% of dynamic nodes. Up to this point, Chord achieved hit rates similar to those of Gnutella, while being faster and more efficient (Fig. 21 and Fig. 23, respectively). This occurs because an increase in Chord peers in non saturated networks increases the hit rates, also slightly increasing network traffic.

It is worth mentioning that some ad hoc networks (such as the ones employed in rescue situations) will exhibit a significant amount of disconnections due to harsh environmental conditions, but there are also more "well-behaved" ad hoc networks. This scenario shows that Chord is more suitable for less dynamic networks, requiring less energy

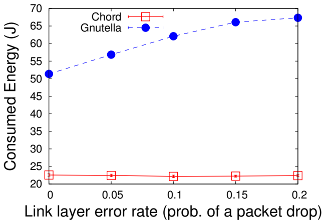


Figure 13. Scen. C: energy per node

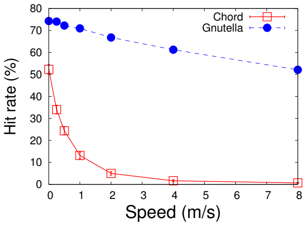


Figure 14. Scen. D: hit rate

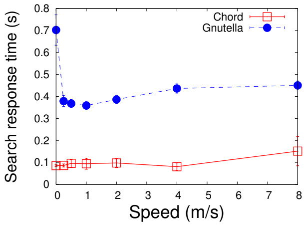


Figure 15. Scen. D: response time

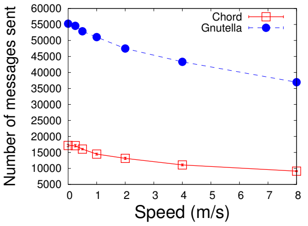


Figure 16. Scen. D: messages sent

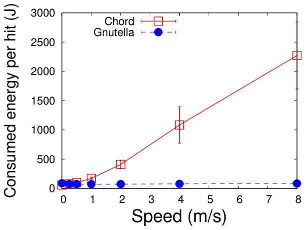


Figure 17. Scen. D: energy per hit

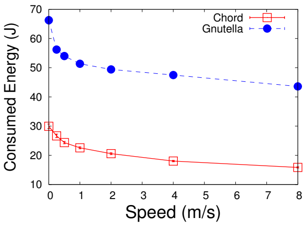


Figure 18. Scen. D: energy per node

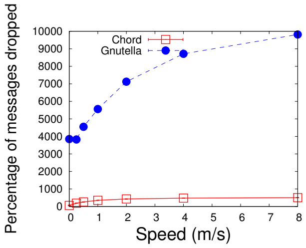


Figure 19. Scen. D: packets dropped

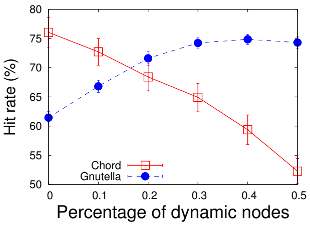


Figure 20. Scen. E: hit rate

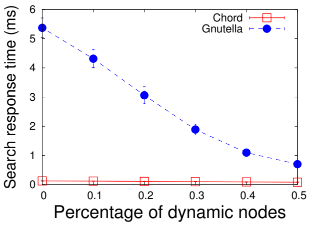


Figure 21. Scen. E: response time

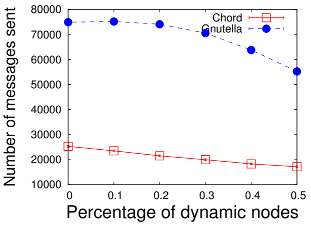


Figure 22. Scen. E: messages sent

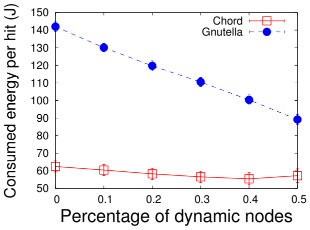


Figure 23. Scen. E: energy per hit

consumption and yielding lower response times when compared to Gnutella. For applications where disconnections are frequent, conversely, Gnutella is the more robust choice.

4 Related Work

Only recently research community realized the synergy between MANETs and P2P networks, thus there are few articles in this context (e.g., [1, 4–8, 10–12, 14]).

Oliveira *et al.* [10] [11] studied an unstructured P2P application running over a MANET where three different ad hoc protocols (DSR, AODV, DSDV) were considered under a number of scenarios.

Franciscani *et al.* [5] concentrated on minimizing the impact of the highly dynamic topology obtained through the combination of P2P networks and MANETs, may have over network resources by proposing algorithms for (re)configuration of these networks.

Ding and Bhargava [4] performed a theoretical comparison between P2P systems over MANETs (broadcast over broadcast; broadcast; DHT over broadcast; DHT over DHT; and DHT) and presented important results in O-notation. Nevertheless, they did not evaluate real P2P systems and did not take into account practical aspects (e.g., mobility and channel error).

5 Conclusion

In this work we studied (through simulation) the performance of content discovery techniques for P2P networks over MANETs. Results show that unstructured protocols are more resilient. This advantage, however, comes at a higher price in terms of energy consumption, network bandwidth, and delay. Structured protocols consume less resources, but are only adequate to MANETs where topology is mostly static, i.e., with low mobility and few node disconnections.

