Techniques of Distributed Caching and Terminal Tracking for Mobile Computing

By

Chiu-Fai Fong

Supervised By :

Prof. John C.S. Lui & Prof. Man-Hon Wong

Submitted to the Division of Computer Science & Engineering
in partial fulfillment of the requirements for the
degree of Master of Philosophy
at the
Chinese University of Hong Kong
June 1997

Techniques of Distributed Caching and Terminal Tracking for Mobile Computing

submitted by

Chiu-Fai Fong

for the degree of Master of Philosophy
at the Chinese University of Hong Kong

Abstract

In the near future, mobile computing systems are expected to support a large number of mobile users with diverse network services. However, due to physical constraints, the limitation in wireless communication bandwidth between the base stations and the mobile computers becomes more severe. Moreover, the rapid growth in the scale of mobile computing systems results in substantial protocol overheads. This may offset other computation and communication activities.

To cope with these problems, we present a novel distributed caching protocol that can reduce the wireless bandwidth consumption as well as improve the response times for data accesses. We also propose a new mobile terminal tracking protocol that can track the mobile computers more effectively. Using these protocols, the contention of the wireless communication channel can be lowered. Therefore, more mobile computers can be supported inside the network and the system capacity is increased.

Our distributed caching protocol is different from the other techniques in the way that we take advantage of the dissemination power of the wireless channel. Unlike the techniques employed in conventional distributed systems, we also address the issues that the resources, specifically the main memory, at the mobile computers are also limited. There are other broadcasting protocols that minimize the response times for data accesses. On the other hand, they cannot balance the tradeoffs between the needs of different mobile computers. It is found that, by doing so, the contention at the wireless channel can be further reduced.

One of the main contributions in our work on mobile terminal tracking is that we propose a new general framework that can model a large class of movement patterns of mobile computers. This framework facilitates us to compare different tracking protocols and address the impacts on them under various movements patterns. Different from the previous work, we also discuss the time-varying property of the location of

Acknowledgments

I would like to thank John Lui and Man-Hon Wong, my thesis supervisors, for their patience, guidance and support. Their creative, insightful comments and experience have provided me with precious help to my thesis. More importantly, I am eternally grateful for their encouragement and spiritual support. These two years should not have been that fruitful without them.

Also, I would like to express my sincere thanks Leana Golubchik of Columbia University who has helped me proofread one of my conference paper, which is now part of my thesis.

Thanks to Edmundo A. de Souza e Silva of the Federal University of Rio de Janeiro for his insightful opinions regarding the mobile terminal tracking problem. His comments have helped me a lot in designing the mathematical framework.

In addition, I would to extend my cordial thanks to Tony Lee who has taught me a lot on other subjects and helped me out during busy periods.

Special thanks to the Croucher Foundation for the Croucher Foundation Studentship. With their generous financial support, I have gained more time and resources in undergoing my research during my graduate study.

Lastly, I am also in debt to my colleagues, Mary Leung, Alan Tung, Peter Lie, Steve Wong, Carson Cheng, Lai-Man Wan, Kelvin Law, Sze-Kin Lam, Dilys Hung, Chan-Man Kuok, Elton Tsang and Shing-Kwong Cheung. They have been resourceful in providing me technical and non-technical information, fun, moral support and other help in these two years.

a mobile computer and exploit it to get a more accurate prediction on the location of the computer. Using this prediction scheme, we present a novel tracking protocol that can minimize the broadcast cost as well as update cost. These all make the terminal tracking, an indispensable process in cellular network, more efficient.

Keywords: Mobile Computing, Cellular Network, Data Caching, Terminal Tracking, Performance Evaluation

This is a Master's Thesis submitted to the Division of Computer Science and Engineering at the Chinese University of Hong Kong on June 27, 1997, in partial fulfillment of the requirements for the degree of Master of Philosophy in Computer Science and Engineering. This thesis is supervised by Prof. John C.S. Lui and Prof. Man-Hon Wong, who are Assistant Professors at the Division.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this thesis, we propose two methods to enhance the system capacity and effectiveness of mobile computing systems by reducing the consumption of system resources, especially the wireless communication bandwidth. First, we present a novel distributed caching protocol for the base station and the mobile computers. This protocol exploits the dissemination power of the wireless channel to deliver information to several mobile computers at the same time. Moreover, it manages to balance the tradeoffs among the data requests from different mobile computers.

Another contribution of our work is to propose a new and general method that minimizes the protocol overhead for mobile terminal tracking. This includes a new prediction scheme that can predict the current location of a mobile computer at a higher accuracy. Using this result, we devise an optimal paging algorithm that searches for a mobile computer at the lowest cost while satisfies the delay constraint at the same time. Lastly, we also present a new updating protocol that is more adaptive than other existing protocols under different user movement patterns.

The personal communication network (PCN) is the underlining communication network system to support mobile computing. It is an integrated, heterogenous system of the wired and the wireless networks, which allows users to move around different locations while maintaining their connectivity with the network system. An overview on the architecture and related issues about this system will be addressed in Chapter 2.

The remainder of this chapter discusses the related work and elaborates our contribution in this thesis. To begin with, we discuss related work and current issues on distributed data caching in Section 1.1. Also, the outline of our approach is presented. Afterwards, we discuss mobile terminal tracking problem and briefly introduce our work in this problem in Section 1.2. Lastly, we present an overview of this thesis in Section 1.3.

1.1 Distributed Data Caching

Information retrieval is one of the standard services provided by computer networks. Therefore, the personal communication system is expected to provide connected users with access to a large amount of information. It is often accomplished by using a database management system inside the network.

Database management systems have been studied widely in the past decades [KS91]. Recently, there are also many research projects on the distributed database management systems. In these systems, data are replicated and stored inside a network of computers so as to reduce the database access times, enhance the data availability and increase fault tolerance under the distributed system environment [BHG87].

Subsequently, with the introduction of client-server systems, data caching techniques are developed to shorten the data access delays at the database client side [FC92]. In these systems, there are abundant amount of system resources like communication bandwidth, disk storage, and memory. The main objective of caching is to reduce the data access rate to the server by rerouting some of the data accesses to the clients' local cache. This is achieved at the expense of other system resources such as disk space at the client side and communication bandwidth. Therefore, the main approach to do caching is to let clients cache all the data that they might access in the future, without being concerned about the exhaustion of system resources.

The personal communication system (PCN) is a new kind of client-server system, in which the mobile computers act as clients to access information at the wired network system. However, there are so many differences between the PCN and the conventional client-server system that deeper investigation is essential to exploit its full potential.

Figure 1.1 shows the general configuration for database access in a mobile computing environment. In this environment, a database management system is running inside the wired network. The mobile computers access the database by sending requests through the base station and wait for the reply.

In general, the mobile computers suffer from resource limitation. They have limited memory, storage, power supply, and so on. More importantly, the wireless communication channel has a much lower bandwidth than a communication channel in the wired network. This will become the bottleneck of the whole personal communication system if the wireless communication bandwidth is not carefully managed. In view of these constraints, our primary objective of using distributed caching is not to improve the performance of data access, but to reduce wireless channel contention.

Research results of data caching in wireless network environments have been re-

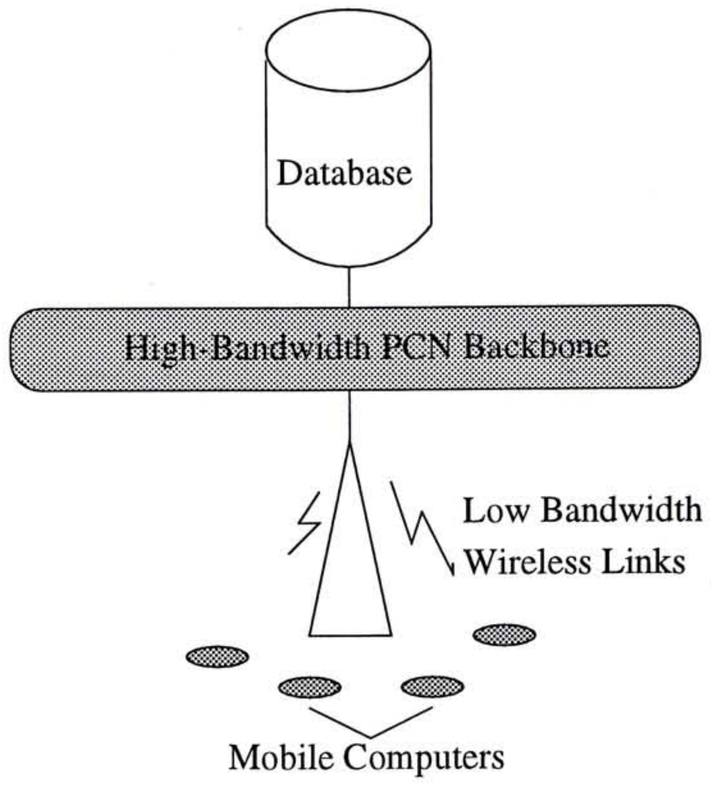


Figure 1.1: The Personal Communication Network Architecture

ported in [HSW94, HW94]. In that work, Huang *et al.* present two cost models for wireless communication, namely the message-based model which counts the number of message exchanges, and the connection-based model which depends on the sizes of the messages sent. Based on these two cost models, a dynamic caching protocol is proposed to decide when a data object should be cached at a mobile computer so as to lower the communication cost. In that protocol, a data object will be cached at a mobile computer if, based on the most recent access history of the mobile computer to that object, the expected cost for read is higher than that for writes, and *vice versa.*

This protocol, however, is analyzed only from the individual mobile computer's point of view. Furthermore, they have not addressed the wireless channel as an effective medium to broadcast information to many mobile computers simultaneously. They have also taken no consideration on the storage limitations of mobile computers. This turns out to be a significant problem because these factors affect the cache selection criteria and thereby the performance of the data access.

There are also other research projects which explored the dissemination feature of the wireless communication. In [BI94, JBEA95], Barbara, Imielinski and Jing *et al.* focus on how the cache invalidation information can be organized and delivered so as to reduce the wireless communication overhead and the power consumption at the mobile computers. The general approach is to summarize the database updates during a past period of time, and then create a forward index for the invalidation information. Nevertheless, the servers have no knowledge of what data the mobile computers are really caching. Therefore, the server needs to broadcast all the data updates. This is wasteful because not all data changes are useful to the mobile computers within a particular cell of a wireless network.

In [AAFZ95, AFZ96, AFZ95], Acharya *et al.* propose to exploit the wireless broadcast as an effective medium to deliver frequently accessed information. They mainly focus on how to organize the data into the broadcast channel so as to achieve a minimum expected time for a mobile computer to access a desired piece of data. The organization depends on the access patterns of the mobile computers to the database. In a nutshell, the more frequently the data is accessed, the more frequently this data is broadcasted. However, as different mobile computers may have different access patterns, some computers might access intensively some data that are not frequently broadcasted by the server. In this situation, the access delay may be very large. To cope with this problem, the authors propose a cache management algorithm at the mobile computers that caches data with the largest access rate to broadcast frequency ratio. However, they have not addressed the update problem in their protocol. Also, they have not addressed that some data should be delivered through the individual channels rather than using the broadcast channel.

One of the main contributions of our work is to propose a novel and effective distributed caching strategy that takes the advantage of the broadcast nature of the wireless communication effectively. Our distributed caching algorithm can reduce the overall bandwidth consumption of the wireless network system and, as a result, the effective capacity of the system is increased. Moreover, we find that by using our proposed strategy, the expected response times for data accesses are further improved. All these attractive gains make the entire system more cost-effective.

It is important to note that in our proposed strategy, we address the problem of selecting a subset of frequently accessed data objects to disseminate by the base station such that caching can be efficiently supported. Therefore, this strategy can be used in conjunction with other techniques, such as those mentioned above [AAFZ95, BI94, JBEA95], to achieve a superior performance.

We first propose a simple protocol, which can be readily extended. In this protocol, the base station periodically selects a subset of data and broadcasts it to all mobile computers within a cell. Upon receipt of the broadcast, each mobile computer will cache a subset of the broadcasted data, based on its individual data access pattern. Therefore, there are two selection processes in this protocol, one at the mobile computers and one at the base station.

If a data object is cached by a mobile computer, all read accesses by this computer can be satisfied by the local cache. On the contrary, if there is any change to this object, the base station has to send an update to the mobile computer. This results in a communication overhead. As a result, caching a data object may not always result in a reduction in communication overhead. It depends on the rate at which the mobile computer reads the data object, and the rate at the data object is updated. In this regard, the objective of the mobile computer is to select a subset of data objects into its limited cache such that the communication overhead is minimized.

The situation at the base station is similar, but with more complications. If the base station broadcasts a data object and some mobile computers cache it, then a reduction in communication can be achieved if the data object is accessed more frequently by those mobile computers than being updated. However, if there is only a few mobile computers which will cache that data object, there will be less reduction in communication overhead; or the overhead may even be higher. The interesting problem is how the base station can know what mobile computers will cache the data object. Given an answer to this problem, the base station will then need to select a subset of data objects to broadcast so as to achieve a maximum saving in communication.

In this thesis, we prove that the two selections are both NP-hard. We therefore propose heuristic algorithms for both sides so as to achieve near-optimal performance. It is shown by experiments that our proposed strategy can achieve a significant improvement in the overall system performance of the wireless network. Detailed discussion will be carried out in Chapter 3.

1.2 Mobile Terminal Tracking

In the mobile computing environment, the mobile computers will release their wireless connections with the base station when communication is no longer needed. This can save power consumption at the mobile computers and also release the precious wireless communication resource for other computers. Nevertheless, if a mobile computer goes to a different cell, the personal communication network will have no information about this movement. In other words, the location information of this mobile computer at the wired network will become out-dated.

This brings the need for *mobile terminal tracking*, which is a mechanism to enable the PCN to locate a disconnected mobile computer and then re-establish a connection to it. The mobile terminal tracking problem has been studied extensively [BNKS94, BNK93, MHS94, Ros96, AH95b] with the target at minimizing the consumption of system resources, especially communication overhead. This is important because the limited system resources are shared by a large number of mobile computers. An effective utilization of these resources can increase the system capacity and also enhance performance.

Currently, a protocol for mobile terminal tracking is adopted inside the European GSM standard and the IS-41 standard in the United States. Under these standards, a cellular network is statically partitioned into a number of subareas. Each subarea contains a number of cells in vicinity with each other. A mobile computer will register with the wired network the subarea where it is currently situated. This subarea is also known as the *registration area* for the computer. The whole architecture for mobility management in as shown in Figure 1.2.

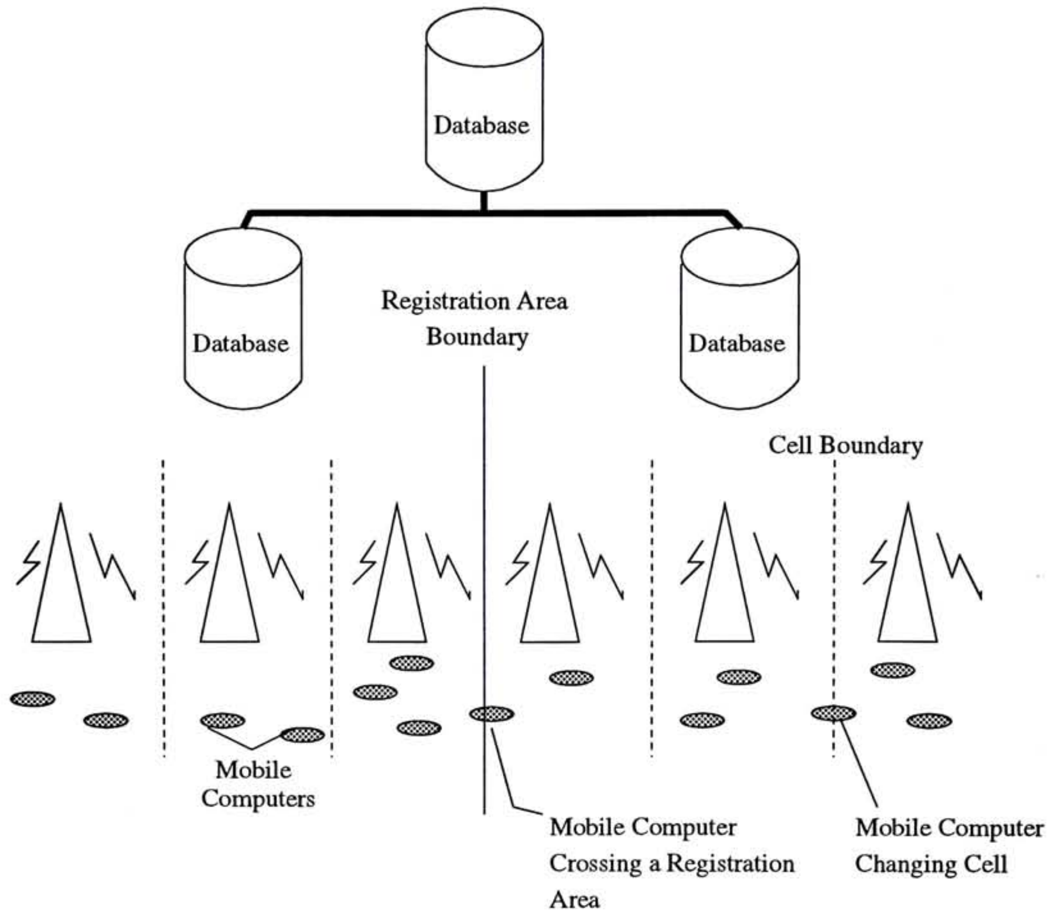

Database
Database
Registration Area Boundary
Database
Cell Boundary
Mobile Computers
Mobile Computer Crossing a Registration Area
Mobile Computer Changing Cell

Figure 1.2: System Architecture for Mobility Management

There are two types of database management system to maintain the location information of the mobile computers, namely the Home Location Register and the Visitor Location Register. In each personal communication network, there is logically one

Home Location Register (HLR) to store the most up-to-date location and user-related information about all the mobile computers. This database is often implemented in a distributed manner over the whole wired network so as to improve the efficiency and load balance of the system. The *Visitor Location Register* (VLR) is implemented inside each registration area. It is used to cache location, authentication, and user access control information for all mobile computers inside its managing area. In GSM, the VLR communicates with the HLR through Signalling System Number 7 (SS7) protocol, which is commonly used in ISDN and current public networks.

If a mobile computer moves from one registration area to another, it will send a location update message to the VLR in the new registration area. The VLR will then contact the HLR for an update. The HLR update its database, notify the old VLR of the mobile computer to delete its old record, and then send an acknowledgment back to the new VLR. Therefore, a location update consumes wireless communication bandwidth, database update overhead and the system signalling cost inside the wired network.

This configuration and protocol, however, provides potential performance problems. First of all, the protocol involves a lot of signalling in the wired network. Moreover, the two-level architecture of the location database system are not effective. There are many research efforts to enhance this architecture [JLLM94, HA95a, LL96, PA96, HS96]. In this thesis, on the other hand, we will look into another problem on mobile terminal tracking. We will look into the problem of location update and paging, which consumes the wireless communication resource.

Whenever a mobile computer moves from one registration area to another, an update message will be sent. If a mobile computer frequently moves across a boundary between two adjacent registration areas, unnecessary overhead is incurred. The reason for this problem is that the boundaries for the registration areas are fixed.

In order to alleviate this problem, other dynamic approaches to assign the registration area are proposed [AH95b, BNKS94, MHS94, TA96]. Rather than fixing the boundaries of the registration areas, one alternative is to draw a new registration area for each update such that the reporting cell will be the center of the new area. This registration area is specified in terms of a number of *rings* of cells around the last reporting cell. When a mobile computer leaves its registration area, it means that it has left its last reporting location for a certain distance. Then, an update message has to be sent. It is called the distance-based protocol [AH95b, MHS94].

Apart from the distance-based protocol, another similar type of protocol, called the time-based protocol [BNKS94, Ros96], is introduced. In this protocol, a specific time limit is set. If a mobile computer has not sent any location information for this time limit, it will send an update message to the network system.

In the above various update protocols, the mobile computer does not need to send update messages for every change in its cell location. As a result, the wired system does not know the exact location of the computer. Whenever the wired system tries to establish a communication channel with the computer, it has to find out where the computer currently is. This involves a *paging* process in which the base stations send polling messages to the cells in the network and wait for the reply of the target computer.

One simple method is to page all the cells in the network at the same time. However, it results in the largest system overhead in wireless communication. Another method is to page cells in two phases. In the first phase, the registration area of the mobile computer is paged first. If the computer cannot be found, other cells will be paged in the second phase. Other paging methods have also been proposed in [AGG96, Ver96, RY95]. These paging methods are based on the probability that the mobile computer being called for will be inside a particular cell at a certain time. These new methods can induce lower system overhead.

In [AH95b, BNKS94, MHS94, Ros96], Akyildiz, Bar-Noy, Madhow and Rose have done mathematical analyses to show the performance of the tracking protocols. In [AH95b, BNKS94], however, they assume that the user has a uniform movement pattern, *i.e.* the user moves from one cell to one of the neighboring cells with even probability and at the same movement rate. Moreover, they use steady state analysis to quantify the performance of their protocol, which is only valid for their uniform movement model. These assumptions may be not be general enough to incorporate realistic situations. A user usually has a unique, non-symmetric movement pattern over the cellular network. In addition, the cost of paging depends on the actual movement of the mobile user. As a result, steady state analyses may not be appropriate.

Besides the time-based and the distance-based protocols in which the mobile computer decides to send updates spontaneously, there is another category of location update protocol described in [LW95]. In that work, Lee and Wong suggest the wired network to periodically broadcast queries to request for the replies from a specified subset of mobile computers. If a mobile computer belongs to the subset, it will respond to the base station by replying *yes* or *no*, which can be encrypted in just one bit. By making a series of different queries, the wired network will get the location information of the mobile computers at a high accuracy. In this way, the update overhead and the paging cost are lowered.

In a nutshell, the problem of mobile terminal tracking can be divided into the following orthogonal problems [RY95]:

1. How to derive reliable time-varying *location probabilities* for a mobile computer, *i.e.* the probability that the computer is currently in a particular cell at a specific time?

2. Given these location probabilities, what is the optimal paging method such that the overall cost is minimized?

3. Given a paging method and the associated costs, what is the optimal update protocol for the mobile computers?

One contribution of this thesis is to develop a mathematical framework that enables us to model general user movement patterns inside network. This model is more general than the one proposed in [AH95b] because it allows non-uniform movement of a mobile terminal inside the cellular network. This is much more realistic as under practical cases where users have general habitual movement patterns.

