ABSTRACT

FENG, FANG. Mobile Movement Patterns and Applications in Wireless Networks. (Under the direction of Professor Douglas S. Reeves).

In a real-life wireless network, the logical movements of mobile nodes on the subnet level are not purely random. Movements of individual mobile nodes have intrinsic patterns determined by regular activities of individual persons. Regularities in people’s group activities also introduce patterns in co-location behavior of multiple mobile nodes. Mobile nodes are able to predict their future behavior using history information, and prediction results can be used to expedite network management processes and reduce the required overhead. Our research focus on characterization and applications of movement and co-location patterns.

We first introduce a fast handoff mechanism with movement prediction for wireless IP networks. Each mobile node records movement history information, and predicts its next subnet before the actual movement. It explicitly notifies the current foreign agent to duplicate and forward packets to the predicted subnet. Simulation with real-life wireless network trace shows that the latency of network-layer handoff and the amount of packet loss are greatly reduced, only with a limited overhead in packet duplication and forwarding.

The topology matching issue for peer-to-peer (P2P) overlay of mobile nodes is also investigated, and a Local Topology Cache mechanism is designed to reduce the overhead of topology matching, and at the same time keep the P2P overlay matched to the physical network topology. As mobile nodes have patterns in their movement and interaction, a mobile node’s neighbors in a P2P overlay that matches the physical network topology when it visited the subnet previously might also be those when the mobile node returns to the same subnet. The mobile node caches the information of its neighbors in a topologically-matched P2P overlay and reuses them when returning to the same subnet, without probing the network again. We simulate this scheme with a real-life wireless network trace, and found the caching mechanism can greatly reduce network probing overhead, while achieving similar efficiency of P2P overlay topology.

We further investigate the co-location behavior of multiple mobile nodes. People’s regular interactions determine that co-location of mobile nodes has regularities. Using real-life wireless network traces, we measure the characteristics of mobile nodes’ co-location behavior, and show that co-location has patterns and is repetitive, which provides the basis of co-location prediction. A Markov-family model is used to dynamically model the
co-location behavior, and a fully distributed co-location prediction method only using a mobile node’s own movement trace and co-location history is proposed. The effectiveness of this co-location prediction method is demonstrated with simulations based on real-life wireless network activity traces. We also utilize the co-location prediction method in the construction of the P2P overlay in a wireless network, and show that it can construct a P2P overlay that matches the physical network topology, without probing the network. This demonstrates that co-location prediction can indeed expedite network management and reduce the associated overhead.
Mobile Movement Patterns and Applications in Wireless Networks

by

Fang Feng

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Computer Engineering

Raleigh, North Carolina

2007

Approved By:

__________________________________________
Dr. Wenye Wang                                Dr. Matthias Stallmann

__________________________________________
Dr. Douglas S. Reeves                          Dr. Arne A. Nilsson
Chair of Advisory Committee
DEDICATION

To my grandmother,

who brought me such vivid and wonderful memories.
Biography

Fang Feng was born in Nanjing, China. He received his B.S. degree in Automation from Tsinghua University, Beijing, China in 2000.

He is currently a Ph.D. candidate in Department of Electrical and Computer Engineering at North Carolina State University. His current research interests include design of network management protocols for wireless IP networks, characterization and application of node movement and interaction patterns, and mobile overlay networks.
Acknowledgements

First I would like to express my deepest appreciation to my advisor, Dr. Douglas S. Reeves. Without his guidance, encouragement and support, this dissertation would not even be possible. His great insight has led me through the long journey of Ph.D. research. I truly thank Dr. Reeves for all his advice and help.

I would like to thank Dr. Arne A. Nilsson, Dr. Wenye Wang, and Dr. Matthias Stallmann for serving on my advisory committee. I always enjoy their excellent lectures on telecommunication network design, wireless networks and algorithms, and appreciate their advice on my Ph.D. research.

Many thanks to all the members of the Cyber Defense Laboratory, Kehang Wu, Qingling Jiang, Pan Wang, Young June Pyun, Young Hee Park, Pai Peng, Kun Sun, Dingbang Xu, Qinghua Zhang, Qin Zhang and etc. I enjoy the collaboration in our group.

Finally, my sincere thanks go to my parents, Jihua Qian and Dae Feng, and my girl friend, Xin Cai, for their love and support over the years.
Contents

List of Figures viii
List of Tables x

1 Introduction 1

1.1 Network Scenario .................................................. 1
  1.1.1 Wireless Networks ............................................. 1
  1.1.2 Peer-to-Peer Overlay ........................................... 4

1.2 Challenges of Node Mobility ........................................ 5
  1.2.1 Handoffs .......................................................... 5
  1.2.2 Topology Matching for Unstructured P2P Overlay ............ 9

1.3 Movement Patterns and Applications ............................... 14

1.4 Contributions of Our Research .................................... 16

1.5 Outline of the Dissertation ....................................... 19

2 Related Work 20

2.1 Related Work on Movement Modeling ............................... 20
  2.1.1 Synthetic Mobility Models ...................................... 21
  2.1.2 Mobility Models Extracted from Actual Movement History .... 24

2.2 Related Work on Handoff ............................................ 27
  2.2.1 Reactive Handoff ............................................... 27
  2.2.2 Proactive Handoff .............................................. 28
  2.2.3 Combined Handoff .............................................. 28

2.3 Related Work on P2P Overlay Topology Optimization ........... 29

3 Explicit Proactive Handoff with Motion Prediction 32

3.1 Introduction ....................................................... 32

3.2 Proactive Handoff with Motion Prediction ................................ 34
  3.2.1 Motivations ..................................................... 34
  3.2.2 System Overview ............................................... 35
  3.2.3 Handoff Process ............................................... 37
  3.2.4 Path Prediction Algorithm .................................... 39
### 5.6.2 P2P Overlay Construction Based on Co-Location Prediction


103

### 5.6.3 Performance Evaluation


104

### 5.7 Summary


109

### 6 Conclusions and Future Work


111

### Bibliography


114
# List of Figures

1.1 Assumed Physical Network Architecture. AP stands for access point, WLAN stands for wireless local area network. ........................................ 3
1.2 Example of a P2P overlay ................................................................ 5
1.3 Mobile IP Network Architecture .................................................. 7
1.4 Message Diagram of Mobile IP Network Layer Handoff .................. 8
1.5 A P2P Overlay Topology that does not Match the Topology of the Physical Network ................................................................. 11
1.6 A P2P Overlay Topology that Matches the Topology of the Physical Network ................................................................. 11
1.7 Change of P2P Overlay due to Change of Physical Network Topology . . . 13

3.1 Message Diagram of a Standard Mobile IP Handoff ...................... 33
3.2 System Architecture of Explicit Proactive Handoff with Motion Prediction . 36
3.3 Message Diagram of a Proactive Handoff with Motion Prediction ..... 37
3.4 A Sample State Transition Diagram ............................................. 40
3.5 CDF Plot of Average Handoff Latency ......................................... 43
3.6 CDF Plot of Prediction Miss Rate ............................................... 45
3.7 CDF Plot of Fraction of Duplicated Packets ................................ 46

4.1 Diagram of Ping-Pong Messages ................................................... 56
4.2 Three Cases of “Pong” Messages Received ..................................... 60
4.3 CDF of Local Topology Cache Miss Rate ....................................... 66
4.4 CDF of the Fraction of Compulsory Misses in Total Cache Misses ...... 67
4.5 CDF of Mobile Node Average Topology Matching Overhead ............ 68
4.6 CDF of the Normalized Search Scope after Topology Matching ......... 69
4.7 CDF of Normalized Average Query Cost per Reached Node after Topology Matching ........................................................... 71
4.8 CDF of the Normalized Average Neighbor Distance after Topology Matching ............................................................... 72

5.1 CDF of Mobile Nodes’ Number of Movements .................................. 80
5.2 CDF of Mobile Nodes’ Number of Visited Access Points ................. 81
5.3 CDF of Number of Unique Nodes a Mobile Node Co-Located with at Any of the APs during the Whole Trace ........................................ 82
5.4 CDF of Number of Unique Nodes a Mobile Node Co-Located with at a Specific AP during the Whole Trace ............................................. 83
5.5 CDF of the Number of APs a Pair of Mobile Nodes Co-Located at ........ 84
5.6 Number of Co-Locations between a Pair of Co-Located Nodes ........... 85
5.7 Number of Co-Locations between a Pair of Co-Located Nodes at an AP . 86
5.8 CDF of the Miss Rate of Naive Co-Location Prediction Methods with Dartmouth Trace ......................................................... 88
5.9 CDF of the Miss Rate of Naive Co-Location Prediction Methods with UNC Trace ................................................................. 89
5.10 Sample State Transition Diagram of a Mobile Node ........................... 90
5.11 An Example of Co-Location Prediction ........................................ 95
5.12 Miss Rate of Co-Location Predictions for Mobile Nodes with Mid/High Mobility in Dartmouth Trace ........................................... 99
5.13 Miss Rate of Co-Location Predictions for Mobile Nodes with High Mobility in Dartmouth Trace ............................................... 100
5.14 Miss Rate of Co-Location Predictions for Mobile Nodes with Mid/High Mobility in UNC Trace ............................................... 101
5.15 Miss Rate of Co-Location Predictions for Mobile Nodes with High Mobility in UNC Trace ..................................................... 102
5.16 CDF of Search Scope of the P2P Overlay Constructed Based on Co-Location Prediction ............................................................ 106
5.17 CDF of a Mobile Node’s Normalized Search Scope in the P2P Overlays ... 107
5.18 CDF of a Mobile Node’s Normalized Average Query Cost per Reached Node in the P2P Overlays ............................................. 108
5.19 CDF of a Mobile Node’s Normalized Average Neighbor Distance ......... 109
List of Tables

4.1 Local Topology Cache Contents ........................................... 53
4.2 Content of TTL-2 “ping” message sent by MN1 .......................... 57
4.3 Content of TTL-2 “pong” message replied by MN2 ................... 57
4.4 Content of TTL-1 “ping” message relayed by MN2 ...................... 57
4.5 Content of TTL-2 “pong” message replied by MN3 ...................... 58
4.6 Content of TTL-1 “pong” message relayed by MN2 ...................... 58

5.1 Content of Co-Location History Cache Entry .............................. 91
5.2 Content of Movement History Cache ......................................... 92
5.3 Content of Pattern Database Entry ........................................... 92
5.4 Movement History Cache, Co-Location History Cache and Pattern Database before Movement ......................................................... 95
5.5 Movement History Cache, Co-Location History Cache and Pattern Database after Movement ......................................................... 96
5.6 Choice of Pattern Matching Parameters ...................................... 98
Chapter 1

Introduction

In the last 20 years, the use of wireless devices has been rapidly expanding, and it continues to grow quickly with the full-scale deployment of wireless networks all over the world. Wireless devices not only eliminate the need for expensive and inconvenient wiring, but also enable users to communicate electronically while moving around. Mobility, while being very popular, raises a variety of technical issues in the network. In this dissertation, we examine some challenges to a wireless network brought by node mobility, investigate the characteristics of mobile nodes’ behavior, and propose some techniques to utilize these characteristics and address the challenges of mobility.

1.1 Network Scenario

1.1.1 Wireless Networks

Wireless networks have been widely deployed around the world and become an essential part of people’s life. They provide wireless voice and data services, so that a user’s network connection is not restrained by a single point of attachment. The most popular categories of wireless networks today are cellular networks and wireless local area networks (WLANs).

Cellular networks evolved from telephone networks originally designed to provide
voice service [1], therefore they have the characteristics of telephone networks. A cellular network usually consists of multiple fixed radio transmitters called *base stations*, connected by cellular telephone exchange switches. The coverage areas of the base stations are called cells, and they overlap with each other. Typically the radius of a cell in a cellular network ranges from 0.5 mile to a few miles.

Wireless local area networks were originally designed as the wireless extension of local area networks [1], so they inherit the characteristics of a local area network. A wireless local area network consists of multiple wireless *access points* (APs) which have radio transmitters to communicate with other wireless devices. An access point also has one or more ports connected to a wired local area network. The coverage radius of an access point is typically in the order of tens of meters, much smaller than that of a base station in a cellular network.

Currently it is believed that cellular networks and wireless local area networks will co-exist as complementary technologies for global wireless coverage, in which cellular networks provide wide-area coverage targeting at highly-mobile users, while wireless local area networks provide local-area broad-band coverage for users with less mobility.

The physical network architecture in our research consists of multiple wireless LANs connected by a wired IP backbone shown as in Figure 1.1.1, and wireless links are only between the access points and the *mobile nodes* (MNs). Each wireless LAN is connected to the Internet through a *gateway*, which is a router that controls the LAN’s access to the Internet. Two mobile nodes can communicate only via the access points, not through their own wireless interfaces directly. The coverage areas of wireless LANs can overlap with each other. This is the typical scenario of campus-wide or enterprise-wide wireless networks. Each wireless LAN is a subnet. There may be only one wireless access point in a single subnet. It is also possible that there are multiple wireless access points in a single subnet.

This network can be abstracted with a layered view [1]. From top to bottom, the layers are as follows.

- **Application layer** is used by user applications for network communication. Data from user applications is passed to this layer and encapsulated by transport layer protocols.

- **Transport layer** is responsible for end-to-end message transmission. It has error control, flow control and fragmentation control. The Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are transport layer protocols.
Network layer is in charge of transferring data from a source to a destination. Internet Protocol (IP) is one of the network layer protocols, and routers or gateways operate at network layer.

Data link layer is responsible for transferring data between entities attached to the same physical link. It detects and possibly corrects errors occurred at the physical link. Wireless LANs and access points operate at data link layer and below.

Physical layer defines the electrical and physical specifications of network devices.
1.1.2 Peer-to-Peer Overlay

A physical network is the network formed by actual devices (such as routers, access points and hosts) that communicate with each other, and the physical links (wired or wireless) by which the devices communicate. The topology of the physical network is the mapping of actual devices and physical links between devices, for example, the location of the devices, and the wiring system.

Hosts in a network can establish virtual connections, such as TCP connections, with each other to exchange information under the control of a specific application. Each of these virtual connections has a lifetime and can be established or removed by the application. At any time instant, the hosts using the application, and the set of such virtual connections between these hosts, can be viewed as an overlay, or an application-layer network. A peer-to-peer (P2P) overlay is an application-layer network built on top of the physical network to provide end-to-end service for its users, and has been widely used for file-sharing and personal communication. In a P2P overlay, there is no dedicated server or client, each peer node acts as server and client simultaneously. This is fundamentally different from the client-server model that has been widely used in computer networks. The topology of a P2P overlay is the mapping of peers and virtual connections between peers. Figure 1.2 shows an simple example of P2P overlay on top of the physical network. Peer1 and Peer2 are in Subnet1, and Peer3 and Peer4 are in Subnet2. Subnet1 and Subnet2 are individual LANs and connected to the Internet via Gateway1 and Gateway2 respectively. This is the topology of the physical network. However, on the application layer, there are virtual connections (shown as red lines) between Peer1 and Peer3, Peer3 and Peer4, and Peer2 and Peer4. These peers and virtual connections form a P2P overlay, whose topology is different from that of the physical network.

Neighbors in the P2P overlay are defined as the peers connected by a single virtual connection set up by the P2P application, and peers communicate with each other using the routing protocols of this specific P2P application. The actual communication is handled by the underlying physical network. However, from the P2P application's view, a peer node's communication with another peer has to go through one or more of its neighbors in the P2P overlay, then through the neighbors' neighbors, determined by the P2P application's routing protocol, until it reaches its destination.

A P2P overlay can be classified as unstructured or structured. In an unstructured
P2P overlay, there is no control over the overlay topology. A peer node chooses arbitrary peers as neighbors, and a query is usually flooded from a peer node to all its neighbors. Structured P2P overlay has a global mechanism to control the overlay topology and object placement, so that any peer can reach an object via an efficient route. Generally unstructured P2P overlay is simpler, easier to implement, more robust and has better support for keyword search. Therefore it is of our special interest through the research.

1.2 Challenges of Node Mobility

Wireless network and node mobility bring great challenges to network management. There has to be a handoff mechanism to support a node’s change of attachment point in the network due to its mobility. Peer-to-peer overlay topology optimization is also more difficult with mobile nodes as the topology of the underlying physical network changes frequently due to node mobility.

1.2.1 Handoffs

A mobile node moves in the wireless network, and possibly connects to different access points in different subnets. When a mobile node changes its attachment point in the network, there has to be a handoff mechanism to ensure that the mobile node is able to continue its communication. From the layered view of the network, handoffs can be categorized as link-layer handoff and network-layer handoff.
Link-Layer Handoff

Link-layer handoff is the process to switch a mobile node from the radio link of one access point to that of another access point. As access points are devices only on the physical layer and data link layer, the handoff between access points is a link-layer handoff. Usually each access point periodically broadcast beacon signals containing its ID and radio link information. A mobile node receives the beacon signals and uses them to get radio link information, signal strength and channel condition of access points nearby. If the signal strength of the serving access point falls below a threshold, the mobile node disassociates with it and actively or passively scans access points nearby on assigned channels. After that it chooses a handoff target access point to associate with using the ID and radio link information of that access point. Usually the latency of a link-layer handoff is in the order of tens or hundreds of milliseconds.

Network-Layer Handoff

When a mobile node moves from a subnet to another one, network-layer handoff is used to re-route the IP packets destined to the mobile node to its new residing subnet. The original Internet Protocol (IP) uses a node’s IP address to determine its point of attachment to the network. The node must reside in the subnet specified by the subnet prefix in its IP address to receive IP packets destined to it, otherwise the packets are undeliverable. To support a mobile node in an IP network, the node’s IP address must be changed, or its routing information has to be updated throughout the network, whenever it moves to another subnet. Neither of these are acceptable. TCP connections are identified by IP addresses and port numbers of the source and the destination. When a mobile node changes its IP address, any of its TCP connections would be broken. To update routing information throughout the whole network frequently for every mobile node is also extremely expensive.

Mobile IP version 4 (MIPv4) [2] was designed as a simple and scalable solution for node mobility. It enables a mobile node to change its attachment point in the network without changing IP address. It is intended to handle mobility issue at the network layer, without changing lower or higher layers. Its architecture is shown in Figure 1.3. In Mobile IP, each mobile node has two IP address. A home address is used to identify the mobile node, and the mobile node’s home network, which the mobile node is permanently registered with, has a subnet prefix matching that of the home address. A care-of address is a temporary
IP address to identify the actual attachment point of a mobile node, when it is visiting a foreign network other than its home network. Mobile IP adds *home agents* (HAs) and *foreign agents* (FAs) into the network architecture to support node mobility and handoff. These are routers in the mobile node’s home network and foreign networks, respectively.

![Figure 1.3: Mobile IP Network Architecture](image)

When a mobile node moves from a foreign network to another foreign network, it need to perform a location registration to handoff to the new one. The message diagram is shown in Figure 1.4. The handoff process is as follows.

1. The mobile node gets a new care-of address from the new foreign agent.
2. The mobile node sends a location registration request with the new care-of address to its home agent, possibly with the assistance of the new foreign agent.
3. The home agent authenticates the mobile node, and creates or modifies the mobility
binding between the mobile node’s home address and the new care-of address.

4. The home agent sends a location registration reply with the lifetime of the mobility binding to the new foreign agent.

5. The new foreign agent forwards the location registration reply to the mobile node.

6. Then the mobile nodes can send packets to its corresponding node (CN) using standard IP mechanism. A corresponding node is a node that communicates with the mobile node. It can be a static node or a mobile node.

7. The home agent intercepts any packet sent from the corresponding node to the mobile node’s home address. It finds out the mobile node’s current care-of address according to its mobility binding. The home agent then routes the packet to the mobile node’s current care-of address using \textit{IP-within-IP encapsulation}, in which a new packet is constructed with the mobile node’s current care-of address as the destination address and the original packet as the payload.

8. The new foreign agent decapsulates the packet delivered to the mobile node’s care-of address and forwards the original packet to the mobile node.

![Figure 1.4: Message Diagram of Mobile IP Network Layer Handoff](image)

Mobile IP also got some improvements. Mobile IP version 6 (MIPv6) [3] removes the need for Foreign Agents, and mobile nodes use IP version 6 (IPv6) [4] features to operate without special support from Foreign Agents. Route optimization [5] updates the
corresponding node with the mobile node’s care-of address, so that packets can be delivered to the mobile node’s care-of address without the help from Home Agent.

Mobile IP, however, still has its own problems. The typical network-layer handoff latency is in the order of 0.1 to 1 second, and is too large for interactive voice or video which need less than 100ms latency and jitter. Packets sent to the mobile node during a network-layer handoff are delivered to the old care-of address of the mobile node, until the mobile node completes the network-layer handoff to register the new care-of address at the Home Agent. These packets are not deliverable, since the latency of link-layer handoff is much smaller and the mobile node has already switched to the radio link of the access point at the new subnet. Since Mobile IP does not buffer packets, these packets are lost and have to be retransmitted by the transport layer protocols such as TCP. This causes disruption of packet delivery during handoff, and wastes network bandwidth as the same packets have to be retransmitted. We need a network-layer handoff mechanism that can significantly reduce the latency and packet loss during a network-layer handoff. Its communication, computation and storage overhead should also be affordable to mobile devices.

### 1.2.2 Topology Matching for Unstructured P2P Overlay

Currently an unstructured P2P overlay is constructed without considering the topology of the underlying physical network. Usually a peer randomly selects its neighbors to form the P2P overlay, and queries are *flooded* to all its neighbors. The most efficient flooding method is that each peer forwards a query to all its neighbors except the one the query came from, if the query has a Time-To-Live (TTL) larger than 0. Throughout this dissertation, TTL refers to the logical hop count in the P2P overlay. And each peer only floods the same query once, which means the peer only sends a query with TTL larger than 0 to all its neighbor in the P2P overlay once. The next time the peer receive the same query, it will simply drop the query even if the query’s TTL is larger than 0. Through this dissertation, this method is used as the method to flood queries.

