Efficient Search and Topology Adaptation in Mobile Peer-to-Peer Networks

Kuan-Kung Huang, Jonathan Chun-Hsien Lu*, and Hsin-Hung Lin
Dept. of Computer Science and Information Engineering
Fu Jen Catholic University, Taipei, Taiwan
*jonlu@csie.fju.edu.tw

Abstract

With the rapid development of wireless technology, more and more people use handheld devices to enjoy their mobility. A mobile ad hoc network (MANET) consists of only mobile devices to share information without any predefined infrastructure. These devices are intrinsically constrained by limited battery capacity, low communication bandwidth, and short transmission range. How to share information efficiently on MANETs is an interesting topic. Peer-to-peer (P2P) networks are another research area that has gained a lot of attention these days, and many different kinds of architectures have been proposed. However, most of these research results only focused on the wired networks. Since P2P networks and MANET have some common characteristics, it is appealing to apply P2P file sharing onto MANETs.

In this work, we propose a new file search and topology adaptation mechanism to make file sharing on MANETs more efficient. We use the neighboring regions to solve the null region problem and design rules for dynamically adjusting the scope of the network to closely match the physical network topology. Simulation results showed that our proposed file search mechanism could achieve excellent performance on MANETs.

Keywords: Peer-to-peer system, ad hoc network, topology adaptation, file search.

1: Introduction

Today many mobile devices such as laptop, Personal Digital Assistant (PDA), cell-phone, and smart phone, etc., come out one after another due to the rapid development in science and technology. More and more people begin to use these devices to enjoy their mobility. These devices are usually equipped with wireless communication capability so that they can retrieve information from the Internet without physical wires. Furthermore, these devices can even form a mobile ad hoc network (MANET) to communicate directly with each other without any predefined infrastructure. Peer-to-peer (P2P) networks are another research area that has received a lot of attention recently. In P2P networks, peers act as both servers and clients simultaneously and can communicate with each other to share resources. One of the important issues in P2P networks is how to share resources efficiently. Since P2P networks and MANETs both lack a fixed infrastructure and all the peers in these two types of networks can join or leave the system frequently, it is appealing to apply P2P file sharing onto MANETs. How to provide quality services for users to share information on P2P systems and MANETs together has become an important research task.

Many P2P system architectures have been proposed, which can be roughly divided into unstructured P2P systems and structured P2P systems. Gnutella [1] is a well-known example of the unstructured P2P systems, in which peers are connected arbitrarily and flooding is the main query mechanism. Due to the constrained resources in most mobile devices, the flooding mechanism is not suitable for implementing file sharing in MANETs. In structured P2P networks, a distributed hash table (DHT) is used to publish and search for files in the system. CAN [2], Chord [3], and Pastry [4] all belong to this kind of P2P systems. Although it takes some time to publish each file to its responsible peer, this DHT-based search mechanism is more efficient and suitable for the mobile environment. However, since the peers in MANETs can move around at will, the routing path to the target peer may need to be frequently rediscovered every time a query is to be processed. This can impose a large load on the system. Because of the constrained resource in most mobile devices, we must have a mapping mechanism that can avoid frequent rerouting in order to reduce the overhead on file management in MANETs.

Mobile peer-to-peer (MP2P) network is a special kind of P2P network which only consists of mobile devices operating in a wireless environment. As opposed to the wired environment, the wireless network usually lacks a fixed infrastructure which can provide steady routing support. The devices in the wireless environment are also more resource constrained, such as limited battery capacity and low communication bandwidth. Peers in the wireless environment need to act as routers for relaying information from one peer to another due to limited transmission range. Besides, the network topology in wireless environment is frequently varying. Because

of these limitations, we are facing challenges different from those in the wired P2P networks, and we must design a new scheme to overcome these inherent shortcomings. In this work, we propose a new file search and topology adaptation mechanism for efficient resource sharing in MP2P networks. The remaining part of this work is organized as follows: Section 2 reviews the related work on MP2P systems. Our proposed mechanism is presented in section 3. Section 4 displays the simulation results, and section 5 gives the conclusion.