This location model is composed of a Markov process whose states represent the time-varying locations of a mobile computer. The rate of state change are the rate at which the computer moves from one cell to another cell. Using this model, we use the uniformization method [Jen53, dSeSG90], which is well-known for its efficiency and numerical stability, to obtain the time-varying state probabilities of the Markov process. By using this method, prediction on the time-varying locations of a user can be more accurate. This allows us to achieve a better minimization on the cost of paging. Furthermore, our model is not just for performance evaluation on the protocols, but also efficient to use in actual implementation.

By using these location probabilities, we devise an optimal paging algorithm which pages for a mobile computer at the minimum system overhead while satisfies certain delay constraint at the same time. This problem has been investigated in [AH95b]. However, since they have assumed the uniform movement pattern, their protocol may not be able to perform well at general movement patterns. In [RY95], the authors has modeled this problem into a constrained optimization problem. However, it is difficult to know the exact complexity for solving the problem. Our method, on the other hand, uses a dynamic programming approach to obtain an optimal paging protocol under different movement patterns. Therefore, we will also quantify the complexity of the algorithm.

Another important advantage of this model is to provide a unified framework to compare the performance of different tracking protocols. Under this framework, we can also analyze the impacts of different movement patterns to the protocols. In Chapter 4, we will derive performance measures for both the time-based method and the distance-based method.

Last but not the least, we have proposed a new protocol, the reverse-guessing protocol. This protocol uses another heuristic based on the prediction result we can obtain in our proposed method. We find that our method outperforms the time-based and the distance-based methods. The main advantage of this protocol is that our reverse-guessing protocol is more adaptive in choosing the right time for the mobile computer to send an update message under general movement patterns.

1.3 Thesis Overview

In this chapter, we have introduced the contribution we present in this thesis, which are distributed data caching and mobile terminal tracking. In Chapter 2, we introduce the personal communication network, which is the underlining network platform for the mobile computing environment. We also discuss the issues and difficulties related to the implementation of such system, namely the resource limitations and mobility of mobile computers. In Chapter 3, we will investigate the use of distributed data caching as a way to solve the contention at the wireless channel. This is done by proposing a protocol that broadcasts data objects and their changes to the mobile computers. In Chapter 4, we discuss the issues on the mobile terminal tracking. We also propose a new protocol and can further lower the system cost to do terminal tracking. In addition, performance analyses and experiments are carried out throughout this thesis to show the performance of our protocols. We conclude our work in Chapter 5.

Chapter 2

Personal Communication Network

Mobile computing has generated much attention in recent years. With the rapid advances in network technologies, the integration of wired and wireless network systems brings the convenience for mobile users to access online information seamlessly while they are moving around different places. This new kind of systems are expected to support a large amount of mobile users in the near future. Moreover, it will support a wide variety of network services, from conventional communication services like telephony, facsimile and teletex, through ordinary computer network activities like electronic mail, remote computer access, file transfer and world-wide-web access, to advanced, communication-intensive applications like video conferencing, and image and video retrieval [MGTH95, MJSL96].

In this chapter, we introduce the personal communication network [LQ95, DN95, FZ94], which is the underlying network system to support the appealing mobile computing environment. In Section 2.1, we present the general architecture of the system. After that, we discuss issues on two unique characteristics of this system that differ itself from the conventional wired networks. We will look into resource limitations and mobility of mobile computers in Sections 2.2 and 2.3 respectively.

2.1 Network Architecture

The personal communication network (PCN) is an integrated system of wired and wireless networks. The wired network forms the backbone of the whole system, which is a heterogeneous network of high speed, integrated service sub-networks: asynchronous transfer mode (ATM) networks, public switched telephone networks (PSTNs), integrated service digital networks (ISDNs), fast Ethernet, etc. The wireless sub-networks—such as wireless local area networks (wireless LANs), wireless public branch exchange (wireless PBX), satellite systems, and so on—provide the communication channels between the wired backbone and the mobile computers (or terminals). The overall architecture of a PCN is illustrated in Figure 2.1.

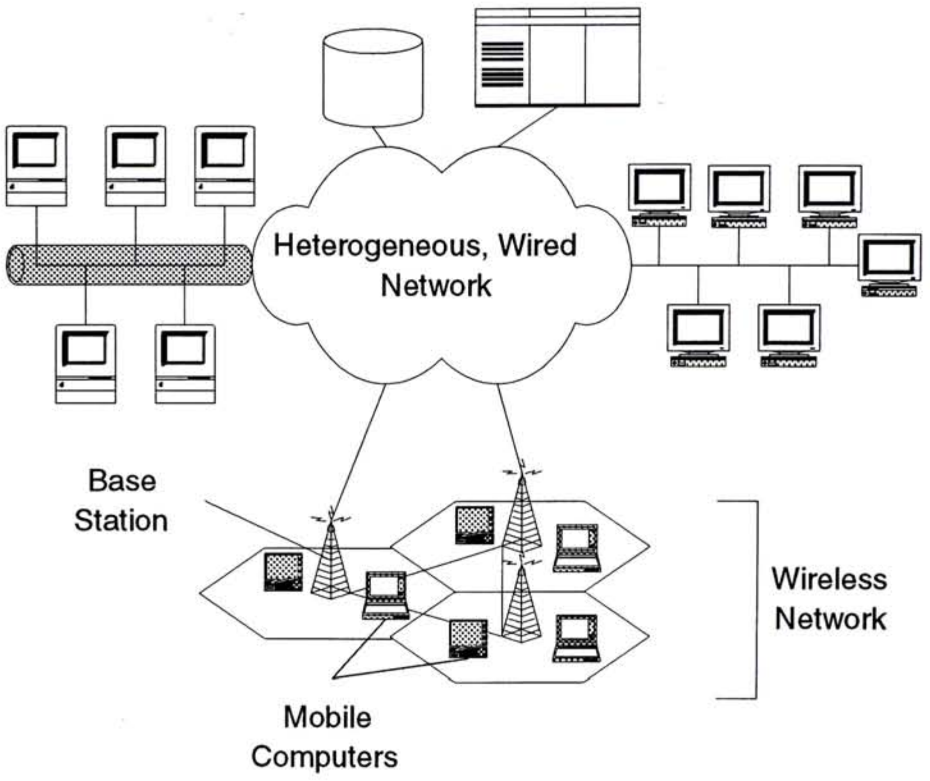


Figure 2.1: The Personal Communication Network Architecture

In general, the wireless communication system employs a cellular architecture. In this configuration, the geographical area covered by the wireless networks is partitioned into *cells*, each of which is managed by a base station. The base station is a static computer with wireless communication capability, which is responsible to gateway messages between the wired network system and the mobile computers within its cell. The major advantage for this cellular architecture is that it allows *spatial frequency reuse*: the same electromagnetic frequency spectrum for the communication can be used again within other cells under the condition that the cells are so distant apart that their communication will not interfere each other.

However, a mobile computer may not be able to connect to the base station at all time even if it is located inside the corresponding cell. This is due to the physical characteristics of electromagnetic waves, traffic congestion at the wireless channel, or high mobility of mobile computers (as will be discussed later). In order to support a more reliable physical connection, *multi-tier* cellular systems are employed. In addition to the ordinary cell partitions, these systems also contain *macrocells* that cover the same geographic area of the ordinary cell partitions. On the other hand, each macrocell covers a larger area and so, it may span over many ordinary cells. It is commonly supported by satellite communication and provides alternative communication channels to the mobile computers so that more reliable communication can be obtained.

In order to support multiple mobile computers in a cell at the same time, some multiple access control schemes are employed at the base station. Currently, some common methods include digital sense multiple access (DSMA) in cellular digital packet data (CDPD) systems, time-division multiple access (TDMA) and frequency-division multiple access scheme (FDMA) in Global System for Mobile Communication (GSM) systems, and code-division multiple access (CDMA) schemes. These schemes divide the wireless communication resource along different domains (time, frequency and code) into numerous, independent communication channels. Each mobile computer will be allocated one or more channels for the communication with the base station, in accordance with the communication requirement of the mobile computer, the channel resource allocation schemes and the channels allocated to other mobile computers inside the cells nearby [PSS95].

Besides those separate channels for each mobile computer, some channels are also reserved for the base station to disseminate control signals and frequently requested information to multiple mobile computers at the same time. All the mobile computers will listen to these channels as well as those allocated for their own use. With this broadcast feature, information can be delivered efficiently to any number of mobile computers within a cell. This broadcast feature is one of the advantages for wireless network systems. By managing the broadcast channels effectively, the system can be scaled up easily.

2.2 Resource Limitations

Despite the above advantage, on the other hand, the wireless system suffers from various resource limitations. One critical limitation is the overall channel bandwidth for wireless communication. To illustrate, the bandwidth of conventional wired networks ranges from $10Mbps$ (Ethernet) to $650Mbps$ (ATM network), while that of wireless networks ranges from $9.6kbps$ (GSM system) to $2Mbps$ (wireless LAN). This difference is due to the physical constraints of electromagnetic waves, the range of spectrum allocated for communication, and the drastically increase in demand for more separate channels to support the exponentially growing number of subscribers to the system. Therefore, wireless communication becomes the contention point for the whole PCN.

To account for portability, mobile computers also suffer from other resource limitations. Examples are power supply, CPU processing power, main memory and secondary storage [KKT93]. Although the advances of hardware technologies have lessened these limitations substantially, there is still a wide gap between mobile computers and the desktop computers. Compared with the personal computers or workstations, the mobile computers is only capable of doing simpler jobs in terms of both computation and storage requirements.

Power management policies are also employed to cope with the power limitation problems. Under these policies, if a mobile computer is idle for a certain period of time, it will go into *sleep* or *doze* mode [FZ94] in which the harddisk will be slowed down or even turned off; the liquid crystal display (LCD) will be off; and the CPU will run at a lower speed. This mechanism can lower the power dissipation at various components of a mobile computer. Also, wireless communication will become semi-functional or even in the off state. In this case, the mobile computer will no longer be connected to the base station at all time. Extra measures, which will be discussed later in this chapter, are required to ensure efficient and correct information exchange.

Besides all these limitations, there are always new applications with higher resource requirements. As a result, the resource management problem in PCNs becomes very challenging. In this thesis, we provide insights to these problems and address on how the scarcity of various resources can be eased by distributed caching.

2.3 Mobility

Another unique characteristic of cellular networks is the mobility of mobile computers. As we have mentioned before, the geographic region covered by the cellular network is divided into many cells. All the communication of mobile computers inside a cell is managed by a specific base station. However, because a mobile user is allowed to move freely inside the whole geographic area, he/she may go through many different cells during the movement. In this case, the mobile computer must switch from the old base station to a new one inside the new cell in order to maintain the ongoing connection with the wired system.

The switch over of an ongoing connection between different base stations is known as the *handover* (or *handoff*) process [Pol96]. This process can be triggered by different kinds of events. Usually, it can be triggered by a drop in the power of the signal from the current base station and a rise in the power of another signal. It involves acquiring a free channel (or several channels) from the new base station, releasing the old channel (channels) to the old base station, and registering the new location information with the *location management system*. All these procedures require resource overheads for control signalling, packet reroute [IDM95, BB94, Joh94, PB94], and database updates in location information.

Apart from the resource overhead, handover may also induce a short disconnection to switch over a new base station. In the worst case, it may also result in a call termination if free channels are not available in the new cell. Moreover, a disconnection may also occur when the mobile computer stays inside a cell because of the wave properties of electromagnetic medium. On the other hand, this disconnection is predictable because the drop in power of a signal is gradual rather than abrupt. This kind of connection with frequent but expected disconnection is termed *weak connectivity* [SKM$^+$93]. It is very different from a low-bandwidth, reliable connection in the way that, for weak connectivity, the channel is usually unreliable. Special measures are necessary to handle the recovery of some network functions, such as network file system and database transactions [SKM$^+$93, MES95, LS95], after connection failures.

Another anomaly that comes with the mobility of mobile computers is the consistency problem for data retrieval [BHG87]. In an ordinary distributed system, the global clock is implemented by a distributed synchronization protocol [Lam78]. However, due to the communication delays between different nodes of the system, the local clocks of these nodes are likely to be different in a particular physical time. As the wired network inside the PCN is also a distributed system, the base stations may also have different local clocks. In the distributed database aspects, which use a similar approach for data updating, the snapshots of replicated data among different base stations may also be different. Consider the case that a mobile computer acquires some data from one base station and then moves to a neighboring cell of a *smaller* logical clock value. Then, if that mobile computer acquires some other data with the new base station, there may be inconsistent among the data inside the local cache of the mobile computer. As a consequence, computation errors or transaction conflicts may occur if strict data consistency is required.

The main reason behind this distributed system anomaly is the dynamic changes in the overall connectivity between the mobile computers and the wired network. Several approaches are proposed to ensure correct distributed system semantics in the PCN. In [AB96, IA94], the authors propose an exactly-once multicast protocol. They have also propose a method in capturing distributed snapshots by causal ordering in [AB94b], and a checkpointing method in [AB94a]. Moreover, in [WL95], a multiversion approach is also proposed to solve the consistency problem.

Besides the handover problem, problems also arise when the mobile computers move across different cells when the connections have not been established. In order to save the power consumption at the mobile computers and increase the availability of wireless channels, a mobile computer will release its connection with the base station if the communication activity is low. When the connection is off, the wired system is no longer able to know the exact location of the mobile computer inside the cellular system.

During this disconnection, the wired system may attempt to contact the mobile computer. For example, another mobile computer wants to establish a connection with the computer; or the base station wants to send messages to that computer. In this situation, it needs to find out the exact location of the mobile computer. This is done by sending polling messages to the cellular system and waiting for the response from the mobile computer. This is called the *paging* process. One simple paging method is to send polling messages to all the cells at the same time. By doing so, we can achieve a minimum delay. On the other hand, as signals are sent all over the cells, the communication overhead will be very large. As mentioned above, the wireless communication bandwidth is very limited. It is of crucial importance to minimize paging cost since the number of mobile computers in the network is expected to be large.

In view of this problem, *mobile terminal tracking* is introduced to lower the system cost by allowing the mobile computers to send their location information to the wired network system periodically. In this way, the wired system would then have some hints on the approximate location of a mobile computer. The broadcast procedure is then modified to send polling messages first to the cells where the target mobile computer is likely to be. In this manner, the expected polling cost is lowered, but another cost for the mobile computers to send update messages is induced. This protocol, therefore, inherits another issue on how to balance the tradeoffs with those two costs. Moreover, it also induces processing overhead at the wired network to manage the location information of the mobile computers inside the cellular network. This will be elaborated in Chapter 4.

Chapter 3

Distributed Data Caching

In this chapter, we investigate the use of data caching as a way to reduce the wireless communication overhead for data delivery in a personal communication network. We propose a strategy that uses the broadcast channel efficiently to support caching at the mobile computers.

In our strategy, the mobile computers will access a database of objects. The base station will choose a particular subset of data objects and support caching of them at the mobile computers. The mobile computers can then choose some of those objects and cache them. As the support of caching incurs a different communication overhead, we need to manage the selections both at the base station (for broadcast) and mobile computers (for caching) carefully.

We will show that the selection processes to minimize the overall wireless communication are NP-hard. As a result, we propose heuristic algorithms to tackle the problems. We also analyze and perform experiments to address the performance of our algorithms and compare them with a conventional client-server caching scheme.

We find that our strategy can lower wireless bandwidth required for information delivery. Moreover, this strategy operates at a near-optimal performance under different system configurations. This increases the precious wireless bandwidth available for other uses. In this way, the system capacity can be increased and thereby the system cost is reduced. Lastly, we find that our protocol can also improve the response time for users' requests. This makes the whole system more efficient and cost-effective.

In the coming section, we present our system model that conceptualizes the wireless network environment, the broadcast protocols and its associated costs. In Section 3.2, we discuss the cache selection problem at the mobile computers. After that, the selection of data objects for broadcast at the base station are presented in Section 3.3. Performance analyses and experiments are done in Sections 3.4 and 3.5 respectively.

Lastly, we will discuss on the mobility issues and their impacts of the distributed caching protocol in Section 3.6.

3.1 System Model

In this section, we introduce our system model and the configuration of the wireless network environment (see Figure 3.1), which is a cellular network running a distributed database management system. There is a stationary server (base station) in each cell which is responsible for processing all requests from mobile computers within the cell. All these stationary servers are connected via a wired network. Below, we illustrate the caching protocol and the cost model which is based on the bandwidth consumption in the wireless network system.

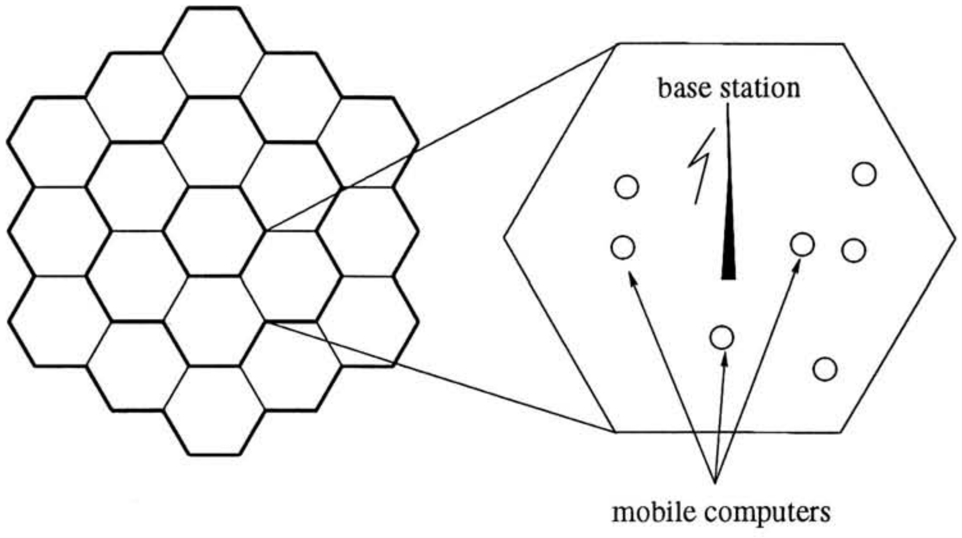


Figure 3.1: Wireless Network Environment for Data Caching

3.1.1 The Wireless Network Environment

As illustrated in Figure 3.1, a wireless network is composed of stationary servers, which are responsible for processing transactions generated by mobile computers within their own cells. The servers themselves form a wired distributed system in which a fully-replicated database resides. The database comprises a set $\mathcal{D}$ of N data objects, $o_1, o_2, \ldots, o_N$. Each object o_j is of a different size s_j (in bytes). We assume that the servers can maintain the consistency among the different copies of data objects through some distributed database replication protocols [Sto79, Tho79].

A mobile computer accesses the database by communicating with the server in its cell through the wireless medium. The server-to-client channel, also known as the *downlink channel*, is a one-to-all *broadcast* channel while the client-to-server channel, namely the *uplink channel*, is a one-to-one channel. Such network can be an implementation of a DSMA network [LQ95].

As the wireless communication within a cell is independent of that within other cells, we concentrate on the management of the wireless communication resource inside a single cell. Within a cell, there are M mobile computers, m_1, m_2, $\dots$, m_M. Note that, the number of mobile computers within a cell may change over time due to the unrestricted mobility of mobile computers. The mobile computers can access objects in the database by communicating with the base station of the cell. On the other hand, we make a simplified assumption that only the server can update the objects in the database. This assumption is equivalent to the situation where the mobile computers send requests to the server to update the data objects on their behalves. Let the *individual read rate* of a mobile computer m_i to object $o_j \in \mathcal{D}$ be λ^r_{ij}. Also, let the *aggregate write rate* of all the servers in the wired distributed system to an object o_j be λ^w_j. Note that, the aggregate write rate of an object can be higher than the sum of the read rates within a single cell because write requests can be generated from many other cells of the whole personal communication network.

It is important to point out that there are approaches to estimate the access rates mentioned above via statistical methods. For example, the access rates can be gathered from statistics of the past habitual access patterns of the mobile computers, as it is very probable that each user will have a particular access pattern over a subset of the data in $\mathcal{D}$. In addition, more reliable statistics can be obtained from users' reservation requests. For example, a stockbroker may set up his mobile computer to get the stock prices at regular intervals. Such *access profiles* can also provide more precise information for determining access statistics.

In order to reduce the wireless bandwidth consumption, some of the data objects in the database will be *cached* in the mobile computers. In other words, data objects are copied at the mobile computers while other mobile computers can still access the copies at the server. Let the size of the cache memory of a mobile computer m_i be c_i bytes. In order to maintain the consistency among the copies of a data object, some consistency protocols are required. In the next section, we will describe an interval-based distributed caching protocol which maintains cache consistency for mobile computers inside a cell.

3.1.2 Caching Protocol

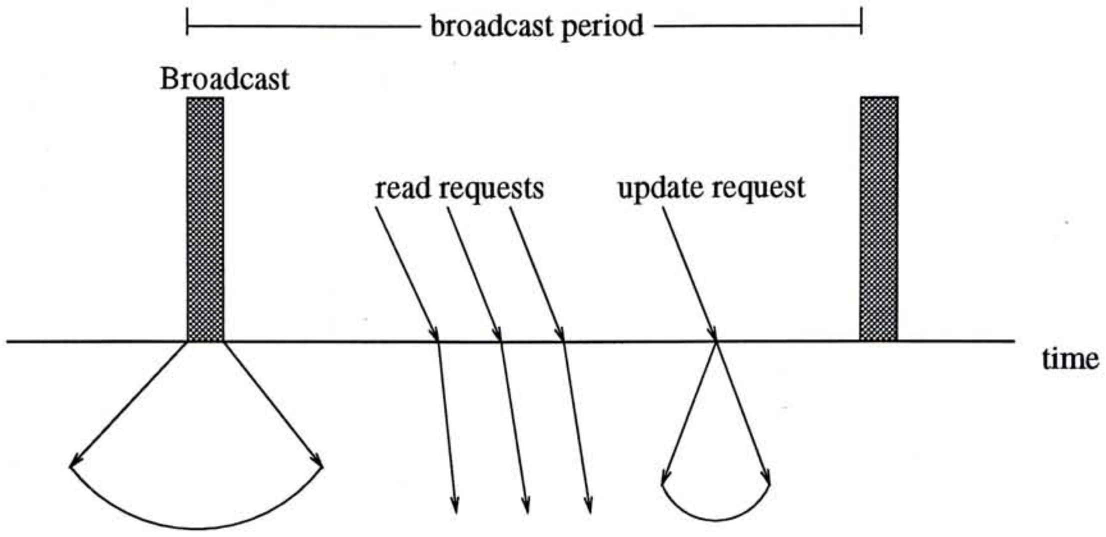


Figure 3.2: The Caching Protocol

The caching protocol is illustrated in Figure 3.2. In this protocol, the time domain is divided into a sequence of *broadcast periods* with constant length $\mathcal{T}$. At the beginning of each broadcast period, data objects of a broadcast set $\mathcal{S}$ (where $\mathcal{S}$ is a subset of the total data set $\mathcal{D}$ in the database) are broadcasted by the server to all mobile computers located inside the cell. Note that, the broadcast set $\mathcal{S}$ may vary in subsequent broadcast periods depending on the access rates of the mobile computers to different data objects. The bandwidth requirement for broadcasting all data objects in $\mathcal{S}$ is given by:

$$U_{broadcast}(\mathcal{S}) = \frac{\sum_{o_j \in \mathcal{S}} b_j}{\mathcal{T}} \tag{3.1}$$

where b_j (in bytes) is the downlink message overhead for delivering the content of data object o_j. In general, the value of b_j is equal to the size of the object, s_j, plus a fixed size message header.