In such an unstructured P2P overlay, there might be mis-match between the topology of the P2P overlay and that of the physical network, which potentially causes unnecessary traffic and latency. Figure 1.5 shows a sample of a P2P overlay topology that does not match the topology of the physical network. Peer1 and Peer2 are in Subnet1 which is connected to the Internet through Gateway1, and Peer3 and Peer4 are in Subnet2 which
is connected to the Internet through Gateway2. A red connection between 2 peers means they are neighbors on the P2P overlay and have virtual connections set up by the P2P application between them. Here we define the distance between two peers as the cost of physical links between them. This cost can be network delay, or bandwidth, and etc., we used the network delay as the cost in this dissertation. With P2P neighbors randomly selected, it is possible that a peer selects a peer far away instead of one nearby as its neighbor, therefore making the connections between P2P neighbors more expensive than needed. In Figure 1.5, Peer1 and Peer2 are not neighbors on the P2P overlay, while Peer2 and Peer3 are neighbors. Since Peer2 and Peer3 are in different subnets and the physical link between them has to go through the Internet, the distance between Peer2 and Peer3 is much larger than that between Peer1 and Peer2, whom are connected with an intra-subnet physical link. Furthermore, a message of the P2P application may have to pass the same physical link multiple times, which wastes the bandwidth of the physical network. If Peer1 wants to send a P2P application message to Peer2, it has to send the message to Peer3 first and Peer3 forwards the message to Peer2. In this case, the message travels through the physical link between Gateway1 and Gateway2 twice. With the query flooded to all neighbors, the same query may reach the same destination peer via different path on the P2P overlay. For example, Peer2 floods a query with TTL=2 to all its neighbors Peer3 and Peer4 on the P2P overlay, and Peer3 or Peer4 forwards the query to all its own neighbors until the TTL is 0. Therefore the same query reaches Peer4 twice, first via the overlay path from Peer2 to Peer4, and then via the overlay path from Peer2 to Peer3 to Peer4. This causes unnecessary traffic. In Figure 1.5, a query with TTL=3 from Peer2 can reach 3 peers, and passes through 5 overlay hops. Therefore the query need to pass through 1.67 overlay hops in average to reach one unique node. Suppose the delay of the physical network between Gateway1 and Gateway2 is 10, and that between two peers or a peer and a gateway in the same subnet is 1. The average distance (delay) between Peer2 and one of its neighbors in this P2P overlay is 12.

For comparison, Figure 1.6 shows a P2P overlay topology that matches the topology of the physical network. The topology of the physical network is the same as in Figure 1.5, however, this time Peer1 and Peer2, Peer1 and Peer3, Peer3 and Peer4 are neighbors in the P2P overlay. Similarly, a query with TTL=3 from Peer2 can reach 3 peers, and passes through 3 overlay hops. Therefore the query need to pass through 1 overlay hop in average to reach one unique node, less than that in the P2P overlay topology that does not match
the topology of the physical network. If the physical network delays are the same as in Figure 1.5, the average distance (delay) between Peer2 and its neighbor in this P2P overlay is 1. Again, it is much smaller than that in the P2P overlay that does not match the topology of the physical network. Therefore, for Peer2, the P2P overlay topology in Figure 1.6 matches the physical network topology better, compared to the P2P overlay topology in Figure 1.5, as the query with same TTL can reach the same number of peers, but the average number of overlay hops the query need to pass through to reach a unique peer and the average neighbor distance are smaller. If a P2P overlay whose topology matches the physical network topology better compared to another P2P overlay, we call the first P2P overlay more topologically matched P2P overlay, and its topology a more efficient P2P overlay topology.
The discussion about topology matching above only refers to an individual peer that initiates the query. The whole P2P overlay is a distributed system with no central control, which means each peer optimizes its neighbors according to the information available locally. When each peer in the P2P overlay optimizes its neighbors, the P2P overlay constructed by this might not the global best one that matches the physical network topology. However, globally it is likely to match the physical network topology better, compared to the P2P overlay constructed by random neighbor selection. The degree that a P2P overlay globally matches the physical network topology, or the global efficiency of the P2P overlay topology, is evaluated by three metrics. The first one is the number of unique peers each peer’s query can reach with a certain TTL, which is called the search scope. The second one is the number of overlay hops this query has to pass through in average to reach a unique peer in the search scope, which is called the average query cost. The third one is the average neighbor distance of each peer. Given two P2P overlay A and B based on the same physical network topology, if in A most peers have bigger search scope, smaller average query cost and smaller average neighbor distance, compared to those in B, globally A is more topologically matched to the physical network than B, and the topology of A is more efficient than that of B.

There have been some topology matching schemes proposed to optimize the topology of unstructured P2P overlay so that it matches the topology of the underlying physical network. However, they usually need to send out probing messages periodically to find out the topology of the physical network and optimize the P2P overlay topology accordingly. Node mobility introduces great dynamics into the network and makes the physical network topology change frequently, as a mobile node can be residing in different subnets at different time. When the topology of the physical network changes, a P2P overlay that previously matches the topology of physical network may not match it any more. Figure 1.7 shows an example. Originally the P2P overlay is the same as in Figure 1.6, then Peer2 moves to Subnet2, and the same P2P overlay does not match the new topology of the physical network. A query with TTL=3 from Peer2 can still reach 3 peers, and in average it has to pass through 1 overlay hop to reached a unique peer. However, the average distance between Peer2 and its neighbor with the new physical network topology is increased to 12. In this case the the P2P overlay topology has to be optimized for Peer2 again, and Peer3 instead of Peer1 becomes Peer2’s neighbor. In the new P2P overlay, the average distance between Peer2 and its neighbor is reduced to 1 again. To deal with this great network
dynamics due to node mobility, the existing topology matching schemes have to probe the physical network more frequently to keep up with the rapid change of physical network topology. This causes huge overhead in network probing, and wastes the precious bandwidth of the wireless network. Therefore a P2P overlay topology optimization mechanism with less network probing overhead is needed for P2P overlay of mobile nodes, and this mechanism also should be able to construct a P2P overlay that matches the topology of the physical network, as least in the same degree as existing topology matching schemes with periodic probing do.

Figure 1.7: Change of P2P Overlay due to Change of Physical Network Topology
1.3 Movement Patterns and Applications

In real-life wireless networks, mobile nodes’ logical movement on subnet level are not purely random, and each mobile node has its own movement patterns. Usually a mobile node is carried by an individual person. As a result, it will be moved according to hourly, daily or weekly patterns corresponding to that person’s regular activities. Significant changes to these patterns are unlikely to be frequent.

Several studies on the activities of mobile nodes in wireless networks confirm this observation. In their study on user activities in a WLAN at Computer Science Department of Stanford University [6], Tang and Baker found that user population can be divided into location-based subcommunities with unique behavior on movement, period of activity and traffic generated. Later they traced user activities in a Ricochet metropolitan-area wireless network for seven weeks [7], and found that a user can be categorized into clusters according to parameters such as the total number of locations it travels and the average number of location it visits within an active day. The more locations a user visits, the closer these locations are. The distance that users move is a Gaussian distribution around the radius of the network. Kotz and Essien traced the activities of mobile users in an IEEE 802.11b wireless campus-wide network at Dartmouth College [8]. Clear weekly and daily patterns of activity and traffic are found although the trace varied widely from hour to hour, day to day, and week to week. Most mobile users in this trace only visit a few locations of their daily routine. Measurement of the wireless campus network at UNC Chapel Hill [9] also shows that each individual mobile node has patterns in its association activities with access points, and its next associated access point can be predicted based on its association history.

Furthermore, the interactions of mobile nodes also have patterns. The structure of social network determines that people tend to dynamically form small communities at specific time and location areas, and the interactions within these communities are much more intense compared with inter-community interactions. The existence of such communities follows regular activities of groups of people, therefore it should also be embedded with intrinsic patterns corresponding to regular group activities. Abrupt change of such patterns is unlikely to be frequent. We define co-location as the scenario that multiple mobile nodes are within a certain distance from each other. For example, two mobile nodes are associated with the same access point, and they can exchange information quickly and
efficiently. Mobile nodes carried by individuals should also have patterns in their co-location and interaction behavior.

Mobile nodes’ movement and co-location patterns could provide crucial information about the behavior of the mobile nodes in the wireless network, and there are a lot of applications that can take advantage of these movement patterns. Based on its movement history and the movement patterns extracted from the history, a mobile node is able to predict its future behavior, such as the next subnet it will move to and the nodes it will co-locate and interact with. The prediction result can be utilized to expedite processes of network management, while reducing the overhead and latency required for information exchange.

For example, a mobile node can predict its next residing subnet and initiate the network-layer handoff process before the actual movement, in order to reduce the latency and packet loss during handoff. If the prediction is correct and the network-layer handoff is initiated early enough, the only handoff latency when a mobile node moves from one subnet to another is the latency of link-layer handoff to switch the mobile node from the radio link of the old access point to that of the new access point. If the prediction is not correct, the handoff preparation is wasted, and the mobile node has to use standard Mobile IP mechanism to perform the handoff. As the prediction of next residing subnet based on movement pattern is not perfect, there is always possibility that the prediction is not correct. We try to minimize the average network-layer handoff latency and packet loss for each mobile node, while keeping the overhead of prediction and handoff preparation reasonable for mobile devices and wireless networks.

A mobile node can also predict its neighbors on a P2P overlay whose topology matches the physical network topology, based on the history information of its neighbors on such a P2P overlay. If the prediction is correct, it can select neighbors to construct a P2P overlay that matches the physical network topology, without probing the network. Therefore the overhead for probing and discovering the physical network topology for topology matching can be reduced. If it is not correct, a topology matching has to be performed to construct a topologically-matched P2P overlay, which need extra overhead for network probing and topology discovery. Since such prediction based on history information is not perfect, it is possible that the prediction is not correct and topology matching is performed. Our goal is to minimize topology matching overhead, which is the average number of overlay hops the probing messages generated by a mobile node have to pass through per time inter-
val. Meanwhile the P2P overlay should be kept matched to the physical network topology, and the P2P overlay topology should be efficient.

A mobile node can even predict the nodes that it will co-locate with at a particular location based on its co-location history with other nodes and its own movement trace. With such prediction, the mobile node is able to optimize the P2P overlay topology without probing the network at all, therefore expediting the network management process and eliminating the overhead for probing and discovering the physical network topology. If the prediction is correct, the P2P overlay topology matches the physical network topology and is efficient, otherwise the P2P overlay topology does not match, resulting in degradation of P2P overlay efficiency. We are targeting at eliminating the need for network probing and topology discovery, while making the P2P overlay as efficient as those constructed by existing topology matching techniques.

In this dissertation, we will show if there is potential for such predictions, describe how to make these predictions, and find the way to utilize the prediction result in network management.

1.4 Contributions of Our Research

This dissertation can be divided into three big parts: explicit proactive handoff with motion prediction, topology matching with Local Topology Cache for peer-to-peer overlay of mobile nodes, and measurement and prediction of mobile nodes’ co-location behavior.

First we proposed a proactive handoff mechanism utilizing prediction of next residing subnet based on movement history. A mobile node prepares network-layer handoff in advance of the actual link-layer handoff. The mobile node then explicitly notifies current foreign agent to forward packets to the predicted subnets before the actual link-layer handoff, therefore reducing the handoff latency and packet loss. A fully distributed, dynamic motion prediction algorithm was used to predict the next residing subnet based on a mobile node’s movement history, and the handoff latency, packet loss, and extra overhead of the proposed scheme were compared to Mobile IP with network-layer handoff initiated right after link-layer handoff. This handoff scheme’s performance is judged by these metrics.

• Improvement of handoff performance

  – Handoff latency: the time interval during a network-layer handoff that a mobile
node cannot receive packets. The smaller the handoff latency, the better this handoff scheme performs.

- **Packet loss**: the average number of packets destined to a mobile node that are lost during a handoff. The less the packet loss, the better this handoff scheme performs.

- **Extra handoff overhead**: the fraction of duplicated packets in all packets sent to a mobile node. The less the extra handoff overhead, the better this handoff scheme performs.

Simulation based on real-life wireless network trace shows that it greatly improves handoff performance compared to Mobile IP, but only introduces slight extra overhead.

Then we focused on the topology matching problem of a P2P overlay of mobile nodes. We designed an on-demand topology matching scheme with Local Topology Cache to reduce the overhead for network probing while making the P2P overlay topology match the physical network topology. The neighbor information of a topologically-matched P2P overlay is cached by a mobile node in the hope that the topologically matched P2P overlay will be the same when it returns to the same location, and these neighbors can be reused without network probing. The effectiveness of this approach is judged by these metrics.

- **Overhead of topology matching**: the average number of P2P overlay hops the probing messages from a mobile node have to pass through per time interval for topology matching. It indicates whether this approach can reduce the overhead of topology matching. The less overhead, the more effective this P2P overlay construction approach is.

- **Efficiency of the P2P overlay topology**
  - **Search scope**: the number of unique peers a mobile node can reach in the P2P overlay by flooding a query with certain TTL. The bigger the search scope, the more effective this approach is.
  - **Average query cost per reached node**: the average number of overlay hops the query message with the same TTL from a mobile node has to pass through to reach a unique peer in the search scope. The less the average query cost per reached node, the more effective this approach is.
– **Average neighbor distance**: the average distance (delay) between a mobile node and its neighbors on the P2P overlay. The less the average neighbor distance, the more effective this approach is.

We used simulation with real-life wireless network trace to demonstrate that the proposed topology matching with caching scheme greatly reduces the overhead for network probing and topology discovery, but still constructs a topologically matched and efficient P2P overlay.

Following the insights we got from research on topology matching with caching, we further investigated the co-location behavior of mobile nodes. The co-location behavior of mobile nodes in two real-life wireless network traces was measured, such as the number of nodes with which a mobile node co-located, the number of locations a particular co-location occurred at, and number of times the same co-location repeated. It shows the repetitiveness and patterns of co-locations, and co-location prediction has great potential. With such observations, we designed a fully-distributed, dynamic method to predict the nodes that a mobile node will be co-locating with in the future. It was only based on the mobile node’s own movement history and its co-location history with other nodes. We found that this prediction method has good accuracy and reasonable computation and storage cost using simulations based on the same traces. The result of co-location prediction was then used to construct P2P overlay of mobile nodes. It totally eliminated the need to probe the physical network for P2P overlay optimization, therefore greatly reducing the corresponding overhead. Similar as the discussion above, the effectiveness of this approach to construct a P2P overlay based on co-location prediction is evaluated by these metrics.

- **Overhead of topology matching**: this approach has 0 overhead of topology matching as it does not probe the physical network at all. This is the minimum overhead of topology matching possible.

- **Efficiency of the P2P overlay topology**
  - **Search scope**: the number of unique peers a mobile node can reach in the P2P overlay by flooding a query with certain TTL. The bigger the search scope, the more effective this approach is.
  - **Average query cost per reached node**: the average number of overlay hops the query message with the same TTL from a mobile node has to pass through
to reach a unique peer in the search scope. The less the average query cost per reached node, the more effective this approach is.

- **Average neighbor distance**: the average distance (delay) between a mobile node and its neighbors on the P2P overlay. The less the average neighbor distance, the more effective this approach is.

Through simulation with a real-life wireless network trace, we found the P2P overlay constructed topologically matches the physical network, and the need for probing the network to do topology matching is totally eliminated.

### 1.5 Outline of the Dissertation

The rest of this dissertation is organized as follows. Chapter 2 discusses the previous work that is related to our research. In Chapter 3 we present an explicit proactive handoff mechanism with Motion Prediction. Each mobile node predicts the next subnet it will move to, and notifies its foreign agent to duplicate and forward packets to the subnet. Chapter 4 introduces a topology matching mechanism with Local Topology Cache for P2P overlay of mobile nodes. In order to expedite topology matching process and reduce associated overhead, each mobile node caches the information of its neighbors on a topologically-matched P2P overlay when visiting a subnet, and uses such information to find neighbors the next time it returns to the same subnet. In Chapter 5 we investigate the characteristics of mobile nodes’ co-location behavior using real-life wireless network activity traces, and discuss the potential to do co-location prediction. A fully distributed co-location prediction method based on a mobile node’s own movement trace and its co-location history with other nodes is proposed, and its performance is evaluated with real-life wireless network traces. Co-location prediction result is initially used to construct an efficient P2P overlay of mobile nodes without probing the network. Chapter 6 is the conclusion and future work.
Chapter 2

Related Work

This dissertation focuses on 3 different problems, enhancement for Mobile IP handoff, topology matching for P2P overlay with mobile nodes, and measurement and prediction of co-location. In this chapter, we first present the related work on modeling of mobile nodes' movements in handoff in Section 2.1. Section 2.2 is the related work on handoff. Section 2.3 discuss the related work on P2P overlay topology optimization.

2.1 Related Work on Movement Modeling

To study wireless networks and design network protocols, it is crucial to have an accurate mobility model of mobile nodes for simulation and evaluation purpose. Mobility models describe the movement patterns of mobile nodes, including how their location, direction, velocity and acceleration change over the time. As mobile nodes’ mobility patterns play a key role in design and evaluation of wireless networks, an unrealistic or inaccurate mobility model may lead to wrong conclusions. It is highly desirable that the mobility model can extract the underlying mechanisms and characteristics that determine mobile nodes’ movements, and depict the actual movements of mobile nodes in real-life wireless networks.

Generally mobility models can be categorized as synthetic models and models extracted from actual movements. Here we present some example of each category.
2.1.1 Synthetic Mobility Models

Synthetic mobility models can be further categorized as pure random models, models with temporal locality, models with inter-node locality and models with geographic constrains.

Pure Random Mobility Models

In pure random mobility models, the direction and velocity of a mobile node’s movement is independently and randomly selected. Random Walk Model, Random Waypoint Model and Random Direction Model are three typical pure random models.

Random Walk Mobility Model is also referred as Brownian Motion. In this model, a mobile node moves from its current location to another location by randomly select its direction from $[0, 2\pi]$ and its velocity from $[v_{min}, v_{max}]$. Each movement is bounded by a constant time interval or a constant distance, and after the movement another \{direction, velocity\} pair is generated for the next movement.

In Random Waypoint Mobility Model [43], a mobile node first randomly selects one location in the simulation area and stays there for a specific amount of time $T_{pause}$, then travels to a random destination with a random velocity uniformly distributed between $[v_{min}, v_{max}]$. After arriving at the destination, it pauses there for a specific amount of time $T_{pause}$ and starts the above process again. Random Waypoint model is simple and widely adopted in the evaluation of MANET, but it can cause non-uniform node distribution and density wave in the simulation area.

Random Direction Mobility Model [44] was proposed to solve the problems of Random Waypoint Model mentioned above. A mobile node first randomly selects a direction, and then randomly chooses a destination along this direction. After arrival at the destination, it pauses there for $T_{pause}$ and repeats the process.

Pure random mobility models are memory-less that the current direction and velocity of a mobile node are independent from its previous direction and velocity, therefore sharp turns and sudden stops/accelerations occur frequently and makes the movement unrealistic. Since a mobile node’s movement are independent from those of other mobile nodes, pure random mobility models fail to capture the interaction characteristics of mobile nodes. In pure random mobility models, a mobile node can move freely in the simulation area without any geographic constraint, which is not true in a real-life scenario. Therefore
mobility models with temporal locality, inter-node locality and geographic constraints were proposed to overcome these problems.

**Mobility Models with Temporal Locality**

In mobility Models with temporal locality, a mobile node’s current movement is dependent on its previous movement. Therefore the acceleration is incremental and direction change is smooth, and the problems of sharp turn and sudden acceleration/stop are solved.

Gauss-Markov Mobility Model [45] assumes that the current velocity of a mobile node is correlated with its previous velocity, and a Gauss-Markov stochastic process is used to model the correlation. A mobile node updates its velocity at fixed time intervals, and the new velocity vector \( \mathbf{V}_t \) is correlated to the previous velocity vector \( \mathbf{V}_{t-1} \).

\[
\mathbf{V}_t = \alpha \mathbf{V}_{t-1} + (1 - \alpha) \mathbf{V} + \sigma \sqrt{1 - \alpha^2} \mathbf{W}_{t-1}
\]

where \( \mathbf{W}_{t-1} = [w^x_{t-1} \, w^y_{t-1}]^T \) is the uncorrelated random Gaussian process with mean 0 and variance \( \sigma^2 \), and \( \alpha = [\alpha^x \, \alpha^y]^T \), \( \mathbf{V} = [v^x \, v^y]^T \), and \( \mathbf{V} = [\sigma^x \, \sigma^y]^T \) are vectors of memory level, asymptotic mean and asymptotic standard deviation of velocity, respectively. The dependency level of current velocity on previous velocity is determined by memory level parameter \( \alpha \), and \( \alpha^x \in [0, 1], \, \alpha^y \in [0, 1] \). When \( \alpha = [0 \, 0] \), the Gauss-Markov model is memory-less and the new velocity \( \mathbf{V}_t \) is only determined by \( \mathbf{W} \). When \( \alpha = [1 \, 1] \), the Gauss-Markov model has strong memory so that the new velocity \( \mathbf{V}_t \) is the same as previous velocity \( \mathbf{V}_{t-1} \). When \( 0 < \alpha^x < 1 \, \text{and} \, 0 < \alpha^y < 1 \), the Gauss-Markov model has some memory and the new velocity is determined by both the previous velocity and a random variable.

**Mobility Models with Inter-Node Locality**

Mobility models with inter-node locality assume that a mobile node’s movement can be correlated with those of its neighboring mobile nodes, therefore they are able to model the scenario that mobile nodes interact with each other in a group.

Reference Point Group Mobility Model [46] organizes mobile nodes into groups. Each group has a logical group center indicating the movement trend of the mobile nodes in
this group, and each individual mobile node randomly moves around a pre-assigned reference point. The movement of the group center can follow a predefined path or be determined by other mobility models such as Random Way Mobility Model. The movement vector of group member $i$ at time $t$ is $\vec{V}_i^t = \vec{V}_G^t + \vec{R} \vec{M}_i^t$, where $\vec{V}_G^t$ is the movement vector of the group center, and $\vec{R} \vec{M}_i^t$ is the random deviation vector of group member $i$, which is a independently and identically distributed (iid) random process with length uniformly distributed in $[0, r_{max}]$ and direction uniformly distributed in $[0, 2\pi)$, where $r_{max}$ is the maximum allowed deviation.

Sanchez and Manzoni proposed three mobility models in which mobile nodes' movements are cooperated [47]. Column Mobility Model describes the scenario that multiple mobile nodes moves around their reference points along a line or column, which is also moving. This model is usually used in search/scanning activities. Each mobile node has a reference point in the line/column, and the reference point of mobile node $i$ at time $t$ is $RP_i^t = RP_i^{t-1} + \alpha_i^t$, where $\alpha_i^t$ is the advance vector, a predefined offset to move the grid of reference points. This offset is got by randomly selecting a distance and angle between 0 and $\pi$. The new position of mobile node $i$ at time $t$ is $P_i^t = RP_i^t + w_i^t$, where $w_i^t$ is a random vector. This means that the mobile node’s new position randomly deviates from its new reference point. Pursue Mobility Model describes the scenario that multiple nodes are trying to catch a single target, and is usually used in target tracking. The new position of mobile node $i$ at time $t$ is $P_i^t = P_i^{t-1} + v_i^t(P_{target}^t - P_i^{t-1}) + w_i^t$, where $v_i^t$ is the acceleration of the mobile node, $P_{target}^t$ is the expected position of the target node at $t$ and $w_i^t$ is a small random vector. Nomadic Community Mobility Model describes the scenario that multiple mobile nodes move from one location to another as a group, and is useful in military or conference applications. All mobile nodes share a single reference point and roam randomly around the reference point using an entity mobility model, and the new position of mobile node $i$ at time $t$ is $P_i^t = RP_i^t + w_i^t$. Nomadic Community Mobility Model differs from Column Mobility Model as all mobile nodes share a common reference point and movement is sporadic.