2: Related Work

Several schemes had been proposed to provide efficient file sharing in MP2P networks. Proximity Regions for Caching in Cooperative MP2P Networks (PReCinCt) [5] worked essentially like the DHT in structured P2P networks, but it modified the mapping method to use geographical information to reduce the mobility overhead. It divided a given geographical area into several small equal-sized regions, and peers in each small region would be collectively responsible for a set of keys. When a peer wanted to publish a file, it hashed the file to a specific key value, and applied a second hash function to map this key to a certain geographical region. The file is then registered to the peers in that region using a geography-aided routing protocol such as the Greedy Perimeter Stateless Routing (GPSR) [6]. Search for a file was done in a similar manner. A file would always be hashed to the same geographical coordinate, which meant the same geographical region. Therefore, this scheme created a balanced indexing load across all the geographical regions even though peers could move here and there. However, it did not address the null region problem, where there might be no peer in a given region to hold the file registration information. How to determine the best size of each small region was not discussed either.

It is also preferable for MP2P systems to always use paths of lower cost to route the information, because most mobile devices are inherently resource constrained. A geography-aided protocol called Obstacle-Free Single-Destination Geocasting Protocol (OFSGP) was proposed in [7]. There the network was partitioned into several equal-sized hexagonal regions and the peer located closest to the center of the region was selected as the manager for this region to handle the routing information. When a peer wanted to send a message to one of the other peers, it had to first find out the region which the destination peer resided in. To do this, the source peer sent the message to the manager in its own region, who would relay the message to three other neighboring managers in the directions that would lead closer to the destination. The new managers who received the routing message would repeat this process until the destination was reached. However, there could exist some obstacles in the network, which might force some regions to be null regions all the time. If the next three neighboring regions all consisted of obstacles, the manager would relay the message to the other three neighboring regions. In case the message could not be forwarded to any of the neighboring regions, it would backtrack to the previous manager who sent it over. Basically, each message would be routed along the shortest path if there existed no obstacles in the system. In case obstacles do exist, OFSGP would explore additional routing direction to try to go around the obstacle to reach the destination if possible.

None of the works so far has provided efficient solutions to the null region problem. Besides, most of the proposed schemes considered a fixed network topology only. However, the peers in MP2P networks are mobile and unpredictable that the network topology could be constantly varying. Therefore, we would like to propose a method to address the null region problem and to map the logical topology closer to the physical topology.

3: Proposed Mechanism

3.1: System Assumptions

Here we describe our assumptions. First, the entire network topology is divided into several equal-sized squares. While the area of the entire network could keep changing, all the peers in the system are assumed to know its current scope at any instant. Second, we assume that every peer in our system is equipped with a capability such as GPS so that it knows the geographical information of itself as well as its neighbors. Next, the devices use the same transmission technology with the same transmission range R. To guarantee that a peer in any given region can communicate directly with the peers in its eight direct neighboring regions, we set the length of the side of the square equal to $R / 2\sqrt{2}$ as shown in figure 1. By this setup, every single peer can easily detect if there is any null region in its direct neighborhood. The peer which is located closest to the center of a square will be selected as the manager of this square. When the manager of a square leaves the square, the peer located second closest to the center of the square will be selected as the new manager, who will then notify all the other peers in the square immediately. When a peer wants to publish a file, it first uses the hash function to create a two-dimensional geographical coordinate. It then calculates the target region which the geographical coordinate belongs to and sends the file registration information to the target region. When this information is routed to the first peer in the target region, that peer will broadcast the registration within the region so that it can be stored in all the other peers in the target region for fault tolerance. To search for a file, a source peer uses the same process

to locate the target region and sends a request to try to retrieve the registration information of the file. Afterwards, the source peer uses the underlying wireless routing protocols, such as DSDV [8], AODV [9], or DSR [10], etc., to forward the message to the file owner to request a copy of the file.