Upon receipt of the broadcast, each mobile computer will select data objects from $\mathcal{S}$ so that its own communication can be minimized. We define $Cache_i(\mathcal{S})$ to be the set of data objects from $\mathcal{S}$ chosen to be cached by a mobile computer m_i. Of course, this caching set should depend on the data access rates of the mobile computer m_i and the size of its cache memory, c_i, such that the expected communication is minimized while the cache of the mobile computer can hold all the objects in $Cache_i(\mathcal{S})$.

When a mobile computer issues a read request, it will first check its local cache memory to see if there is a copy of the data object. If there is, the read request can be satisfied locally and thus no communication is required. On the other hand, if there is no cache copy of the data object in the cache, the mobile computer will send a request message to the server and wait for a reply. This request induces a communication overhead of $(b_{query} + b_j)$ bytes, where b_{query} is a constant uplink overhead of the request message which includes the object index, and b_j is the overhead of the reply message (with the content of the data object o_j). Therefore, the wireless bandwidth requirement for supporting all read queries to non-cached data objects is given by

$$U_{read}(\mathcal{S}) = \sum_{i=1}^{M} \sum_{o_j \notin Cache_i(\mathcal{S})} \lambda_{ij}^{r}(b_{query} + b_j). \tag{3.2}$$

Whenever a data object in the broadcast set $\mathcal{S}$ is updated, the change must be reflected to the caches at the mobile computers immediately so as to maintain consistency among the copies. There are two methods to accomplish this, the invalidation method and the update method. The *invalidation* method only invalidates the copies of the data objects at the mobile computers and the server does not send the new value to any mobile computer in the cell. This method has a constant message overhead of $b_{invalid}$ which includes an index of the data object. After invalidation, the object is dropped by all mobile computers and thus subsequent read requests incur communication overhead. In the *update* method, the server not only invalidates the old copies of the data object at the mobile computers, but also broadcasts the new value of the updated object. This operation costs a downlink message with an overhead of b_j. In spite of the higher communication overhead as compared to the invalidation method, the cached copies at the mobile computers will remain valid and so the subsequent message overheads for serving the read requests are saved. Although our distributed caching method can be applied to both of these methods, for the sake of simplicity in presentation, we choose the update method in this paper. The bandwidth requirement for broadcasting the *update notification* is

$$U_{update}(\mathcal{S}) = \sum_{o_j \in \mathcal{S}} \lambda_j^w b_j. \tag{3.3}$$

To summarize, the bandwidth requirement for supporting our basic caching protocol for a broadcast set $\mathcal{S}$ is

$$U(\mathcal{S}) = U_{broadcast}(\mathcal{S}) + U_{read}(\mathcal{S}) + U_{update}(\mathcal{S}). \tag{3.4}$$

Note that, in this thesis, we assume that each mobile computer can maintain its cache consistency even after crossing a cell boundary or being powered off. This can be enforced by using some handshaking protocols ranging from a simple approach (such as discarding all the cache content once disconnection occurs) to other more elegant methods reported in [WL95].

The issues now are how the server selects the broadcast set $\mathcal{S}$ and how each mobile computer m_i selects the caching set $Cache_i(\mathcal{S})$. We will describe these two selection processes in the coming sections.

3.2 Caching at Mobile Computers

In this section, we define the cache selection algorithm used at the mobile computers, *i.e.* the approach to obtain the caching set $Cache_i(\mathcal{S})$ in a mobile computer m_i, given that the broadcast set is $\mathcal{S}$. This selection algorithm will have a significant impact to the wireless bandwidth consumption because the bandwidth consumption depends on what the mobile computers will cache. Presumably, all mobile computers try to optimize for themselves selfishly, *i.e.* each mobile computer will try to cache all those objects such that its communication with the server is minimized. Note that, the data caching not only reduces the user charges of the mobile computers for network access but also achieves better database access times. The complexity of the optimal selection process, however, is NP-complete. This is illustrated by the following theorem.

Theorem 1 *Given a set of read rates λ^r_{ij} by mobile computer m_i to all data objects o_j in the database $\mathcal{D}$, the problem of finding the caching set $Cache_i(\mathcal{S})$ for a given $\mathcal{S} \subseteq \mathcal{D}$ so as to minimize the communication for the mobile computer m_i is NP-complete.*

Proof: We prove this theorem by showing the equivalence of this problem with the 0/1 knapsack problem [GJ79], which is NP-complete.

Consider the 0/1 knapsack problem instance:

INSTANCE: A finite set $\mathcal{U}$, a *size* $s(u) \in \Re^+$ and a *value* $v(u) \in \Re^+$ for each $u \in \mathcal{U}$, a size constraint $B \in \Re^+$, and a value goal $K \in \Re^+$.

QUESTION: Is there a subset $\mathcal{U}' \subseteq \mathcal{U}$ such that

$$\sum_{u \in \mathcal{U}'} s(u) \leq B \text{ and } \sum_{u \in \mathcal{U}'} v(u) \geq K?$$

Let $\mathcal{U} = \mathcal{S}$, $s(u) = s_j$ and $v(u) = \lambda^r_{ij} b_j$ for $u = o_j$, $B = c_i$ and the bandwidth gain $G = K$, we have the following client caching problem instance:

INSTANCE: A set of data objects $\mathcal{S}$, a data object size $b_j \in \Re^+$ and a read rate $\lambda^r_{ij} \in \Re^+$ for each object $o_j \in \mathcal{S}$, a caching buffer size constraint

$c_i \in \Re^+$, and a saving of communication bandwidth $G \in \Re^+$ by reducing the read requests.

QUESTION: Is there a subset $Cache_i(\mathcal{S}) \subseteq \mathcal{S}$ such that

$$\sum_{o_j \in \mathcal{S}} s_j \leq c_i \text{ and } \sum_{o_j \in \mathcal{S}} \lambda_{ij}^r b_j \geq G?$$

As these two problems are equivalent, the client caching problem is NP-complete. ∎

Therefore, some heuristic methods can be used instead. Here, we make the approximation that the mobile computers use a greedy approach to cache data objects. That is, each mobile computer chooses the data object of a higher read rate with a higher preference unless its cache does not have enough space to accommodate that object. In this case, that object will be dropped and the object with the next higher read rate will be considered. This selection ends until all the data objects in the broadcast set $\mathcal{S}$ is tested. The caching algorithm of each mobile computer is presented in Figure 3.3.

```
For each mobile computer $m_i$ receiving a broadcast set $\mathcal{S}$:

/* Sort the data objects in decreasing read rates */
Arrange the elements in $\mathcal{S}$ into a sequence $\mathcal{O} = \langle o_{s_1}, o_{s_2}, \ldots, o_{s_j}, \ldots o_{s_{|\mathcal{S}|}} \rangle$
        such that $\lambda_{is_j}^r \geq \lambda_{is_{j+1}}^r \forall 1 \leq j < |\mathcal{S}|$

/* Empty the cache */
$Cache_i(\mathcal{S}) \leftarrow \phi$
$AvailBuf \leftarrow c_i$

/* Store the objects if there is enough space */
FOR $k \leftarrow 1$ to $|\mathcal{S}|$ DO
        IF $AvailBuf \geq s_{s_k}$ THEN
                $Cache_i(\mathcal{S}) \leftarrow Cache_i(\mathcal{S}) \cup \{o_{s_k}\}$
                $AvailBuf \leftarrow AvailBuf - s_{s_k}$
        ENDIF
ENDFOR
```

Figure 3.3: Client Caching Algorithm

The complexity of this client caching algorithm can be calculated as follows. The time required to construct the sequence $\mathcal{O}$ using sorting is $O(|\mathcal{S}| \log |\mathcal{S}|)$. Since each iteration in the loop takes $O(1)$ time, the loop takes $O(|\mathcal{S}|)$ time. The average case complexity of the algorithm is $O(|\mathcal{S}|\log|\mathcal{S}|)$ and, in the worst case it is $O(N\log N)$ when $|\mathcal{S}| = N$.

3.3 Broadcasting at the Server

One major difference between wireless and wired networks is that the communication mode is asymmetric [AAFZ95] — the channel from the base station to the mobile computers is a broadcast channel; while the one from the mobile computer to the base station is a one-to-one communication channel. Given this asymmetric communication mode, the server can send its information to virtually any number of mobile computers. It is therefore reasonable to use the downlink channel for broadcasting information of general interest.

In order to reduce the communication bandwidth consumption (*i.e.* minimize the objective function in Equation (3.4)), the server participates in the distributed caching protocol and exploits the broadcast capability of the downlink channel. This scenario is very different from the previous caching cases in which the server is not involved in decision making and only the clients are optimizing for themselves selfishly. To be specific, the server controls caching at the mobile computers by selecting data objects into the broadcast set $\mathcal{S}$ such that the overall wireless bandwidth consumption $U(\mathcal{S})$ is minimized.

From the server's perspective, putting copies of data objects at the mobile computers' caches can result in a saving of message overhead for read queries. However, if there are any updates of the objects in $\mathcal{S}$, the server must utilize some channel resources to notify the mobile computers. Taking both of these factors into consideration, a data object can be favorably chosen into the broadcast set $\mathcal{S}$ if some mobile computers will cache this object and, as a result, the communication overhead saved by reducing the number of all the read requests from those computers is *higher* than the cost of supporting caching this object. Let us introduce the concept of *bandwidth gain*, which is the difference between the communication overhead saved by issuing fewer read requests and the bandwidth required to support caching. Let $G(\mathcal{S}, o_j)$ denote the bandwidth gain of a data object $o_j \in \mathcal{S}$, *i.e.* the reduction in bandwidth consumption due to o_j. Mathematically, it is given by

$$G(\mathcal{S}, o_j) = \begin{cases} \sum\limits_{\{i|o_j \in Cache_i(\mathcal{S})\}} \lambda_{ij}^{r}(b_{query} + b_j) - (\frac{b_j}{T} + \lambda_j^{w} b_j) & \text{if } o_j \in \mathcal{S} \\ 0 & \text{otherwise} \end{cases}. \tag{3.5}$$

The first term above is the bandwidth saved by issuing fewer read requests from all the mobile computers which decide to cache o_j; the second term above is the cost of supporting broadcasts and updates of that data object. Using this definition, we can rewrite the bandwidth consumption equation (Equation 3.4) as follows:

$$\begin{aligned} U(\mathcal{S}) &= U(\oslash) - \sum_{o_j \in \mathcal{S}} G(\mathcal{S}, o_j) \qquad (3.6) \\ U(\oslash) &= U_{read}(\oslash) = \sum_{i=1}^{M}\sum_{j=1}^{N} \lambda_{ij}^{r}(b_{query} + b_j) \end{aligned}$$

where $U(\oslash)$ is the total bandwidth consumption for the non-caching case, *i.e.* $\mathcal{S}$ is an empty set.

Note that, if a broadcast set is optimal, then all the data objects belonging to this set must have positive gain. Moreover, this set must be *maximal* in the sense that there will be no object outside this set which, if included in the set, will also have a positive gain. The following theorem describes this formally.

Theorem 2 *If a broadcast set $\mathcal{S}$ is optimal, then this set must satisfy the following properties:*

Positive Gain Property: $\forall o_j \in \mathcal{S}, G(\mathcal{S}, o_j) \geq 0.$

Maximal Set Property: $\forall o_j \notin \mathcal{S}, G(\mathcal{S} \cup \{o_j\}, o_j) \leq 0.$

Proof: We prove the above theorem by contradiction.

Positive Gain Property: Suppose that there is an optimal broadcast set $\mathcal{S}$ which does not satisfy the Positive Gain Property. Then there exists an object $o_j \in \mathcal{S}$ such that $G(\mathcal{S}, o_j) < 0$. Given Equation (3.6), removing this object from $\mathcal{S}$ will result in a strictly lower bandwidth requirement, *i.e.* $U(\mathcal{S}) > U(\mathcal{S} \backslash \{o_j\})$. This contradicts with the assumption that $\mathcal{S}$ is optimal. So $\mathcal{S}$ must satisfy the Positive Gain Property.

Maximal Set Property: Suppose that there is an optimal broadcast set $\mathcal{S}$ which does not satisfy the Maximal Set Property. Then there exists an object $o_j \notin \mathcal{S}$ such that $G(\mathcal{S} \cup \{o_j\}, o_j) > 0$. Given Equation (3.6), adding this object to $\mathcal{S}$ will result in a strictly lower bandwidth requirement, *i.e.* $U(\mathcal{S}) > U(\mathcal{S} \cup \{o_j\})$. This contradicts with the assumption that $\mathcal{S}$ is optimal. So $\mathcal{S}$ must satisfy the Maximal Set Property. ∎

Note that these two properties are only necessary conditions for an optimal broadcast set. A broadcast set, on the other hand, may not be optimal even if it satisfies these properties. Here is a counter-example:

Example 1 Consider a simple scenario where there are three equal size data objects $\{o_1, o_2, o_3\}$ in database $\mathcal{D}$. The aggregate write rate of each object is equal to 3, and there are two mobile computers $\{m_1, m_2\}$ querying the database at the following rates (Table 3.1):

	o_1	o_2	o_3
m_1	5	6	3
m_2	5	3	6

Table 3.1: Access Rate Assignment for Example 1

Assume that each mobile computer can cache only one object at a time. For simplicity, let both b_{query} and $b_{broadcast}$ for each object be 0, and b_j be 1 for all $1 \leq j \leq 3$. Let $\mathcal{S}$ be $\{o_2, o_3\}$. Then m_1 will cache o_2 while m_2 will cache o_3. Both of these objects result in positive bandwidth gains of 3. Moreover, if o_1 is added to $\mathcal{S}$, it will result in a negative bandwidth of -3. As a result, this set satisfies both of above mentioned properties. However, the set $\{o_1\}$ is the optimal caching set, and of course it also satisfies the theorem.

From this point of view, we cannot construct an optimal broadcast set by merely satisfying those properties. The main problem is that the bandwidth gain of a data object depends on the current content of the broadcast set. Due to limited size of cache memories, the mobile computers cannot hold all broadcasted objects. As a result, a broadcasted data object may not be cached by all mobile computers which have an interest in it. This makes it difficult to compute the bandwidth gains $G(\mathcal{S}, o_j)$ for the data objects. The following theorem gives the complexity of finding an optimal broadcast set $\mathcal{S}$.

Theorem 3 *Given the read rates λ^r_{ij} and the write rates λ^w_j for all data objects in $\mathcal{D}$, and the client caching algorithm in Figure 3.3, the complexity of finding an optimal broadcast set $\mathcal{S}$ so as to minimize the cost function in Equation 3.4 is NP-hard.*

Proof: Please refer to Appendix A. ■

Given this result, it is impractical to compute the optimal broadcast set. Instead, we explore other more efficient algorithms at a reasonable expense so as to achieve near-optimal performance. In the following two sections, we propose two strategies.

3.3.1 Passive Strategy

The Passive Strategy is commonly used in a conventional client-server database systems. In this strategy, the base station does not perform any selection. It simply broadcasts what the mobile computers want to cache. In other words, given its cache size constraint, each mobile computer chooses all the data objects it wants to access and the server then broadcasts the requested data objects. The broadcast set can therefore be computed as

$$\mathcal{S} = \bigcup_{i=1}^{M} Cache_i(\mathcal{D}). \tag{3.7}$$

Note that the resulting broadcast set does not, in general, have the Positive Gain nor the Maximal Set properties given above. However, the major advantage of this algorithm is its simplicity and its low complexity, which is $O(MN)$.

3.3.2 Active Strategy

The Active Strategy exploits the broadcast feature of the wireless channel. It aims at constructing a broadcast set that conforms to the Positive Gain Property. Here, we assume that the mobile computers use the selection strategy described in the previous section. The server selects data objects in two consecutive phases, namely, the *growing phase* and the *shrinking phase.* At the beginning of the growing phase, the broadcast set is initialized to be empty. At each iteration, an object will be added to the broadcast set, if it results in a positive bandwidth gain. This phase terminates when all the objects are tested. As objects are added to the broadcast set, some of the objects that were cached by mobile computers earlier will be dropped in subsequent iterations to make room for those objects that are more favorable. Some of these objects may have negative gains after this phase. The main objective of the shrinking phase is to remove the data objects with negative gains. This procedure is very similar to the growing phase, except that it proceeds in a reverse manner. The resulting algorithm is as given in Figure 3.4.

The complexity of the algorithm can be found as follows. In this algorithm, the calculation of $G(\mathcal{S}, o_j)$ requires computation of the caching set $Cache_i(\mathcal{S})$ for each mobile computer. However, in the growing phase, only one data object is added in each iteration. Similarly, in the shrinking phase, data objects are removed from the broadcast set one at a time. The computation of $Cache_i(\mathcal{S})$ can be simplified and the complexity can be reduced to $O(\log N)$. Therefore, the whole phase takes $O(MN \log N)$ time. Altogether, the time complexity for the algorithm is $O(MN \log N)$. Note that, the

```
/* Initialize with an empty broadcast set */
$\mathcal{S} \leftarrow \phi$

/* Growing Phase */
FOR each $o_j \in \mathcal{D}$ DO
        /* Collect objects with positive bandwidth gains */
        IF $G(\mathcal{S} \cup \{o_j\}, o_j) > 0$ THEN
                $\mathcal{S} \leftarrow \mathcal{S} \cup \{o_j\}$
        ENDIF
ENDFOR

/* Shrinking Phase */
FOR each $o_j \in \mathcal{S}$ DO
        /* Remove objects with negative bandwidth gains */
        IF $G(\mathcal{S}, o_j) \leq 0$ THEN
                $\mathcal{S} \leftarrow \mathcal{S} - \{o_j\}$
        ENDIF
ENDFOR
```

Figure 3.4: The Active Strategy for Assigning Broadcast Set $\mathcal{S}$

caching algorithm used by mobile computers must also be implemented at the server so as to evaluate the gain. As the server must take $O(N \log N)$ time to determine the caching set of each mobile computer, the lowest sensible computational complexity for all M mobile computers will be at least $O(MN \log N)$ time. In this sense, the algorithm is an efficient one.

It is easy to verify that the broadcast set generated by the above algorithm has the *Positive Gain Property.* The shrinking phase ensures that each object that remains in the set $\mathcal{S}$ will have a positive gain and therefore, the property follows. However, the resulting broadcast set does not have the Maximal Set Property. This implies that there are data objects outside the broadcast set $\mathcal{S}$ which would have positive gain, if they were added to the set. It is possible to add these objects to the set by iterating through the above algorithm at most $O(N)$ times. However, the complexity of the algorithm will then be $O(MN^2 \log N)$, which is higher by $O(N)$.

3.4 Performance Analysis

In this section, we derive analytically some performance measures for comparing the efficiency of the strategies. Although we have defined the *total* bandwidth requirement (see Equation (3.4)) previously, it is also desirable to compare the bandwidth requirements for uplink and downlink channels separately. In addition, we are interested in the total bandwidth consumption, $U(\mathcal{S})$, and the expected response time, $R(\mathcal{S})$ of accessing a data object, given a system configuration (*i.e.* the wireless bandwidth, the number of mobile computers, ... etc.) and the broadcast set $\mathcal{S}$. Besides these performance measures, we also derive a lower bound to the optimal bandwidth consumption. Note that, this bound is very important because it gives us a quantitative measure on how close our algorithm is as compared to the optimal solution.

3.4.1 Bandwidth Requirements

In Section 3.1, we have defined the bandwidth requirement as the communication bandwidth required to support three operations of our caching protocol, namely, the periodic broadcast, the read requests to non-cached objects, and the update notification. However, we have not investigated the bandwidth consumption of each of the uplink and downlink channels separately. This section derives the uplink bandwidth requirement for *all* mobile computers within the cell and the downlink bandwidth requirement for the server.

For the periodic broadcast and update notification, these two operations are server-initiated and require no acknowledgment from mobile computers. Therefore, these operations only use up bandwidth in the downlink channel. For the read requests to non-cached objects, however, the mobile computers need to send their requests using the uplink. Furthermore, a reply from the server is required for each of these requests. This utilizes the downlink channel. To summarize, the bandwidth requirement, $U^{up}(\mathcal{S})$, for uplink channels of all mobile computers is

$$U^{up}(\mathcal{S}) = \sum_{i=1}^{M} \sum_{j \notin Cache_i(\mathcal{S})} \lambda_{ij}^{r} b_{query} \tag{3.8}$$

and the bandwidth requirement, $U^{down}(\mathcal{S})$, for the downlink channel is

$$U^{down}(\mathcal{S}) = U_{broadcast}(\mathcal{S}) + U_{update}(\mathcal{S}) + \sum_{i=1}^{M} \sum_{j \notin Cache_i(\mathcal{S})} \lambda_{ij}^{r} b_j. \tag{3.9}$$

As the mobile computers may not be able to cache their most frequently accessed objects in the Active strategy, it is expected that the bandwidth consumption of the uplink channel in this strategy will be higher than that in the Passive strategy. On the other hand, the bandwidth consumption of the downlink channel will be lower in the Active strategy.

3.4.2 Lower Bound on the Optimal Bandwidth Consumption

Since the caching strategies cannot obtain the optimal solution, it is desirable to estimate their performance. Our approach is to devise an *unrealizable* lower bound on the bandwidth requirement analytically and compare it with the resulting bandwidth requirement of the proposed algorithms.