Mobility Models with Geographic Constraints

Mobility Models have to represent the characteristic of real-life mobile nodes that their movements are subject to the geographic features in the simulation area. Vehicles or pedestrians have to move in predefined pathways and can be blocked by obstacles, therefore
their movements are pseudo-random.

Pathway Mobility Model [50] restricts mobile nodes’ movements to predefined pathways, and a random graph is used to model the map of simulation area. The vertices of the graph stand for the buildings and edges represent the streets connecting buildings. A mobile node is first randomly placed in a vertex of the graph. It then randomly chooses a destination vertex and moves to the destination along the shortest path. After arrival at the destination, the mobile node pauses for $T_{\text{pause}}$ and repeats the above process.

In Obstacle Motility Model [51], obstacles are placed in the simulation area and a Voronoi graph is computed according to the buildings and obstacles. Pathways connecting buildings are constructed and mobile nodes can only move along these pathways.

2.1.2 Mobility Models Extracted from Actual Movement History

Synthetic mobility models are simple and easy to implement, but it has not been validated that they are accurate to model movements of real-life mobile nodes. Mobility Models extracted from actual movement history can capture characteristics of real-life movements, provide a better insight about the underlying patterns in the movements, and predict future movements of mobile nodes. Many of these models use a macroscopic view and focus on mobile nodes’ movements relative to abstract areas, or locations.

Order-$k$ Markov-family model assumes that the probability that a mobile node moves to a specific location only depends on the context of $k$ most recent locations the mobile node has visited, and the probability distribution is stationary. Random variable $X$ is defined as the mobile node’s location and $X(i, j)$ is string $X_i X_{i+1} \ldots X_j$. If entire simulation area is $A$ and the the mobile node has a movement history of $X(1, n) = a_1 a_2 \ldots a_n$, its next location $X_{n+1}$ has the following properties for all $i \in [1, n]$.

$$P[X_{n+1} = a | X(1, n) = a_1 a_2 \ldots a_n]$$
$$= P[X_{n+1} = a | X(n - k + 1, n) = a_{n-k+1} \ldots a_n]$$
$$= P[X_{i+k+1} = a | X(i + 1, i + k) = a_{n-k+1} \ldots a_n]$$

And the transition probability matrix $M$ of length-$k$ location strings in this Markov chain
model can be estimated by the number of strings \( N(a_m \ldots a_n) \) observed in history.

\[
P[X_{n+1} = a|X(n - k + 1, n) = a_{n-k+1} \ldots a_n] \\
= M[X(n - k + 1, n), X(n - k + 2, n + 1)] \\
\approx \frac{N(a_{n-k+2} \ldots a_n a)}{\sum_{a_j \in A} N(a_{n-k+2} \ldots a_n a_j)}
\]

Mobile Motion Prediction (MMP) algorithms [20] extend the Markov-family models by including timing information in the Markov chain. In this model, a mobile node enters a new state when it moves to a location. If it stays there for more than a specified period of time, this state is a stationary state. Otherwise it is a transitional state. A movement track models the movement of a mobile node on a regular route, starting from and ending at different stationary states. A movement circle models the mobile node’s long-term regular behavior, and starts from and ends at the same stationary state. MMP attempts to correlate current movement with movement circles and movement tracks stored in the pattern database using state matching, velocity matching and frequency matching techniques.

Compression-family models [48] are based on the incremental parsing algorithm proposed by Ziv and Lampel [49]. Such algorithm is popular in text compression applications. The LZ parsing algorithm partitions strings into distinct substrings, and constructs a LZ tree with each nodes represents a substring associated with Statistics. Similar as in Markov-family models, the transition probability matrix can be estimated by the number of substrings observed in the entire history \( L \).

\[
P[X_{n+1} = a|L] = \frac{N_{LZ}(s_m a, L)}{N_{LZ}(s_m, L)}
\]

where \( N^{LZ}(s', s) \) is the number of times that \( s' \) occurs as the prefix of substring of substrings parsed from the LeZi algorithm. LeZi based models differ from Markov-family models as the strings is not required to be of a fixed length.

Song et al. [39] evaluated the accuracy of these history-based models with real-life network traces, and found simple order-2 Markov model has good accuracy compared with more complex ones. For mobile nodes with long trace length, it can achieve about 72%
accuracy of prediction.

A mobility model using real-life mobility characteristics was proposed in [52]. A spatial process and a temporal process are used in collaboration to generate node trajectories according to the parameters extracted from Dartmouth College’s wireless network activity trace.

[53] described a method to estimate a mobile node’s geographic movement from its trace of associations with access points. The speed and pause time of a mobile node are found to follow a log-normal distribution in Dartmouth College’s wireless network activity trace. Based on their observations from the real network activity trace, the authors proposed a synthetic mobility model, which categorizes nodes to mobile or stationary, and characterize mobile nodes’ movements according to the region transition matrix and the waypoint matrix.

Hsu et al. [55] studied WLAN traces and investigated the encounters between users. They found that the total number of encounters of each mobile node follows BiPareto distribution. The Small World model was used to understand the characteristics of mobile nodes’ encounters. Via encounters WLAN users form connected Small World graphs, in which nodes within small clusters are well connected and nodes in different clusters are sparsely connected. They proposed an information diffusion scheme to spread information among users through their encounters, and showed that the information can reach most of the node population even if a high percentage of users do not participate in the information diffusion.

Students’ mobility and contact patterns were inferred from class schedules and rosters in [54]. The authors found that students interact with different sets of students at different time. Their contact graph is well connected and the distance between 2 nodes on the contact graph is small, which suggests a small-world model. The characteristics of students contact patterns were studied from the perspective of Delay Tolerant Network, virus spread and information aggregation.

The inter-contact time of mobile devices was investigated in [56], and actual network activity traces show that its distribution has a heavy tail as in power law. This is different from the exponential decay as suggested in many mobility models such as random way-point model or the model assuming the locations of mobile nodes are i.i.d. in a bounded region.

A time-variant community mobility model was proposed in [57] to capture mobile nodes’ preference of visiting location and periodic re-appearance at the same location, which
were observed in wireless network activity traces. The authors modeled the average time before a node moves to a random location and the average time before two nodes move to the same location to study the routing performance in delay-tolerant networks or mobility-assisted packet forwarding.

[58] presented a realistic mobility model by combining WiFi association trace and map of the measured space. The users’ movements are modeled as a second-order Markov chain, and the transition probability matrix of locations is generated from the trace. This model was evaluated by comparing the transition probabilities at intersections in the model to actual observations from the trace, and the model has high correlation and low error compared to the actual trace.

2.2 Related Work on Handoff

A number of schemes have been proposed to solve the problems of Mobile IP during handoff. Here we focus on dedicated fast handoff schemes, which can be categorized as reactive, proactive, or a combination of the two, according to whether the packets sent to the MN start to arrive at the new subnet after or before the link-layer handoff, respectively.

2.2.1 Reactive Handoff

In simultaneous 802.11 and MIPv4 handoff [10], when an MN associates with a new AP, it sends MIPv4 registration information to the new FA. This information is contained within 802.11 frames, and initiates a MIPv4 location registration process. This scheme reduces handoff latency, but does not solve the problems caused by the latency gap between link-layer handoff and network-layer handoff.

[11] utilizes a filtering database and a MAC bridge connecting WLANs. When an MN associates with a new WLAN, its MAC address is broadcast locally by the new AP. It is received by the MAC bridge and stored in the filtering database, along with the corresponding port. Before the MN completes network-layer handoff, the MAC bridge relays MAC frames for the MN from the old WLAN to the new WLAN. Scalability and reliability may be problems for this scheme, due to its dependence on a centralized bridge mechanism.
2.2.2 Proactive Handoff

The Daedalus project [12] uses IP multicast and buffering to reduce packet loss. Each base station and its neighboring base stations form a multicast group. When an MN connects to a base station, it registers a corresponding multicast address at the home agent (HA). Using this address, packets are multicast to and buffered at the base stations of this multicast group. If an MN switches to a neighboring base station, it can receive packets before performing a network-layer handoff to register a new multicast address.

E. Shim et al. used neighborcasting [13] to achieve low latency handoff. An MN can transfer its old FA information to the new FA, which then constructs a neighbor table. Before link-layer handoff, the MN notifies the old FA to forward duplicated packets to all neighboring FAs. Network-layer handoff latency can be reduced significantly in this scheme.

R. Hsieh et al. proposed a seamless handoff architecture for Mobile IP [14]. A decision engine is added to the architecture of hierarchical Mobile IPv6. A handoff is initiated by an MN when it receives beacon signals from neighboring access routers (ARs). And the decision engine uses location tracking information of the MN and load information of these ARs to determine handoff time and target. Packets sent to the MN are initially forwarded to the new AR by the old AR, and simulcasted to both ARs after the old AR requests simulcast. This scheme successfully reduces network-layer handoff latency and packet loss, but it is centralized, requires extra signaling and imposes a bound on the speed of MNs.

2.2.3 Combined Handoff

Low latency handoff [15] [16] utilizes the L2 trigger mechanism (described in section 3.2.2). In pre-registration handoff, an MN performs network-layer handoff before the link-layer handoff, based on information from the L2 trigger. In post-registration handoff, an MN first completes link-layer handoff. It continues using the old FA and care-of address (CoA) through a bi-directional tunnel between the old FA and the new FA. The combined handoff method first tries a pre-registration handoff. If it fails, post-registration handoff is used.

Simultaneous Binding [2] [18] aims at decoupling network-layer handoff from link-layer handoff, by enabling an MN to bind to multiple subnets simultaneously. An MN can
retrieve the FA/CoA information from the beacon signals of APs, and register multiple CoAs with the HA before or after the link-layer handoff. The HA or correspondent node (CN) forwards duplicated packets to these CoAs simultaneously until the MN completes network-layer handoff.

Y. Gwon et al. introduced Mobile Initiated Tunneling Handoff [19]. Before disconnecting from the old AP, an MN can initiate a handoff by sending the old FA a handoff request containing the new FA’s information got from the mobile pre-trigger. Or it can initiate the handoff process after connecting to the new AP and receiving a link-up trigger, and send the new FA a handoff request containing the old FA’s information. A bi-directional tunnel is set up between these two FAs for the MN until it completes Mobile IP registration. This scheme achieved low latency and low loss handoffs with less requirements on L2 triggers and access networks than other fast handoff schemes.

Generally the proactive approach provides better performance, since packets are forwarded to the new subnet in advance. But these proactive schemes can cause unnecessary handoff preparations and forward too many duplicated packets. Neighborcasting forwards duplicated packets to all neighbor FAs without considering the MN’s moving direction. With simultaneous binding, an MN binds to any subnet from which it can receive a beacon signal, although in fact it may not directly move to all these subnets. As argued in the shadow cluster concept [17], an MN has influence near its current location and along its anticipated path, and resource reservation should be made for the MN based on the predicted demands.

2.3 Related Work on P2P Overlay Topology Optimization

To provide better scalability than unscalable Gnutella-like systems, many peer-to-peer systems have been proposed to use Distributed Hash Tables (DHT). They provide hash-table-like semantics system wide and construct structured P2P overlays with tightly controlled data placement and topology. Lookup operations based on DHT typically requires \(O(\log n)\) steps compared with \(O(n)\) steps needed by Gnutella. CAN [29], Chord [30], Pastry [31] and Tapestry [32] are examples of such systems. The authors of CAN further proposed a binning scheme [33] to partition the system into bins of nodes relatively close to each other in term of network latency. Each CAN node measures its round trip time (RTT) to a set of landmark nodes, and orders the landmarks with increasing RTT as the represen-
tation of the bin the node belongs to. The coordinate space is partitioned into equal sized portions, one for each bin. A new node joins the CAN at a random point in the portion associated with its bin. Nodes topologically close to each other are likely to have the same RTT ordering of landmarks, and will be assigned to the same portion of the coordinate space. Therefore neighbors in the coordinate space of CAN are likely to be topologically close on the physical network. If the content is replicated over the system, binning based server selection is used to select topologically close server to retrieve the content.

M. Castro et al. proposed an improved design in Pastry [34] [35] which uses topology-aware routing and new P2P overlay maintenance protocols to reduce the cost of overlay construction and operation. They compared three approaches of topology-aware routing: proximity routing, topology-based node ID assignment and proximity neighbor selection. Proximity routing constructs the P2P overlay without consideration about the topology of the physical network, but the node closest to the destination in the physical network is selected as the next hop. Topology-based node ID assignment maps the P2P overlay’s logical ID space onto the physical network so that neighboring nodes in the ID space are close in the physical network. This is similar to the approach used in binning of CAN [33]. Proximity neighbor selection constructs topology-aware P2P overlay by choosing topologically closest nodes from all nodes with IDs in the desired portion of the ID space and only adds them into the routing table. They concluded that proximity neighbor selection is the most promising one as it preserves load balance and robustness of random node ID assignment while greatly reducing the per-hop delay.

Topology matching problem has also been investigated in the scenario of unstructured Gnutella-like P2P overlays in the design of Gia [36]. The authors of Gia added new mechanisms into Gnutella, which dynamically adapt the P2P overlay topology and search algorithm to exploit node heterogeneity. A dynamic topology adaptation protocol is used to put most nodes close to high capacity nodes and ensure that high degree nodes have the capacity to handle queries. Each node computes its satisfaction level and search for appropriate neighbors to connect until fully satisfied. Search in Gia is biased random walk that directs queries to neighbors with highest capacity. Simulation shows that Gia can provide a total system capacity 3 to 5 orders of magnitude larger than that of Gnutella, and greatly improves the scalability of peer-to-peer system. Although Gia is designed to deal with a somewhat different topology matching problem, it provides some insights on topology adaptation of P2P overlays.
Location-Aware Topology Matching (LTM) [37] was proposed to solve the problem of topology mis-matching between the P2P overlay network and the physical network and reduce unnecessary traffic. It constructs an efficient overlay by disconnecting inefficient and redundant connections and choosing physically closer nodes as P2P neighbors. Each node floods a TTL2-detector periodically, which can provide its receiver network delay measurement within two P2P overlay hops. If a node receive the TTL2-detector from the same source along multiple paths, it removes the connection to one of its neighbors if it has the largest delay among the hops in the these paths. A node can also probe the delay to peers 2 hops away, and switch connections to peers with smaller delay. Their simulation shows that LTM can greatly reduce query cost and average neighbor distance. The authors claimed that LTM can deal with both static and dynamic environment, but the overhead is high in the latter case because TTL2-detectors are flooded periodically. For a mobile environment, the performance will be much worse as both node mobility and transiency add great dynamics to the P2P overlay. LTM requires peers to be synchronized, which also introduces extra overhead.

Topology Aware Grouping (TAG) [38] exploits underlying physical network topology to build efficient overlay networks for application layer multicast. It uses a path matching algorithm to traverse the overlay data delivery tree to determine the best parent for a new node. In order to reduce additional delay, the number of duplicated packets and the number of extra hops needed over a unicast path, the parent of a new TAG node is selected as the node whose shortest path from the multicast source has longest prefix of the new node’s own shortest path. The shortest path information is acquired by tools such as traceroute or from topology servers. Path matching is performed when a node leaves or the network conditions change.
Chapter 3

Explicit Proactive Handoff with Motion Prediction

Mobile IP has been widely accepted, but lacks a fast handoff mechanism. In this chapter, we introduce an explicit proactive handoff scheme with motion prediction. Since each user has patterns of movement, a mobile node predicts its future motion and explicitly notifies its old foreign agent which subnet it is likely to handoff to. During a handoff, the old foreign agent duplicates and forwards packets to the predicted subnets. With our scheme, network-layer handoff latency can be reduced to the level of link-layer handoff latency, and the number of packets lost during handoffs is also minimized. With a real network activity trace, we demonstrate that this scheme is able to predict motion accurately, with only a small overhead in bandwidth consumption and computation.

3.1 Introduction

Wireless local area networks (WLANs) have become extremely popular in these years. Link-layer mechanisms provide support for link-layer handoff, which is used to switch a mobile node (MN) from the radio link of one access point (AP) to that of another access point. For WLANs connected by an IP backbone, Mobile IP [2] is the protocol for location
management and network-layer handoff. This updates the routing information for the MN, to reflect movement from one subnet to another subnet. The message diagram for handoffs is shown in Figure 3.1.

![Message Diagram of a Standard Mobile IP Handoff](image)

Figure 3.1: Message Diagram of a Standard Mobile IP Handoff

Mobile IP, however, still has problems. First, it uses IP packets to transfer mobility management information. The latency of network-layer handoff is in the order of 0.1 to 1 second [21], 10 times larger than link-layer handoff latency. This cannot meet the requirement of delay-sensitive or real-time traffic, such as interactive voice and video which require delay and jitter to be less than 100 milliseconds. The second problem is handoff disruption. Mobile IP does not buffer packets sent to an MN during handoffs. During a network-layer handoff, all packets destined to the mobile node are delivered to its old subnet until the mobile node completes location registration to register the new care-of address at the HA. But these packets are not deliverable at the old subnet, as the latency of link-layer handoff is much smaller and the mobile node has switched to the radio link of an access point in the new subnet. Therefore, these packets may be lost and need to be retransmitted by the transport layer protocol. The third problem is that the two types of handoffs are coupled.
Because of the latency gap between them, packet loss will occur even after the completion of link-layer handoff.

A mobile node has patterns in its movements, and such patterns can be utilized to predict its future behavior and assist handoff. With prediction, it is possible to prepare network-layer handoff before link-layer handoff to reduce latency and packet loss. The MN’s optimal handoff strategy should also be adjusted dynamically according to the prediction to minimize handoff cost. Handoff decisions based on movement prediction eliminates the need to wait for beacon signals from other subnets, and assists the discovery of the handoff target in an environment of overlapping coverage areas and changing wireless channel conditions.

Based on these observations, we propose an explicit proactive handoff scheme with motion prediction for Mobile IPv4. It adopts a proactive approach to prepare network-layer handoff before link-layer handoff. Each MN records its movement patterns and predicts its future subnets. Before link-layer handoff, the MN explicitly notifies its current foreign agent (FA) of the predicted subnets. The current FA then duplicates packets sent to this MN, and forwards them to these predicted subnets. Network-layer handoff latency is close to that of link-layer handoff, and packet loss is reduced by buffering forwarded packets at FAs. This scheme is fully distributed because the network-layer handoff is controlled by the MN, and FAs are notified only if the MN recommends packet forwarding. There is extra bandwidth required to forward the duplicated packets, which is a cost. However, this cost is much less than that of other proactive handoff schemes.

The rest of this chapter is organized as follows. In Section 3.2, the proposed scheme is described. We evaluate its performance with simulation results in Section 3.3, and Section 3.4 is the conclusion and discussion of future work.

### 3.2 Proactive Handoff with Motion Prediction

#### 3.2.1 Motivations

We wish to minimize the effort and optimize the performance of network-layer handoff when an MN is moving among WLANs connected by an IP backbone. Each MN predicts its future movement based on the recorded movement patterns, and uses this to prepare network-layer handoff to reduce handoff latency and packet loss.
Handoff decision based on movement patterns eliminates the need for the MN to wait for beacon signals from neighboring subnets, which typically takes hundreds of milliseconds. Handoff target discovery only based on received signal strength can be a problem when the MN is in the overlapping coverage areas of multiple subnets and wireless channel conditions have great fluctuation. Movement patterns can provide another criterion to find the most possible handoff target in such a case.

Movement pattern and motion prediction are only concerned with network-layer movement, i.e., the logical movement between subnets. Physical or geographic location is of no importance to our scheme. We speculate that network-layer movement is simpler and more predictable than geographic movement, and is more meaningful for our purposes. For example, since subnets are of different shapes and sizes, an MN’s geographic movement parameters may not directly correspond to its regular activities, which on the other hand can be inferred by its network-layer movement.

It is also desirable to impose as few modifications to Mobile IPv4 as possible, and to require the minimum amount of additional capabilities from the link layer. After describing the method, we will return to these stated motivations.

### 3.2.2 System Overview

The system is assumed to have multiple WLANs connected by a wired IP backbone as shown in Figure 3.2. Mobile IPv4 is used for location management among subnets. The proposed proactive handoff scheme is used to reduce the latency and packet loss of network-layer handoff. Authentication and authorization of users are issues that must be handled by separate protocols.

The proposed scheme utilizes the L2 (layer 2) “trigger” mechanism [22]. The “Mobile Trigger” notifies the MN about an impending link-layer handoff, for example, link-layer notifies the mobile node when the signal strength from its serving access point is less than a threshold. And the “Link-Up Trigger” informs the FA that an MN has connected to the radio link of its subnet, for example, the access point can logically broadcast a link layer frame to announce an MN has associated with it. Beacon signals (link-layer frames) periodically transmitted by each AP contain information for both link-layer and network-layer handoffs, such as the ID of the AP, the AP’s frequency hopping or direct sequence parameters, the subnet ID and the IP address of the FA in that subnet, etc. Therefore an
MN does not need to wait for an FA advertisement to initiate location registration when it connects to the radio link of an AP in another subnet.

Two new messages are needed at the network layer. The first, the Forwarding Request message, is sent from an MN to the FA of its current subnet once the movement is predicted and the MN decides to handoff. This message contains the ID(s) and FA IP address(es) of the subnet(s) to which duplicated packets should be forwarded. The second, the Stop Forwarding message, is sent by the MN’s HA to its previous FA once the MN completes location registration. This informs the previous FA to stop forwarding duplicated packets.

The MN is responsible for recording network-layer movements. From this movement history, the MN will predict its future motion using the path prediction algorithm. Each MN maintains a First-In-First-Out (FIFO) movement history cache. An entry in the
movement history cache records the ID of a subnet to which the MN was previously attached, the start time of this attachment, and the IP address of the FA in that subnet. After the MN connects to the radio link of a new subnet, it creates a movement history cache entry of this connection, according to the beacon signal. It also has a pattern database to store movement patterns, which are sequences of movement history cache entries.

3.2.3 Handoff Process

Figure 3.3 shows the message diagram of a handoff. The handoff process is described as the following:

1. The MN moves to a subnet boundary. The Mobile Trigger informs the MN about an
impending link-layer handoff when the received signal strength from the current AP falls below the threshold level.

2. The MN, with the information in its movement history cache and pattern database, uses the path prediction algorithm to predict the subnet or subnets to which it is likely to move. (If the history data is not sufficient to make a prediction, the MN will randomly choose one of the neighboring subnets as the predicted subnet. If it is first time that the MN associates with the current subnet, it will not do any proactive handoff, and the standard Mobile IPv4 mechanism will be used.) The number of predicted subnets can be pre-defined.