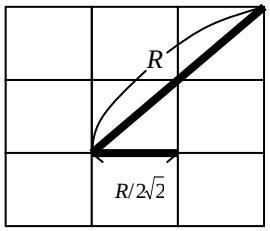

Figure 1: Size of a region.

3.2: Coverage Method

When a manger peer *M* detects a null region in its direct neighborhood, it will try to cover this null region by sending a coverage message to notify the managers of all the other neighboring regions of this null region. If this null region had already been covered by another neighboring region, the manager in the covering region will return an *already_covered* message, and *M* will cancel its coverage notification and record the information about the covering region. If the null region has not been covered, M will cover it by asking all the peers in *M*'s region to be collectively responsible for holding all the registration information of files belonging to the null region. All the registration information and the file requests destined for a null region will be redirected to any peer in its covering region. Since the covering region is always adjacent to the null region, a message destined for the null region can be routed to its covering region following almost the same routing path, which consumes almost no extra bandwidth.

Figure 2 gives a simple example where the network topology is comprised of nine regions. Region r5 is the only null region, and the managers in all of its four direct neighboring regions: r2, r4, r6 and r8 are all trying to cover it by sending the coverage messages to the other managers in all of r5's neighboring regions. The one with the smallest region id (r2 in this case) wins the tie and takes the coverage. The other three regions will record the fact that r5 is currently covered by r2.

r1	**r2**	**r3**
r4	**r5**	**r6**
r7	**r8**	**r9**

Figure 2: Coverage for the null region r5.

3.3: Routing Protocol

Each region in our system has eight direct neighboring regions while OFSGP has six, so next we try to modify the OFSGP routing algorithm. Similar to OFSGP, we consider the null regions as obstacles in the system, and we try to send the message along additional directions to route around the null region. When a peer wants to send a message to a destination region, it first calculates that which direction will provide the shortest distance to the destination, and try to forward the message to the neighboring region in that direction. If the neighboring region in that direction is a null region, the peer will choose the second best direction to route the message, etc. All the relaying peers repeat this process to choose the direction to route the message until the destination is reached, or there is no feasible routing path to the destination. When there is no null region, all the messages will be routed to the destination along the shortest path. Even though there may exist some null regions, the messages will be routed around along to the destination if possible.

Figure 3 shows a network topology comprised of sixteen regions to illustrate our routing protocol. Suppose that a peer in region r1 wants to send a message to the destination region r16 and region r6 is a null region. First, the source peer in region r1 will calculate the shortest distances of routing to region r16 through different neighboring region and record them in a table (as shown in Table 1). It then first tries to forward the message to the neighboring region that gives the minimum distance, which is region r6 in this example. Because region r6 is a null region, it then tries the second best region (region r2 or r5) to forward the message. Suppose that it sends the message to region r5. The peer in r5 which received the message will repeat the process and forward the message to region r10. The message will later be routed to region r11, and reach the destination in region r16 in the end.

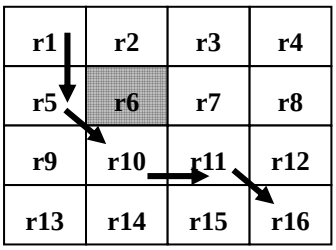

Figure 3: A routing example from r1 to r16.

Next Region	r6	r2	r5
Distance in number of hops	3	4	4

Table 1: Table of shortest distances from r1 to r16 in figure 3.

3.4: Dynamic Adjustment of Network Scope

Because peers in MANETs can move frequently, the boundary of physical area these peers occupy may keep changing. In this section we describe how the

network scope can be expanded or reduced.