As described before, the bandwidth requirement is made up of three components. The periodic broadcast and update notification are for supporting objects for data caching at mobile computers while the read request part is for supporting individual queries to non-cached objects at mobile computers. In determining the lower bound, we ignore the broadcast and update components and assume that they have zero bandwidth contributions. We want to find a lower bound on the bandwidth consumption in supporting individual queries.

The formulation of the lower bound on bandwidth consumption comes from the unrealistic assumption that each mobile computer can cache the most highly accessed objects in the database. By doing so, the mobile computer can minimize the bandwidth consumption to the fullest extent and thus a lower bound is obtained. It is unrealizable because the server will not always broadcast all these objects to the mobile computers.

We begin the evaluation by introducing a few more parameters. Let $\bar{c}$ be the average cache memory size of mobile computers, and let $\bar{s}$ be the average object size (the mean value of s_j). The values read rates (λ_{ij}^r) are distributed according to a general probability density function $p(\lambda)$. On the average, a mobile computer can hold $\bar{c}/\bar{s}$ objects. These objects are cached and thus the corresponding read cost is zero in terms of bandwidth consumption. For each of the remaining $N - \bar{c}/\bar{s}$ objects in the database, there is an expected read cost of $(b_{query} + \bar{b_j})$, where $\bar{b_j}$ is the mean value of b_j. To find the bandwidth consumed by this mobile computer, we need to find the expected rate of accessing these remaining *non-cached* objects.

We assume that the read rates of the objects of a mobile computer are independent of each other. Given this assumption, the *hottest* $\bar{c}/\bar{s}$ data objects are cached by a mobile computer. As a result, the average read rate of the non-cached objects is lowered. These non-cached data objects correspond to the $(1 - \bar{c}/N\bar{s})^{th}$ portion of the database. Probabilistically, there is an expected upper bound φ on the access rates λ_{ij}^r such that the probability of accessing a non-cached object is

$$Prob[\text{cache miss}] = \int_0^{\varphi} p(\lambda)\, d\lambda = 1 - \frac{\bar{c}}{N\bar{s}}. \tag{3.10}$$

The value of φ can be determined if a particular probability density function $p(\lambda)$ is determined. The expected read rate of a mobile computer of a non-cached object is then given by

$$E[\lambda_{ij}^r | o_j \text{ not in cache}] = \frac{\int_0^{\varphi} \lambda p(\lambda)\, d\lambda}{Prob[\text{cache miss}]}. \tag{3.11}$$

Hence, the total read request rate of all mobile computers within a cell to the non-cached objects is

$$E[\lambda_{ij}^r | o_j \text{ not in cache}](N - \frac{\bar{c}}{\bar{s}})M = \left(\int_0^{\varphi} \lambda p(\lambda)\, d\lambda\right) MN \tag{3.12}$$

and the lower bound on the total bandwidth consumption for this read request is

$$\left(\int_0^{\varphi} \lambda p(\lambda)\, d\lambda\right) MN(b_{query} + \bar{b_j}). \tag{3.13}$$

This lower bound varies linearly with the number of mobile computers and the number of data objects. We can also observe that as the cache size increases, φ will decrease and thus, the lower bound value will also decrease. The wireless bandwidth requirement for the non-caching case is

$$E[\lambda_{ij}^r]MN(b_{query} + \bar{b_j}) = \left(\int_0^{\infty} \lambda p(\lambda)\, d\lambda\right) MN(b_{query} + \bar{b_j}). \tag{3.14}$$

We can easily observe from the above two equations that the overall wireless bandwidth consumptions vary linearly with the average value of the individual read rate. Moreover, the linearity constant is determined by the distribution of the individual read request rates and the sizes of caches at the mobile computers.

In the average case analysis above, we assume that the mobile computers can cache all different objects they like. However, it is not always possible. One situation is that for some of the objects, the aggregate write rate is higher than the total read rate. These objects will not be selected into the broadcast set by the algorithm. By taking this into account, we can consider a smaller number of data objects in our estimation and obtain a tighter and better lower bound.

3.4.3 The Read Response Time

There are two scenarios in reading a data object o_j in the database, namely, the cached case and the non-cached case. In the cached case, a copy of the requested data object is already inside the cache of the requesting mobile computer. So access to that object requires no wireless resource. In the non-cached case, on the other hand, the data object is not in the mobile computer's cache. A request message is needed to be sent uplink to the server. The server has to reply by sending the data object back to the mobile computer through the downlink channel. In regard to the high processing power of computers nowadays, we may assume that the service time at the server (or more specifically, the wired system) is negligible compared to the round-trip delay of the data access.

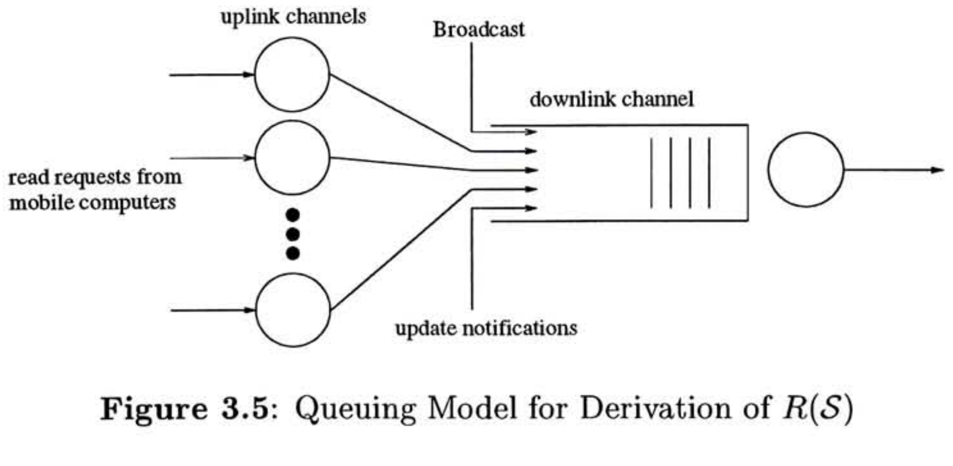


Figure 3.5: Queuing Model for Derivation of $R(\mathcal{S})$

Figure 3.5 illustrates our queuing model for deriving the read response times. Since the uplink channel is one-to-one, the channel will not be interfered by other mobile computers. Assume that the mobile computer can issue only one request at a time. Therefore, the uplink request will be sent immediately once the request is generated. This channel is modeled by a server node. As the request message should be of fixed size, b_{query}, therefore, the message can be delivered in a fixed amount of time. Let the bandwidth of the uplink channel be B_{up}. Therefore, the uplink request takes b_{query}/B_{up} time. Moreover, the arrival traffic of the read requests to the server is the aggregate read requests from the mobile computers, with a b_{query}/B_{up} delay.

In order to answer the requests to various mobile computers, the server sends the contents of the requested data objects back to the mobile computer through the downlink channel. The transmission delay for an object o_j is b_j/B_{down} where B_{down} is the bandwidth of the downlink channel. However, since the broadcast channel is shared by many requests at the same time, some arbitration procedure must be enforced. In this paper, we assume the first-come-first-serve discipline for accessing the downlink communication channel. Under this discipline, a newly arrived reply message has to wait for the preceding messages to be delivered first. This incurs a waiting time, which we are going to derive.

The downlink channel is serving *multiple classes* of requests: reply to read queries of mobile computers, update notifications, and periodic broadcasts. We model the read query traffic from each mobile computer by a Poisson process. Therefore, the aggregate read queries from all the mobile computers within the cell also form an aggregated Poisson process. This is also the case for update notifications. For periodic broadcasts, however, the inter-arrival times are fixed and this complicates the solution of the model. In order to alleviate this problem, we use an approximation that the time period $\mathcal{T}$ is exponentially distributed. By making this assumption, we can model the broadcast channel as an M/G/1 queue with multiple classes of requests.

Consider the simpler case in which an M/G/1 queue serves a single class of requests whose expected arrival rate is λ. Let t and $t^{(2)}$ be the first and second moments of the service times of the requests. Let T be the system time of a request passing through the queue. Using the PASTA property [Wol82] and Little's result [Lit61], the expected number of pending requests when a new request arrives at the system is λT and the expected number of requests in service inside the queue is $\rho = \lambda t$, where ρ is also the system utilization of the M/G/1 system. Therefore, the amount of unfinished work in the queuing system is given by [Kle75]

$$(\lambda T - \rho)t + \rho\frac{t^{(2)}}{2t}. \tag{3.15}$$

The first term in Equation (3.15) is the expected unfinished work of the requests waiting in the queue and the second term is the expected unfinished work of the request [Cox62] currently in service when a new request arrives. By extending this result to the case for multiple classes of requests, we can have the expected total amount of unfinished work in our model which is given by

$$\begin{aligned} W &= (\lambda_r T_r - \rho_r)t_r + \rho_r \frac{t_r^{(2)}}{2t_r} + (\lambda_u T_u - \rho_u)t_u + \rho_u \frac{t_u^{(2)}}{2t_u} \\ &\quad + (\lambda_b T_b - \rho_b)t_b + \rho_b \frac{t_b^{(2)}}{2t_b} \end{aligned} \tag{3.16}$$

where

T_r, T_u, T_b are the expected system times for delivering a read reply, update, and broadcast message respectively;

t_r, t_u, t_b are mean service times to deliver a read reply, update, and broadcast message respectively;

$t_r^{(2)}, t_u^{(2)}, t_b^{(2)}$ are the second moments of service times to deliver a read reply, update, and broadcast message respectively;

$\rho_r = \lambda_r t_r, \rho_u = \lambda_u t_u, \rho_b = \lambda_b t_b$ are the channel utilizations to deliver read reply, update, and broadcast messages respectively;

$\lambda_r = \sum_{i=1}^{M} \sum_{o_j \notin Cache_i(\mathcal{S})} \lambda_{ij}^r$ is the aggregate read rate from all mobile computers;

$\lambda_u = \sum_{o_j \in \mathcal{S}} \lambda_j^w$ is the aggregate write rate to all objects in $\mathcal{S}$; and

$\lambda_b = 1/\mathcal{T}$ is the rate of periodic broadcast.

In Equation (3.16), the expected response times for reply to read request (T_r), the expected response time of update notifications (T_u) and the expected response time of periodic broadcasts (T_b) are given by the following equations (by the first-come-first-serve queuing discipline):

$$T_r = W + t_r \tag{3.17}$$

$$T_u = W + t_u \tag{3.18}$$

$$T_b = W + t_b \tag{3.19}$$

Equations (3.16) through (3.19) form a system of equations with four unknowns. Substituting Equations (3.17), (3.18) and (3.19) into Equation (3.16), we have

$$\begin{aligned} W &= (\lambda_r (W + t_r) - \rho_r)t_r + \rho_r \frac{t_r^{(2)}}{2t_r} + (\lambda_u (W + t_u) - \rho_u)t_u + \rho_u \frac{t_u^{(2)}}{2t_u} \\ &\quad + (\lambda_b (W + t_b) - \rho_b)t_b + \rho_b \frac{t_b^{(2)}}{2t_b}. \end{aligned} \tag{3.20}$$

Since $\rho_r = \lambda_r t_r$, $\rho_u = \lambda_u t_u$ and $\rho_b = \lambda_b t_b$, it can be simplified into

$$W = \rho_r W + \rho_r \frac{t_r^{(2)}}{2t_r} + \rho_u W + \rho_u \frac{t_u^{(2)}}{2t_u} + \rho_b W + \rho_b \frac{t_b^{(2)}}{2t_b}. \tag{3.21}$$

By changing the subject of the above equation to W, and then substitute it back into Equation (3.17), we can obtain the following closed form expression to the expected system time T_r:

$$T_r = \frac{\rho_r \frac{t_r^{(2)}}{2t_r} + \rho_u \frac{t_u^{(2)}}{2t_u} + \rho_b \frac{t_b^{(2)}}{2t_b}}{1 - \rho_r - \rho_u - \rho_b} + t_r \tag{3.22}$$

The expected read response time for non-cached objects is then given by

$$\text{Uplink Time} + \text{Downlink Time} = \frac{b_{query}}{B_{up}} + T_r. \tag{3.23}$$

Unconditioning the probability of the accessing the non-cached objects, we have the overall expected response times, $R(\mathcal{S})$, of reading a data object:

$$\begin{aligned} R(\mathcal{S}) &= Prob[\text{cache miss}] \times (\frac{b_{query}}{B_{up}} + T_r) \\ &= \frac{\sum_{i=1}^{M} \sum_{j \notin Cache_i(\mathcal{S})} \lambda_{ij}^r}{\sum_{i=1}^{M} \sum_{j=1}^{N} \lambda_{ij}^r} \times (\frac{b_{query}}{B_{up}} + \frac{\rho_r \frac{t_r^{(2)}}{2t_r} + \rho_u \frac{t_u^{(2)}}{2t_u} + \rho_b \frac{t_b^{(2)}}{2t_b}}{1 - \rho_r - \rho_u - \rho_b} + t_r) \end{aligned} \tag{3.24}$$

3.5 Experiments

In this section, we carry out experiments on various test cases to compare the performance of the Active strategy, the Passive strategy, and the case where no caching is used. Here, we assign values for the broadcast period ($\mathcal{T}$) of the caching protocol, the read rates (λ_{ij}^r) and the write rates (λ_j^w) for the data objects, and the query overheads b_{query} and b_j to do the update requests and broadcast. Then we use the Passive and Active strategies to generate the broadcast set $\mathcal{S}$ for each problem instance. Using these broadcast sets, we evaluate the wireless bandwidth consumptions using Equations (3.4) and (3.14), and the expected read response times using Equation (3.24) for different strategies. We also use the lower bound formulated in Equation (3.13) to estimate the effectiveness of these strategies.

There are four experiments and they differ in the access patterns of the mobile computers to the database. The aim of these experiments is to demonstrate how adaptive our distributed caching algorithm is under different workloads and read/write ratio. The following table summarizes the access models of the four experiments:

Experiment	Access Pattern
1	Uniform access to all objects in the database
2	"Single Hot Spot" access at a particular subset of the database
3	"Multiple Hot Spot" access at different partitions of the database
4	"Single Hot Spot" access, varying read/write ratio

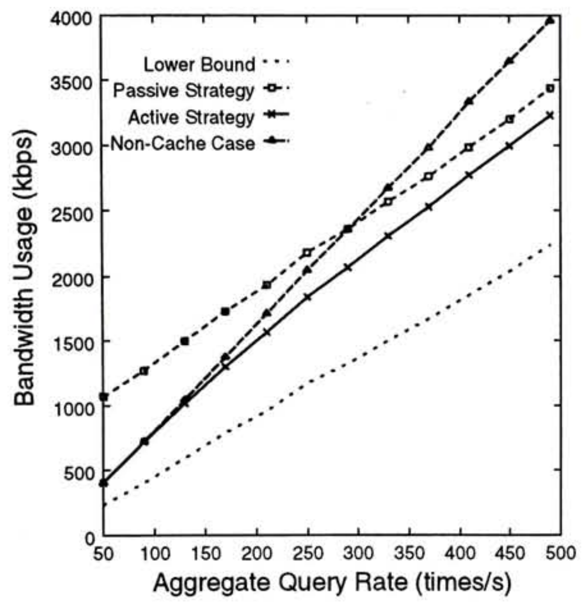
Table 3.2: Database Access Patterns for Different Experiments

The first experiment models the case where all mobile computers access the data objects in the database uniformly. In the second experiment, we model a more realistic situation in which a particular subset (*hot spot*) of the data objects in the database is highly referenced. For example, this model represents the case where all mobile computers are running a similar kind of application. In the third experiment, we model the case where different kinds of users are using different applications and therefore, they are accessing different subsets of data objects in the database. The last experiment aims at showing the performance of different strategies under different read/write ratios. In real situations, the access patterns will probably be a composition of the above cases.

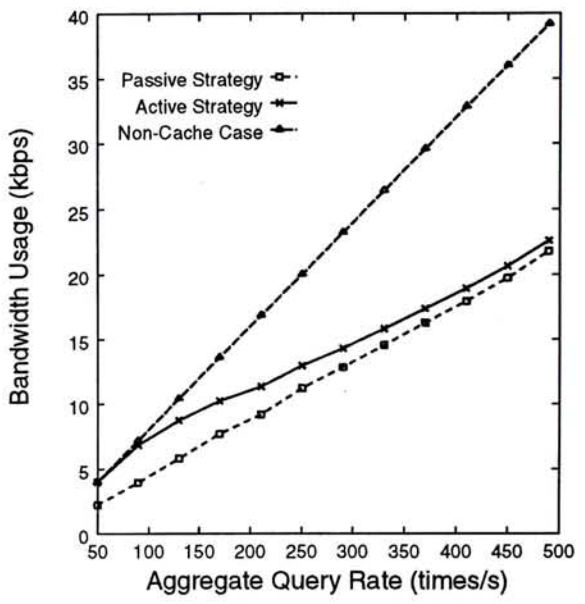
Experiment 1. In this experiment, there are 1000 data objects in the database. The sizes of the data objects are exponentially distributed with a mean value of 1K bytes. Within a cell, there are 50 mobile computers which access the database by exchanging messages with the base station. The cache sizes of the mobile computers vary exponentially with a mean value 200K bytes. Let $b_j = s_j$ for all data objects in the database. Moreover, let the bandwidth of the downlink channel and the uplink channel be $2Mbps$ and $19.2kbps$, respectively, which are typical bandwidths for wireless systems nowadays.

In this experiment, the mobile computers will access all the objects in the database uniformly. The read requests of each mobile computer i for an object j form a Poisson process with a parameter λ_{ij}^r. The performance of the system will be evaluated with an increasing aggregate read rate. Although we have specified the number of mobile computers here, it is worth noting that this gradual increase of the read rate has a similar effect as increasing the number of mobile computers and/or the activity of the computers. The total write rate to all of the objects is exponentially distributed with the mean kept at 10% of the total read rate.

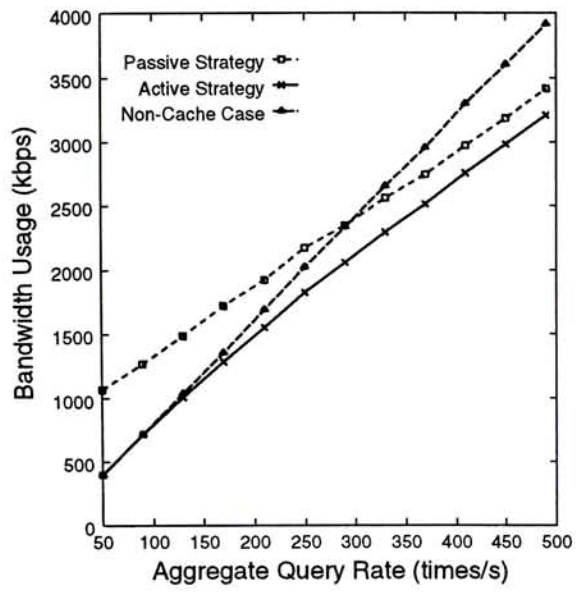
The bandwidth consumption is shown in Figure 3.6(a). From this figure, we see that



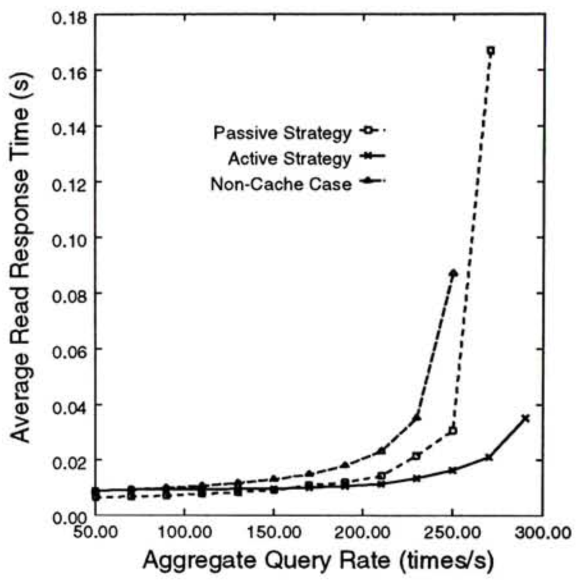
(a) Overall Bandwidth Consumption



(b) Uplink Bandwidth Consumption



(c) Downlink Bandwidth Consumption



(d) Read Response Times

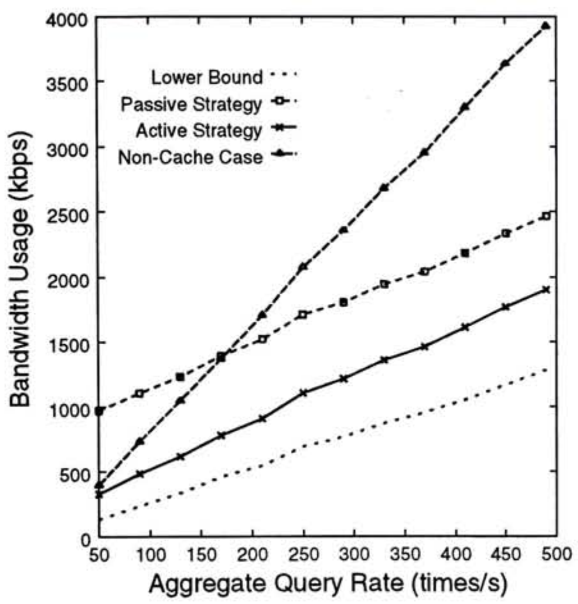
Figure 3.6: Experiment 1: Results for the Uniform Access Model

the Active strategy is always better than the Passive strategy or the non-caching case. However, as the total read rate increases, the performance of the Active strategy gradually approaches that of the Passive strategy or the non-caching case, whichever is lower. The reason is that when the total read rate is high, more objects may be included in the broadcast set, and the overhead of the periodic broadcast may be higher. Also, the activities for querying the non-cached objects are also more frequent. However, note that the performance of the non-caching case degrades much faster; it performs worse than both strategies at high read rates. This justifies the use of distributed caching to reduce the wireless network bandwidth consumption.