3. If there is a predicted subnet or subnets, the MN sends the ID(s) and FA IP address(es) of the predicted subnet(s) to the current FA. They are sent in a Forwarding Request message.

4. Data packets sent to the MN during the handoff are first tunneled to the current FA by the HA, or directly by the CN with route optimization. The current FA first decapsulates these packets, and then duplicates and forwards them to the FA(s) of the predicted subnet(s) using IP-within-IP encapsulation. The predicted FA(s) decapsulate and buffer these data packets in anticipation of the arrival of the MN.

5. If the MN continues moving, it eventually performs a link-layer handoff to connect to the radio link of an AP in the new subnet. A new entry for this movement is created in the movement history cache. The MN then compares the ID(s) of the predicted subnet(s) with the ID from the beacon signal of the new AP.

6. If the prediction was not correct, the duplicated packets sent to the predicted FA will not reach the MN. These packets will eventually have to be retransmitted, as is normally the case without proactive handoff. Therefore the scheme falls back to the standard Mobile IPv4.

7. If the prediction was correct, the Link-up Trigger informs the FA of the new subnet to deliver its buffered data packets to the MN.

8. The MN registers its new care-of-address with the HA using Mobile IPv4 mechanisms.

9. The HA sends a Stop Forwarding message to the old FA to finish the handoff process.
Our scheme can also work in conjunction with other schemes to determine the handoff target. A separate mechanism can initially choose a certain subnet as the handoff target according to a variety of criteria such as the functionality of this subnet. Then our scheme will capture such a regular behavior and use it to predict the future network-layer movement of this MN.

The extra overhead of our scheme includes the cost to record movement history and do prediction, the transmission and processing cost of Forwarding Request and Stop Forwarding messages, and the cost to duplicate and forward data packets.

### 3.2.4 Path Prediction Algorithm

There have been a number of mobility models, such as the fluid flow model and the random walk model. These models describe the aggregate behavior of the MNs, and each MN’s movement is independent and random from the system’s view. These models also describe movement in terms of the change in the physical location of the terminals, rather than in terms of the change in the subnets they attach to. Therefore, we adopted an improved version of the motion prediction algorithm proposed by Liu and Maguire [20]. This method is designed for tracking individual MNs using historical movement patterns, and is based on logical movements rather than geographical movements. Here logical movement means the movement on subnet level, which is the change of associations with different subnets.

In this model, an MN enters a state when it connects to the radio link of a subnet. If it stays connected to this subnet for more than a specific period of time, this state is a stationary state. Otherwise it is a transitional state. A movement track (MT) models the movement of an MN on a regular route, starting from and ending at different stationary states. A movement circle (MC) models the MN’s long-term regular behavior, and starts from and ends at the same stationary state. Figure 3.4 shows an example of state transition diagram. The round nodes are stationary states, and the square nodes are transitional states. $S_i$ means the mobile node is connected to the radio link of subnet $i$. Here $S_1$ and $S_2$ are stationary states while $S_3$, $S_4$, $S_5$, $S_6$ and $S_7$ are transitional states. There are 2 MTs and 3 MCs in this diagram. $MT_1$ is $S_1 \Rightarrow S_3 \Rightarrow S_4 \Rightarrow S_2$, $MT_2$ is $S_2 \Rightarrow S_5 \Rightarrow S_1$. $MC_1$ is $S_1 \Rightarrow S_3 \Rightarrow S_4 \Rightarrow S_2 \Rightarrow S_5 \Rightarrow S_1$, which is $MT_1 + MT_2$. $MC_2$ is $S_1 \Rightarrow S_6 \Rightarrow S_1$, and $MC_3$ is $S_2 \Rightarrow S_7 \Rightarrow S_2$. 

For prediction purposes, the algorithm attempts to correlate current movement (i.e., a sequence of subnets that has just been visited) with past movement (i.e., a sequence of subnets that was visited at some previous time and stored in the pattern database). There are three types of matching for correlation analysis, two of them from [20], and one proposed by us. The first type is state-matching from the motion prediction algorithm in [20], which indicates the fraction of states in common between the current sequence and a past sequence. In Equation 3.1, $m_s$ is the number of identical states and $N_s$ is the total number of states.

$$\mu = \frac{m_s}{N_s} \quad (3.1)$$

The second type of matching is velocity-matching from the motion prediction algorithm in [20]. This indicates the similarity between the movement velocities of the current sequence and a past sequence. In Equation 3.2, $(t_{i+1} - t_i)_j$ is the interval between state $i$ and state $i + 1$ of sequence $j$.

$$\eta = \sum_{i=1}^{N_s-1} \left| \frac{(t_{i+1} - t_i)_{\text{current}} - (t_{i+1} - t_i)_{\text{past}}}{N_s - 1} \right| \quad (3.2)$$

We propose a third type of matching, called occurrence-matching. Let $T_{\text{current}}$ be the starting time of the current sequence, $T_{\text{past,L}}$ be the time of the last occurrence of some past sequence, and $\tau_{\text{past,k}}$ be the $k$th interval between two consecutive occurrences of this past sequence. Equation 3.3 computes how close the interval between the current sequence and the last occurrence of a past sequence matches the interval between any two consecutive

---

**Figure 3.4: A Sample State Transition Diagram**

MT1

S3

S4

MC1 = MT1 + MT2

S6 MC2

S1

MT2

S5

S2 MC3

S7
occurrences of the past sequence.

\[ \Phi = \min_k \frac{|T_{\text{current}} - T_{\text{past},L} - \tau_{\text{past},k}|}{\tau_{\text{past},k}} \]  

(3.3)

When an MN connects to the radio link of a new subnet,

1. The mobile node enters a new transitional state.

2. If it stays attached long enough, this state becomes stationary, and the MN starts pattern detection.

3. The mobile node uses state-matching to correlate the current sequence in the movement history cache with each MT and MC stored in the pattern database.

4. If there is no match, the current sequence is added to the pattern database, otherwise the information of this MC/MT in the pattern database is updated.

5. Then the current sequence is removed from the movement history cache if necessary.

When a Mobile Trigger fires, the MN initiates the handoff process and starts motion prediction. It correlates the current sequence in the movement history cache to the stored MCs and MTs, using (in order) state-matching, then velocity-matching, and after that occurrence-matching, until the number of MCs/MTs that match the current sequence is no more than the number of subnets need to be predicted. The number of predicted subnets is a parameter of the path prediction algorithm, and can be adjusted as required by the MN. The subnets at the corresponding positions of the final matched MCs/MTs are the predicted subnet. Generally the more subnets predicted by the MN, the less likely that the subnet it actually moves to is not any of the predicted subnets. However, this also increases the number of subnets that the old FA need to duplicate and forward packets to, which increases the overhead of proactive handoff. The MN also maintains a transition probability matrix among the subnets. If the current movement does not match any previously-stored pattern, the mobile node predicts the subnets with the highest transition probability from the previous subnet.
3.3 Performance Evaluation

There are several ways to evaluate the performance of the proposed proactive handoff scheme. To evaluate its effectiveness, we use handoff latency and prediction miss rate. To evaluate its efficiency, we use the fraction of duplicated packets in all packets, which indicates how many extra data packets are generated by our scheme.

3.3.1 Simulation Scenario

For purposes of evaluation, we simulated handoff behavior of a set of mobile nodes. This simulation is based on an actual trace taken from the campus-wide wireless network of Dartmouth College [8]. It recorded the activities of almost 2000 MNs for an academic term in an IEEE 802.11b network which covers 161 buildings and contains 81 subnets. The trace includes characteristics of both residential and work-related movements, since it recorded the network activity in both dorms and academic buildings.

The trace contains time-stamps and information on MNs’ association and reassociation activities with access points, but lacks details about the network topology, or reliable disassociation signaling. It also does not distinguish a direct handoff with the case that an MN was turned off, moved to another subnet and turned back on. Due to the limitations of the trace, we made the following assumptions in our simulation:

1. Each subnet has one and only one AP.
2. Each transition from a subnet to another subnet is a direct handoff.
3. Each MN stays at a subnet until it associates with another subnet.
4. Two subnets are neighbors if and only if there is at least one transition between them.
5. Reassociation to the same subnet is not a handoff.

There is no training period in our simulation. For purposes of computing packet loss and duplication rates, we used a simulated CBR traffic, with a packet interarrival time of 3.75ms for easier comparison with other proposed schemes. Since our scheme is for highly mobile users, we only show the simulation result for the mobile nodes with at least an average of 6 handoffs daily; there were 585 such MNs. For performance comparison purposes, we assume link-layer handoff latency is 50ms, and Mobile IP registration latency
is uniformly distributed between 200ms and 700ms. We simulated the performance of our scheme for the case of one predicted subnet, and two predicted subnets.

### 3.3.2 Handoff Latency

Figure 3.5 shows the distribution of the average network-layer handoff latency of each MN, using our scheme with prediction of one or two subnets. It can be seen that the latency of our scheme is much less than that of Mobile IP, and quite close to link-layer handoff latency. With prediction of two subnets, the latency of our scheme is less than 160ms for about 80% of the MNs. This is 3X faster than normal Mobile IP handoff latency.

![Empirical CDF](image)

**Figure 3.5: CDF Plot of Average Handoff Latency**

The average time to compute a prediction was less than 0.15 ms per handoff on a Pentium IV 1.7GHz PC with 256MB memory. Therefore we speculate that the algorithm is more than fast enough to support proactive handoff, even on a PDA-type device.
3.3.3 Prediction Miss Rate

The prediction miss rate is defined as the probability that the actual next subnet of the mobile node is not any of the predicted subnets. In Figure 3.6, we show the distribution of prediction miss rate of each MN, using our scheme with prediction of one or two subnets. With prediction of two subnets, about 80% of the MNs have a prediction miss rate less than 0.3. With prediction of one subnet, about 80% of the MNs have a prediction miss rate of less than 0.46. Compared with the miss rate of randomly selecting one or two neighboring subnets, our proposed scheme has a much lower miss rate. Only about 8% of the MNs have a miss rate less than 0.3 when randomly selecting 2 neighbor subnets. Neighborcasting sends packets to all neighboring subnets, and so should represent the lower limit of the prediction miss rate. Its miss rate is shown by the leftmost line in the figure, and is greater than zero because a miss is incurred when a movement from one specific subnet to another occurs for the first time. Our result using prediction of two subnets is quite close to this limit. This confirms the value of predicting MN movement based on past history. We believe that prediction using more complete traces, with information about network topology and MN power-down events, should do even better.

Using our scheme with prediction of two subnets, the average number of packet losses per handoff due to prediction misses is about 19, compared with 133 for Mobile IP and 5 for neighborcasting.

3.3.4 Extra Handoff Overhead

Section 3.2 discussed the extra overhead of the proposed scheme. Since our scheme is fully distributed, movement recording and prediction do not incur any cost to the network. The number of Forwarding Request and Stop Forwarding messages is trivial compared with the potential number of duplicated data packets, so we focus on measuring these duplicated data packets.

Figure 3.7 shows the distribution of the fraction of duplicated packets in the 585 MNs, based on CBR traffic sources. With prediction of two subnets, the fraction of duplicated packet is less than 0.011 for about 90% of the MNs, and less than 0.007 for 80% of the MNs. This is much better than neighborcasting. For instance, only about 45% of the MNs have a fraction of duplicated packets less than 0.007 with neighborcasting. Our scheme is
The prediction miss rate is the probability that the actual next subnet of the mobile node is not any of the predicted subnets. A curve closer to the upper left corner means less possibility of prediction misses and better performance of the prediction algorithm. Neighbor-casting has the lowest prediction miss rate, which is the compulsory miss rate of prediction algorithms based on movement history.

Figure 3.6: CDF Plot of Prediction Miss Rate

even more efficient when there are multiple MNs leaving the same subnet at the same time and a burst of duplicated data packets occurs.

### 3.4 Summary

In this chapter we proposed an explicit proactive handoff scheme based on the movement patterns of mobile nodes. An MN can anticipate a handoff from the L2 trigger, and use locally stored movement patterns to dynamically predict the next subnet. As a result, handoff latency and packet loss rate are dramatically reduced, with a cost of a small number of duplicated packets. This scheme works in a fully distributed fashion and introduces much less duplicated packets than another proactive handoff scheme, neighbor-casting. It also eliminates the need to wait for beacon signals, and solves the problem of handoff
The fraction of duplicated packets is the fraction of packets that were duplicated and forwarded to the predicted subnets in all the packets transmitted for a mobile node.

Figure 3.7: CDF Plot of Fraction of Duplicated Packets

target discovery when the coverage areas of multiple subnets overlap with each other. It does not require specific radio technology, or special routing techniques such as multicast. It keeps the current Mobile IP infrastructure and augments it to improve performance.
Chapter 4

Topology Matching with Local Topology Cache

4.1 Introduction

A peer-to-peer (P2P) overlay is defined as a logical network built on top of the physical network, in which all nodes have the same capabilities and responsibilities. There is no dedicated server or client, and each peer node acts as server and client simultaneously. It simplifies the network architecture and greatly improves the performance, reliability and robustness of the network. P2P overlays have been widely adopted for file-sharing, as we have witnessed the fast growth of numerous P2P systems such as Napster [23], Guntella [24], KzZaA [25], eDonkey [26], and BitTorrent [27]. Meanwhile, wireless/mobile network is another area that we have seen fastest growth in these years. Mobile devices can provide high-speed and cost-effective Internet access to users wherever they move, and have become an indispensable part of people’s life. To support P2P protocols on mobile devices would be a natural combination of them. [59] discussed a dynamic P2P overlay for voice systems on mobile devices, and [60] talked about design and implementation of P2P protocol for mobile phones. But node mobility also brings great challenges to P2P overlay construction and maintenance.
Current P2P systems construct and maintain the overlay without considering the topology of the underlying physical network. For an unstructured P2P overlay, usually the neighbors of a node in the P2P overlay are randomly selected, and flooding or limited scope flooding is used for queries. Here the flooding method is that a peer forwards the query to all its neighbors except the one that the query came from, when the query has a TTL larger than 0. But a peer only flood the same query once, which means if the peer receives the same query later, if will drop the query even if the query has a TTL larger than 0. Serious mis-matching exists between the topology of the P2P overlay network and the topology of the underlying physical network. Ripeanu et al. [28] found that more than 40% Gnutella nodes resides within the top 10 autonomous systems (ASes), but only 2-5% Gnutella connections link nodes within the same AS, which means far-away nodes instead of those in the same AS are selected as neighbors. Since the delay and cost of inter-AS connections are much larger than those of intra-AS connections, the P2P overlay has very inefficient connections. It is also possible that a message sent from a peer node to another one passes through the same physical link multiple times because of the mis-matched topology, which incurs unnecessary delay and transmission cost. It is even worse that a query can be flooded to the same destination along different overlay paths, which causes unnecessary traffic. All of these impose great stress on the physical network and limit the scalability of the P2P overlay. With mobile nodes, the problem is even more severe. Mobile devices usually have limited bandwidth, processing capability and battery power, therefore the penalty caused by inefficient P2P overlay topology has much greater negative impact on the performance of mobile devices.

The scalability problem of unstructured P2P overlay can be alleviated by utilizing the heterogeneity of nodes. In KaZaA, ordinary nodes are connected to super nodes with greater bandwidth and processing power, and the network is organized in a multi-level hierarchy with super nodes residing in the higher tiers. This eliminates the problem of blind flooding as the queries go to super peers first, but does not address the problem of topology mis-matching. Some topology matching techniques such as proximity routing and proximity neighbor selection have been proposed to base the P2P overlay construction and operation on the constrains of underlying physical network topology. Mobility of nodes introduces great dynamics to the physical network, and makes topology matching much more difficult. As shown by the example in Section 1.2.2, node mobility changes the topology of the physical network, and the P2P overlay has to be changed accordingly so that it still matches the new
physical network topology. None of the techniques mentioned above addressed the topology matching problem in a mobile environment. Although some of them might be able to deal with node mobility by periodically refreshing routing table or neighbor list, the overhead and delay of information exchange among a large amount of nodes can be prohibitive. However, as observed in multiple real-life wireless network traces [6] [7] [8], nodes’ movement has intrinsic patterns, and these patterns can be utilized to make prediction about nodes’ future behavior and assist topology matching.

In order to solve the topology mis-matching problem and improve the operation efficiency of P2P overlay with mobile nodes, we propose a topology matching scheme utilizing the movement pattern and topology history. The goal is to reduce the overhead of topology matching for P2P overlay of mobile nodes, while still making the P2P overlay match the physical network topologically, as much as the existing topology matching techniques do. When a mobile node moves to a specific subnet for the first time, it performs one topology matching operation, to probe the physical network and construct a P2P overlay that matches the physical network topology. After that the mobile node records the subnet ID and the information of its neighbors in this P2P overlay into its Local Topology Cache. The next time when the mobile node moves to the same subnet, it is possible that the topology of the physical network near this subnet is similar, and the P2P neighbors stored in the Local Topology Cache are still the mobile node’s neighbors in a topologically-matched P2P overlay. Therefore the mobile node can connect to these neighbors directly without performing topology matching operation again. This means that topology matching operations are only performed when the topology of the physical network nearby actually changes between the mobile node’s two consecutive visits to a subnet, and the cached neighbors are no longer those should be in a topologically-matched P2P overlay. This scheme is fully distributed and only need local information available to the mobile node, and the caching mechanism can work with any topology matching techniques. Our simulation results based on real-life wireless network traces [8] show that the overhead of probing and topology matching with Local Topology Cache is much less than existing topology matching schemes in a mobile environment, while still being able to construct a topologically matched P2P overlay.

The rest of this chapter is organized as follows. In Section 4.2, the proposed topology matching scheme with Local Topology Cache is presented. We evaluate its performance with simulation results in Section 4.3, and Section 4.4 is the summary of this chapter.
4.2 Topology Matching with Local Topology Cache

4.2.1 Motivations and Goals

As discussed in Section 1.2 and Section 4.1, the topology mis-matching between the P2P overlay and underlying physical network is very severe and forces the P2P overlay operate inefficiently. Although there have been some topology matching techniques proposed for both structured and unstructured P2P overlays, they also have a fair amount of overhead. Topology matching itself can generate a substantial amount of messages for probing and discovering the physical network topology. For the purpose of illustration, let’s suppose LTM [37] is used. If the average peer life time is 10 minutes and 0.3 query is issued by each peer per minute in average, the optimal frequency to perform LTM is twice per minute. In a Gnutella network each node has an average of 3.4 neighbors [28]. And as observed in a P2P overlay of 4500 users and 850,000 files with Zipf distribution, most query targets can be located if the query has reached 120 nodes [40], which translates to about 4 hops. In this case, if each query has TTL=4, in average a node’s queries pass through about 40 overlay hops per minute. A node’s detector messages pass through 23 overlay hops per minute in average to probe the physical network. This shows that the communication overhead of topology matching can be comparable to that for query. A mobile environment brings even greater challenges to the existing topology matching techniques as none of them was designed with node mobility in mind. To deal with the frequent change of network topology caused by node mobility, most of them have to periodically probe the physical network, and update thr P2P overlay at a higher frequency, and this can cause even greater overhead of network probing for topology matching. It is highly desirable to have an on-demand topology matching technique which only updates when the topology of the physical network actually changes, and this is likely to reduce the overhead of topology matching.

Compared with the transiency caused by nodes’ lifetime, mobility introduces a different kind of dynamics to the network topology. A node’s lifetime is random and hard to predict as it depends on the specific P2P application, while its mobility has intrinsic patterns since a mobile device is usually carried by an individual person who has some extent of regularity in his activities. Moreover, multiple nodes’ movement are correlated as the people carrying them have regularity in group movements. Studies of wireless networks
by Tang [6] [7] and Kotz [8] confirm this observation. Therefore change of physical network topology caused by node mobility is usually repetitive and predictable. When a mobile node returns to a subnet it has visited, it is quite possible that the topology of physical network close to this subnet is similar to that when the node visited there previously, and the mobile node’s previous neighbors in a topologically-matched P2P overlay during its last visit to this subnet are still its neighbors that should be in a topologically-matched P2P overlay. The mobile node can reestablish virtual connections to these stored neighbors to construct a P2P overlay that matches the physical network topology. Therefore the overhead of topology matching can be reduced while the P2P overlay still matches the physical network topology, as much as that using existing topology matching techniques.

There is also a huge gap between the network layer and the application layer in current designs of P2P overlay. Nodes’ activities on the network layer are totally ignored in the construction and maintenance of the P2P overlay. However, they do provide useful information about nodes’ activities on the application layer. Utilizing movement and topology history information in topology matching would be one of the steps to close this gap.

Based on the motivations, we propose this topology matching scheme in which each mobile node caches the information of its neighbors in a topologically-matched P2P overlay when visiting a subnet previously. The topology matching operation is only executed when there is a cache miss and new topology matching is necessary. Our target is to reduce the overhead of topology matching in a mobile environment, and also keep the P2P overlay topologically matched to the physical network. The degree that a P2P overlay globally matches the physical network topology, or the global efficiency of the P2P overlay topology, is evaluated by three metrics. The first one is the number of unique peers each peer’s query can reach with a certain TTL, which is called the search scope. The second one is the number of overlay hops this query has to pass through in average to reach a unique peer in the search scope, which is called the average query cost. The third one is the average neighbor distance of each peer. Given two P2P overlay A and B based on the same physical network topology, if in A most peers have bigger search scope, smaller average query cost and smaller average neighbor distance, compared to those in B, globally A is more topologically matched to the physical network than B, and the topology of A is more efficient than that of B. We are targeting at reducing the overhead of topology matching with the caching mechanism, while still constructing a P2P overlay that most mobile nodes
have search scope as large as that using existing topology matching techniques, and average query cost and average neighbor distance as low as those using existing techniques. When the cache hits, a mobile node constructs the P2P overlay by reusing the stored neighbors without performing topology matching operation, and this reduces the overhead of topology matching, which is the gain of the proposed scheme for correct prediction of the physical network topology. If these neighbors cannot form a P2P overlay that matches the physical network topology, the mobile node’s search scope will be reduced, the average query cost and average neighbor distance will be increased. This is the penalty of the proposed scheme for incorrect prediction of the physical network topology. In this section we will first give an overview of the proposed scheme and then describe its mechanisms in detail.

4.2.2 Overview

Here we are focusing on the topology matching problem for unstructured Gnutella-like P2P overlays with mobile nodes. As argued in [36], the DHT based P2P systems are not as robust as Gnutella-like designs when the nodes are transient. They can provide exact-match lookups with good performance, but it is not easy to support keyword searching with DHT. DHT based systems can guarantee that a file can be found even if there is only one copy in the system, but typical requests in a peer-to-peer file sharing system are for well-replicated files. Therefore for typical P2P applications such as file sharing, Gnutella-like system might be a better choice over structured systems based on DHT. They are also relatively simple to implement in current Internet. We also assume that the neighborhood relationship of peer nodes are symmetric, which means if node A is a neighbor of node B, node B is also a neighbor of node A.