3.4.1: Scope Expansion. To do scope expansion, the peers residing in the boundary regions must keep monitoring the number of the peers moving out of the boundary. If a manager discovers that there are peers located outside for a period of time, it will send a probe message to the managers in the other regions along the same boundary line to calculate the total number of peers located out of the boundary on this side. For example, suppose region r3, r6 and r9 in figure 4 all detect the presence of other peers to the right. Suppose the manager in r3 starts the scope expansion process by preparing a probe message which contains the number of peers outside r3, and sending the probe down to r6. r6 adds its number of outside peers to the probe, and forwards it downward. This is repeated until the probe reaches the other corner region along the boundary, and the probe travels reversely back to the initiator (r3 in this case). Because we would like to have at least two peers responsible for each region in an ideal situation, the network scope will not be expanded unless the total number of outside peers is at least twice the number of the regions on that side of the boundary. If r3 decides to expand the topology, it will broadcast the new scope to all the peers in the system. Any new files generated from now on will be hashed based on the new scope.

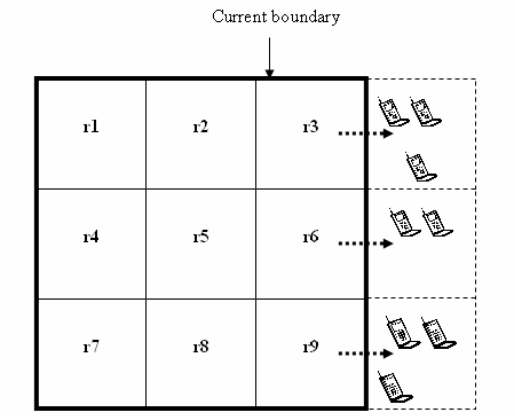

Figure 4: Example of the scope expansion.

3.4.2: Scope Reduction. When peers move around, null regions may be created around the boundary. If all the regions along a boundary side become null and stay null for an extended period of time, the network scope should be reduced accordingly. Because the scope reduction can be done only when the regions along a boundary side are all null, the reduction process can only be initiated by a corner region. The procedure is similar to that for scope expansion. If a corner region stays null for a while, the manager responsible for this corner region will send out a *reduction_request* message to be forwarded to the managers in each of the regions along the same boundary side to check if these regions are all null at the same time. If yes, the network scope will be reduced, and the new scope is made known to all the other peers in the system by broadcast.

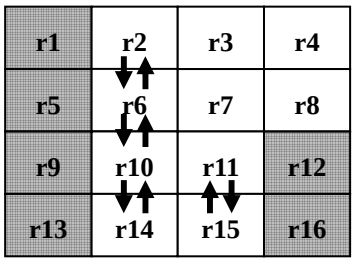

Figure 5: Example of the scope reduction.

Figure 5 gives an example of the scope reduction. Suppose regions r1, r5, r9, r13, r12 and r16 are all null, and regions r2, r6, r10, r14, r11 and r15 are their covering regions, respectively. Because region r2 covers the corner region r1, r2 will initiate the reduction process by sending a *reduction_request* message to r6. If r6 agrees to reduce, it then relays the message to r10. This process is repeated until the message reaches region r14, which covers another corner region r13, r14 will reply a *reduction_agreed* message, which will traverse reversely back to r2 to confirm the reduction of the scope. Separately, the reduction request initiated by region r15 will be rejected by r7 because the region r8 to the right of r7 is not null.

4. Simulation Results

We have implemented a simulator in Java Platform, Standard Edition [11] to evaluate the performance of our proposed methods. We first study the influence of the peer density and the number of routing directions employed. The network is divided into 8 regions * 8 regions. A number of peers ranging from 32 to 192 are generated and randomly located in the network. There are totally 12,288 files evenly distributed to the peers in the system. Each peer will issue the same amount of queries and the total number of queries issued in the system is equal to 3072. The movement probability is set to 0.5.