As future work, we will work towards developing new

P2P content discovery techniques tailored to MANETs. Also, a study on best policies for defining logical neighbors in ad hoc networks is another direction for future work.

References

[1] J. Borg. A comparative study of ad hoc & peer to peer networks. Master's thesis, University College London, 2003.

[2] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *4th annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 85–97, Dallas, USA, 1998.

[3] J.-C. Cano and P. Manzoni. A performance comparison of energy consumption for mobile ad hoc network routing protocols. In *8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'00)*, pages 57–64, San Francisco, CA, August 29-September 1 2000. IEEE Computer Society.

[4] G. Ding and B. Bhargava. Peer-to-peer file-sharing over mobile ad hoc networks. In *2th IEEE Annual Conference on Pervasive Computing and Communications Workshops*, pages 104–108, Orlando, Florida, March 2004.

[5] F. P. Franciscani, M. A. Vasconcelos, R. P. Couto, and A. A. F. Loureiro. Peer-to-peer over ad-hoc networks: (re)configuration algorithms. *Journal of Parallel and Distributed Computing (JPDC)*. Special Issue.To appear. Also appeared in 17th IEEE International Parallel and Distributed Processing Symposium 2003 (IPDPS'03).

[6] Y. C. Hu, S. M. Das, and H. Pucha. Exploiting the Synergy between Peer-to-Peer and Mobile Ad Hoc Networks. In *HotOS-IX: Ninth Workshop on Hot Topics in Operating Systems*, Lihue, Kauai, Hawaii, May 18–21 2003.

[7] A. Klemm, C. Lindemann, and O. P. Waldhorst. A special-purpose peer-to-peer file sharing system for mobile ad hoc networks. In *IEEE Semiannual Vehicular Technology Conference (VTC2003-Fall)*, October 2003.

[8] G. Kortuem. Proem: a middleware platform for mobile peer-to-peer computing. *SIGMOBILE Mobile Computing and Communicatin ReviewCommun*, 6(4):62–64, October 2002.

[9] X. Li and J. Wu. *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks*, chapter Searching Techniques in Peer-to-Peer networks. CRC Press, 2005. To appear.

[10] L. B. Oliveira, I. G. Siqueira, and A. A. F. Loureiro. On the performance of ad hoc routing protocols under a peer-to-peer application. *Journal of Parallel and Distributed Computing(JPDC)*. Special Issue on the D̈esign and Performance of Networks for Super-, Cluster-, and Grid-Computing.̈To appear.

[11] L. B. Oliveira, I. G. Siqueira, and A. A. F. Loureiro. Evaluation of ad hoc routing protocols under a peer-to-peer application (WCNC'03). In *IEEE Wireless Communications and Networking Conference*, pages 1143–1148, New Orleans, USA, March 2003.

[12] M. Papadopouli and H. Schulzrinne. Effects of power conservation, wireless coverage and cooperation on data dissemination among mobile devices. In *2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 117–127. ACM Press, March 2001.

[13] C. E. Perkins and E. M. Royer. Ad hoc on-demand distance vector routing. In *2nd IEEE Workshop on Mobile Computing System and Applications*, pages 90–100, New Orleans, LA, Feb. 1999.

[14] R. Schollmeier, I. Gruber, and M. Finkenzeller. Routing in peer-to-peer and mobile ad hoc networks: A comparison. In *International Workshop on Peer-to-Peer Computing*, 2002.

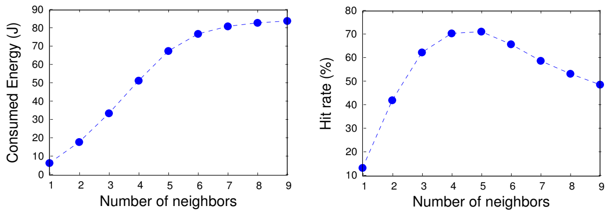
[15] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions on Networking*, 11(1):17–32, 2003.

[16] D. Talia and P. Trunfio. Toward a synergy between P2P and grids. *IEEE Internet Computing*, 7(4):94–95, 2003.

[17] L. Zhou and Z. J. Haas. Securing ad hoc networks. *IEEE Network*, 13(6):24–30, 1999.

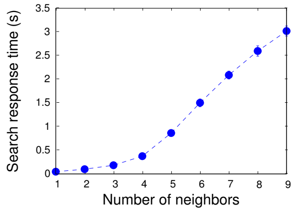
Appendix: Gnutella Neighbors

In Gnutella, a peer sends a copy of a query to all its neighbors. This increases hit rates and resiliency, but incurs more traffic and energy consumption. We analyzed this trade-off by varying the number of Gnutella neighbors. Results for energy consumption, hit rate and response time are shown in Figs. 24(a), 24(b) , and 24(c), respectively. Note that Gnutella reached the highest hit rate with 4 and 5 neighbors. Also note that response time and energy consumption for 5 neighbors is much higher than for 4. Thus, we opted to use 4 neighbors in our simulations.



(a) Energy consumption

(b) Hit rate



(c) Response time

Figure 24. Simulation results for Gnutella.