Each mobile node maintains its own Local Topology Cache to record the information of its neighbors in a topologically matched P2P overlay when visiting a subnet previously. The basic content of a Local Topology Cache entry is shown in Table 4.1. Each Local Topology Cache entry corresponds to a subnet the mobile node has visited previously. It contains the ID of the subnet visited and information of each neighbors in a topologically matched P2P overlay, which the mobile node was connected to by a virtual connection when staying at this subnet. Such information includes the “home” IP address of the P2P neighbor and its residing subnet ID when the cache entry is recorded.

The basic topology matching process with Local Topology Cache is shown in Al-
Table 4.1: Local Topology Cache Contents

<table>
<thead>
<tr>
<th>SubnetID</th>
<th>Topology-Matched P2P Neighbors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neighbor1</td>
<td>...</td>
</tr>
<tr>
<td>SubnetID1</td>
<td>...</td>
</tr>
<tr>
<td>IP Addr1</td>
<td>SubnetID1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>NeighborN</td>
<td>IP AddrN</td>
</tr>
<tr>
<td>SubnetIDN</td>
<td></td>
</tr>
</tbody>
</table>

Algorithm 1 Topology Matching with Local Topology Cache

1: MN arrives at a subnet
2: search cache for SubnetID
3: if SubnetID exists in cache then
   4: check entry validity
   5: if valid entry then
      6: connect to cached neighbors
      7: update cache entry
   8: else
      9: seek candidate neighbors
     10: topology match within candidate neighbors
    11: update cache (SubnetID, matched neighbors)
 12: end if
13: else
14: seek candidate neighbors
15: topology match within candidate neighbors
16: create new entry (SubnetID, matched neighbors)
17: end if

The mobile node moves to a subnet, and gets the ID of this subnet from the beacon signals transmitted by wireless access points. It then searches its Local Topology Cache for an entry corresponding to this subnet ID. If such an entry is located, the mobile node checks the validity of this entry by sending a “ping” message directly to the home IP address of each neighbor stored in this entry. Every neighbor replies with a message containing the ID of the subnet in which it is currently residing, and this subnet ID is compared with the subnet ID associated with this neighbor stored in the cache entry. If for each of the neighbors, the current subnet ID is the same as the one cached, this Local Topology Cache entry is valid and we define it as a cache hit. Otherwise the cache entry is not valid and this is a non-compulsory miss. The mobile node disconnects from all its current neighbors and sets up virtual connections with these cached P2P neighbors, therefore ends the topology matching procedure. If this cache entry is not valid, one or more of the stored neighbors are
not at the corresponding subnets stored in the cache entry. The mobile node first searches candidate P2P neighbors using standard mechanism of the P2P application or some location hints, and then executes a topology matching operation within these candidates to select neighbors that can form a topologically matched P2P overlay. After that the Local Topology Cache entry is updated with the new neighbors’ IP addresses and IDs of residing subnets. In the case that there is no cache entry corresponding to the ID of the subnet the mobile node has just moved to, the mobile node has never visited this subnet previously, and this is regarded as a compulsory cache miss as there is no neighbor information available for this mobile node at this location. The mobile node then searches candidate P2P neighbors and performs one topology matching operation within the candidates to find neighbors that can form a topologically matched P2P overlay. A new cache entry is created to store the subnet ID and the information of resulting P2P neighbors.

There is no limit to some specific types of topology matching operation that our local topology caching mechanism can work with. Actually the caching mechanism is just used to assist topology matching operation and reduce its frequency and overhead, regardless of what the exact topology matching technique is. Here we used a modified version of Location-Aware Topology Matching (LTM) [37] as discussed in Section 2.3 to demonstrate the performance of Local Topology Caching. We used the mechanism of original Gnutella to search candidate neighbors.

### 4.2.3 Caching Strategies

We used basic cache replacement and retrieving strategies for the initial part of this research project. For each Local Topology Cache entry corresponding to a specific subnet a mobile node has visited, only one set of previous neighbors that formed a topologically-matched P2P overlay is stored. For each subnet, the size of cache is one. When a cache miss occurs, the mobile node performs topology matching operation, gets a list of neighbors that forms a topologically matched P2P overlay, and uses them to replace the set of neighbors stored in the cache. Therefore the mobile node always retrieves the newest set of previous neighbors in a topologically matched P2P overlay when visiting the same subnet, regardless of the exact time when the information was stored. Although this is simple and does not require much memory space for Local Topology Cache, it may have negative impact on the cache hit rate as the cache is not able to store and retrieve neighbor information before
the mobile node’s last visit to the subnet. Advanced caching strategies can be used to store multiple sets of topologically matched neighbors in a single entry, and to retrieve an appropriate set according to the state of the mobile node when the cache is accessed. Song et al. [39] found that a simple Markov chain model has good performance to predict the next movement of a mobile node, and similar model may also be used to retrieve the appropriate neighbor set. In the simulation part, we used the basic caching strategies.

4.2.4 Topology Matching Operation

The topology matching operation is performed to find neighbors that can form a topologically-matched P2P overlay within a mobile node’s candidate neighbors. The basic idea is similar as in LTM [37]. In LTM, each node floods TTL-2 detectors periodically, and the initiator of topology matching is actually the source node of the TTL-2 detectors. One-way delay is used as the measurement of network distance. The receivers of TTL-2 detectors can calculate delay from the source and intermediate node the detector has passed through. Based on the delay measurement, a mobile node removes neighbors that cannot form a topologically-matched P2P overlay. The biggest problem of LTM is that it is not on-demand and topology matching has to be performed periodically even if the topology of physical network has not changed, which wastes network resources. The other problem of LTM is the requirement of synchronization among nodes to perform topology matching, which is difficult to achieve among a large population of mobile nodes and incurs extra overhead. Using one-way delay as distance measurement would have problems with asymmetric links, which are quite common in wireless networks.

Our topology matching operation is purely on-demand, and is only executed when there is a miss in Local Topology Cache and topology matching operation is necessary. No periodic updating or refreshing is required, therefore the number and overhead of topology matching operations needed is much less. This is achieved by adopting an active approach instead of the passive approach used in LTM. The initiator of topology matching is the node who actually probes its vicinity and removes neighbors that cannot form a topologically-matched P2P overlay. No synchronization among nodes is required as each node can probe the network and do topology matching independently. We also use Round Trip Delay (RTT) as the measurement of network delay to deal with the problem caused by asymmetric links.

The topology matching operation has three steps. In the first step, the node
executing topology matching operation probes the network in its vicinity using “ping-pong” messages. After that, in order to avoid the case that the same query is flooded to the same peer via different overlay paths, it searches neighbors that cannot form a topologically-matched P2P overlay, and put them into a list to be removed later. It also puts its neighbors with larger delay measurements to the list to be removed, and adds other peer nodes with smaller delay measurements as neighbors, so that the average delay between neighbors can be reduced. In the last step, the node removes the neighbors in the removal list. Now we use the following example to show how the topology matching operation works.

**Probing with “Ping-Pong” Messages**

The node initiating topology matching operation first probes the physical network topology. It gets the delay of P2P virtual connections within a radius of 2 overlay hops using “ping-pong” messages. The diagram of the these messages is shown in Figure 4.1.

![Diagram of Ping-Pong Messages](image)

Figure 4.1: Diagram of Ping-Pong Messages

Suppose node MN1 with IP address $IP_1$ initiates topology matching operation at time $T_1$ by sending a TTL-2 “ping” message with a unique message ID to each of its candidate neighbors. The content of the TTL-2 “ping” message is shown in Table 4.2, with message ID set as 1, source IP address set as $IP_1$, initial timestamp set as $T_1$, both IP
Table 4.2: Content of TTL-2 “ping” message sent by MN1

<table>
<thead>
<tr>
<th>msg ID</th>
<th>src IP</th>
<th>init TS</th>
<th>inter IP</th>
<th>inter TS</th>
<th>TTL count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IP1</td>
<td>T1</td>
<td>N/A</td>
<td>N/A</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 4.3: Content of TTL-2 “pong” message replied by MN2

<table>
<thead>
<tr>
<th>msg ID</th>
<th>dest IP</th>
<th>src IP</th>
<th>init TS</th>
<th>inter IP</th>
<th>inter TS</th>
<th>TTL count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IP1</td>
<td>IP2</td>
<td>T1</td>
<td>N/A</td>
<td>N/A</td>
<td>2</td>
</tr>
</tbody>
</table>

address and timestamp of intermediate node set as N/A, and TTL set as 2. When one of MN1’s candidate neighbors MN2 with IP address IP2 receives this TTL-2 “ping” message, it replies a TTL-2 “pong” message to MN1 with the same message ID it received. The content of this TTL-2 “pong” message is shown in Table 4.3, in which the message ID is 1, IP2 is the source IP address, IP1 is the destination IP address, initial timestamp is set as the initial timestamp T1 received from the TTL-2 “ping” message, both IP address and timestamp of intermediate node are N/A and TTL is 2. MN1, the initiator of topology matching, receives this TTL-2 “pong” message at T2. Although this message is received with TTL=1, MN1 does not relay it as the destination IP address is MN1’s IP address IP1. Therefore MN1 can get $T2 - T1$ as the RTT measurement to its candidate neighbor MN2.

Table 4.4: Content of TTL-1 “ping” message relayed by MN2

<table>
<thead>
<tr>
<th>msg ID</th>
<th>src IP</th>
<th>init TS</th>
<th>inter IP</th>
<th>inter TS</th>
<th>TTL count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IP1</td>
<td>T1</td>
<td>IP2</td>
<td>T3</td>
<td>1</td>
</tr>
</tbody>
</table>

At the same time, since MN2 receives a TTL-2 “ping” message from MN1, it also relays a TTL-1 “ping” message to each of its topologically matched neighbors. It keeps the same message ID, source IP address and initial timestamp, appends its IP address IP2 to the intermediate node IP address field, and fills the intermediate node timestamp field with T3, the time when this TTL-1 “ping” message is transmitted. The content of the TTL-1 “ping” message relayed by MN2 is shown in Table 4.4. When MN2’s topologically matched neighbor MN3 receives this relayed “ping” message, it replies MN2 with a TTL-2 “pong” message shown in Table 4.5. It keeps the same message ID 1, sets TTL to 2, and
Table 4.5: Content of TTL-2 “pong” message replied by MN3

<table>
<thead>
<tr>
<th>msg ID</th>
<th>dest IP</th>
<th>src IP</th>
<th>init TS</th>
<th>inter IP</th>
<th>inter TS</th>
<th>TTL count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IP1</td>
<td>IP3</td>
<td>T3</td>
<td>IP2</td>
<td>N/A</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 4.6: Content of TTL-1 “pong” message relayed by MN2

<table>
<thead>
<tr>
<th>msg ID</th>
<th>dest IP</th>
<th>src IP</th>
<th>init TS</th>
<th>inter IP</th>
<th>inter TS</th>
<th>TTL count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IP1</td>
<td>IP3</td>
<td>T3</td>
<td>IP2</td>
<td>T4</td>
<td>1</td>
</tr>
</tbody>
</table>

uses MN1’s IP address IP1, its own IP address IP3 and MN2’s IP address IP2 as the destination IP address, source IP address and intermediate node IP address respectively. The initial timestamp field is set with T3, the intermediate node timestamp that MN3 receives in the TTL-1 “ping” message relayed by MN2. The intermediate node timestamp is set as N/A. After MN2 receives this TTL-2 “pong” message, it relays a TTL-1 “pong” message to MN1 according to IP1, the destination IP address in the TTL-2 “pong” message. As shown in Table 4.6, the TTL-1 “pong” message keeps the most value unchanged, but sets the intermediate node timestamp with T4, the time when MN2 receives the TTL-2 “pong” message from MN3. When MN1 receives this relayed TTL-1 “pong” message, it does not relay it again as the TTL=0. Therefore MN1 can determine that MN3 is 2 hops away via candidate neighbor MN2 according to the source and intermediate node IP address, and the RTT between MN2 and MN3 is $T4 - T3$.

Alternatively, to reduce the probing overhead, each node can store a measurement of RTT corresponding to each of its neighbors, and replies probes with its neighbor list and RTT measurements. Therefore, the node performs topology matching only need to use TTL-1 “ping” messages to get the topology and delay measurement within the 2-hop radius. One disadvantage of this is that the stored RTT measurements may become stale due to network dynamics. This approach is not used in this research.
Finding Neighbors with Smaller Distance and Neighbors that Cannot Form a Topologically-Matched P2P Overlay

After probings using “ping-pong” messages, the topology matching initiator MN1 gets the un-optimized P2P overlay topology and the RTT of each P2P connections within the radius of 2 overlay hops from itself. Then it starts to try replacing neighbors with those have smaller network delay measurement and finding neighbors that cannot form a topologically-matched P2P overlay. All the candidate neighbors to which MN1 has to remove are first put into a list, and after the completion of this step, these candidate neighbors are removed from MN1’s neighbor list and corresponding connections are cut. MN1 may receive multiple “pong” messages with the same source IP address, and this can be reduced to the case that one or two “pong” messages are received from the same source IP address. Here we discuss the case that one or two “pong” messages are received from source node MN3 with IP address $IP_3$, and there are three different combinations.

- **Only one “pong” message with TTL=0.** As shown in Figure 4.2a, in this case the length of MN1’s shortest path to MN3 on the P2P overlay is 2, and MN1 probes the RTT of its direct connection to MN3 by using a TTL-1 “ping” message. MN1 can also find the intermediate peer nodes along the path to MN3 by checking the intermediate node IP address field in the received “pong” messages whose source IP address is $IP_3$. Here we suppose the intermediate node is MN2. It then compares the RTT between MN1 and MN3 to the RTT between MN1 and MN2, and the RTT between MN2 and MN3. If the RTT between MN1 and MN3 is the smallest among the three, MN1 adds MN3 as its neighbor, and puts MN2 into the removal list. The result is that MN1 can still reach MN2 and MN3 within 2 overlay hops, but its distance to the neighbor is smaller that before. By this way, nodes with less network delay measurement are selected as the neighbor of the node initiating topology matching.

- **One “pong” message with TTL=0 and one “pong” message with TTL=1.** As shown in Figure 4.2b, this means there are a 1-hop path and a 2-hop path from MN1 to MN3, and let’s suppose the intermediate node along the 2-hop path is MN2. Therefore MN1, MN2 and MN3 form a 3-hop loop, and there is one connection that is redundant and causes unnecessary query messages. MN1 then compares the RTT between MN1 and MN2, the RTT between MN1 and MN3, and the RTT between MN2 and MN3. If
either the RTT between MN1 and MN2 or that between MN1 and MN3 is the largest one among the three, MN1 puts the corresponding neighbor to the list to be removed later.

- **Two “pong” messages with TTL=0.** As shown in Figure 4.2 c, in this case MN1 has two 2-hop paths to MN3, and let’s suppose the intermediate nodes are MN2 and MN4 respectively. MN1, MN2, MN3 and MN4 form a 4-hop loop, and a redundant connection need to be removed to open the loop and stop unnecessary query messages. If either the RTT between MN1 and MN2 or that between MN1 and MN4 is the largest one among the RTTs of the 4 connections in this loop, MN1 puts the corresponding neighbor to the list to be removed later.

![Figure 4.2: Three Cases of “Pong” Messages Received](image)

**Neighbor Removal**

The final step is to remove the neighbors that are stored in the removal list. The candidate neighbors left are those who can form a topologically-matched P2P overlay for the node performing topology matching. This opens the 3-hop and 4-hop loops in the P2P overlay within a radius of 2 hops from the node initiating topology matching operation, therefore the case that the same query reaches the same peer via different paths caused by short loops are almost eliminated. As observed in [28], more than 95% nodes in Gnutella are less than 7 hops away from each other, therefore a query message is unlikely to have a TTL close to 7. It is possible that loops of more than 4 hops still exist in the P2P overlay topology, but with such constraint of TTL, it is unlikely that a query message reaches the same peer multiple times via different paths. Opening these loops also reduces the
redundancy in the P2P overlay in some degree, but we speculate it will not cause severe problems as there are still loops with more hop counts.

The TTL of “ping” messages can also be adjusted as a configuration parameter to limit the scope of topology matching. Larger TTL means that the topology within a larger radius from the node can be matched, but the cost is greatly increased topology matching overhead.

4.2.5 Overhead of Topology Matching with Local Topology Cache

The proposed topology matching scheme with Local Topology Cache performs topology matching operation only when it is necessary. If the mobile node does not move to another subnet, or if it moves to another subnet but the movement results in a hit of Local Topology Cache, no topology matching operation is needed. Compared with the periodic refreshing that existing topology matching schemes have to use in a mobile environment, this is a huge reduction in the overhead of topology matching. In LTM, only to deal with the dynamics caused by node lifetime with an average of 50-60 minutes, optimally the topology matching operation need to be performed at each node once every 3 minutes. But in a real wireless network, the frequency of nodes’ inter-subnet movements is much less than once every 3 minutes [41]. With Local Topology Cache, the frequency of topology matching operation is further reduced.

For each topology matching operation performed by a node, the communication overhead is about 2.5 times of LTM as the node actively probes its vicinity instead of passively receiving TTL-2 detectors. But this eliminates the requirement of synchronization among nodes, and enables the on-demand execution of topology matching operation, which greatly reduces the frequency of topology matching operation. If each node stores RTT measurements to its neighbors and TTL-1 “ping” messages are used for probing, the traffic overhead for each topology matching operation can be reduced to about the same as LTM.

With Local Topology Cache, the latency of topology matching is also reduced. A topology matching operation need to probe nodes within a radius of 2 overlay hops from the mobile node who initiates topology matching, but with Local Topology Cache, only nodes within a radius of 1 overlay hop have to be probed if there is a hit in the cache. Since the number of nodes in the 1-hop radius is much less than that in the 2-hop radius, the number of probe messages and associated latency with Local Topology Cache is greatly reduced.
4.3 Performance Evaluation

Since the proposed topology matching scheme with Local Topology Cache is based on the assumption that mobile nodes have patterns in their movements and the physical network topology close to a subnet is unlikely to have significant change between two visits of a mobile node, it is a must to evaluate its performance in a real network scenario. In this section we first discuss the metrics used for performance evaluation purpose, and then describe the simulation scenario. The simulation result is presented in the third part of this section, together with analysis and discussion.

4.3.1 Metrics for Performance Evaluation

As the proposed scheme is a combination of caching techniques and topology matching mechanisms, we need to evaluate its performance from both the aspect of Local Topology Cache and that of topology matching part. Naturally, one of the most important metrics to evaluate cache performance is the miss rate. The percentage of compulsory and non-compulsory misses in all cache misses is also a metric to judge the effectiveness of cache. From the aspect of topology matching, we also need to know how effective it is in constructing a topologically-matched P2P overlay. We have to measure how much reduction in the average query cost and average neighbor distance, and if there is any shrink of the search scope, compared to an unstructured P2P overlay with random neighbor selection. In the combination of the two parts, we need to find out how much topology matching overhead the caching mechanism can reduce, and whether and to what degree caching causes the P2P overlay to be less topologically-matched to the physical network, compared to existing topology matching techniques. Since mobile nodes have different levels of mobility, we have to evaluate the performance together with these mobility levels.

Here we present the metrics used to evaluate the performance of the proposed scheme.

i) Cache miss rate: We define a miss in the Local Topology Cache as the event that a mobile node does not have a cache entry for the subnet ID it is looking for, or the cache entry corresponding to the subnet ID is invalid. The first case is a compulsory miss since it means that the mobile node has never visited this subnet. The second case is a non-compulsory miss, and it is caused by the change of topology in the physical
network. The lower the miss rate is, the more effective Local Topology Cache would be to reduce the number of topology matching operations.

ii) **Topology matching overhead**: One of the major goals of our scheme is to reduce the overhead of topology matching, therefore the amount of reduction achieved is one major metric for performance evaluation. A mobile node’s topology matching overhead is measured by the average number of overlay hops the probing messages from this mobile node have to pass through per time interval. The probing messages include the TTL-2 detectors in LTW, and “Ping/Pong” messages in the proposed scheme. With the P2P overlays topologically matched to the physical network on the same level, the less topology matching overhead, the better the performance of the proposed scheme would be.

iii) **Search scope**: We define the search scope of a node as the number of unique peer nodes it can reach within a certain overlay hop count in the P2P overlay using flooding query. The larger the search scope, the less hop count needed to search for an object in the P2P overlay. Usually the nodes have larger search scopes if they have larger number of neighbors, but this would also greatly increases the cost of queries. The advantage of topology matching is that the search scope is kept while the query cost is greatly reduced.

iv) **Average query cost per reached node**: It is the average number of overlay hops the query from a mobile node has to pass through to reach a unique peer in the mobile node’s search scope. It is calculated as the number of overly hops a query with the specific overlay hop count passes through divided by the number of unique nodes in the search scope. When a mobile node sends the query to one of its neighbors, the query pass through 1 overlay hop. When this neighbor forwards the query to one of its own neighbor (neighbor of neighbor of the node initiating query), there is another 1 overlay hop passed. The less average query cost per reached node, the more the P2P overlay is topologically-matched to the physical network.

v) **Average neighbor distance**: This is another metric to evaluate the level of that a P2P overlay is topologically-matched to the physical network. It shows whether physically closer nodes are selected as P2P neighbors. The distance here is defined as the RTT between the peers in the P2P overlay. The lower the average neighbor distance, the
more efficient the P2P overlay is.

The goal of the proposed topology matching scheme with Local Topology Cache is to reduce the overhead of topology matching, while achieving average query cost per reached node and average neighbor distance as low as those using existing topology matching techniques, and search scope as large as that using existing techniques. We evaluate the cache miss rate with different mobility levels, and the overhead needed for on-demand topology matching with or without caching and periodic LTM. We also compare search scope, average query cost per reached node and average neighbor distance for the cases of random neighbor selection, topology matching without caching, and topology matching with Local Topology Cache.

4.3.2 Simulation Scenario

The performance of our proposed scheme has to be evaluated in a real wireless network scenario, and fortunately, Dartmouth College provides a good archive of wireless network trace data consisted of real-life activities in its campus-wide wireless network [41]. The IEEE802.11b wireless network at Dartmouth College covers the whole campus of about 200 acres with 623 access points in more than 160 buildings. The population on campus is consisted of about 5,500 students and 1,900 faculties and staffs. This trace provides the characteristics of wireless network activities associated with different parts of people’s life, as it covers all kinds of buildings such as academic buildings (classrooms and labs), administrative buildings (offices), libraries, residential buildings (dorms), and etc. This is an ideal basis for our simulation.