Figure 6 shows that the hit rate increases when more routing directions were explored for each query. However, when there are too few peers in the system, the hit rate could not reach 1 as there may exist many null regions which cause some areas to be isolated from the rest of the network. As there are more peers in the system, the hit rate increases as well because null regions gradually disappear. Figure 7 presents the total amount of bandwidth consumed by the queries. We assume that a query message traversing one hop consumes one unit of bandwidth. While the number of routing directions increases, both the bandwidth and the hit rate increase as well because the peers try to explore more directions to reach the destination. When there are a sufficient number of peers (128 and 192 peers in this case) in the system, very few null regions exist and most queries can be

satisfied in trying one or two directions in routing, so the consumed bandwidth stays the same as the number of routing directions grows beyond two. However, when there are not enough peers in the system, the consumed bandwidth increases as more routing directions are tried to lift the hit rate.

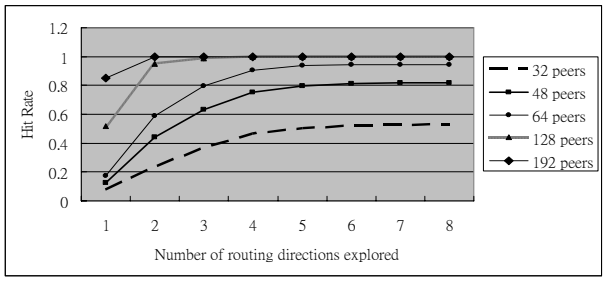

Figure 6: Hit rates under different density and number of routing directions.

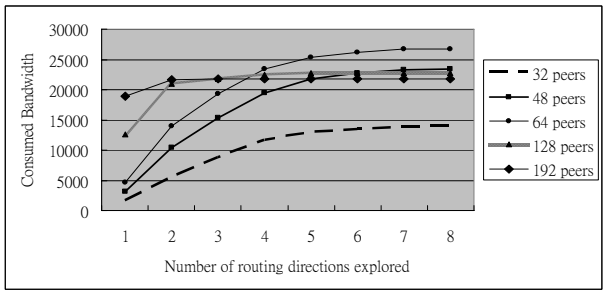

Figure 7: Consumed bandwidth.

Next we investigate how well our routing protocol can route around an obstacle consisting of a group of null regions. Figure 8 shows a network topology where an L-shaped area of null regions exists in the middle. Again the moving probability is set equal to 0.5. Figure 9 displays the hit rates for both the cases of normal topology and the topology with an L-shaped obstacle. The results show that the hit rate becomes lower with an obstacle, and trying more routing directions still could not increase the hit rate to be the same as the normal topology when there are not enough peers to help route around the obstacle. When there is sufficient number of peers, the hit rate can go up to 1 even in the existence of obstacle. Figure 10 shows that consumed bandwidth. For routing around the L-shaped obstacle, the consumed bandwidth will increase as well.

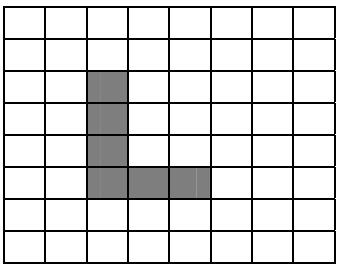

Figure 8: An L-shaped obstacle in the middle of the system.

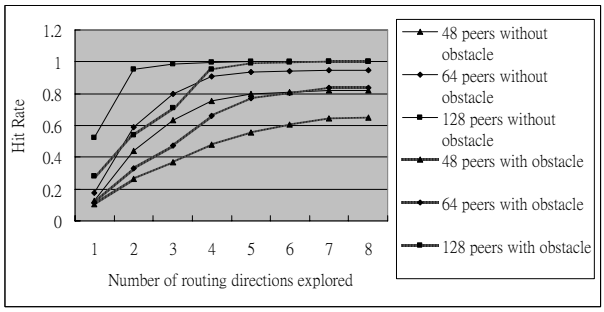

Figure 9: Hit rates for network with L-shaped obstacle shown in figure 9.