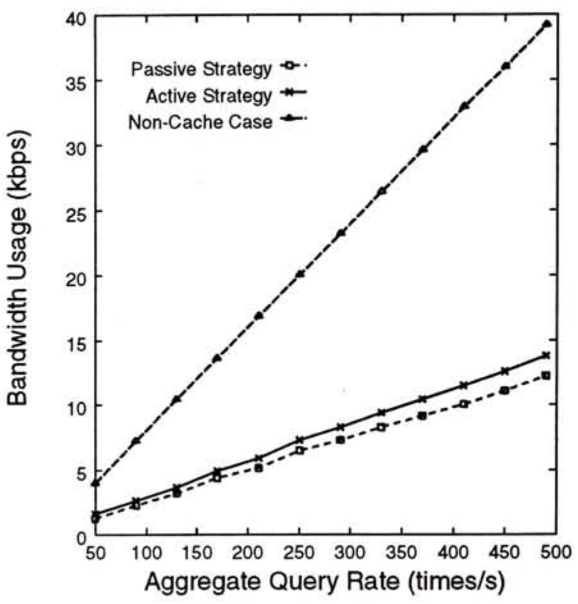
Figures 3.6(b) and 3.6(c) show the bandwidth consumption for the uplink and downlink channels respectively. For the bandwidth consumption of the uplink channel, the bandwidth usage of the Active strategy is higher than that of the Passive strategy. It is because in the Active strategy, the mobile computers cannot cache all their frequently accessed data objects. The non-caching case is the worst because the mobile computers cannot cache anything and so, all queries require uplink requests. On the other hand, the bandwidth consumption of the downlink channel under the Active strategy is better. It is because the server has also considered the resource management at the broadcast channel. So the resulting bandwidth consumption in this channel is lowered. This reduction not only compensates for the increased bandwidth overhead in the uplink channel, but also improves the overall response time to the read requests of mobile computers, as shown in Figure 3.6(d).

In Figure 3.6(d), we see that when the aggregate read rate is low, the read response times of both strategies are comparable. This is due to the fact that when the downlink channel utilization is low, the response is very fast. In this range, it may be a good tradeoff to cache some data objects; this will result in a small increase in bandwidth consumption but will reduce the response times, as in the case of the Passive strategy. On the other hand, when the channel utilization is high, the response times increase exponentially. In this scenario, the only way to lower the response time is to reduce the channel bandwidth consumption. In this case, the Active strategy performs better.

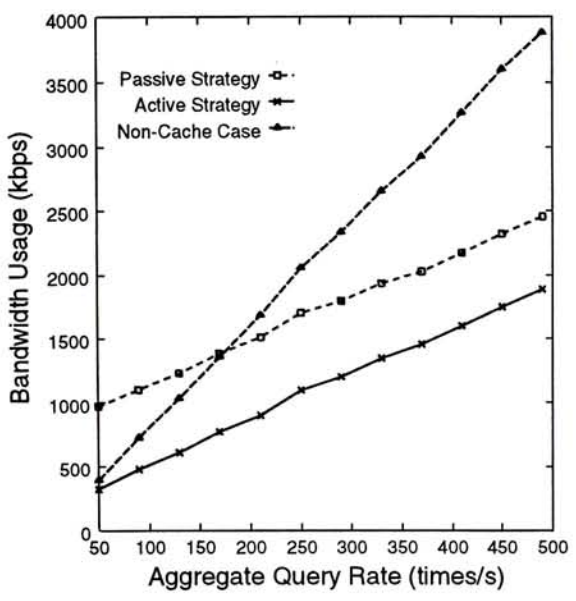
Experiment 2. The configuration of this experiment is almost the same as that of Experiment 1. In this experiment, however, we make a more *realistic* assumption that the computers are running the same kind of application. This means that the mobile computers will access a particular subset of the database at a much higher rate. We model this access pattern by using the 80/20 access model. That is, 80% of the total read and write queries will go to 20% of the database objects. Without loss of generality, we assign the first 200 data objects of the database with 80% of the total access rates.



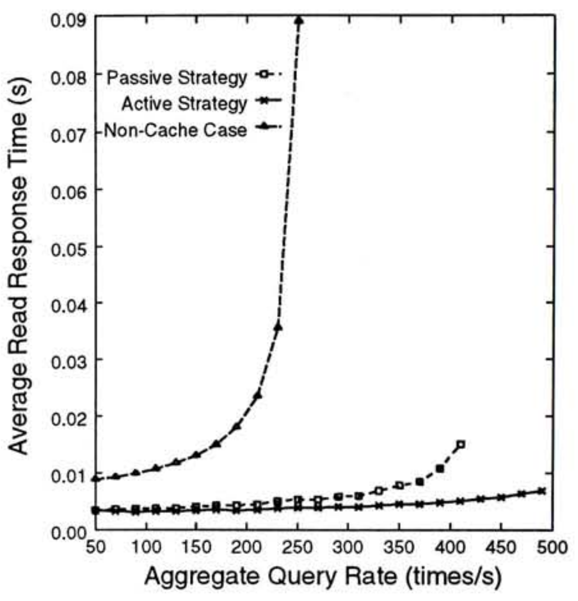
(a) Overall Bandwidth Consumption



(b) Uplink Bandwidth Consumption



(c) Downlink Bandwidth Consumption



(d) Read Response Times

Figure 3.7: Experiment 2: Results for the "Single Hot Spot" Access Model

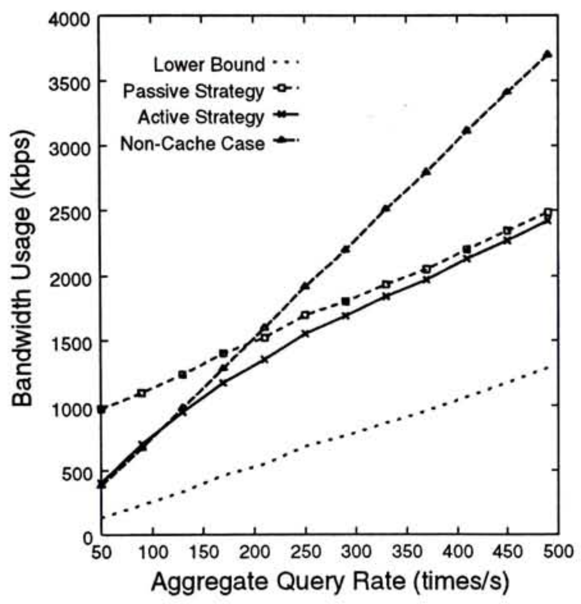
The result for bandwidth consumption for this experiment is shown in Figure 3.7(a). For all three cases, the bandwidth consumption for this scenario is much lower than the bandwidth consumption in the previous experiment. The reason for this is that the read requests are concentrated at a particular subset of the database and, this subset will be more likely to be broadcasted and cached. Moreover, we can also see that the bandwidth requirement is closer to the lower bound bandwidth requirement. This means that the performance of the Active strategy is closer to the optimal, as compared to the Passive strategy. Under this access model, the performance of the Active strategy is the best, while the Passive strategy is better than the non-caching case at high read rates. We can also observe that, as the bandwidth consumption is lowered, the saturation point for the downlink channel moves to a higher read rates and therefore, the read response times of the Active strategy, as shown in Figure 3.7(d), is much lower than that of the Passive strategy and the non-caching case even at high read rates.

Experiment 3. In this experiment, we model the situation where different kinds of mobile computers are running different kinds of applications. Here, we divide the mobile computers into five equal classes, each of which accesses a separate subset of data objects. The bandwidth consumption of this experiment is shown in Figure 3.8(a). From the figure, we observe that the performance of both the Active and the Passive strategies are very near at high read rate. However, at low read rate the performance of the Active strategy is much better. It is because the access rates are so evenly distributed that there is little difference in selecting whatever subset of objects into the broadcast set.

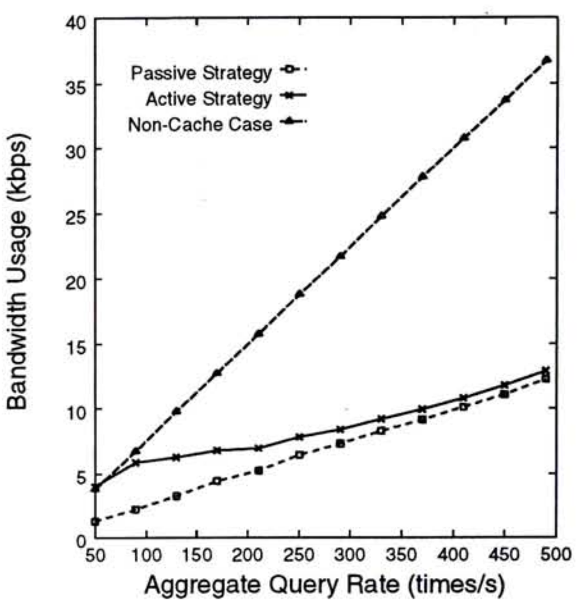
The performance for the non-caching case, however, is always much worse than the caching strategies. Figure 3.8(d) shows the response times of different cases. The performance of both strategies are nearly the same. It is because different kinds of mobile computers are accessing different data objects. The access rates are then nearly uniformly distributed that the Active strategy cannot choose any objects with higher bandwidth gains, so it would result to have a comparable performance as the Passive strategy. However, it is important to point out that the Active strategy is very adaptive to various workloads and therefore, it may be the algorithm of choice.

Experiment 4. In this experiment, we explore the performance of the different strategies under various read-write ratios. We first fix the total read rate to be 300 requests per time unit, then we vary the total write rate from 0 to 600 requests per time unit, reach results in a write-to-read ratio of 0 to 2. Note that, this range is possible as write requests can be issued by other stationary servers in the wired-network.

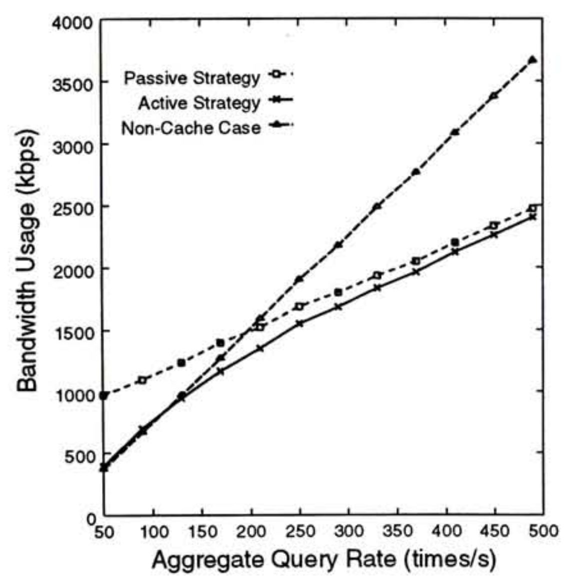
Figure 3.9 shows that the performance of the Active strategy and the non-caching



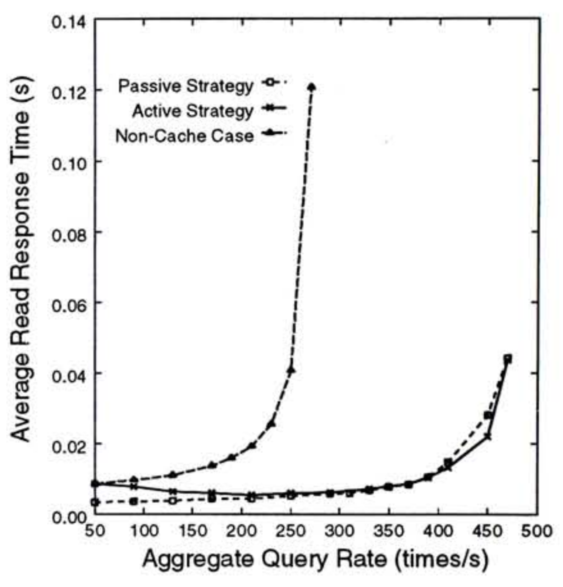
(a) Overall Bandwidth Consumption



(b) Uplink Bandwidth Consumption



(c) Downlink Bandwidth Consumption



(d) Read Response Times

Figure 3.8: Experiment 3: Results for the "Multiple Hot Spot" Access Model

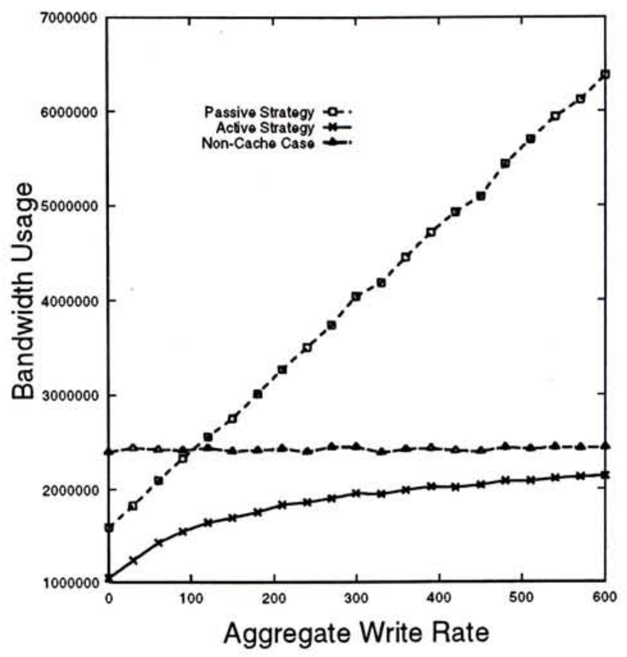


Figure 3.9: Experiment 4: Bandwidth Requirements under different Read-Write Ratios

case remains relatively constant with increasing write rates. Comparing it to the Passive strategy which does not take any account on the cost of update notification. Moreover, the bandwidth consumption of the Active strategy is the lowest amount of the three strategies, therefore, the Active strategy is much more favorable.

3.6 Mobility Concerns

In a personal communication network, the mobile computers can move around the geographical region without any restriction. Therefore, the number of mobile computers inside a cell will change over time. Moreover, because each mobile computer has its own database access pattern, the overall database access pattern of the mobile computers inside a certain cell will inevitably change.

This phenomenon have significant impacts on the performance of our distributed caching protocol, which relies on the database access information of the mobile computers to optimize for the overall performance. As the access patterns change, the performance of the protocol will deviate from the optimal one.

It is, however, worth noting that the base station of a cell is able to keep track of the movement of the mobile computers into and out of the cell. It is because, if a mobile computer switches to another cell, it needs to do the handover procedure in order to maintain the access to the database. This provides a means for the base station to keep track of the movements of the connected computers.

Using this information, the base station in fact can re-execute our protocol so as to get an optimal result for the new configuration. One method to do so is to re-execute the Active strategy every time at the beginning of each broadcast. However, this places heavy burden on the base station. It may also not be cost-effective if the overall database access pattern does not change rapidly as compared to the rate of data broadcasts.

Another better way is to re-execute the Active strategy at the time *in need*, rather than for every broadcast. In fact, if the access pattern changes, the performance of the system will deviate from the expected value which have been computed by the base station in the last time of protocol execution. If this deviation is large, it is a very reliable hint for the base station to re-execute the Active strategy. By using this method, the near-optimal performance can be obtained at all time.

Chapter 4

Mobile Terminal Tracking

In this chapter, we explore the ways to reduce bandwidth overhead of the mobile terminal tracking process, which is indispensable for the wired network to locate a particular mobile computer inside a cellular network. We propose a new mechanism to predict the current location of the mobile computers at a higher accuracy. Incorporating with this prediction mechanism, we propose a dynamic programming approach in deriving an optimal broadcast protocol that can attain a minimal expected broadcast cost. Lastly, we also propose a new update protocol to lower the tracking cost of a mobile computer.

Our mechanism to predict the current location of a mobile computer is based on the uniformization method [Jen53, dSeSG92]. This method acts on a continuous-time Markov chain model whose state space consists of the cell locations that a mobile computer can be situated in. The result of this method will then be in the form of a vector of *location probabilities*, which are the probabilities that the mobile computer is situated in the corresponding cells. The method can capture the time-varying change in these location probabilities. As a result, a better accuracy for the prediction comparing to other schemes that only evaluate the stable state probabilities for the chain.

Given this improved prediction on the current location of a mobile user, we can devise an optimal broadcast protocol. Research efforts have reported to derive the optimal protocol [RY95] by transforming it into a continuous variable optimization problem. However, the complexity for solving the mathematical formulation is not yet known. Instead, we will propose a dynamic programming approach to derive the optimal set. The complexity for this algorithm can therefore be addressed.

Another contribution made is that we propose a new update protocol to trigger the mobile computer to send an update message to the base station. We have compared it with the common approaches, like the distance-based approach and the time-based approach, and find that our approach can reach a lower resource requirement. Also, this approach is more adaptive under different movement patterns of the mobile users.

This chapter is organized as follows. In Section 4.1, we present our mathematical model for the prediction of the current location of a mobile computer. The evaluation method will also be presented. Next, in Section 4.2, the problem on optimal paging is then analyzed and the algorithmic approach for deriving the optimal paging protocol is presented. In Section 4.3, we present the time-based update and the distance-based protocols. Also, we will use transient analysis to evaluate the protocol. In Section 4.4, we propose our new update protocol and the rationale behind it. Lastly, in Section 4.5, we carry out experiments and show the effectiveness of our protocol method as compared to the existing methods and proposals.

4.1 Movement Model

In this section, we present a movement model for a mobile computer inside a cellular network. Theoretically, the movement pattern of a mobile computer is independent of others. So, from now on, we consider only a single mobile computer moving inside the network. In our model, we consider a cellular network which consists of a collection $\mathcal{C}$ of N cells, which are labelled $c_1, c_2, \ldots, c_N$ respectively. The cells are closely packed in an orderly hexagonal fashion as shown in Figure 4.1. A mobile computer can move freely around the network. The network fully covers the whole geographic region so that the mobile computer will not lose connection to the network whenever it is inside the region.

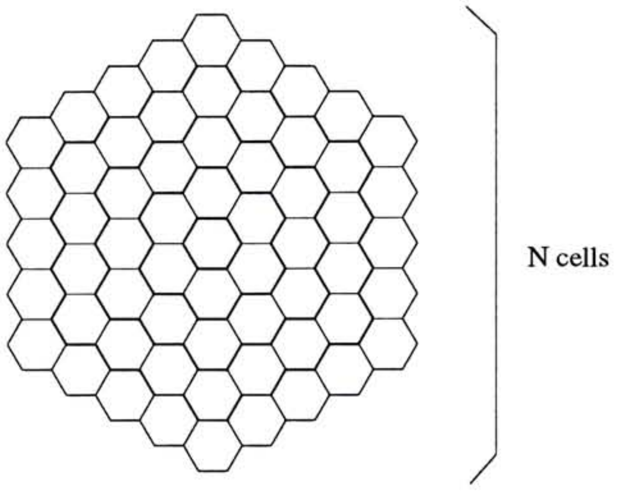


Figure 4.1: A Cellular Network Topology under our analysis

The mobility of this computer can be represented by an $N \times N$ *mobility matrix*

$\mathbf{Q} = [q_{ij}]$ where

$$q_{ij} = \begin{cases} \text{the } \textit{rate} \text{ at which the mobile computer moves from } c_i \text{ to } c_j & \text{for } i \neq j \\ -\sum\limits_{j:j\neq i} q_{ij} & \text{otherwise} \end{cases} \tag{4.1}$$

This matrix can be input as a user specification, gathered from statistics on the user's past routes, or a combination of the above two. The formulation of this mobility matrix allows us to model different kinds of movement patterns. Because each element in the matrix defines the rate at which the mobile computer crosses a cell boundary, the value of q_{ij} can be greater than zero only if the cells c_i and c_j are next to each other (or $i = j$). Under this restriction, the mobility matrix $\mathbf{Q}$ is always a sparse matrix. Although we formulate our mobility matrix using the hexagonal geometry for the cellular network, it can be easily seen that our model is general enough to model different geometric patterns of the cellular networks.

Note that from the definition of q_{ii}, the matrix $\mathbf{Q}$ becomes a transition rate matrix which governs a continuous time Markov process representing the movement of the mobile computer. Let this process be $\mathcal{X}_m = \{X_m(t) : t \geq 0\}$ whose state space consists of all possible cell locations $\{c_1, c_2, \ldots, c_N\}$ of the cellular network. The subscript m means that the last location update takes place at cell c_m, *i.e.* $X_m(0) = c_m$; and t is the time elapsed from the last location update.

Now we are going to derive the *transient* location probabilities for that Markov process. It is important, as in many location update schemes, the cost minimization is done using these probabilities. Let the transient location probability vector of the process $\mathcal{X}_m$ be $\boldsymbol{\pi}_m(t) = <\pi_{1|m}(t), \pi_{2|m}(t), \ldots, \pi_{N|m}(t)>$, where $\pi_{i|m}(t)$ is the probability that the computer is located at cell c_i at time t since the last report of location at cell c_m. Note that the initial probability vector $\boldsymbol{\pi}_m(0)$ is a unit vector where $\pi_{i|m}(0) = 0$ for all $i \neq m$ and $\pi_{m|m}(0) = 1$. It is important to point out that the Markov process will be restarted with a new initial probability vector whenever a location update takes place. In this way, it will become more accurate to predict the current location of the mobile computer as the probability vector is reset once new location information is obtained.

Using this notation, we mean that the location probabilities of a user only depends on the time t elapsed and at the last reporting cell c_m. This is in fact a simplified model. A more accurate model is to include the moving direction of the mobile computer. For example, along a two-way highway, a car moves in a predictable direction. It is very improbable that the car will move in opposite direction suddenly. By including the direction of movement into the state space of the process, we can then model this kind of information. However, the size of the state space will increase by $O(N_d)$, where N_d is the number of possible directions of movement. This increase the complexity of the computation as the mobility matrix will become $O(N_d^2)$ larger.

Here, we use the uniformization method [dSeSG96, Jen53], which is well-known for its efficiency and numerical stability, to find the location probabilities of the mobile computer at a particular time. In our case, we now have a continuous time Markov process $\mathcal{X}_m$ which is governed by the initial location probability vector $\boldsymbol{\pi}_m(0)$ and transition rate matrix $\mathbf{Q}$. By uniformization, $\mathcal{X}_m$ can be transformed into a discrete time Markov chain with the same state space $\mathcal{C}$ whose transition events are governed by a Poisson process with rate $\Lambda = \max\{-q_{ii} : 1 \leq i \leq N\}$. The step transition matrix for this Markov chain can be computed as $\mathbf{P} = \mathbf{I} + \mathbf{Q}/\Lambda$. By conditioning on the number of transitions over the time interval $(0, t)$, we can establish the following equations:

$$\boldsymbol{\pi}_m(t) = \sum_{n=0}^{\infty} e^{-\Lambda t} \frac{(\Lambda t)^n}{n!} \mathbf{p}_m(n) \tag{4.2}$$

where

$$\begin{cases} \mathbf{p}_m(n) = \mathbf{p}_m(n{-}1)\mathbf{P} \\ \mathbf{p}_m(0) = \boldsymbol{\pi}_m(0) \end{cases} \tag{4.3}$$

is the probability vector for the discrete Markov chain after n transitions. The element $p_{i|m}(n)$ of the vector $\mathbf{p}_m(n)$ is the probability that the mobile computer is located at cell c_i after n transitions since its last report of location at cell c_m.