The syslog part of Dartmouth trace provides nodes’ movement information, including the MAC address of the mobile node, the timestamp of the movement, and the name of the access point. It also contains the timestamp when a mobile node turns off and leaves the wireless network, inferred by a predetermined interval of inactivity. Therefore we can use the “on-off” activities of mobile nodes to simulate their lifetime on the P2P overlay. Analysis of Dartmouth wireless network trace [8] reported that the median length of a wireless session between on and off is 16.6 minutes and 71% of the sessions finish within one hour. There is also a list of access points with geographical (x,y,z) coordinates for each access point. Since detailed information about the network topology is not available, we assume that the network is rather “flat”, in which each building has a wireless subnet, and
these subnets are connected to each other by a wired IP backbone. The delay between access points is approximated by the geographical distance calculated from their (x,y,z) coordinates, and we assume the delay of the wireless link between an access point and a mobile node to be a constant.

We used the Dartmouth trace between January 1 and January 31, 2003 to do the simulation. To keep the simulation time at a reasonable level, we only simulated the activities of mobile nodes whose MAC address is within the range of 00022d000000 and 00022dffffff, and in the simulation we observed 1528 such mobile nodes, which form a P2P overlay. We categorize these mobile nodes to low-mobility, medium-mobility and high-mobility nodes. There are 548 low-mobility nodes, each of them has at most one movement per day in average, and 714 medium-mobility nodes with more than one but at most ten movements per day in average, and 166 high-mobility nodes with more than ten movements per day in average.

In our simulation, these 1528 mobile nodes forms a Gnutella-like unstructured P2P overlay, and the neighborhood relationship between nodes is symmetrical. Each mobile node first checks its Local Topology Cache for an appropriate entry when moving to a subnet. If it is a cache hit, the mobile node connects to the neighbors stored in the entry. If it is a cache miss, the mobile node retrieves a set of candidate neighbors with standard Gnutella mechanisms. Each peer has 10 random nodes as neighbors and then perform topology matching operation to remove the neighbors that cannot form a topologically-matched P2P overlay. 10 is an arbitrary number we used in the simulation. There is no training or warm-up period in the simulation, and the results are collected from the beginning of the simulation.

4.3.3 Simulation Results

Here we present the results of our simulation. These results are used to evaluate the effectiveness of topology caching and the efficiency of the topologically-matched P2P overlay. We also compared the P2P overlay efficiency of the case that each peer in the P2P overlay has 10 random neighbors, topology matching without caching and topology matching with caching to see if overlay efficiency is reduced by caching mechanism.
Miss rate of Local Topology Cache

A miss of Local Topology Cache is the event that there is no entry for the specific subnet, or there is such an entry, but it is not valid as one or more stored neighbors are not at the corresponding subnets recorded in this entry. Miss rate is the most important metric to evaluate the performance Local Topology Cache, and we compared the miss rates of low-mobility, medium-mobility and high-mobility nodes. Fig. 4.3 shows the CDF of Local Topology Cache miss rate for the three categories of mobile nodes. From this figure we can find that the miss rate of Local Topology Cache is quite high for low-mobility and medium-mobility nodes. This is a reasonable finding as the low-mobility nodes do not have sufficient movement and topology history data to support Local Topology Caching, and the topology of the physical network in the vicinity of subnets they previously visited is likely to have great changes due to the long intervals between movements. On the other hand, high-mobility nodes have decent miss rates, and for about 70% of them the miss rate is less than 0.36. This means that for 70% high-mobility nodes, they only need to execute topology matching operation for less than 36% of their movements. The ratio of the total number of cache misses to that of cache accesses for all of the 1528 mobile nodes is about

![CDF of Local Topology Cache Miss Rate](image)

Figure 4.3: CDF of Local Topology Cache Miss Rate
0.26, which means mobile nodes with very high mobility have particularly low cache miss rate. As PDA or smart-phone type mobile devices become more and more popular, the mobility of mobile devices is likely to increase, and Local Topology Cache is quite effective for them. Fig. 4.4 shows what fraction of cache misses is compulsory, which is caused when the mobile node has never visited the specific subnet and there is no previous topology information available. Generally mobile nodes with lower level of mobility have a large fraction of compulsory misses in total cache misses, while nodes with high mobility only have a small fraction of compulsory misses. This is good and shows that Local Topology Cache has room to be further improved with advanced caching strategies to reduce the cache miss rate, when the node has high mobility level.

**Overhead of Topology Matching**

We evaluated the overhead of topology matching using the average number of overlay hops the probing messages from a mobile node have to pass through per second. We compared the topology matching overhead of our on-demand scheme with or without Local Topology Cache, and periodic LTM. As the median interval between mobile nodes
“on-off” activities is 16.6 minutes, the optimal LTM frequency is twice per minute. Here in the simulation we used LTM frequency of once per minute for the comparison purpose. Fig. 4.5 shows the overhead of topology matching for the three cases. The graph shows the topology matching overhead for 618 mobile nodes with at least 3 movements per day in average. For most mobile nodes, the overhead for on-demand topology matching is about \(\frac{1}{10} - \frac{1}{8}\) of that for periodic LTM. And the topology matching overhead with caching is less than half of that without caching for about 50% mobile nodes.

**Search Scope**

The goal of topology matching is to reduce the query cost while keeping the search scope about the same. Here in the simulation, we define the search scope as the number of unique peer nodes a mobile node can reach by flooding a query with TTL=4 to all its neighbors, each neighbor also forwards the queries to all of its neighbors except the one the query came from, and etc., until the TTL is 0. Topology matching opens loops with the length of 3 or 4 overlay hops, therefore a node used to be reachable within 4 hops might only be reached with more than 4 hops, and this can possibly shrink the search scope. We
The search scope is the average number of unique peers a mobile node can reach with a TTL=4 query. A mobile node’s normalized search scope is 1 when each peer in the P2P overlay has 10 random neighbors. The closer the curve to the lower-right corner, the bigger the search scopes are, and the better the performance of topology matching scheme is. Topology matching without and with Local Topology Cache have almost identical search scopes in this graph.

Figure 4.6: CDF of the Normalized Search Scope after Topology Matching

 normalize a mobile node’s average search scope with its average search scope when each peer in the P2P overlay has 10 random neighbors. Therefore a mobile node’s normalized search scope is 1 when each peer in the P2P overlay has 10 random neighbors. The reason to use normalized search scope is that we would like to use CDF graph to show the distribution of search scope in mobile nodes. However, if we need to compare 2 curves in the same CDF, the points with the same Y-coordinates in these 2 curves do not represent the same mobile node. Therefore we have to normalize the search scopes with that when each peer has 10 random neighbors, in order to compare the mobile node’s change of search scope in a CDF graph. We measured the normalized search scope after topology matching without caching, and topology matching with Local Topology Cache, and the result of 618 mobile nodes with at least 3 movements per day in average is shown in Fig. 4.6. We can find that topology matching only slightly shrinks the search scope. Less than 5% mobile nodes has a
normalized search scope less than 0.9. According to the findings of real P2P systems with 4,500 nodes in [40], almost all objects queried can be found after the query messages have reached 120 unique nodes. As above 90% of the 618 mobile nodes have search scopes of more than 120 nodes, this slight shrink of search scope is unlikely to cause any significant reduction in query success rate. Topology matching without and with Local Topology Cache have almost identical search scopes, as their curves overlap, which means caching does not shrink the search scope of a mobile node.

**Average Query Cost per Reached Node**

The average query cost per reached node is an important metric to evaluate the efficiency of topologically-matched P2P overlay. Here in the simulation the average query cost per reached node is the average number of overlay hops a TTL-4 query from a mobile node has to pass through to reach a unique node in its search scope. It is calculated as the overlay hops a TTL-4 query from the mobile node passes through, divided by the number of unique nodes in the mobile node’s search scope. Similar as search scope, we normalize a mobile node’s average query cost per reached node with that when each peer in the P2P overlay has 10 random neighbors. We measured the average query cost per reached node in the mobile node’s search scope, and compared topology matching without and with Local Topology Cache. The result of 618 mobile nodes with at least 3 movements per day in average is shown in Fig. 4.7. The average query cost per reached node is greatly reduced by topology matching compared to random neighbor selection. For about 80-90% of the mobile nodes, the query cost per reached node is less than half of that with random neighbor selection. This proves that topology matching can greatly reduce the traffic cost of queries. Topology matching with Local Topology Cache only has a slightly higher query cost per reached node compared with the case of no caching, which means caching still keeps the P2P overlay matched to the physical network topology.

**Average Neighbor Distance**

Average neighbor distance shows whether closer nodes are selected as P2P neighbors, and it is another metric to measure the efficiency of topologically-matched P2P overlay. The neighbor distance here is defined as the RTT between a pair of neighbors, which is ap-
The average query cost per reached node is the average number of overlay hops a TTL-4 query from a mobile node has to pass through to reach a unique node in its search scope. A mobile node’s normalized average query cost per reached node is 1 when each peer in the P2P overlay has 10 random neighbors. The closer the curve to the upper-left corner, the less average query cost is, and the better the performance of topology matching scheme is.

Figure 4.7: CDF of Normalized Average Query Cost per Reached Node after Topology Matching

proximated by the geographic distance between the access points they are associated with in the simulation. Again, the average neighbor distance of a mobile node is normalized with that when each peer in the P2P overlay has 10 random neighbors. We measured the normalized average neighbor distances after topology matching without and with Local Topology Cache. From Fig. 4.8 we can find that the average neighbor distance after topology matching is about 45-65% of that before it. This shows that closer nodes are indeed selected as P2P neighbors, and the average delay of P2P hops is greatly reduced. Topology matching without and with Local Topology Cache almost have the same average neighbor distance, again, it shows that caching keeps the P2P overlay matched to the physical network topology.
The average neighbor distance is the average RTT between a mobile node and its neighbors. A mobile node’s normalized average neighbor distance is 1 when each peer in the P2P overlay has 10 random neighbors. The closer the curve to the upper-left corner, the less average neighbor distance is, and the better the performance of topology matching scheme is.

Figure 4.8: CDF of the Normalized Average Neighbor Distance after Topology Matching

4.4 Summary

We have proposed a topology matching scheme with Local Topology Cache for P2P overlays of mobile nodes. The goal is to reduce the overhead of topology matching, while still making the P2P overlay match the physical network topology. In a highly dynamic P2P overlay of mobile nodes, a mobile node uses Local Topology Cache to store the information of its neighbors in a topologically-matched P2P overlay when visiting a subnet previously, and re-uses these neighbors directly when it returns to the same subnet and the Local Topology Cache hits. Topology matching operation is only performed when the cache misses. Based on a trace of real wireless network activities, we evaluated the performance of the proposed scheme with simulation. We found that topology matching with Local Topology Cache greatly reduced the overhead of topology matching, and still constructed a P2P overlay that matches the physical network topology. Compared to the P2P overlay constructed by random neighbor selection, it only slightly shrinks the search scope, but
greatly reduces the average query cost and average neighbor distance. Caching significantly reduces the overhead of topology matching, especially for mobile nodes with high-mobility, and its search scope, average query cost and average neighbor distance are about the same as topology matching without caching.
Chapter 5

Co-Location Patterns and Co-Location Prediction

5.1 Introduction

In last chapter we discussed the topology matching scheme with Local Topology Cache. This scheme is based on the assumption that each mobile node has movement pattern, and a mobile node’s neighbors in a P2P overlay that matches the physical network topology during the mobile node’s visit to a subnet are also those during its next visit to the same subnet. The effectiveness of this scheme was demonstrated by simulation based on traces of real-life wireless network activities, which also confirms that our assumption is valid. We would like to further investigate mobile nodes’ co-location behavior and look for insights that how co-location behavior can affect the topology of a wireless network. Co-location is defined as the scenario that two or more mobile nodes are within a specific distance from each other. In this dissertation, co-location is two or more mobile nodes are associated with the radio link of the same subnet at the same time.

It has been observed that the movements of mobile nodes have intrinsic patterns [6] [7] [8] as a mobile node is usually carried by an individual who has some extent of regularities in his activities. Furthermore, each individual also has regularities in his
interaction with other person at specific time and locations, and the mobile device carried by him has patterns of co-location with mobile devices carried by other people. Therefore co-location of mobile nodes is usually repetitive and predictable. There can be a distributed method to predict a mobile node’s co-locations with other nodes based on its co-location history information, and the method should also be able to dynamically adjust to the co-location history. The co-location prediction result can be utilized to construct a P2P overlay that matches the physical network topology similar as discussed in Chapter 4. In this case, there is no need to probe the network to discover the physical network topology, and it can significantly reduce the overhead to construct such a P2P overlay that matches the physical network topology. If the prediction is correct, a mobile node knows what nodes are residing in the same subnet, and can construct a P2P overlay that matches the physical network without probing the network. This is the gain of correct co-location predictions. If the prediction is not correct, the P2P overlay is constructed based on wrong assumption of physical network topology, which results in that the P2P overlay does not match the physical network topology. This is the penalty of incorrect co-location predictions.

Study of mobile nodes’ movement patterns can be based on synthetic movement models or models extracted from actual movement trace. A lot of synthetic mobility models have been proposed for simulation of wireless networks. They describe the aggregated behavior of mobile nodes, and each mobile node’s movement is random with some constraints. Most of them also focus on the modeling of the geographic movement of mobile nodes, such as the direction and velocity of the movement. Random Walk Mobility Model, Random Way Point Mobility Model [43], and Gauss-Markov Mobility Model [45] are some of the well-known examples. There are also some group mobility models proposed for the case that a mobile node’s movement is associated with that of other mobile nodes in a group, including Reference Point Group Mobility Model [46], Column Mobility Model, Pursue Mobility Model and Nomadic Community Mobility Model [47]. Little work has been done to validate these synthetic models with real-life network traces, therefore their applicability in a real wireless network is still questionable. Some models based on actual movement history focus on the logical movement of mobile nodes among different locations, and they can be categorized to Markov-family and compression-family. Markov-family models assume that each mobile node’s movement can be modeled by a Markov chain, while compression-family models such as LeZi-Update [48] utilize an incremental parsing algorithm by Ziv and Lempel [49], which is popular in text compression. Mobile nodes’ movements were also studied
with real network activity trace, and a few models were proposed according to the characteristics of mobile nodes’ movements extracted from these traces. All of the work referred above either focuses on the aggregated behavior of mobile nodes, or only investigates the movements of a single mobile node without considering the interaction among them, or clusters mobile nodes only using static group membership, which is not realistic in actual wireless network activities. To the best of our knowledge, there still lacks a method to dynamically describe the co-location patterns of mobile node pairs, and the possibility of co-location prediction has not been studied either.

In order to keep the amount of information exchanged among mobile nodes at a reasonable level, co-location prediction also has to be done in a fully distributed way based on local information available to a mobile node. If a mobile node has access to the entire movement history information of other mobile nodes, obviously it can make co-location prediction based on such information and its own movement history. But in this case each mobile node has to track the movement of all other mobile nodes in the system, the overhead needed would be prohibitively high, and the latency to disseminate movement information would also be huge. Therefore a mobile node has to predict its co-location probability with another node only according to its own movement history and its co-location history with that node, which is the partial movement history of that node the predicting node collected when co-locating and interacting with it previously. Compared to the existing movement predictors using entire movement history [20] [39], such constraint on partial movement history introduces great challenge to the design of co-location predictor.

In the previous work on topology matching with Local Topology Cache for P2P overlay of mobile nodes, we proposed a topology matching scheme to exploit the regularities in the movement of mobile nodes. A mobile node caches the information of its neighbors in a P2P overlay that matches the physical network topology, and uses this for topology matching. The predicting node always assumes that its neighbors in a topologically-matched P2P overlay during its last visit to a subnet will also be those when it returns to the subnet next time. Caching is essentially one of the simplest ways to predict future behavior based on history information. Although this method is effective for its own purpose, it is an over-simplified method and does not always work with predicting mobile nodes’ actual co-location behavior. In a particular subnet, a mobile node can co-locate with different sets of nodes at different time and occasions, which is determined by its regular activities with different sets of nodes. Using caching directly for co-location prediction still has a lot of
Based on these observations, we first investigate the co-location behavior of mobile nodes with real-life wireless network activity traces. We would like to find out how many nodes a mobile node can co-locate with, how many locations the co-location of a particular pair of nodes can occur at, how many mobile nodes can co-locate at a particular location, and how repetitive the co-location can be. Based on these findings, we evaluate the possibility to use co-location history information to predict future co-locations. Then we propose a co-location prediction method based on the mobile node’s own movement history and its co-location history with other nodes. Each mobile node tracks its co-location events with other mobile nodes, records them in the co-location history cache, extracts co-location patterns and stores these patterns into the pattern database. When the mobile node moves to a subnet it has visited previously, a Markov family model is used to predict the mobile nodes that will be co-locating with it at this location, according to the information in the co-location history cache and pattern database.

The rest of this chapter is organized as follows. Motivations and goals of the proposed method are presented in Section 5.2. Section 5.3 analyzes the characteristics of mobile nodes’ co-locations, and the proposed co-location prediction method is described in detail in Section 5.4. The performance of the co-location prediction method is evaluated in Section 5.5. The co-location prediction method is utilized to construct an efficient P2P overlay in Section 5.6. Section 5.7 summarizes this chapter.

5.2 Motivations and Goals

Our first motivation is to investigate the characteristics of mobile nodes’ co-location behavior, i.e., the scenario that multiple mobile nodes are residing at the same subnet at the same time. Movement of an individual mobile node has intrinsic patterns determined by the regularities in an individual person’s activity. Multiple mobile nodes can reside in the same subnet at the same time. This behavior may be repetitive and predictable, due to the repetitive activity of a group of people, and the group of people’s limited circle of friends and acquaintances. So far most work on mobility modeling and patterns only focus on the behavior of individual mobile nodes, and there has been little work on the co-location patterns of multiple mobile nodes. By using activity traces collected...
from real-life wireless networks, we would like to analyze the characteristics of mobile nodes' co-location behavior, to show that multiple mobile nodes have co-location patterns, and to prove co-location is repetitive and predictable.

To find a way to model co-location behavior and predict future co-locations is also one of the motivations of this research work. We are targeting at designing a fully-distributed co-location prediction method with good accuracy and reasonable overhead. The prediction should only be based on the partial movement history information of other nodes, which is the co-location history the predicting mobile node collected. Access to the entire movement history of other node can make co-location prediction much easier, but it takes a lot of network bandwidth and time to get the entire movement history of every node in the network. There will also be threats on privacy and security if a mobile node can collect the entire movement history of other mobile node. Only collecting partial movement history information for co-location prediction is likely to keep the amount of information exchanged at a low level, and will not compromise the privacy and security of mobile nodes.

To utilize co-location prediction in network management is the ultimate motivation of our research. There are many applications that can potentially benefit from co-location prediction result. We plan to first utilize the result of co-location prediction to construct unstructured P2P overlay of mobile nodes, in order to expedite the overlay construction process and reduce the overhead needed for network probing and topology discovery. Whether the P2P overlay matches the physical network topology has a key role in the overall performance of P2P overlay, therefore it is essential to construct a P2P overlay that matches the physical network topology. A mobile node can construct such a P2P overlay according to the result of co-location prediction, as co-locating nodes are closest to each other and should be neighbors in the P2P overlay. Instead of probing the physical network topology in the vicinity again and again to find good neighbors, a mobile node can simply predict which node is a good candidate of neighbor. Therefore a co-location predictor with good accuracy is likely to reduce the overhead of network probing and information exchange for overlay topology optimization, while the P2P overlay topology constructed can be as efficient as that of other topology optimization schemes.
5.3 Measurements and Analysis of Co-Location Behavior

Although there have been a lot of synthetic mobility models proposed, and a few of them do integrate inter-node locality [46] [47], it is always a question whether mobile nodes in real life move in the same way as described by these synthetic models. On the other hand, measurements and analysis of real life wireless network can provide insights about how mobile nodes actually move and interact. We base our analysis of mobile nodes' co-location behavior on the wireless network activity trace collected at the wireless campus networks of Dartmouth College [41] and UNC Chapel Hill [42]. The Dartmouth wireless network covers the whole campus of about 200 acres with 623 access points in more than 160 buildings. The wireless campus network at UNC Chapel Hill has 498 access points in 110 buildings. We would like to use multiple traces to avoid any bias that may be included in a single trace, and these 2 traces are the most comprehensive ones available right now.

In our investigation, we assume that each subnet only has one access point. Therefore co-location is as the event that multiple mobile nodes are associated with the radio link of the same access point at the same time. A 30-day trace from Dartmouth data is used in this investigation, and we tracked 1521 mobile nodes. For the UNC data, we used a 28-day trace and tracked 1208 nodes' movements. Both the traces show that mobile nodes have regularities in their co-location behavior.

5.3.1 Mobility Levels

Figure 5.1 shows the distribution of mobile nodes' number of movements. In Dartmouth trace, about 50% mobile nodes moved among access points for less than 60 times in 30 days, which translates to average 2 movements daily. About 18% mobile nodes are highly mobile and moved more than 300 times in 30 days. For the UNC trace, about 66% mobile nodes moved less than 56 times in 28 days, or twice daily in average, and about 14% mobile nodes moved more than 280 times in 28 days. Here we would like to categorize mobile nodes according to their level of mobility. Mobile nodes that moved less than twice a day in average are nodes with low mobility, those moved between 2 to 10 times a day in average are nodes with medium mobility, and those moved more than 10 times a day in average are nodes with high mobility.
5.3.2 Number of Access Points Visited

Generally an individual’s logical movements are associated with his intentions, and the individual visits a location with purpose. Since a mobile node’s movement has regularities determined by the regular activities of the person carrying it, usually a mobile node only visits a limited part of the whole wireless network. Therefore the mobile node only co-locates and interacts with other nodes at a limited number of access points. This means analysis and prediction of co-location behavior is much easier as the mobile node only need to be tracked at a limited number of locations. Figure 5.2 shows the distribution of the number of unique access points a mobile node visited during the traces. From this graph we can find that about 90% of the mobile nodes in Dartmouth trace only visited less than 17 out of the 623 access points, while in UNC trace, about 90% of the mobile nodes only visited less than 25 out of the 498 access points. Mobile nodes in UNC trace generally visited more access points than those in Dartmouth trace did, possibly because UNC’s wireless network has more access points in a building in average. In both traces, the number of access points a mobile node visited is usually less than 5% of the total number of access points in the wireless network. This proves that a mobile node only visits a few
The closer the curves to the upper-left corner, the greater potential of co-location prediction, as a mobile node usually only visits less locations, and there is less movement and co-location information to record.

Figure 5.2: CDF of Mobile Nodes’ Number of Visited Access Points

specific locations, determined by the regularity in the activity of the person carrying it. If we need to record the movement and co-location history information and do co-location prediction based on the history information, the amount of the information need to record is limited.