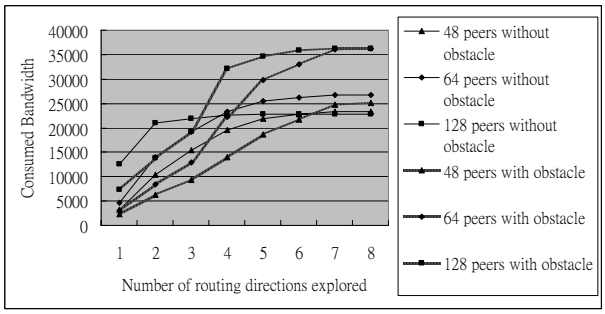

Figure 10: Consumed bandwidth for network with L-shaped obstacle shown in figure 9.

To study the effects of a peer's moving speed, we next assume a 20-region * 20-region network topology with peers placed only in the 10-region * 10-region area at the center in the beginning. There are totally 200 peers and 12,000 files evenly distributed on the peers. We assume that at the end of each time unit a peer will move out of its current region with a probability of 0.9. A moving speed of k means that a peer can move to a new region at most k regions away if it decides to move. Figure 11 shows that with a higher moving speed, peers would spread out from the initial center area more quickly, and the hit rate dropped to the steady state value much faster.

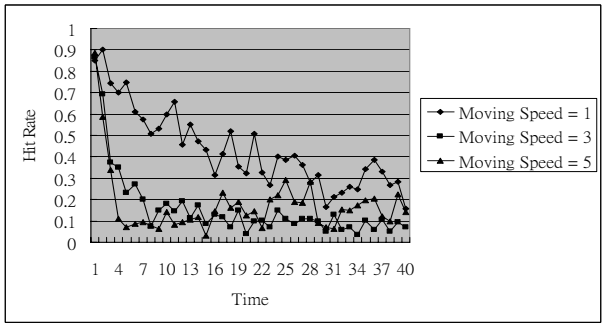

Figure 11: Hit rates under different moving speeds.

5. Conclusion

With the rapid development on wireless technology, MANETs have become an interesting research topic. P2P networks are another important

research topic which has received a lot of attention. Because these two types of system share similar characteristics, more and more research has focused on applying P2P file sharing onto MANETs, but few of them addressed the null region problem. In this work, we proposed a new mechanism to solve the null region problem and modify the OFSGP algorithm to route the messages around obstacles in the system. In addition, we proposed a scheme to dynamically adjust the scope of the network to make the logical topology closely match the physical topology. Simulation results showed that our coverage method and routing protocol could efficiently solve the null region problem and provided excellent performance.

References

[1] "Gnutella," http://www.gnutella.com/

[2] S. Ratnasamy, P. Francis, M. Handley and R. Karp, "A Scalable Content-Addressable Network," *in Proceedings of the ACM SIGOMM Conference*, 2001.

[3] I. Stoica, et al., "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications," *IEEE/ACM Transactions on Networking*, Vol. 11, No. 1, February 2003.

[4] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems," *in Proceedings of the IFIP/ACM Middleware Conference*, Nov. 2001.

[5] M. Joseph, M. Kumar, H. Shen and S. Das, "Energy Efficient Data Retrieval and Caching in Mobile Peer-to-Peer Networks," *Pervasive Computing and Communications Workshops*, 2005.

[6] B. Karp, and H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *in Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom 2000)*, August 2000.

[7] C. Y. Chang, C. T. Chang, and S. C. Tu , "Obstacle-Free Geocasting Routing Protocol for Ad-Hoc Wireless Networks," *ACM/Baltzer Journal of Wireless Networks*, 2003.

[8] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance vector routing (DSDV) for mobile computers", *Computer Communications Review*, pp. 234-244, October 1994.

[9] C. Perkins, E. Royer, and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing," IETF RFC 3561, 2003.

[10] D. Johnson and D. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, Kluwer Academic Publishers, 1996.

[11] http://java.sun.com/