Note that the infinite series causes difficulty in actual computation. To alleviate this problem, we do the summation up to a particular number of terms, say, M. The associated error bound $\varepsilon(M)$ for each element $\pi_{i|m}(t)$ in the probability vector is given by

$$\varepsilon(M) = \left\| \sum_{n=M}^{\infty} e^{-\Lambda t} \frac{(\Lambda t)^n}{n!} \mathbf{p}_m(n) \right\|_{\infty} \leq \sum_{n=0}^{\infty} e^{-\Lambda t} \frac{(\Lambda t)^n}{n!} = 1 - \sum_{n=0}^{M} e^{-\Lambda t} \frac{(\Lambda t)^n}{n!} \tag{4.4}$$

using the property that $||\mathbf{p}_m(n)||_\infty \leq 1$, where $||\cdot||_\infty$ is the max-norm of a vector. It is worthwhile to note that the error bound does not depend on the movement pattern of the user. Rather, it just depends on the maximum rate at which the user moves. Moreover, it is easy to precompute the number of terms required to attain a specified error bound. Therefore, we can tradeoff the prediction accuracy with processing time.

Lastly, we define $\boldsymbol{\pi} = <\pi_1, \pi_2, \ldots, \pi_N>$ to be the steady location probability vector of the mobile computer in the network, where π_m represents the steady probability that the mobile computer is at cell c_m. This probability vector, which is the solution to the equation $\boldsymbol{\pi} = \boldsymbol{\pi} P$, can be found by standard iterative methods [Ste94].

In a large personal communication network where the number of cells is large and the number of subscribers is huge, the size of the mobility matrix $\mathbf{Q}$ will then be very large. This requires large system overhead to do the prediction. To alleviate this problem, we may use only *a portion* of the matrix for computation. This can be done by realizing that the mobile computer cannot move too far away from the last reporting location before the next call arrives.

The reduction in the computation model can be performed as follows. Around the last reporting cell of the mobile computer, we designate an area of possible cell locations where the mobile computer can be located in until the next call arrives. This area can be organized in rings of cells. We specify this area by the center location (last reported cell of the computer) and the number of surrounding rings, D. In general, there are $K = 3(D+1)D+1$ cells in the area. We truncate the transition rate matrix Q and reformulate the Markov chain $\mathcal{X}_m$ such that the state space will only consist of the cells inside the area. Although computation error may be introduced in this reduced problem, it can be justified with a substantial decrease in computation requirements.

4.2 Optimal Paging

When a mobile computer has turned off its connection for a long time, the prediction of its location at the wired network system will become inaccurate. It may be due to the inaccuracy of the measurement on the movement statistics of the mobile user, the computation error at the wired network, and more likely, the abrupt change of the movement pattern of the mobile user. In any case, *paging* becomes a necessary procedure for the network system to locate the mobile computer in order to allow a connection to be re-established. In this paging process, the wired network system sends the polling signals over the cells through the wireless channel. If the target computer listens to this channel and detects the polling signal, it will respond to the wired network system by sending an acknowledgement message. Because paging consumes computer resources on system signalling and wireless communication, we want to minimize the number of cells to page.

In this section, we investigate a method to minimize the expected number of cells to page. We will use the location probability vector that can be obtained from the prediction method described in Section 4.1. Let the cells $c_1, c_2, \ldots, c_N$ be the N cells in the cellular network. We minimize the expected number of cells to page by using the location probability vector $\boldsymbol{\pi}_m(t) =<\pi_{1|m}(t), \pi_{2|m}(t), \ldots, \pi_{N|m}(t)>$, which is the vector when a call arrival at time t since the mobile computer report its location at cell c_m in the last time.

We also consider the paging delay bound, D, for paging, which models the maximum tolerable time the caller is willing to wait for. This delay bound is expressed in terms of polling cycles [AH95a]. In particular, the wired network is allowed to poll for mobile computer for at most D polling cycles. In each cycle, a group of cells will be paged simultaneously and then the wired network will wait for the response from the target mobile computer. If the computer has not replied for a specific time, another group of cells will be paged in the next cycle. This process continues until all the cells are paged or the mobile computer is found. In the former case, the paging process will report failure, meaning that the mobile computer cannot be contacted. Obviously, if the required delay bound is D, we can divide the cells by at most D groups, one for each polling cycle.

In order to minimize the number of cells to page, the network has to page the cells in non-increasing order of location probability. More formally, let the sequence $<i_1, i_2, \ldots, i_N>$ be the ordered sequence of indices to the location probability vector such that $\pi_{i_1|m}(t) \geq \pi_{i_2|m}(t) \geq \ldots \geq \pi_{i_N|m}(t)$. For a delay bound of D, we partition the cells into D groups by specifying $D-1$ indices, $l_1, l_2, \ldots, l_{D-1}$, to the ordered index sequence. In this way, the cells $<c_{i_1}, c_{i_2}, \ldots, c_{i_{l_1}}>$ form the first partition of cells and will be paged in the first polling cycle. The next partition is $<c_{i_{l_1}+1}, \ldots, c_{i_{l_2}}>$, then $<c_{s,l_2+1}, \ldots, c_{i_{l_3}}>$, and so on. The last partition is the cells $<c_{i_{l_{D-1}}+1}, \ldots, c_{i_N}>$. For notational convenience, we let $l_0 = 0$, $1 < l_1 \leq l_2 \leq \ldots \leq l_{D-1} \leq N$ and $l_D = N$. Let the expected number of cells to be paged by the wired network be $P(t, D, N)$. This function can be defined as

$$P(t, D, N) = \sum_{j=1}^{D} l_j \sum_{k=l_{j-1}+1}^{l_j} \pi_{i_k}(t). \tag{4.5}$$

Our aim here is to find the values of indices $l_1, l_2, \ldots, l_{D-1}$ such that $P(t, D, N)$ attains its minimum value, which we denote as $P_{min}(t, D, N)$. Note that the above re-ordering of cells is a requirement for the paging overhead to be minimized. It can be proved as follows.

Theorem 4 *The configuration of the cells in non-increasing order of location probability is a necessary requirement for the number of cell to be paged by the wired system to be minimum.*

Proof: We prove it by contradiction. Suppose that there exists an optimal paging process that does not conform to the high-probability-first ordering. In other words, there is an index sequence $<i_1, i_2, \ldots, i_N>$ such that there exists a pair of indices, a and b such that $a < b$ and $\pi_{i_a|m} < \pi_{i_b|m}$. Also assume that we have found the values for the indices $l_1, l_2, \ldots, l_{D-1}$ such that the cost function in Equation (4.5) is now minimum. We analyze this paging process in two cases.

Case 1: Suppose that the cells c_{i_a} and c_{i_b} are in the same partition, say in the partition $<c_{l_{n-1}+1}, \ldots, c_{l_n}>$ for some $1 \leq n \leq D$. In this case, if we swap the indices i_a and i_b in the index sequence, the cost $P(t, D, N)$ will remain the same, since the value of the term $l_n \sum_{k=l_{n-1}+1}^{l_n} \pi_{i_k|m}(t)$ in Equation (4.5 will remain the same.

Case 2: Suppose that the cells c_{i_a} and c_{i_b} are not in the same partition. In other words, c_{i_a} is in the partition $<c_{l_{m-1}+1}, \ldots, c_{l_m}>$ while c_{i_b} is in $<c_{l_{n-1}+1}, \ldots, c_{l_n}>$ for some $1 \leq m < n \leq D$. In this case, if we swap the indices i_a and i_b in the index sequence, the change in the cost function (Equation (4.5)) will be given by

$$\begin{aligned} & \left[l_m \pi_{i_a|m}(t) + l_n \pi_{i_b|m}(t)\right] - \left[l_m \pi_{i_b|m}(t) + l_n \pi_{i_a|m}(t)\right] \\ = & \quad (l_n - l_m)(\pi_{i_b|m}(t) - \pi_{i_a|m}(t)) \end{aligned} \tag{4.6}$$

Since $\pi_{i_b|m}(t) > \pi_{i_a|m}(t)$ and, from $m < n$, $l_m \leq l_n$, the above difference is greater than or equal to zero. In other words, we can obtain a lower value for the cost function $P(t, D, N)$ by exchanging the indices i_a and i_b in the index sequence.

In conclusion, by swapping the indices i_a and i_b in the index sequence, we can obtain equal or even lower value for the cost function $P(t, D, N)$. This contradicts with the pre-supposition that a minimum cost can be attained even if $\pi_{i_a|m} < \pi_{i_b|m}$ for some $a < b$. Therefore, the theorem is proved. ■

Now, we are going to devise methods to find the values of $l_1, l_2, \ldots, l_{D-1}$ such that $P(t, D, N)$ attains its minimum value, $P_{min}(t, D, N)$. The assignment to the values of $l_1, l_2, \ldots, l_{D-1}$ means that we partition the cells into D groups. However, in some cases, we can partition the cells into less than D groups and the minimum cost is reached simultaneously. In order to capture this characteristic, some of the indices can have the value N and so some empty partitions are created.

One simple method to find the values for the indices is evaluate the cost function for all possible values $l_1, l_2, \ldots, l_{D-1}$ exhaustively. However, this method has a very high complexity, as shown by the following theorem.

Theorem 5 *The number of possible assignments of values to the indices $l_1, l_2, \ldots, l_{D-1}$ is exponential in N.*

Proof: In this assignment problem, we have to assign $D - 1$ indices with $N - 1$

possible values $(2, 3, \ldots, N$, assuming that $N \geq 2$ for the problem to be meaningful). In general, there may be d partitions of cells, where $d \leq D$ for the delay bound D. In this regard, $1 < l_1 < l_2 < \ldots < l_{d-1} < N$ and $l_d, l_{d+1}, \ldots, l_{D-1}$ are all equal to N. For the first $d-1$ indices, there are thus $\binom{N-2}{d-1}$ combinations for values. Now, we vary the value of d from 1 to D, we have the following expression for the number of possible combination of values for the indices $l_1, l_2, \ldots, l_{D-1}$:

$$\sum_{i=1}^{D} \begin{pmatrix} N-2 \\ i-1 \end{pmatrix} = \sum_{i=0}^{D-1} \begin{pmatrix} N-2 \\ i \end{pmatrix} \tag{4.7}$$

which is exponential in N if the values of D approaches N. ■

In order to reduce the complexity, we propose a new method to evaluate the indices. This method uses the dynamic programming approach to solve the problem. Consider the simplest case where $D = 1$. In this case, the wired network can only page in one polling cycle and therefore only one partition can be formed. As a result, the expected number of cells to page should be equal to the total number of cells in the wireless network, N. Mathematically, we have

$$P_{min}(t, 1, N) = N \tag{4.8}$$

Consider the other case where $D > 1$. In this case, we divide the problem into two parts. The first part is the problem of dividing the first k cells in the sequence $< c_{i_1}, c_{i_2}, \ldots, c_{i_N} >$ into $D-1$ partitions. This is equivalent to finding the value of $P_{min}(t, D-1, k)$. The second part is to form *one* partition for the last, remaining $N-k$ cells. Under this formulation, if the target mobile computer is inside one of the first k cells, then the expected minimum number of cells to page is $P_{min}(t, D\text{-}1, k)$. On the other hand, if the computer is in one of the last $N-k$ cells, the number of cells to page is N. By total probability, the expected minimum number of cells to page can then be given by

$$P_{min}(t, D, N) = \min_{0<k\leq N} \left\{ P_{min}(t, D-1, k) + N \sum_{j=k+1}^{N} \pi_{i_j|m}(t) \right\} \tag{4.9}$$

Therefore, we can obtain the minimum expected number of cells to page by using this recursive equation as well as the initial condition (Equation (4.8)). Moreover, the set of indices $l_1, l_2, \ldots, l_{D-1}$ can be obtained by backtracking the recursion. The complexity of this dynamic programming method is $O(ND)$, as proved by the following theorem.

Theorem 6 *The complexity of the recursive formulation Equations (4.8) and (4.9) is $O(ND)$.*

Proof: In Equation (4.9), the summation $\sum_{j=k+1}^{N} \pi_{i_j|m}(t)$ can be evaluated beforehand and stored in a lookup table. It can be done complexity of $O(N)$. For the recursion, in order to evaluate the minimal cost, we need to evaluate the function for all values of k. Therefore, the complexity for one recursive step is $O(N)$. Lastly, there are D recursive steps in the evaluation. Therefore, the complexity for the whole algorithm is $O(N) + O(ND) = O(ND)$, which is significantly lower than the exhaustive searching method. ■

Therefore, using our algorithm, we can derive the paging procedure efficiently. In particular, given a specific probability vector $\boldsymbol{\pi}_m$, the minimum number of cells to page is the highest when the delay bound D is 1, at which the number of cells to page is N, since only one partition can be formed. Another special case is that there is no delay bound, *i.e.* $D = \infty$. In this case, the expected number of cells to page is minimum. This can be achieved by paging the cells one by one, in descending order of location probability. The minimum cost can be given by

$$P_{min}(t, \infty, N) = \sum_{k=1}^{D} k\pi_{i_k|m}(t) \tag{4.10}$$

However, the maximum delay by this paging method will be N. This is intolerable if the wireless network is large and there are many cells in the network.

4.3 Transient Analysis

In this section, we present two common protocols, namely the time-based protocol [Ros96] and the distance-based protocol [HA95b]. Furthermore, we demonstrate the practicability of employing transient analysis to evaluate the performance of these protocols. Figure 4.2 shows the general location update activity of a mobile computer.

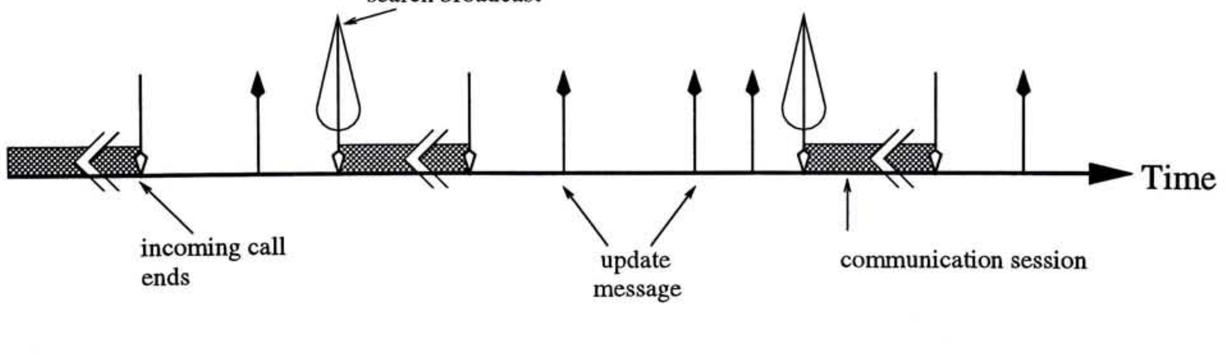


Figure 4.2: Location Update Activity of a Mobile Computer

In this figure, the shaded regions are the durations in which the mobile computer has established a connection with the wired network subsystem. We do not address the mobility management in this region. It is related to the handover problem [Pol96] which accounts for the communication throughput and reliability.

Instead, we focus on the operations on the non-shaded region. Each of these regions begins at the end of a communication session of the mobile computer and the wired system. It lasts until another incoming call arrives or the mobile computer tries to contact the base station spontaneously. In the former case, the wired network subsystem will search for the mobile computer inside the cellular network, *i.e.* paging is carried out, which has already been described in the last section. Note that it will result in a system overhead that is related to the location probabilities of the mobile computer at the time of paging. For the latter case, on the other hand, only the communication cost from the mobile computer is induced. We ignore this case because, it corresponds to the communication establishment issue rather than mobile terminal tracking.

One extreme case is that the mobile computer does not send any update message to the base station. If the disconnected period between two consecutive communication sessions is long, the wired network subsystem will no longer be able to predict the current location of the mobile computer at a reasonable accuracy. Moreover, the distribution of the location probabilities among different cells may become less skew. These make the *actual* paging cost higher.

In the other extreme, a mobile computer sends update messages to the wired network whenever it changes its current location. This method reduces the paging call to the minimum. Nevertheless, the sending of location update messages also incurs an *update cost*, which accounts for the power consumption at the computer side, the overhead of the communication channel, and the processing at the static system. Obviously, this protocol increases the update cost to the maximum. No matter which of these two extremes is better, it is very likely that we can reach a lower cost by varying the frequency of the location updates by the mobile computer.

There are many location management protocols [MHS94, BNK93, BNKS94] in which a mobile computer does not need to send a location update message for every change in its cell location. Therefore, the wired system has to page for the mobile computer, as described in the previous section. As the time since the last location update increases, the prediction will become inaccurate. As a result, the cost of paging will also become higher. To solve this problem, the mobile computer will continually send some update messages to the wired system, indicating that its current location is at cell, say, c_m. In this way, the Markov process for location prediction is reinitialized and the location probability vector $\boldsymbol{\pi}_m(0)$ is reset to $\pi_{m|m}(0) = 1$ and $\pi_{i|m}(0) = 0$ for all

$i \neq m$. Therefore, we can maintain the accuracy of the prediction at the wired network and the skewness of the distribution of the location probabilities among different cells. The paging cost at the next arrival of incoming call can then be lowered.

Nowadays, different protocols are proposed to balance the tradeoffs between the update cost and the searching cost so that the overall cost can be minimized. Here, we will investigate two commonly proposed protocols, the time-based protocol and the distance-based protocol. For each of these protocols, we illustrate the use of transient analysis to evaluate their performance.

In these analyses, we make a simplification that we use a registration area approach to do paging. Specifically, a registration area is associated with the mobile computer. This area is assigned around the last reporting location of the mobile computer. The paging process proceeds in two phases. In the first phase, the cells inside the registration area are paged first. If the mobile computer is not found, all other cells will be paged in the second phase. This paging method is similar to the one used in the GSM standard, except that we assign the registration area dynamically depending on the last reporting cell of the mobile computer.

Here, we also make a simplification that the system overhead of paging one cell is a constant, C; and the paging cost is additive in terms of the number of cells being paged by the wired system. In general, the paging cost also includes the wireless communication induced, the signalling overhead inside the wired system, and the database transaction overhead at the HLR and VLRs. A more accurate cost model is the one that depends on the distance of the cell paged from the last reporting cell of the mobile computer. As we will see later, our analysis can be easily extended to this general model.

Since we use the registration area approach for paging here, the paging cost can vary with the size of the registration area. For an area of with distance less than or equal to d from the last reporting cell location, there are $3d(d+1)+1$ cells in the area. Therefore, the cost of paging this area is $(3d(d+1)+1)C$. On the other hand, if the mobile computer is not found in the first phase of paging, the rest of the cellular network will be paged. This results in a paging cost of NC, where N is the total number of cells in the network.

Another cost for mobile terminal tracking is the update cost, U, which is the cost for the mobile computer to send the location update message to the wired system as well as the system cost for the wired network to handle the update message.

In the next subsections, we present and analyze the time-based protocol and the distance-based protocol respectively. The performance metric we use here is the *cost* *rate*, which is the cost expended per unit time for the mobile terminal tracking.

4.3.1 The Time-Based Protocol

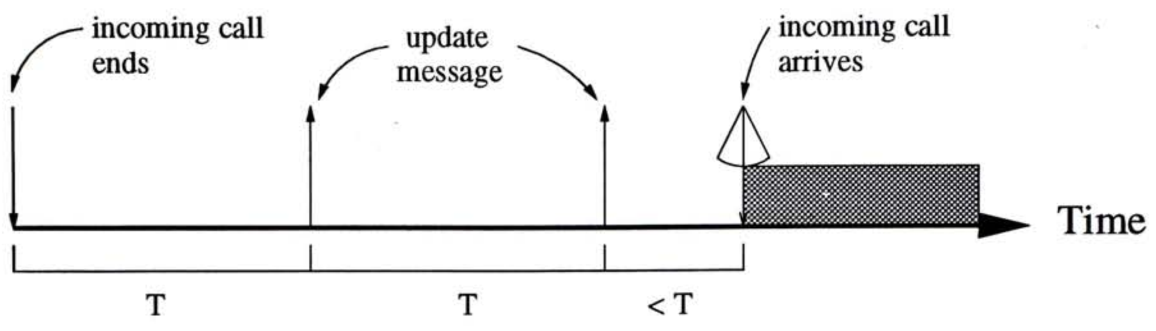


Figure 4.3: Illustration for the Time-Based Protocol

The time-based protocol is illustrated in Figure 4.3. In this protocol, the time domain is divided into consecutive periods. Each period is initiated by an update event, which can be an end of a call session or a sending of a location update message by the mobile computer. It is terminated when another location update message is sent or paging from the wired system occurs.

In this time-based protocol, a time limit T is specified *a priori* for limiting the period length. If the period lasts for this time limit without any paging event, the mobile computer will send an update message to the wired network. As a result, a new period is initiated and another Markov process with a new reporting cell is generated for the location prediction.

On the other hand, if there is an incoming call arriving at the wired network before the time limit, the network subsystem will page for the mobile computer. When the mobile computer receives this paging signal, it will reply the subsystem, and a communication session will be started. This period ends with a paging cost.

One obviously advantage for this protocol is simplicity. We need only to implement a timer inside a mobile computer so as to trigger the events of location updates. However, this protocol has some deficiencies. Consider the simple case that a mobile computer moves inside a cellular network symmetrically and uniformly, *i.e.* moves to any one of the neighboring cells with equal probability and at the same rate over all the network. In this case, we can optimize it by identifying an optimal time limit T.

However, if the mobile computer moves at a higher rate, the expected distance traveled by the mobile computer between two subsequent location updates will be larger. Consequently, the location probability vector and the update cost will change.

In this regard, we need to readjust the value of the time limit. In general, moreover, a mobile computer may have non-uniform movement pattern. Therefore, we need to have a separate optimal value for the mobile computer at each of the cell in the network. This makes the implementation more complicated. More importantly, as the movement pattern of the mobile computer changes over time, the system has to re-compute all the optimal values. This increases the maintenance cost vastly.