5.3.3 Number of Co-Located Nodes

We would also like to see how many unique nodes a mobile node can ever co-locate with at any of the locations in the duration of the traces. Generally prediction of future co-location behavior would be easier when the prediction is made out of a small set of nodes. If a mobile node indeed only co-locates with a limited set of nodes, it is possible to find a co-location prediction method that is low-cost and effective. Figure 5.3 shows the distribution of the number of nodes a mobile node co-located with. In Dartmouth trace, about 90% mobile nodes co-located with less than 90 nodes, while about 90% mobile nodes in UNC trace co-located with less than 270 mobile nodes. In both traces, generally a mobile
The closer the curves to the upper-left corner, the greater potential of co-location prediction, as a mobile node usually only co-locates with less nodes, and co-location prediction only is made from these nodes.

Figure 5.3: CDF of Number of Unique Nodes a Mobile Node Co-Located with at Any of the APs during the Whole Trace

node co-located with less than 1/4 of the whole node population, which is a good news for co-location prediction.

5.3.4 Number of Nodes a Mobile Node Co-Located with at an Access Point

We further investigate the number of unique nodes a mobile node can co-locate with at a particular access point in the duration of the traces, and speculate that the number should also be a small fraction of the total number of unique nodes it co-located with at any of the access points. Since a mobile node will possibly predict its co-locating nodes based on the information that which nodes it has co-located with at the same access point, such prediction will be easier if the mobile node has only co-located with a limited set of nodes at this access point previously. Figure 5.4 confirms our speculation. 90% of mobile nodes in Dartmouth trace co-located with less than 22 nodes at any access point, which is
The closer the curves to the upper-left corner, the greater potential of co-location prediction, as a mobile node usually only co-locates with less nodes at a particular location, and co-location prediction is made from these nodes.

Figure 5.4: CDF of Number of Unique Nodes a Mobile Node Co-Located with at a Specific AP during the Whole Trace

about 25% of the total number of co-located nodes. In UNC trace, 90% of mobile nodes co-located with less than 36 nodes, about 13.3% of the total number of co-located nodes.

5.3.5 Number of Access Points a Pair of Nodes Co-Located at

As a mobile node usually only visits a limited number of access points, the co-location of a pair of mobile nodes can only occur at even less access points, with which both of them are associated at the same time. The less number of access points a pair of mobile nodes can co-locate at, the easier to make co-location predictions. Figure 5.5 shows the distribution of the number of access points at which a particular pair of mobile nodes co-located. From this graph, co-locations of most pairs of nodes only occur at a specific access point, as around 70% of node pairs co-located at only 1 access point in both the Dartmouth and UNC trace. Generally a pair of mobile nodes in UNC traces co-located at more access points than that in Dartmouth trace, possibly because the UNC wireless
The closer the curves to the upper-left corner, the greater potential of co-location prediction, as co-locations of a pair of mobile nodes usually only occurs at less locations, and co-location prediction only need to be made at these locations.

Figure 5.5: CDF of the Number of APs a Pair of Mobile Nodes Co-Located at

network has a higher density of access points per building in average. This measurement proves that co-location of a pair of mobile nodes can only occur at a very limited number of locations, determined by the regularities in the interaction between the persons carrying them. It shows that co-location prediction can indeed be low-cost and effective.

5.3.6 Number of Co-locations between a Pair of Co-located Nodes

We also want to know how many co-locations actually occurred between a particular pair of co-located nodes, and how many of these co-locations occurred at a particular access point. These would indicate how repetitive co-locations can be and if there is potential to do co-location prediction. Figure 5.6 shows the distribution of the total number of co-locations between a pair of mobile nodes that ever co-located in the duration of the traces. Figure 5.7 shows the distribution of the number of co-locations between a pair of co-located nodes at a particular access point. The solid lines show the co-location numbers in UNC trace, and the dashed lines show the co-location numbers in Dartmouth trace. The
The closer the curves to the lower-right corner, the greater potential of co-location prediction, as a pair of mobile nodes usually have more co-locations, and co-location is more repetitive.

Figure 5.6: Number of Co-locations between a Pair of Co-located Nodes

The blue lines show the co-location numbers of all pairs of co-located nodes, while the red lines show the co-location numbers of the node pairs with one node of mid/high mobility.

We found that the majority of pairs of co-located nodes only co-located a few times, which means the co-locations are not that repetitive. However, the majority of mobile nodes in both traces are not that mobile either, which might be the reason why co-locations are not repetitive. From Figure 5.1, 50% - 60% of the mobile nodes moved less than twice a day in average. If one of these mobile nodes co-locates with another node, the number of their co-locations will be low. From Figure 5.6 and Figure 5.7, we can also find that a pair of nodes with one node of mid/high mobility is likely to have more co-locations than a pair of nodes with all levels of mobility. If both of the nodes in a pair have medium or high level of mobility, they will have even more co-locations. Therefore co-locations between nodes with medium or high mobility are more repetitive, and have the potential to be predicted. On the other hand, the repetitiveness of co-locations for mobile nodes with low mobility level is not as important, since these mobile nodes do not move frequently and their co-locating
The closer the curves to the lower-right corner, the greater potential of co-location prediction, as a pair of mobile nodes usually have more co-locations at a specific location, and co-location at a specific location is more repetitive.

Figure 5.7: Number of Co-Locations between a Pair of Co-Located Nodes at an AP

nodes do not change as significantly.

5.3.7 Summary of Co-Location Behavior

In this section we measured and analyzed the co-location behavior of mobile nodes. About half of all mobile nodes have low mobility, while only about 15% mobile nodes have high mobility. Generally a mobile node visits less than 5% of all access points, and co-locates with a small fraction of the total node population. At a particular access point, mobile node usually only co-locates with about 15% to 25% of the nodes it ever co-located with. A pair of co-located nodes only co-locate at a few access points. The number of co-locations between a pair of co-located nodes is limited, but if nodes with low mobility are filtered out, the co-locations are more repetitive. All of these show that a mobile node only co-locates with a limited number of nodes at specific access points. Co-locations are repetitive and have regularities, which provide the basis for co-location prediction.
5.4 Co-Location Prediction

5.4.1 Introduction

For co-location prediction, first we have to define the criteria to evaluate the co-location prediction method. Generally a prediction method can be evaluated by its accuracy and its cost. The accuracy can be measured by the miss rate of the co-location prediction method. Here a miss of co-location prediction is defined as the case that a node with which the mobile node actually co-locates is not any of the predicted nodes. The lower the miss rate, the more accurate the co-location prediction method is. The cost of co-location prediction has two parts, one is the cost of making prediction itself, the other is the cost of the application that uses co-location prediction result. If the prediction is correct, there is no need to probe the network for nodes at the same location, in this case the application using co-location prediction has 0 cost in probing the physical network. When there is a miss, potentially the application using co-location prediction has to probe the physical network, and the corresponding cost is the penalty of miss in co-location prediction. If the application chooses to use the co-location prediction result even though there is a miss, the penalty is the decrease of the effectiveness/efficiency of that application. Let $R_{\text{miss}}$ be the miss rate of co-location prediction, $C$ be the cost of the application when the co-location prediction hits, and $P$ be the penalty of that when there is a miss in co-location prediction, the application using co-location prediction has a utility function $U = P \cdot R_{\text{miss}} + C \cdot (1 - R_{\text{miss}})$.

The method that a mobile node finds its co-locating node is out of the scope of our research. Potentially the co-locating nodes are within the radio range of each other, and are able to hear each other’s associations with the access points at the location. Therefore the mobile nodes are able to track which nodes they have co-located at their current location. There might also be some external ways to find out co-locating nodes, for example, the access points can broadcast the identity of newly associated nodes to all the nodes in its coverage area. Here we assume each mobile node knows every node that it co-located with at the current location.

Before discussing any method of co-location prediction, we would like to discuss the naive co-location prediction methods. There are two questions to be answered. What is the best accuracy we can achieve in co-location prediction based on previous co-location history regardless of the cost? And what is the bottom line we can do with limit on cost?
If cost is not an issue, obviously the best we can do in enhancing co-location prediction accuracy is to predict every node the mobile node has co-located with at the same access point previously as the nodes the mobile node will be co-locating with this time. The only misses are those nodes the mobile node has not co-located with at this access point, which are compulsory misses based on co-location history. The bottom line of co-location prediction we can do with limited cost is to cache every node that the mobile node co-located with during its last visit to the same access point, and use them as the prediction result. The size of cache is infinite in this bottom line method.

Figure 5.8 shows the distribution of the miss rate of naive co-location prediction methods for a mobile node in Dartmouth trace, and Figure 5.9 shows that for a mobile node in UNC trace. We can find the compulsory miss rate for co-location prediction in both traces is rather low, which means it is possible to do co-location prediction with good accuracy. The miss rate of caching is much higher than the compulsory miss rate, and between them is the room for improvements on co-location prediction methods.

The closer the curves to the upper-left corner, the more accurate the prediction is. The area between the curves is the room for improvement with the co-location prediction method.

Figure 5.8: CDF of the Miss Rate of Naive Co-Location Prediction Methods with Dartmouth Trace
The closer the curves to the upper-left corner, the more accurate the prediction is. The area between the curves is the room for improvement with the co-location prediction method.

Figure 5.9: CDF of the Miss Rate of Naive Co-Location Prediction Methods with UNC Trace

5.4.2 Overview of Co-Location Prediction Method

When a mobile node visits a location, co-location prediction is performed to predict the nodes it will be co-locating with at this location, based on its co-location history at this location and its own movement trace. We would like to use a Markov-family mobility model in the co-location prediction method. The rationale of using Markov-family model is based on the observation that people usually have regular daily or weekly activities, and their movements tend to follow certain pathways. It has also been shown that low-order Markov-family predictors have accuracy as good as those of more complex compression-based predictors [39]. A mobile node’s co-locations with other nodes are assumed to be associated with their regular interaction. The regular interaction can be identified by a series of locations that the mobile node has visited right before the co-location. The series of locations visited can be modeled by a Markov chain, and a state in the Markov chain is the location that this mobile node is visiting. A sample state transition diagram is shown in Figure 5.10, in which a mobile node moves from $State_1$ (visiting $Location_1$) through $State_2$, 

![Cumulative Distribution](cumulative_distribution.png)
State\textsubscript{3}, State\textsubscript{4}, and goes back to State\textsubscript{1}. We assume the current state only depends on the recent \(k\) states, and the distribution of state is stationary, which means the distribution is the same as long as the recent states are the same. Therefore the mobile node’s co-locations with each of the co-located node is associated with this Markov chain.

Figure 5.10: Sample State Transition Diagram of a Mobile Node

Suppose the entire simulation area is \(S\), the entire node population is \(N\), and the predicting mobile node is \(N_p\). The node with which \(N_p\) co-locates at time \(t\) is a random variable \(C_t\), and the location \(N_p\) visits at time \(t\) is also a random variable \(X_t\). From the characteristic of Markov chain model we have

\[
P[C_t = n_{co} | X_1 = s_1, X_2 = s_2, \ldots, X_t = s_t] \\
= P[C_t = n_{co} | C_{t-k+1} = n_{co}, X_{t-k+1} = s_{t-k+1}, \ldots, X_t = s_t] \\
= P[C_{t+k} = n_{co} | C_{t+1} = n_{co}, X_{t+1} = s_{t-k+1}, \ldots, X_{t+k} = s_t]
\]

Time \(t - k + 1\) is the time when the most recent co-location between \(N_p\) and \(n_{co}\) occurred. This Markov chain model is of order \(k\). The probability of co-location with \(n_{co}\) at time \(t\) can be estimated by the number of times that co-locations with \(n_{co}\) occurred right after \(N_p\) moves through such length-\(k\) sequence of states previously.

\[
P[C_t = n_{co} | C_{t-k+1} = n_{co}, X_{t-k+1} = s_{t-k+1}, X_{t-k+2} = s_{t-k+2}, \ldots, X_t = s_t] \\
= M[n_{co} s_{t-k+1} \ldots s_t, s_{t-k+1} \ldots s_t n_{co}] \approx \frac{\text{Num}(s_{t-k+1} \ldots s_t n_{co})}{\sum_{n_a \in N} \text{Num}(s_{t-k+1} \ldots s_t n_a)}
\]

where \(\text{Num}(s_i \ldots s_j n_a)\) is the number of co-locations with \(n_a\) occurred right after \(N_p\) moved through the sequence of states \(s_i \ldots s_j\) in the history.

A mobile node performing co-location prediction records its own movement trace,
which is a sequence of the locations it just visited. It also extracts the characteristic movement tracks (CMTs) corresponding to a co-located node from its movement trace and stores them in a pattern database. For the mobile node making co-location prediction, a characteristic movement track corresponding to a co-located node is a sequence of locations visited by the predicting mobile node between two consecutive co-locations with this particular co-located node.

The model above does not take into account the timing information of movements and co-locations available to the predicting mobile node, and the Markov-family model can be extended by utilizing such information. Optionally a time stamp can be recorded when the predicting mobile node stores its movement history and co-location events with other mobile nodes. This extension increases the complexity of the co-location predictor, but is likely to give more accurate predictions.

5.4.3 Co-Location History Cache, Movement History Cache and Pattern Database

Each mobile node has an FIFO Co-Location History Cache of length \( l \) for each of the nodes it has co-located and interacted with, and stores the sequence of co-location events with a node in the Co-Location History Cache entry for this particular node. Suppose the predicting mobile node is \( N_p \), and one mobile node \( n_{co} \) with home IP address \( IP_{Addr_{co}} \) co-located with \( N_p \) at location \( s_i \) and time \( \tau_i \) previously, \( i \in [t - l + 1, t] \). Table 5.1 shows the content of Co-Location History Cache entry for \( n_{co} \).

<table>
<thead>
<tr>
<th>Co-Location History Cache Entry</th>
<th>Home IP Addr</th>
<th>Sequence of Co-Location Events with ( n_{co} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( IP_{Addr_{co}} )</td>
<td>( s_{t-l+1}, \tau_{t-l+1} )</td>
<td>( s_{t-l+2}, \tau_{t-l+2} )</td>
</tr>
</tbody>
</table>

A mobile node maintains an FIFO Movement History Cache of length \( m \) to store its own movement trace. If the mobile node visits location \( s_i \) at time \( \tau_i \), the content of its Movement History Cache is shown in Table 5.2.

A pattern database is also maintained at a mobile node for each node co-located previously to store the characteristic movement tracks that this mobile node moved between.
their co-locations. It also stores the occurrence count corresponding to each characteristic movement track. Each of the characteristic movement track is a sequence of locations visited by this mobile node, starting from a location that these two nodes co-located at and ending at the next location that they co-located at. Table 5.3 shows the content of the pattern database entry for a co-located node with home IP address $IP_{Addr_{co}}$.

<table>
<thead>
<tr>
<th>Home IP Addr</th>
<th>Characteristic Movement Tracks</th>
<th>Occurrence Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>$IP_{Addr_{co}}$</td>
<td>CMT1 $s_i, \tau_i$ $s_{i+1}, \tau_{i+1}$ $\ldots$, $s_j$, $\tau_j$ Count1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CMT2 $s_p, \tau_p$ $s_{p+1}, \tau_{p+1}$ $\ldots$, $s_q$, $\tau_q$ Count2</td>
<td></td>
</tr>
</tbody>
</table>

### 5.4.4 Pattern Matching Techniques

For the purpose of co-location prediction, a mobile node tries to correlate its current movement trace (sequence of locations just visited) with the characteristic movement tracks (sequence of locations visited previously and stored in the pattern database) of the nodes it has co-located with at the same location. Correlation is also used when a mobile node tries to extract new characteristic movement tracks from its movement trace and store them in the pattern database. Three types of matching techniques can be used here.

#### State Matching

State matching compares the locations in two sequences, and indicates the similarity of these two sequence of locations. Suppose the two sequences of locations have $m_s$ identical locations at the same position of the sequences out of $m_t$ locations in total, the state matching co-efficient $\mu$ is

$$\mu = \frac{m_s}{m_t}$$
The closer the state matching co-efficient to 1, the more similar that the two sequences of locations are. If $\mu$ is more than a state matching threshold, the two sequences are regarded as state matched.

**Occurrence Matching**

Occurrence matching is used to match two state matched sequences. It indicates how close the time interval between occurrence of the first sequence and the last occurrence of the second sequence matches the time intervals between any two consecutive occurrences of the second sequence. Occurrence matching captures the regularity in the intervals between two consecutive co-locations between a pair of mobile nodes. Let $t_n$ be the starting time of the first sequence, $\tau_l$ and $\tau_k$ be the time of the last and $k$th occurrence of the second sequence, we have the occurrence matching co-efficient $\Phi$

$$\Phi = \min_k \frac{|(t_n - \tau_l) - (\tau_{k+1} - \tau_k)|}{\tau_{k+1} - \tau_k}$$

The smaller the occurrence matching co-efficient, the more similar the occurrence intervals of the two sequences are. If $\Phi$ is less than an occurrence matching threshold, the two sequences are regarded as occurrence matched.

**Velocity Matching**

A third type of matching technique is velocity matching and is also used for two state matched sequences. It compares the velocity of movement among locations in two sequences of locations, and indicates the similarity of velocity in these two sequences. If $m_t$ is the total number of locations in each of the two sequences, $t_i$ is the time that the mobile node visited location $s_i$ in the first sequence, and $\tau_i$ is the time that the mobile node visited the same location $s_i$ in the second sequence, we have the velocity matching co-efficient $\eta$

$$\eta = \frac{\sum_{i=1}^{m_t-1} |(t_{i+1} - t_i) - (\tau_{i+1} - \tau_i)|}{m_t - 1}$$
The smaller the velocity matching co-efficient, the more similar the moving velocities in the two sequences are. If \( \eta \) is less than a velocity matching threshold, the two sequences are regarded as velocity matched.

### 5.4.5 Co-Location Prediction Process

As discussed above, a mobile node makes co-location prediction only based on its own movement trace and the co-location history at the same location. The number of the nodes that are predicted to be co-locating with the predicting mobile nodes is an adjustable parameter in the co-location prediction method. Generally the larger this number is, the less chance that a co-location prediction miss will occur. However, a large number of nodes as co-location prediction result is likely to increase the overhead of the application that uses the co-location prediction result. On the extremity, a mobile node can predict all the nodes it has co-located with at this location to be the nodes it will co-locate with this time, and the co-location prediction misses are only compulsory misses that cannot be avoided for a prediction method based on co-location history information. From our measurement of co-location behavior, a mobile node with medium or high mobility usually has 10 or more co-located nodes at a location it has visited, and this number is too much for many of the applications that can potentially use the co-location prediction result, such as the peer-to-peer network. Therefore there is a trade off between the co-location prediction miss rate and the extra overhead for the application that uses the co-location prediction result. Just a few would be a reasonable number for the nodes predicted to be co-locating with the predicting mobile node.

Here is a simple example of the co-location prediction process, shown in Figure 5.11. A square represents the mobile node is residing at the particular location, and a circle connected with the square represents the mobile node co-locates with the particular node when residing at this location. The mobile node visited \( s_1 \) and co-located with Node1 at time \( \tau_1 \), visited \( s_2 \) and co-located with Node1 at time \( \tau_2 \), then visited \( s_3 \) and co-located with Node2 at time \( \tau_3 \), after that visited \( s_4 \) again and co-located with Node1 at time \( \tau_4 \). Then it makes another movement at time \( \tau_5 \), from \( S_1 \) to \( S_2 \). Table 5.4 and Table 5.5 show the content of movement history cache, co-location history cache, and pattern database before and after the movement for this example.

When a mobile node is about to leave a location, it tries to extract characteristic
movement tracks. The following operations are performed for each of the nodes co-located at this location during this stay. In the example the co-located node at \( s_1 \) is Node1.

i) Find the location and time of the last co-location with the particular node before this stay from the corresponding co-location history cache entry. In the example, it is \((s_2, \tau_2)\).

ii) Extract the sequence of locations visited from the last co-location to the current location according to movement history cache. It is the current movement trace. In the example, it is \((s_2, \tau_2)(s_3, \tau_3)(s_1, \tau_4)\).

iii) Correlate the current movement trace to each stored characteristic movement track, first using state matching and optionally using occurrence and velocity matching. If there is a match, increase the occurrence count of the matched characteristic movement track by 1. Otherwise add the current movement trace into the pattern database as a new characteristic movement track. In the example, \((s_2, \tau_2)(s_3, \tau_3)(s_1, \tau_4)\) is a new characteristic movement track, and is added into the pattern database with occurrence count 1.
Table 5.5: Movement History Cache, Co-Location History Cache and Pattern Database after Movement

<table>
<thead>
<tr>
<th>Movement History Cache</th>
<th>Co-Location History Cache</th>
<th>Pattern Database</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Home IP Addr</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IP Addr</td>
</tr>
<tr>
<td>(s₁, ṭ₁) (s₂, ṭ₂) (s₃, ṭ₃) (s₁, ṭ₄) (s₂, ṭ₅)</td>
<td>(s₁, ṭ₁) (s₁, ṭ₄)</td>
<td>Characteristic Movement Tracks</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CMT#</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CMT₁</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CMT₂</td>
</tr>
</tbody>
</table>

When a mobile node enters a new location, it records the current time and location into the movement history cache (in the example, (s₂, ṭ₅)), and tries to make co-location predictions. It performs the following operations for each of the nodes co-located at this location previously. In the example, the co-located nodes at s₂ is Node1.

i) Find the location and time of the last co-location with this particular node from the corresponding co-location history cache entry. In the example, it is (s₁, ṭ₄).

ii) From the movement history cache, find the current movement trace, which is the sequence of locations visited from the last co-location to the current location. In the example, it is (s₁, ṭ₄)(s₂, ṭ₅).

iii) Correlate the current movement trace with the characteristic movement tracks of this particular node stored in the pattern database, first using state matching and optionally using occurrence and velocity matching. If there is a match, this particular node is a candidate co-locating node at this location, and it is recorded together with the occurrence count of the matched characteristic movement track. In the example, the current movement trace (s₁, ṭ₄)(s₂, ṭ₅) matches CMT₁ (s₁, ṭ₁)(s₂, ṭ₂) with occurrence count 1. Therefore Node1 is a candidate co-locating node.

If the number of candidate co-locating nodes is more than the number of nodes need to be predicted, the candidate co-locating node with the least occurrence count of the matched characteristic movement track is removed. This is done until the number of candidate co-locating node is no more than the number of nodes need to be predicted. Therefore nodes with higher co-location probability right after such sequence of visited locations are chosen.
to be the co-location prediction result.

5.5 Performance Evaluation of Co-Location Prediction

The co-location prediction method proposed is based on the assumption that co-locations between a pair of mobile nodes are associated with their regular interactions, which can be identified by the sequence of locations one of these mobile nodes visited right before the co-location. We have to evaluate its performance in a real wireless network scenario based on real-life network activity traces. Here we first discuss the criteria used to evaluate the performance of the co-location prediction method, then describe the scenario of simulation and evaluation, and at last present and analyze the simulation result.