Transient Analysis of the Time-Based Protocol

In our derivation of the cost rate, we consider a single time period that begins with a location update and ends with either another location update or paging. We will first derive the expected tracking cost of the period and then the expected length duration. The expected cost rate is then the ratio of the expected cost and the period length.

If the time period terminates with a location update, *i.e.* the mobile computer sends an update message to the base station, then an update cost of U is incurred. This period will also ends at time T, which is the time limit for the time-based protocol.

In another case, if there is a call arrives at time t $(0 < t \leq T)$ in the period, the static network will page for the mobile computer. If the mobile computer remains inside the location area, the cost of polling is KC, where C is the overhead for polling a single cell, where $K = 3(d+1)d+1$. However, if the computer is outside the location area, the *overall* cost of polling for the computer is NC. Let $\mathbf{r}_m = <r_{i|m} : i = 1, 2, \ldots, N>$ be a column vector such that $r_{i|m} = 1$ if cell c_i is at a distance less than or equal to d from c_m, and $r_{i|m} = 0$ otherwise. Using the location probabilities derived previously, the probability of the mobile computer staying inside the area is $\boldsymbol{\pi}_m(t)\mathbf{r}_m$ and that of outside the area is $1 - \boldsymbol{\pi}_m(t)\mathbf{r}_m$. Therefore the polling cost when an incoming call arrives at time t is

$$KC[\boldsymbol{\pi}_m(t)\mathbf{r}_m] + NC[1 - \boldsymbol{\pi}_m(t)\mathbf{r}_m] = NC - (N{-}K)C\boldsymbol{\pi}_m(t)\mathbf{r}_m. \tag{4.11}$$

Assume that the arrivals of the incoming calls form an indepedent Poisson process with parameter μ. The probability distribution function of the time for the call arrival is $P(t) = 1 - e^{-\mu t}$. Subsequently, the probability that the time period terminates with an incoming call is given by $1 - e^{-\mu T}$, while the probability that the period terminates with the delivery of an update message by the mobile computer is $e^{-\mu T}$, where T is the time limit for the time-based protocol. Unconditioning update and paging cases, we have the expected cost of tracking the mobile computer for one single period:

$$C_m^T(T) = Ue^{-\mu T} + \int_0^T \{NC - (N{-}K)C\boldsymbol{\pi}_m(t)\mathbf{r}_m\}\, \mu e^{-\mu t}\, dt \tag{4.12}$$

Substitute Equation (4.3) into the above formula and then rearrange the expression, we now have

$$\begin{aligned}
C_m^T(T) &= Ue^{-\mu T} + \int_0^T \left\{ NC - (N{-}K)C \left[\sum_{n=0}^{\infty} e^{-\Lambda t} \frac{(\Lambda t)^n}{n!} \mathbf{p}_m(n)\mathbf{r}_m \right] \right\} \mu e^{-\mu t}\, dt \\
&= Ue^{-\mu T} + NC(1 - e^{-\mu T}) \\
&\quad -(N{-}K)C \sum_{n=0}^{\infty} \left[\int_0^T e^{-\Lambda t} \frac{(\Lambda t)^n}{n!} \mu e^{-\mu t}\, dt \right] \mathbf{p}_m(n)\mathbf{r}_m \\
&= Ue^{-\mu T} + NC(1 - e^{-\mu T}) \\
&\quad -(N{-}K)C\mu \sum_{n=0}^{\infty} \left(\frac{\Lambda}{\Lambda+\mu} \right)^n \left[\int_0^T e^{-(\Lambda+\mu)t} \frac{[(\Lambda+\mu)t]^n}{n!}\, dt \right] \mathbf{p}_m(n)\mathbf{r}_m.
\end{aligned} \tag{4.13}$$

Noting that

$$(\Lambda+\mu) e^{-(\Lambda+\mu)t} \frac{[(\Lambda+\mu)t]^n}{n!} \tag{4.14}$$

is the density function for an $(n+1)$-th stage Erlang distribution, we can arrive at the following identity:

$$\int_0^T e^{-(\Lambda+\mu)t} \frac{[(\Lambda+\mu)t]^n}{n!}\, dt = \sum_{i=n+1}^{\infty} \frac{[(\Lambda+\mu)t]^i}{i!} \cdot \frac{e^{-(\Lambda+\mu)t}}{\Lambda+\mu} \tag{4.15}$$

As a result, we now have

$$\begin{aligned}
C_m^T(T) &= Ue^{-\mu T} + NC(1 - e^{-\mu T}) \\
&\quad -(N{-}K)C\mu \sum_{n=0}^{\infty} \left(\frac{\Lambda}{\Lambda+\mu} \right)^n \left[\sum_{j=n+1}^{\infty} \frac{[(\Lambda+\mu)T]^j}{j!} \cdot \frac{e^{-(\Lambda+\mu)T}}{\Lambda+\mu} \right] \mathbf{p}_m(n)\mathbf{r}_m \\
&= Ue^{-\mu T} + NC(1 - e^{-\mu T}) \\
&\quad -(N{-}K)C\mu \sum_{n=0}^{\infty} \sum_{j=n+1}^{\infty} \left(\frac{\Lambda}{\Lambda+\mu} \right)^n \left[\frac{[(\Lambda+\mu)T]^j}{j!} \cdot \frac{e^{-(\Lambda+\mu)T}}{\Lambda+\mu} \right] \mathbf{p}_m(n)\mathbf{r}_m \\
&= Ue^{-\mu T} + NC(1 - e^{-\mu T}) \\
&\quad -(N{-}K)C\mu \sum_{j=1}^{\infty} \sum_{n=0}^{j-1} \left(\frac{\Lambda}{\Lambda+\mu} \right)^n \left[\frac{[(\Lambda+\mu)T]^j}{j!} \cdot \frac{e^{-(\Lambda+\mu)T}}{\Lambda+\mu} \right] \mathbf{p}_m(n)\mathbf{r}_m \\
&= Ue^{-\mu T} + NC(1 - e^{-\mu T}) \\
&\quad -(N{-}K)C\mu \sum_{n=1}^{\infty} \sum_{j=0}^{n-1} \left(\frac{\Lambda}{\Lambda+\mu} \right)^j \left[\frac{[(\Lambda+\mu)T]^n}{n!} \cdot \frac{e^{-(\Lambda+\mu)T}}{\Lambda+\mu} \right] \mathbf{p}_m(j)\mathbf{r}_m \\
&= Ue^{-\mu T} + NC(1 - e^{-\mu T}) \\
&\quad -(N{-}K)C\mu \sum_{n=0}^{\infty} \sum_{j=0}^{n} \left(\frac{\Lambda}{\Lambda+\mu} \right)^j \left[\frac{[(\Lambda+\mu)T]^{(n+1)}}{(n+1)!} \cdot \frac{e^{-(\Lambda+\mu)T}}{\Lambda+\mu} \right] \mathbf{p}_m(j)\mathbf{r}_m
\end{aligned}$$

$$\begin{aligned} &= Ue^{-\mu T} + NC(1-e^{-\mu T}) \\ &\quad -(N\!-\!K)C\mu T\sum_{n=0}^{\infty} e^{-(\Lambda+\mu)T}\frac{[(\Lambda+\mu)T]^n}{n!}\left[\frac{\sum_{j=0}^{n}\frac{\Lambda^j}{(\Lambda+\mu)^j}\mathbf{p}_m(j)\mathbf{r}_m}{n+1}\right]. \end{aligned} \quad (4.16)$$

Lastly, this can be written in the following form:

$$\begin{aligned} C_m^T(T) &= Ue^{-\mu T} + NC(1-e^{-\mu T}) \\ &\quad -(N\!-\!K)C\mu T\sum_{n=0}^{\infty} e^{-(\Lambda+\mu)T}\frac{[(\Lambda+\mu)T]^n}{n!}f(n) \end{aligned} \quad (4.17)$$

where $f(n)$ can be evaluated recursively as

$$\begin{cases} f(n+1) = \dfrac{n+1}{n+2}f(n) + \left(\dfrac{\Lambda}{\Lambda+\mu}\right)^{n+1}\dfrac{\mathbf{p}_m(n+1)\mathbf{r}_m}{n+2} \\ f(0) = \mathbf{p}_m(0)\mathbf{r}_m \end{cases}. \quad (4.18)$$

For the duration of the time period, if the period ends with a location update, the period should be T. Otherwise, if the period ends with an arriving call, the length t of the period can be defined by the probability density function $p(t) = \mu e^{-\mu t}$. Therefore, the distribution of the duration of the time period is exponential truncated at T with rate μ; and the expected duration of the period is given by

$$D_m^T(T) = \int_0^T t\mu e^{-\mu t}dt + Te^{-\mu T} = \frac{1-e^{-\mu T}}{\mu}. \quad (4.19)$$

Note that both the expected cost and the expected duration of each time period are conditioned on the last reporting cell location, c_m, of the mobile computer. By removing this condition, we have the *expected cost rate* for this time based protocol with parameter T as the following:

$$E_T(T) = \frac{\sum_{m=1}^{N} C_m^T(T)\pi_m}{\sum_{m=1}^{N} D_m^T(T)\pi_m} \quad (4.20)$$

In this protocol, we want to find an optimal parameter T^* such that the average expected cost $E_T(T^*)$ is minimized. However, in our mobility model, we allow a non-uniform computer movement pattern over the cellular network. Instead of finding a single optimal value of the time limit for this protocol, it would be theoretically better to find an optimal vector of time limits $\mathbf{T}^* = <T_i^* : 1 \le i \le N>$ such that each element T_i^* represents the optimal value when the mobile computer is in cell c_i, *i.e.* $C_i^T(T_i^*)/D_i^T(T_i^*)$ is minimum for all $1 \le i \le N$. These values can be found for each cell separately because they are independent of each other.

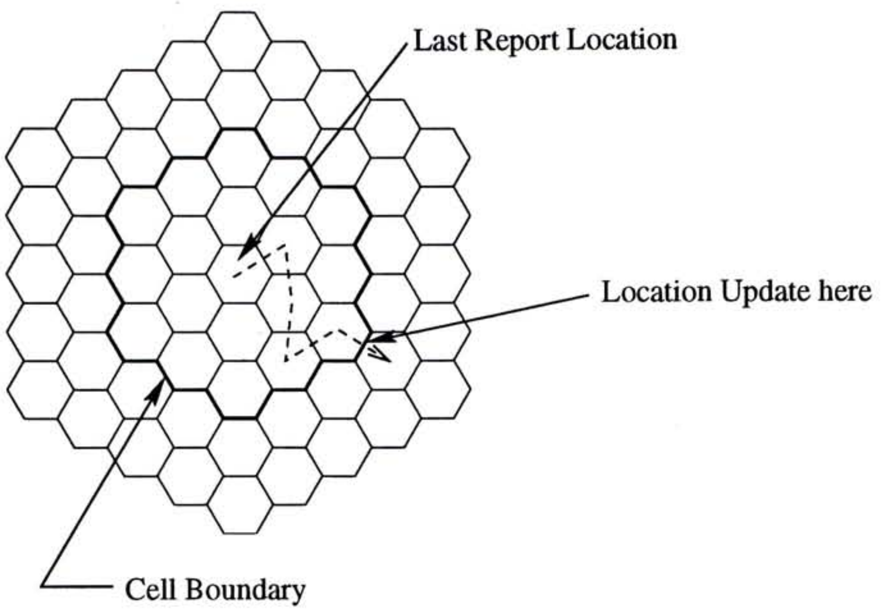


Figure 4.4: Illustration for the Distance-Based Protocol

4.3.2 Distance-Based Protocol

Figure 4.4 illustrates the distance-based protocol. In this protocol, the mobile computer will listen to the broadcast channel from the base station to locate itself inside the cellular network, even though there is not a communication channel between them. A distance limit is defined in this protocol. When the mobile computer detects that it has gone farther away from the last reporting location than the distance limit, it will send a location update message. In other words, we can consider it as a registration area assigned around the mobile computer's last reporting cell.

When an incoming call arrives, the wired system will page over the cellular network. On the other hand, we can see that in this protocol, the mobile computer will definitely be inside the registration area since if it were not, an update notification will be sent. Therefore, it is not necessary for the wired system to page outside the location area. In this regard, the paging cost can be lowered. Note that it the distance limit is set to zero, this protocol will become the always-update protocol.

All in all, the distance-based protocol has an advantage over the time-based protocol, which is that the mobile computer is within the registration area. This can lower the paging cost at the wired network substantially. However, it still inherits the disadvantages of the time-based protocol. The distance limit, which is the optimization heuristic in the distance-based protocol, varies with the movement pattern as well as the moving rate of the mobile computer. It means that we need different values of the distance limit at different reporting cells in the network.

Transient Analysis of the Distance-based Protocol

The general method for the mathematical derivation of the cost rate is the same as the one presented in the time-based protocol. We derive the expected cost rate by finding the expected cost for a single period and then the expected length of the duration of the period.

The probability distribution of the event of location update, however, is not as simple as the one in the time-based protocol, which is the truncated exponential distribution in the time-based protocol. In this distance-based protocol, the criterion for the mobile computer to send an update message is for the computer to move away from its last reporting cell by more than a distance limit apart. In our mobility model, we use the cell location of the mobile computer as the state in the Markov process. Therefore, the location update can be viewed as the event that the Markov process changes from one state, which correspond to a cell location within the distance bound, to another state, which correspond to a cell location that is out of the bound.

With this insight, we can model the time for the location update as the *first passage time* for the Markov process to change some states (cells within the distance bound) to some other states (cell out of the distance bound). The distribution of the time is a *phase distribution* [Nel95], of which the probability density function is denoted as $f_m(t)$. By deriving the expression for $f_m(t)$, we can get the expression for the expected cost for a location update period, $C_m^D(d)$, given that the last location update is at cell c_m. The expression is as shown in the following:

$$C_m^D(d) = \int_0^\infty U f_m(t) e^{-\mu t} dt + \int_0^\infty KC\mu e^{-\mu t}[1 - F_m(t)]dt \qquad (4.21)$$

where $F_m(t)$ is the probability distribution function for $f_m(t)$. The first term is the expected cost for location update, and the second term is the expected cost for paging. As we have assumed in the analysis of the time-based method, the incoming calls form an independent Poisson process with the parameter μ. Therefore, the first term of the above expression denotes the case that there is no call arrived at the period and the period ends with a location update, which results in an update cost U. While the second term corresponds to the case that there is an arriving call and there is no location update during the period. For this case, the paging cost is KC, where $K = 3d(d+1)+1$ is the number of cells within the distance limit. Note that in the distance-based protocol, the mobile computer should be within the distance limit. Therefore, the wired system needs not page the cells outside the region. Using similar argument, we can also derive the expected length, $D_m^D(d)$, of the period given that

that the last reporting cell is c_m. The expression is

$$D_m^D(d) = \int_0^\infty t f_m(t) e^{-\mu t} dt + \int_0^\infty t \mu e^{-\mu t}[1 - F_m(t)]dt. \qquad (4.22)$$

Note that the phase distribution function $f_m(t)$ also depends on the last reporting cell. The reason can be easily followed from the fact that, in the distance-based protocol, the time until the next location update is dependent on the mobility of the mobile computer. Because our movement model allows different movement patterns across the geographical area, this mobility in turn depends on the starting location, c_m, of mobile computer. Now, the remaining task is to find the expression of $f_m(t)$.

The phase distribution function can be derived as follows. From our movement model described in Section 4.1, we reconstructed a new continuous time Markov process of which the state space consists of $K+1$ states, where $K = 3d(d+1)+1$ is the number of cells inside the registration area. The first K states correspond to the K possible cell locations in $\mathcal{R}_m = \{c_{r_1}, c_{r_2}, \ldots, c_{r_K}\}$ within a distance limit d from c_m. The last state corresponds to the case that the mobile computer is outside the registration area. The time at which the mobile computer leaves location area can then be determined as the first passage time to the (K+1)-st state in this new process.

From the original mobility matrix $\mathbf{Q}$, we construct a truncated version of the transition rate matrix $\mathbf{R}_m = [r_{ij}^m]$ for this new Markov chain. Without loss of generality, let the first K states be $s_1^m, s_2^m, \ldots, s_K^m$ which correspond to K cell locations $c_{r_1}, c_{r_2}, \ldots, c_{r_K}$ respectively, and the extra state s_{K+1}^m be the state of the mobile computer is out of the distance bound. The transition rate matrix $\mathbf{R}_m$ for this chain has the following form:

$$\mathbf{R}_m = \begin{bmatrix} \mathbf{U}_m & \mathbf{A}_m \\ \mathbf{0} & 0 \end{bmatrix} \qquad (4.23)$$

where $\mathbf{U}_m = [u_{ij}^m]$ is a $K \times K$ matrix whose elements correspond to the rates at which the mobile computer move across different cells within the registration area. Therefore, $u_{ij}^m = q_{r_i,r_j}$ in $\mathbf{Q}$ for $i \neq j$. The matrix $\mathbf{A}_m = [a_i^m]$ is a K-element column vector where a_i^m is the rate at which the mobile computer goes out of the registration area from cell c_{r_i}, *i.e.*

$$a_i^m = \sum_{k: c_k \notin \mathcal{R}_m} q_{r_i,k} \qquad (4.24)$$

In this analysis, we are only interested in the movement of the mobile computer when it is inside the location area. Therefore, we can construct this Markov chain by reducing the state of the mobile computer being out of the distance limit be an *absorbing state.* Using this construction, the elements in the last row, which correspond to the outgoing rate of absorbing state, s_{K+1}^m, are all zeros.

Using uniformization, we transform the above Markov process into a discrete version and get the step transition matrix $\mathbf{S}_m = \mathbf{I} + \mathbf{R}_m/\Lambda_m$ for the new discrete chain, where $\Lambda_m = \max\{-u_{ii} : i = 1, 2, \ldots, K\}$. $\mathbf{S}_m$ will then have the following form:

$$\mathbf{S}_m = \begin{bmatrix} \mathbf{V}_m & \mathbf{B}_m \\ \mathbf{0} & 1 \end{bmatrix} \tag{4.25}$$

Let $\boldsymbol{\pi}'_m(t) = <\pi'_{i|m}(t) : 1 \leq i \leq K>$ be the location probability vector whose element $\pi'_{i|m}(t)$ represents the probability that the mobile computer is located inside cell c_{r_i} in the location area at time t. This probability is given by

$$\boldsymbol{\pi}'_m(t) = \sum_{n=0}^{\infty} e^{-\Lambda_m t} \frac{(\Lambda_m t)^n}{n!} \boldsymbol{\pi}'_m(0) \mathbf{V}_m^n. \tag{4.26}$$

The probability distribution function $F_m(t)$ of the time until the location of the mobile computer exceeds the distance bound (*i.e.* going to the absorbing state, s_{K+1}) is defined by

$$F_m(t) \equiv 1 - \boldsymbol{\pi}'_m(t)\mathbf{e} = 1 - \sum_{n=0}^{\infty} e^{-\Lambda_m t} \frac{(\Lambda_m t)^n}{n!} \boldsymbol{\pi}'_m(0) \mathbf{V}_m^n \mathbf{e}. \tag{4.27}$$

where $\mathbf{e}$ is a column vector whose elements are all 1's. Differentiate Equation (4.27), we get the probability density function as follows:

$$\begin{aligned}
f_m(t) &= \frac{\partial}{\partial t}\left\{1 - \sum_{n=0}^{\infty} e^{-\Lambda_m t} \frac{(\Lambda_m t)^n}{n!} \boldsymbol{\pi}'_m(0)\mathbf{V}_m^n \mathbf{e}\right\} \\
&= \frac{\partial}{\partial t}\left\{1 - e^{-\Lambda_m t}\boldsymbol{\pi}'_m(0)\mathbf{e} - \sum_{n=1}^{\infty} e^{-\Lambda_m t} \frac{(\Lambda_m t)^n}{n!} \boldsymbol{\pi}'_m(0)\mathbf{V}_m^n \mathbf{e}\right\} \\
&= \Lambda_m e^{-\Lambda_m t}\boldsymbol{\pi}'_m(0)\mathbf{e} - \sum_{n=1}^{\infty} \frac{\partial}{\partial t}\left\{e^{-\Lambda_m t}\frac{(\Lambda_m t)^n}{n!}\right\}\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n \mathbf{e} \\
&= \Lambda_m e^{-\Lambda_m t}\boldsymbol{\pi}'_m(0)\mathbf{e} + \sum_{n=1}^{\infty}\left\{\Lambda_m e^{-\Lambda_m t}\frac{(\Lambda_m t)^n}{n!} - \Lambda_m e^{-\Lambda_m t}\frac{(\Lambda_m t)^{n-1}}{(n-1)!}\right\}\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n \mathbf{e} \\
&= \sum_{n=0}^{\infty}\Lambda_m e^{-\Lambda_m t}\frac{(\Lambda_m t)^n}{n!}\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n \mathbf{e} - \sum_{n=0}^{\infty}\Lambda_m e^{-\Lambda_m t}\frac{(\Lambda_m t)^n}{n!}\boldsymbol{\pi}'_m(0)\mathbf{V}_m^{n+1} \mathbf{e} \\
&= \sum_{n=0}^{\infty}\Lambda_m e^{-\Lambda_m t}\frac{(\Lambda_m t)^n}{n!}\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n(\mathbf{I} - \mathbf{V}_m)\mathbf{e} \\
&= \sum_{n=0}^{\infty}\Lambda_m e^{-\Lambda_m t}\frac{(\Lambda_m t)^n}{n!}\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n \mathbf{B}_m
\end{aligned} \tag{4.28}$$

where $\mathbf{B}_m = (\mathbf{I} - \mathbf{V}_m)\mathbf{e}$ can be followed from the general property of the stochastic matrix $\mathbf{S}_m$ that the sum of elements in a row is always equal to 1.