5.5.1 Performance Evaluation Criteria

The two major criteria to evaluate any prediction method are accuracy and cost. Prediction miss rate is defined as the probability that an actual co-located node is not any of the nodes predicted to be co-locating, and indicates the accuracy of the co-location prediction method. We also have to pay attention to the percentage of compulsory misses in the total number of misses. It shows the room for improvement of the co-location prediction method, as we can never have a miss rate lower than the compulsory miss rate using a co-location prediction method based on history information. The cost of the prediction method includes the computation and storage overhead to collect/store movement and co-location history information, and the computation overhead to match mobile nodes’ recent movement trace with stored patterns.

5.5.2 Simulation Scenario

The simulation is still based on the real-life wireless network activity traces collected at Dartmouth College [41] and UNC Chapel Hill [42]. Both of these traces are collected from wireless campus networks having hundreds of access points and thousands of mobile nodes, which provide an ideal and realistic environment for the simulation and evaluation of the co-location prediction method. We simulated the co-location behavior of 1521 mobile nodes in a 30-day Dartmouth trace and 1208 mobile nodes in a 28-day UNC
trace. Each access point is regarded as a location. All the associations with an access point for less than 3 seconds are filtered out to avoid the bias that might be introduced by the ping-pong effect due to fluctuation in wireless link.

5.5.3 Simulation Results

When discussing the performance of the co-location prediction method, we have to keep in mind that the performance is related to the mobile node’s level of mobility. If a mobile node is almost non-mobile, the number of its co-locations with other nodes will be low, and it is difficult to do co-location prediction as there has to be a certain number of previous co-location to extract characteristic movement tracks and train the predictor. Co-location prediction is not as important for almost non-mobile nodes, as they do not move frequently and their co-located nodes do not change as much. Therefore we would like to discuss the accuracy of the co-location prediction method according to the mobile node’s level of mobility.

In the simulations all three types of pattern matching techniques are used, and we found the parameters shown in Table 5.6 are reasonable choices for pattern matching parameters, and give a satisfactory co-location prediction miss rate. Here we present the simulation result based on this combination of parameters. We simulated the co-location prediction method with 3, 4, or 5 nodes predicted to be co-locating in each prediction.

<table>
<thead>
<tr>
<th>state matching threshold</th>
<th>occurrence matching threshold</th>
<th>velocity matching threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.95</td>
<td>0.2</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Figure 5.12 shows the distribution of the co-location prediction miss rate of the mobile nodes in Dartmouth trace. These mobile nodes have mid or high mobility levels and moved at least twice per day in average. There are 752 such mobile nodes. The miss rates of the co-location prediction method with 3, 4 or 5 as the number of nodes predicted to be co-locating are shown in the graph, and compared to the miss rate of caching and the compulsory miss rate. Co-location prediction with larger number of predicted nodes have smaller miss rate, but not significantly better. The miss rate of co-location prediction is lower than that of caching. For example, about 75%-80% of these mobile nodes have a miss
rate of 0.4 or less with co-location prediction, but with caching, only about 46% mobile nodes have a miss rate of 0.4. This is a good indication of the accuracy of the co-location prediction method, as usually the number of nodes cached is larger than the number of nodes predicted. The curve of co-location prediction is still a bit far away from the curve of compulsory misses. However, one should also note that the number of nodes predicted is significantly less than the number of co-located nodes in the compulsory miss case. Another possible reason why the miss rate of co-location prediction is not exceptionally low is that a lot of mobile nodes are still not mobile enough. Mobile nodes that have high level of mobility moves more than 10 times a day in average, and the distribution of their co-location prediction miss rate is shown in Figure 5.13. There are 267 nodes with high level of mobility. Compared to Figure 5.12, caching performs worse, with around 70% mobile nodes have miss rate only less than 0.62. However, co-location prediction performs better, with around 70% mobile nodes have co-location prediction miss rate less than 0.3. This shows co-location prediction will perform better if nodes are more mobile.

The closer the curve to the upper-left corner, the less the co-location prediction miss rate, and the more accurate the co-location prediction method is.

Figure 5.12: Miss Rate of Co-Location Predictions for Mobile Nodes with Mid/High Mobility in Dartmouth Trace
The closer the curve to the upper-left corner, the less the co-location prediction miss rate, and the more accurate the co-location prediction method is.

Figure 5.13: Miss Rate of Co-Location Predictions for Mobile Nodes with High Mobility in Dartmouth Trace

Simulation result with UNC trace is similar. Figure 5.14 and Figure 5.15 shows the distribution of co-location prediction miss rate for 416 nodes with medium/high level of mobility and 160 nodes with high level of mobility respectively. The co-location prediction method has reasonably low miss rate, especially for mobile nodes with high level of mobility. For co-location prediction with 3, 4 or 5 predicted nodes, the co-location miss rate is less than 0.3 for 70% mobile nodes. The miss rate curve of co-location prediction in UNC trace is also closer to the curve of compulsory miss compared to that in Dartmouth trace. The possible reason might be nodes in UNC trace are more mobile.

From these graphs, we can find that the co-location prediction has reasonably good accuracy, especially for mobile nodes with high level of mobility. For mobile nodes with low mobility level, the miss rate of co-location prediction is high. However, co-location prediction is not very important for them anyway, since they do not move frequently and there is no frequent and significant change in co-locating nodes.

The computation and storage overhead of the co-location prediction method is also
The closer the curve to the upper-left corner, the less the co-location prediction miss rate, and the more accurate the co-location prediction method is.

Figure 5.14: Miss Rate of Co-Location Predictions for Mobile Nodes with Mid/High Mobility in UNC Trace

very reasonable. It took a Pentium IV 2.8GHz workstation a few hours to simulate a trace around 30 days with 1200-1500 mobile nodes, and the total memory consumption for the 1200-1500 nodes is about 1.2-1.5GB, which is about 1MB per mobile node. This co-location prediction method is fully distributed, as a mobile node only need its own movement trace and information of its previous co-locations with other nodes. Therefore the computation and storage overhead is very affordable for today’s mobile devices. Furthermore, co-location prediction is not a time critical or real-time requirement, a few seconds is more than enough to a perform co-location prediction.
The closer the curve to the upper-left corner, the less the co-location prediction miss rate, and the more accurate the co-location prediction method is.

Figure 5.15: Miss Rate of Co-Location Predictions for Mobile Nodes with High Mobility in UNC Trace

5.6 P2P Overlay Construction with Co-Location Prediction

5.6.1 Introduction

To have a P2P overlay topology that matches the physical network topology, an ideal P2P overlay should be constructed in the way that a node chooses peers nearby instead of those far away as its neighbors. Here the distance between a pair of peers is the RTT between them. Existing topology matching techniques construct such P2P overlay by probing the physical network and removing neighbors that cannot form a P2P overlay that matches the physical network topology. When a mobile node is co-locating with other nodes, these co-located nodes are by far the closest nodes to the mobile node, as they are essentially in the same subnet and the distance between them is only the RTT of an intra-subnet link. Therefore these co-located nodes are the natural choices of neighbors for the mobile node in a P2P overlay. If the mobile node can predict the nodes it will be co-locating with, the mobile node is able to construct a P2P overlay that matches the physical
network topology without probing the network. If the prediction is correct, the mobile node can construct a topologically-matched P2P overlay without probing the network, therefore reducing the overhead for P2P overlay construction. This is the gain of using co-location prediction result to construct P2P overlay. If the prediction is not correct, the mobile node constructs the P2P overlay with an incorrect assumption of the physical network topology, the result is that the P2P overlay is not as topologically-matched as expected. This is the penalty of prediction miss. Our target is to eliminate the need of network probing when constructing a P2P overlay that matches the physical network topology. Meanwhile, the P2P overlay constructed has to be topologically-matched, as much as those with existing topology matching techniques do. The degree that the P2P overlay matches the physical network topology is evaluated by search scope, average query cost per reached node, and average neighbor distance, as discussed in Section 4.3.

We have proposed a co-location prediction method based on a mobile node’s own movement trace and its co-location history with other nodes, and demonstrated its effectiveness using real-life wireless network activity traces. In this section, we will utilize the co-location prediction method to construct a P2P overlay that matches the physical network topology, without probing the network. This is also a demonstration of one application of co-location prediction.

5.6.2 P2P Overlay Construction Based on Co-Location Prediction

A mobile node records its own movement trace and its co-location history with other nodes, and predicts its future co-locations based on this information as described in Section 5.4. When a mobile node moves to a location, it first removes all of its neighbors in the P2P overlay. Then it predicts the nodes that it will be co-locating with at this location, and sets up virtual connections to them and adds them as its neighbors in the P2P overlay. Since these predicted nodes will be at the same location, the mobile node also adds a couple of other nodes not at the same location as its neighbors, which is used to provide connections to other locations. Therefore the P2P overlay topology is like what in the small-world model. Nodes form small local clusters so that nodes are well connected within a cluster, but only sparsely connected across clusters.

When choosing P2P neighbors, a mobile node always gives higher priority to nodes that it has co-located with. After adding nodes predicted to be co-locating as neighbors, it
first chooses neighbors from nodes which it has co-located with but are not predicted to be co-locating this time. These nodes provide connections to the outside of the mobile node’s local cluster. Only if such nodes are not available, the mobile node can add a random node as neighbor, or find a neighbor through some external way.

Compared to other topology matching techniques such as those described in Chapter 4, constructing P2P overlay based on co-location prediction is likely to have a smaller average neighbor distance, as most of the neighbors of a mobile node are predicted to be at the same location. However, it does not open the loops in the P2P overlay as the topology matching operation does in Section 4.2.4. Therefore a mobile node is likely to have a smaller search scope compared to that with other topology matching techniques, as it queries can reach the same node via multiple overlay paths. However, the biggest advantage of constructing P2P overlay based on co-location prediction is that it does not need to probe the network at all. Topology matching in a mobile environment need a lot of overhead to probe the physical network since the nodes are moving and the topology of the physical network changes frequently. As discussed in Section 4.2 and 4.3, with topology matching the probing messages generated by a mobile node typically have to pass through anywhere from 2 to 100 overlay hops per minute to probe the network, sometimes the number of overlay hops the probing messages pass through can even get close to that for querying the P2P overlay. This is a huge overhead for P2P overlay of mobile nodes, as radio link only has limited bandwidth and mobile devices have limited power supply from batteries. Constructing the P2P overlay based on co-location prediction totally eliminated the need to probe the network, therefore reduces the overhead of overlay construction, and saves the network bandwidth and mobile devices’ battery power.

5.6.3 Performance Evaluation

We simulated the construction of P2P overlay based on co-location prediction using the real-life wireless network activity trace from Dartmouth College [41], and compared the P2P overlay topology with those using topology matching techniques. With topology matching, a mobile node first finds 10 random nodes as neighbors, and then removes the neighbors that cannot form a topologically-matched P2P overlay using the method described in Section 4.2. With co-location prediction based P2P overlay construction, a mobile node predicts 5 nodes as those it will be co-locating with, and adds them as its neighbor. It also
adds other nodes to provide connections outside of the local cluster, and its co-located nodes have higher priority to be its neighbors. For comparison purpose, P2P overlay construction based on co-location prediction has about the same number of neighbors per mobile node in average as that of topology matching techniques. The distance between two mobile nodes, which is defined as the RTT in this dissertation, is approximated by the geographic distance between the access points they are associated with.

Similar as in Section 4.3, the performance of constructing P2P overlay based on co-location prediction is evaluated by its overhead and the degree that the P2P overlay matches the physical network topology. Because it does not need to probe the network, compared to topology matching techniques described in Chapter 4, it has 0 overhead for network probing and greatly reduces the communication overhead for P2P overlay construction. The computation and storage overhead is also very affordable for mobile devices. Here we would like to focus on the following metrics to evaluate the efficiency of the P2P overlay topology.

**Search Scope**

The search scope is defined as the number of unique peers a TTL=4 query from a mobile node can reach using flooding. Here flooding is that a mobile node sends the query to each of its neighbors, and the neighbor also forwards the query to all its neighbors except the on the query came from, until the query’s TTL=0. Each peer only forwards the query once, which means the peer will drop the query it already forwarded, even if the query’s TTL is larger than 0. The larger the search scope, the less hops needed to search for an object in the P2P overlay, and the more the P2P overlay matches the physical network topology. Figure 5.16 shows the distribution of search scope of a mobile node in the P2P overlay constructed based on co-location prediction. Similar as in Section 4.3.3, we also normalize a mobile node’s search scope with that when each peer in the P2P overlay has 10 random neighbors for comparison purpose. The distribution of the normalized search scope in P2P overlays constructed based on co-location prediction and topology matching techniques is shown in Figure 5.17. The search scope of a mobile node in the P2P overlay constructed based on co-location prediction is about 15-25% smaller than that with topology matching techniques. Since P2P overlay construction based on co-location prediction does not open loops in the P2P overlay, this is expected. However, this shrink of search scope would
not significantly reduce the probability of reaching an object in the P2P overlay within 4 overlay hops. According to [40], in a P2P overlay of 4500 users and 850,000 files with Zipf distribution, most query targets can be located if the query has reached 120 unique nodes. As shown in Figure 5.16, about 80% of the mobile nodes in the P2P overlay constructed based on co-location prediction have a search scope of more than 120 unique nodes, and for these nodes most of the objects in the P2P overlay can be reached within its search scope.

![Cumulative Distribution](image)

Figure 5.16: CDF of Search Scope of the P2P Overlay Constructed Based on Co-Location Prediction

**Average Query Cost per Reached Node**

Average query cost per reached node is the average number of overlay hops a TTL=4 query from the mobile node has to pass through to reach a unique peer in its search scope. It is calculated as the number of overlay hops a TTL=4 query from the mobile node passes through divided by the number of unique peers in the mobile node’s search scope. The less it is, the more the P2P overlay matches the physical network topology. A mobile node’s average query cost per reached node is normalized with that when each peer in the P2P overlay has 10 random neighbors. The distribution of normalized average query cost per reached node in the P2P overlays constructed based on co-location prediction
The search scope is the number of unique peers a TTL=4 query from the mobile node can reach. A mobile node’s normalized search scope is 1 when each peer in the P2P overlay has 10 random neighbors. In the P2P overlay constructed by co-location prediction, 5 nodes are predicted to be co-locating nodes, and each mobile node has about the same number of neighbors as that in topology matching for comparison purpose. The closer the curves to the lower-right corner, the bigger the search scope, and the more topologically-matched the P2P overlay is.

Figure 5.17: CDF of a Mobile Node’s Normalized Search Scope in the P2P Overlays

and topology matching techniques is shown in Figure 5.18. We can find that co-location prediction reduces average query cost per reached node for about 50% compared to random neighbor case, and performs about as good as topology matching.

**Average Neighbor Distance**

Average neighbor distance shows whether nodes closer in the physical network are actually neighbors in the P2P overlay. The lower the average neighbor distance, the more the P2P overlay matches the physical network topology. There the distance between two peers is the RTT between them, and is approximated by the geographic distance between the access points these nodes are associated with. The P2P overlay constructed with co-location prediction should have a smaller average neighbor distance than existing topology
Average query cost per reached node is the average number of overlay hops a TTL=4 query from a mobile node has to pass through to reach a unique peer. A mobile node’s normalized average query cost per reached node is 1 when each peer in the P2P overlay has 10 random neighbors. In the P2P overlay constructed by co-location prediction, 5 nodes are predicted to be co-locating nodes, and each mobile node has about the same number of neighbors as in topology matching for comparison purpose. The closer the curves to the upper-left corner, the less the average query cost per reached node is, and the more the P2P overlay matches the physical network topology.

Figure 5.18: CDF of a Mobile Node’s Normalized Average Query Cost per Reached Node in the P2P Overlays

matching techniques. The reason is that the co-locating nodes are closest to each other as they are associated with the radio link of the same access point. Again, a mobile node’s average neighbor distance is normalized with that when each peer in the P2P overlay has 10 random neighbors. The distribution of normalized average neighbor distance in P2P overlays constructed with co-location prediction and topology matching techniques is shown in Figure 5.19. We can find that co-location prediction significantly reduces average neighbor distance, and for about 50% of mobile nodes, its average neighbor distance is less than half of that using existing topology matching techniques.

Overall, P2P overlay construction based on co-location prediction greatly reduces average neighbor distance compared to topology matching techniques, and achieves about
A mobile node’s average neighbor distance is measured by RTT, and approximated by the geographic distance between the nodes. The normalized average neighbor distance is 1 when each peer in the P2P overlay has 10 random neighbors. In the P2P overlay constructed by co-location prediction, 5 nodes are predicted to be co-locating nodes, and each mobile node has about the same number of neighbors as in topology matching for comparison purpose. The closer the curves to the upper-left corner, the less average neighbor distance, and the more the P2P overlay matches the physical network topology.

Figure 5.19: CDF of a Mobile Node’s Normalized Average Neighbor Distance

the same average query cost per reached node. It has slightly smaller search scope, but that does not significantly reduce the probability to locate an object. It totally eliminates the need to probe the physical network, therefore saves wireless network bandwidth and mobile device’s battery power.

5.7 Summary

In this chapter we investigated mobile nodes’ co-location behavior. Co-location is defined as an event that multiple mobile nodes are associated with the radio link of the same subnet at the same time. A mobile node is usually carried by a person who has regularities in his activities, therefore a mobile node has movement patterns associated with
these regularities. As a person also has regularities in his interaction with other individuals, a mobile node has pattern of co-location with other nodes.

We measured the co-location behavior of mobile nodes in real-life wireless networks with the network activity trace collected at Dartmouth College and UNC Chapel Hill. The result shows that a mobile node usually only visits a small part of the whole network, and co-locates only with a small fraction of the whole node population. The co-location of a pair of mobile nodes only occurs at a few particular locations, and at a specific location a mobile node only co-locates with a few nodes. The co-locations are repetitive for mobile nodes with medium or high level of mobility. All of these show that there is potential to predict co-locations.

Based on these observations, we proposed a fully distributed method for co-location prediction based on a Markov family model. The co-location between a pair of mobile nodes at a particular location is associated with their regular interaction, identified by a sequence of locations one of the nodes visited between co-locations. A mobile node extracts characteristic movement tracks right before its co-locations from its movement trace and stores them in a pattern database. To predict future co-locations, it uses pattern matching techniques to correlate its current movement trace to the stored characteristic movement tracks, and uses the nodes corresponding to the correlated characteristic movement tracks as the prediction output. Simulation with Dartmouth trace and UNC trace shows that this co-location prediction method has good accuracy, especially for mobile nodes with higher level of mobility. The cost of co-location prediction is also affordable for mobile devices.

This co-location prediction method has the potential to be used in various applications to optimize the network topology. We utilized co-location prediction when constructing the P2P overlay. It eliminates the need to probe the network, saving wireless network bandwidth and mobile devices’ battery power. P2P overlay constructed based on co-location prediction matches the physical network topology. It has smaller average neighbor distance, about the same query cost and only slightly smaller search scope compared to those constructed using topology matching techniques.
Chapter 6

Conclusions and Future Work

This research is based on the observation that the movements of individual mobile nodes have intrinsic patterns determined by individual person’s regular activities, and the co-location events of multiple mobile nodes also have specific patterns caused by the regularities in the interactions inside groups of people. Mobile nodes are able to predict their future behaviors according to the history information, and such prediction results can be utilized to expedite network management processes and reduce the associated overhead.

In Chapter 3, we presented a fast handoff scheme for wireless IP networks with prediction of next subnet. Each mobile node records its movement history information including the ID of the subnets it was connected to, the IP addresses of the foreign agents, and the start time of those connections. When about to leave a subnet, it predicts the next subnet using pattern matching techniques based on its movement history, and explicitly notifies the current foreign agent to duplicate and forward packets to the predicted subnet. Therefore the latency and number of packet loss of handoff are greatly reduced, only with a very moderate overhead in packet duplication and forwarding compared with other fast handoff mechanisms. Simulation with real-life wireless network trace shows that next-subnet prediction has low miss rate, and average latency of network-layer handoffs is reduced to less than 1/3 of Mobile IP handoff latency, while the overhead of packet duplication and forwarding is very limited.

A topology matching mechanism with Local Topology Cache was proposed for P2P overlay of mobile nodes in Chapter 4. The performance of unstructured P2P overlay
is determined by the degree that the P2P overlay matches the physical network topology. In a P2P overlay of mobile nodes, which has great dynamics in physical network topology, existing topology matching mechanisms have to periodically probe the network, which incurs excessive overhead. Since mobile nodes have movement patterns and co-location patterns, it is possible that a mobile node’s neighbors in a P2P overlay that matches the physical network topology when visiting a subnet previously, are still those should be in a topologically-matched P2P overlay when the mobile node returns to this subnet. Therefore the mobile node can cache the information of its neighbors in a topologically-matched P2P overlay, and reuse them without probing the network again. We used a real-life wireless network trace in the simulation, and found that the Local Topology Cache can significantly reduce the overhead of topology matching. Local Topology Cache has a fairly low miss rate for highly-mobile nodes. The P2P overlay constructed by the caching mechanism matches the physical network as much as that using existing topology matching techniques, while the overhead of network probing is greatly reduced.

We investigated the co-location behavior of mobile nodes in Chapter 5 and found that mobile nodes’ co-location is repetitive and has patterns. A mobile node usually only visits a small fraction of the whole network, and co-locates with a small fraction of the whole node population. Co-location between a pair of nodes only occurs at a couple of locations, and at a particular location only a limited number of pairs of nodes can co-locate. A Markov-family model was used to describe the characteristics of co-location events, and a fully distributed co-location prediction method was proposed based on a mobile node’s movement trace and its co-location history information. We simulated the co-location prediction method with real-life wireless activity traces, and evaluated its accuracy and cost. The proposed method has good co-location prediction accuracy, and its computation and storage overhead is still affordable for a mobile node. The co-location prediction result was used in a P2P overlay of mobile nodes to optimize the P2P overlay topology. A mobile node chooses the predicted co-locating nodes as its P2P neighbors, to form small local clusters that are well connected within. It also chooses a couple of other nodes not at the same location as neighbors to provide connections across the local clusters. To construct the P2P overlay with co-location prediction has no overhead for network probing, while other topology matching techniques need a significant amount of overhead to probe the network topology. The P2P overlay topology constructed with co-location prediction result matches the physical network topology. Compared to other topology matching techniques,
it achieved significantly less average neighbor distance and almost the same average query cost, with only a slightly smaller average search scope that does not reduce query success probability much.

Our future work includes utilizing co-location prediction in other applications, such as information dissemination. We also plan to analyze the characteristics of the communities formed by dynamically clustering mobile nodes according to co-location patterns, and look for connections between characteristics of communities and those of individual mobile nodes.
Bibliography