Substitute Equations (4.27) and (4.28) into (4.21) and do the integration, we arrive at the following equation for the expected cost in the period:

$$\begin{aligned}
C_m^D(d) &= \int_0^\infty U\left[\sum_{n=0}^{\infty} \Lambda_m e^{-\Lambda_m t}\frac{(\Lambda_m t)^n}{n!}\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{B}_m\right]e^{-\mu t}dt \\
&\quad + \int_0^\infty KC\mu e^{-\mu t}\left[\sum_{n=0}^{\infty} e^{-\Lambda_m t}\frac{(\Lambda_m t)^n}{n!}\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{e}\right]dt \\
&= \sum_{n=0}^{\infty}\left[\int_0^\infty e^{-(\Lambda_m+\mu)t}\frac{(\Lambda_m t)^n}{n!}dt\right](U\Lambda_m\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{B}_m + KC\mu\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{e}) \\
&= \sum_{n=0}^{\infty}\left(\frac{\Lambda_m}{\Lambda_m+\mu}\right)^n\left[\int_0^\infty e^{-(\Lambda_m+\mu)t}\frac{[(\Lambda_m+\mu)t]^n}{n!}dt\right] \\
&\quad \times (U\Lambda_m\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{B}_m + KC\mu\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{e}) \\
&= \sum_{n=0}^{\infty}\left(\frac{\Lambda_m}{\Lambda_m+\mu}\right)^n\frac{\Gamma(n+1)}{n!}\left(\frac{U\Lambda_m\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{B}_m + KC\mu\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{e}}{\Lambda_m+\mu}\right) \\
&= \sum_{n=0}^{\infty}\left(\frac{\Lambda_m}{\Lambda_m+\mu}\right)^n\frac{\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n(U\Lambda_m\mathbf{B}_m + KC\mu\mathbf{e})}{\Lambda_m+\mu}
\end{aligned} \tag{4.29}$$

where $\Gamma(n) = \int_0^\infty t^{n-1}e^{-t}dt$ is the Gamma function. Note that n is an integer in the series and so we can use the identity that $\Gamma(1+n) \equiv n!, n = 0, 1, \ldots$.

Using similar approach, we can find the expected period length. Substitute Equations (4.27) and (4.28) into Equation (4.22), we have

$$\begin{aligned}
D_m^D(d) &= \int_0^\infty t\left[\sum_{n=0}^{\infty} \Lambda_m e^{-\Lambda_m t}\frac{(\Lambda_m t)^n}{n!}\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{B}_m\right]e^{-\mu t}dt \\
&\quad + \int_0^\infty t\mu e^{-\mu t}\left[\sum_{n=0}^{\infty} e^{-\Lambda_m t}\frac{(\Lambda_m t)^n}{n!}\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{e}\right]dt \\
&= \sum_{n=0}^{\infty}\left[\int_0^\infty e^{-(\Lambda_m+\mu)t}\frac{\Lambda_m^n t^{n+1}}{n!}dt\right](\Lambda_m\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{B}_m + \mu\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{e}) \\
&= \sum_{n=0}^{\infty}\left(\frac{\Lambda_m}{\Lambda_m+\mu}\right)^n\frac{\Gamma(n+2)}{n!}\left(\frac{\Lambda_m\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{B}_m + \mu\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n\mathbf{e}}{\Lambda_m+\mu}\right) \\
&= \sum_{n=0}^{\infty}\left(\frac{\Lambda_m}{\Lambda_m+\mu}\right)^n\frac{(n+1)\boldsymbol{\pi}'_m(0)\mathbf{V}_m^n(\mathbf{B}_m\Lambda_m + \mathbf{e}\mu)}{(\Lambda_m+\mu)^2}.
\end{aligned} \tag{4.30}$$

Lastly, the average expected cost rate for the distance-based protocol is then given by

$$E_D(d) = \frac{\sum_{m=1}^N C_m^D(d)\pi_m}{\sum_{m=1}^N D_m^D(d)\pi_m}. \tag{4.31}$$

In this protocol, we manage to find the optimal value d^* such that the average expected cost $E_D(d^*)$ is minimized. Again, because our movement model allows a non-uniform movement pattern of a mobile computer inside the network, it is advantageous to evaluate an optimal vector $\mathbf{d}^* = <d_i^* : 1 \leq i \leq N>$ where each element d_i^* represents the optimal distance value for the cell c_i. Since the values for each element in the vector is independent, we can derive the optimal value for each cell separately.

4.4 The Reverse-Guessing Protocol

In the studies of the time-based protocol and the distance-based protocol, we find that the performance of these protocols are dependent on both the movement patterns of the mobile computers and the paging methods (which is also shown numerically in Section 4.5). Whilst in realistic situations, the mobility is non-uniform across a geographic region of the wireless network. Therefore, we have to use different heuristic values when the mobile computer is located in a different cell. Moreover, it is very likely that the mobility configuration of a mobile computer will also change over time. As a result, the wired network system has to recompute all the optimal heuristics periodically. This requires a great computational efforts for the mobility tracking since the number of subscribers in a personal communication network is expected to be tremendous.

Another problem for these protocols is that these protocols has an implicit assumption that the mobile computer has a symmetric movement pattern [BN95]. It is, however, obvious that if the computer move to a different place, the value of the heuristic should be different. This problem is hidden by the simplified models used in other analyses. Therefore, it is difficult to establish the relationship between the heuristic values and the cost of mobile terminal tracking.

We are now motivated to devise a new protocol that are more adaptive to the changing mobility configuration of a mobile computer. Indeed, the ideal protocol for mobile terminal tracking is the one that can prompt the mobile computer to send an update message whenever the expected cost for paging the mobile computer is higher than the cost of location update.

This ideal protocol, however, is unrealizable because the value of the expected cost depends on the on-line movement of the mobile computer as well as the prediction of the wired network system. Also, the location probabilities of the mobile computer changes with time. This requires unmeasurable computation overhead on the calculation of the expected cost all the time.

Here, we propose a new algorithm, the *reverse-guessing* algorithm, that accounts for the paging cost and the update cost in a more direct manner. In this protocol, we assume that the mobile computer will also have the statistics about its movement patterns. It will also use the *same* prediction method as the wired network system so that it can know what the wired system predict on its current location.

The protocol operates as the following. At certain time intervals, the computer evaluates its location probability in the same way as the wired network (as presented in Section 4.1). Then we rearrange cells such that the location probabilities are in decreasing order of magnitudes. Lastly, the mobile computer then looks up the position of its current cell in the sequence, which we term it as the *rank*. If the rank is greater than a predefined limit for the protocol, then an update message will be sent. The protocol is presented in a more formal manner in Figure 4.5.

At time t since the last location update at cell c_m:

1. Evaluate the location probability $\boldsymbol{\pi}_m = <\pi_{1|m}, \pi_{2|m}, \ldots, \pi_{N|m}>$ using Equation (4.3) and the same mobility matrix as recorded at the wired network
2 Sort the cells in order of non-increasing location probabilities, *i.e.* evaluate the sequence of indices $<i_1, i_2, \ldots, i_N>$ such that $\pi_{i_1|m} \geq \pi_{i_2|m} \geq \cdots \geq \pi_{i_N|m}$.
3. Observe the current location c_k.
4. Look up k in the sequence $<i_1, i_2, \ldots, i_N>$ such that $k = i_r$, which r is the rank.
5. If $r >$ *THRESHOLD*, send location update message; otherwise end.

Figure 4.5: The Reverse-Guessing Protocol

In this protocol, we need extra storage space at the mobile computer to store the movement statistics. Also, we need the processing overhead at the mobile computer to compute the location probability. It is the major drawback of the protocol. However, as technology evolves, the problem of deficiency in disk space and processor power at the mobile computer will become less critical than the scarcity of the wireless media. It is a good tradeoff for the overhead of tracking at the expense of the processing and storage overhead at the computer. Moreover, because each mobile computer only needs to take care of its own mobility, the overhead is relatively small.

The length of the period can be varied according to different criteria. In general, the more frequent the protocol is invoked, the lower the tracking overhead can be achieved. However, more computation overhead is induced. For example, a timer can be set for a specific time limit. If the time limit expires, the protocol is started. The protocol can also be executed at each of the change in the cell location of the mobile computer. However, if the mobile computer moves quickly, the computation may be very frequent. This protocol can also be augmented with the distance-based method. Instead of the sending the update message directly, the protocol is invoked.

The main reason to choose the rank as the heuristic for this protocol can be followed from Section 4.2. In the analysis of our paging protocol, we can see that the cost of paging is approximately directly related to the rank of the mobile computer's current cell in the location probability sequence. In other words, if the value of the rank is larger, the actual paging cost will be higher. Therefore, we can use the rank as a more direct estimate of the paging cost.

There are many advantages for this protocol. First of all, it is an online protocol. The mobile computer will keep track of its current location and use it to obtain a more precise estimate of the paging cost. Moreover, The value of the heuristic (rank) for our reverse-guessing protocol is insensitive to different mobility patterns. This results in a more adaptive optimization as compared to the time-based and the distance-based protocols. Lastly, the protocol involves only the calculation of the location probability, which is efficient enough to be implemented inside a mobile computer. Moreover, we can lower the complexity of the prediction at the mobile computer by the reducing the problem size, as described in Section 4.1.

4.5 Experiments

In this section, we are going to do some numerical comparisons on the protocols described in Section 4.3. There are two experiments in this section. The main objective of the first experiment is to show the soundness of our transient analysis. Moreover, we want to compare the performance of the time-based and the distance-based protocols. In the second experiment, we want to compare the performance of different protocols and show that the performance of our proposed reverse-guessing protocol is better than the others.

Experiment 1. In this experiment, we want to illustrate the soundness of our proposed transient analysis. In order to do so, we first obtain analytically the performance results of the time-based and the distance-based protocols from Equations (4.20) and (4.31) respectively. Then, we use simulation to emulate the actual movements of a mobile computer and compare the simulation results with the theoretical ones.

The configuration of the cellular network in this experiment consists of 721 cells, which is made up of closely packed cells in 16 layers. The incoming calls to the mobile computer is a Poisson process with rate $\mu = 0.1$ calls per unit time. Moreover, the mobile computer will follow a uniform and symmetric movement pattern. In particular, the probability for the mobile computer to go from one cell to another is once per unit time. While the move rate Λ is equal to once per unit time. The cost for paging a cell is a constant C of 10 units; and the cost for the mobile computer to send a location update message is $U = 100$ units.

It is worth noting that, in our transient analysis, we assume that the paging protocol is using a registration area approach. This approach has two steps. In the first step, cells inside the registration area are pages first. In the target mobile computer is not found, the other cells will be polled in the next step.

Figure 4.6 shows the results for this experiment. In part (a), we show the results of the time-based protocol with using the transient analysis. Since we use a two-step paging here, different curves in the graph represents different sizes of the registration area. For instance, the registration area will have only one cell if the distance is 1. For distance equals to 3, the size of the registration will be 19, using the formula $K = 3d(d-1) + 1$.

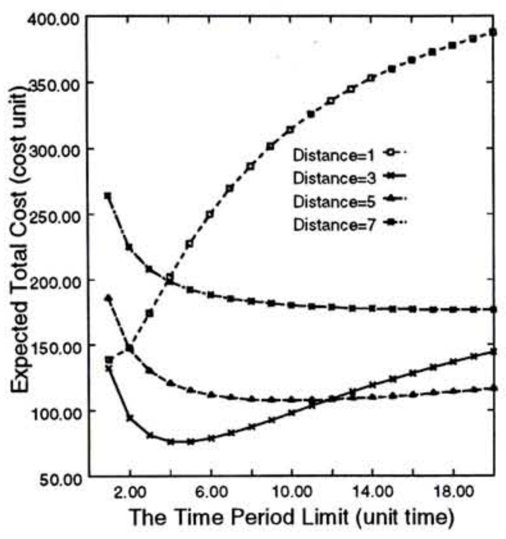
Compare with part (b) which is the simulation result. The evaluation using our transient analysis is quite accurate. This proves the soundness of our analysis of the time-based protocol.

From these curves, we can see that different sizes of the location areas result in different performance output. Moreover, the trends along different time limits are also different among the curves. In particular, the location area with the registration are of 3 layers of cells achieves the lowest cost at $T = 4.0$ units. Lastly, we can also see that the minimum points may not exist in some sizes (*e.g.*, distance = 7) of registration area.

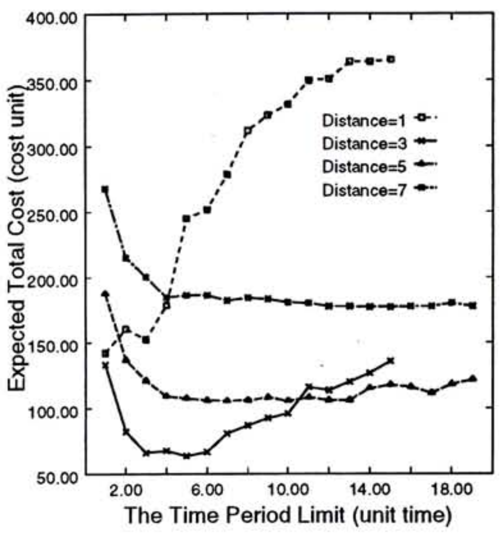
Lastly, by comparing the results of parts (c) and (d), we can see that our transient analysis on the distance-based protocol is also very accurate as compared to the simulation result. This proves the soundness of using our transient analysis technique to determine the performance of this protocol. Also, we can see that there exists an optimal point at distance limit of 2.

Experiment 2. In this experiment, we compare the performance results of the time-based protocol, the distance-based protocol and the proposed reverse guessing protocol. As we have not evaluated the performance of the reverse-guessing protocol analytically, we will use simulation in this part of comparison.

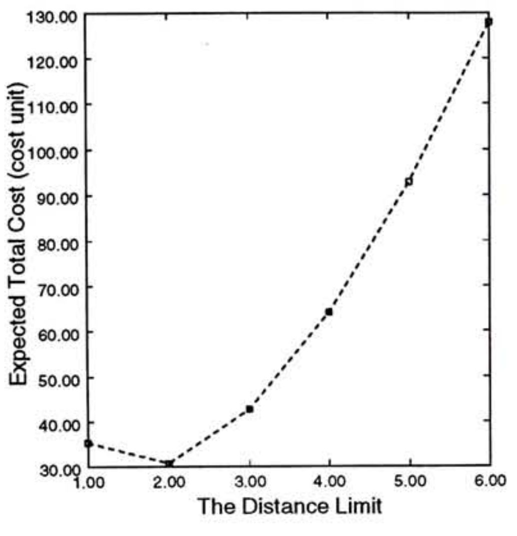
This experiment is divided into four cases, each of them differs from each other by the mobility pattern. In the first case—the uniform mobility—the mobility of the mobile computer is uniform and symmetric across the whole cellular network. This



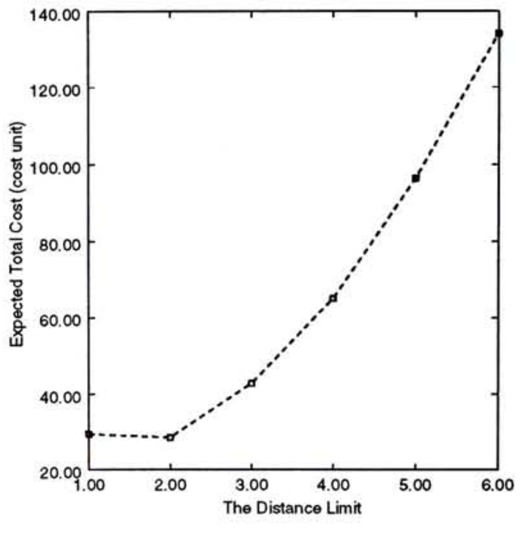
(a) Time-Based Protocol (Transient Analysis)



(b) Time-Based Protocol (Simulation)



(c) Distance-Based Protocol (Transient Analysis)



(d) Distance-Based Protocol (Simulation)

Figure 4.6: Comparison of Analytic Results with Simulation ones

case occurs when the mobile user is inside a prosperous area of a city, where probably random walk is the most realistic model. In the second case—the random mobility—the move rate of the computer from one cell to another cell is generated randomly from uniform distribution of move rate. In the third, or the zone case, the computer has a tendency to stay inside a geographical area. In the last case, the circle case, the computer moves along a circular path. Many common user movement patterns can be generated by superimposing the different movement patterns mentioned above.

In all these cases, we carried out five protocols. The first one corresponds to the case that the mobile computer will not send any update message. The second one corresponds to the case that the computer sends an update whenever it enters a different cell. The last three protocols are time-based, distance-based and reverse-guessing protocols respectively.

The network configuration is the same as the previous experiment, in which the network is composed of 721 cells. The only different is the assignment of the move rates from one cell to another cell, which are now geographically dependent according to the different cases mentioned above. For a fair comparison, however, we have normalize the move rates in such a way that the maximum move rate is set to be once per unit time.

Protocol	Uniform	Random	Zone	Circle
No Update	24.6926	15.2988	7.8195	3.09715
Always Update	100.679	100.683	100.677	100.681
Time-Based	18.9124(9)	12.1886(13)	7.6240(18)	3.09715(∞)
Distance-Based	23.1667(8)	15.9603(8)	7.6983(11)	3.03874(11)
Reverse-Guessing	13.4576(30)	9.13518(30)	5.79265(30)	3.02828(30)

Table 4.1: Simulation costs for different update protocols under various movement pattern in Experiment 2

The performance results for the experiment is tabulated in 4.1. In this table the minimum costs for different protocols under different movement patterns are recorded. Also, the optimal parameter values (*i.e.* time period limit for the time-based protocol, the distance limit for the distance-based protocol, and the threshold for the rank in the reverse-guessing protocol) are recorded inside the brackets.

From this result, we can see that the performance results of different protocol vary under different movement patterns. Moreover, the optimal parameter settings for the time-based and the distance-based protocol also change with different movement patterns.

The most important of all, however, is that the performance of the reverse-guessing protocol is always the best among all five protocols, regardless of the movement patterns under investigation. Moreover, the optimal value of the heuristic, the threshold for the rank, does not change in any pattern as well. As a result, our proposed reverse-guessing protocol is suitable for many types of mobility patterns. It is also shown the optimal setting is invariant to different user movement patterns values such as the ratio of the update cost to the cost of paging. This is extremely useful because the wired system does not need to record the optimal settings of the mobile computers and, more importantly, it does not need to re-adjust the settings if the movement patterns of the computers change over time.

Chapter 5

Conclusions & Future Work

The personal communication network is the basis platform to support the mobile computing environment, whose mission is to provide the subscribers with computer network services seamlessly wherever they are located. It is done by integrating wired network systems with wireless ones.

In this system, nevertheless, there are many differences from the conventional client-server systems. As the technology advances, the resources for wired computer networks are increasing with rapid rates; while, at the same time, the improvement in wireless network resources is still hindered by the intrinsic physical properties for electromagnetic media and the portability concerns of mobile computers. This widening gap causes significant bottlenecks at the wireless network part. To improve the situation, software techniques are therefore indispensable.

In this thesis, we have presented a novel distributed data caching protocol to enhance the efficiency of information delivery at the wireless network system. We have also proposed some techniques to lower the system cost for mobile terminal tracking. In these investigations, we primarily focus on the minimization on the use of wireless communication resource, which is the most critical scarce resource that has to be managed carefully. This is done at the expense of other more abundant resources. Nonetheless, we have also considered other factors, such as the limited storage space and processor power at the mobile computers, system signalling inside the wired networks, workloads on the systems, among others.

In this chapter, we will summarize our work and discuss on some interesting issues. In the coming section, we will summarize our work on using distributed data caching to reduce the communication overheads at the wireless network. After that, we will consolidate our work on mobile terminal tracking.

5.1 Distributed Data Caching

Data caching is an important technique in client-server systems. It improves the performance of data access by replicating copies of frequently accessed data at the clients. In view of this, it is favorable to deploy this technique in the mobile computing environment so as to reduce the communication of the limited wireless channels. This environment, however, has some characteristics different from the wired networks. In the wireless network, the base station can use the dissemination channel to deliver information to virtually infinite number of mobile computers within a cell. Yet, the bandwidth of this wireless channel is very limited. Moreover, various resources such as cache memory at mobile computers is also constrained by portability reasons.

In this thesis, we regard the caching problem as a different optimization problem. Instead of optimizing for the access performance at the mobile computers, we optimize, from the base station's point of view, for the overall bandwidth consumption of the wireless channels. We have also considered the storage constraints in the mobile computers. This optimization is accomplished by exploring the dissemination nature of the downlink channel from the base station to the mobile computers.

We have proposed a novel distributed caching protocol where the base station broadcasts a set of data objects to the mobile computers within a cell. This protocol involves the base station to select a set of data objects to broadcast and the mobile computers to select and cache a subset of objects from this broadcast. We have proved that, however, in order to minimize overall communication, both the selection processes at the base station and the mobile computers are NP-hard. Therefore, we propose strategies for the base station and the mobile computers to attain a near-optimal performance.

For the mobile computers, we have proposed a greedy method to do cache selection. This method can attain a guaranteed performance at an $O(N \log N)$ complexity. At the base station, we have also proposed an efficient and effective heuristic strategy, the Active strategy, for deriving a near-optimal solution at an $O(MN \log N)$ time. In addition to proposing these algorithms, we have also analyzed their performance and compared them to a conventional client-server caching protocol, the Passive strategy.

In the performance analysis, we have proposed an unattainable lower bound for overall communication bandwidth consumption. By comparing the output of our strategies with this bound, we can get insights on the real performance for our proposed method. Moreover, we have derived closed form expressions of the performance metrics for our protocol, which are the wireless communication bandwidth consumption and response time.

Using the above performance metrics, we have shown that our proposed Active strategy can reduce the wireless communication overhead and, at the same time, the expected response times for the database accesses are shortened. As a result, we can accommodate more mobile computers inside a cell as well as increase the overall system performance using this protocol.

This research is based on the rationale that, because the broadcast channel can be shared by many mobile computers at the same time, we need to manage the use of this channel such that data objects of general interest should be broadcasted. The base station takes the role as an arbitrator to balance the tradeoffs among different individual requests from the mobile computers. This belief, in general, can be readily employed in other broadcasting protocols.

In this thesis, we have chosen a simple synchronous broadcast protocol to demonstrate our idea. However, the general idea is compatible with other methods. For example, we may use asynchronous broadcast instead of synchronous. We can also deliver the invalidation information in batches, as suggested in [BI94]. Lastly, we can also extend it to support the allocation of communication bandwidths to different data objects, such as the one in [AAFZ95].

5.2 Mobile Terminal Tracking

Mobile terminal tracking is an essential mechanism for the personal communication network to location a disconnected mobile computer so as to re-establish a connection to the computer. Currently, research efforts have been made so as to reduce the system overhead for this process. This problem can be divided into three issues. First, we need a reliable and efficient prediction method to locate a moving mobile computer. Second, given a prediction on the possible locations, an optimal paging strategy is required to minimize the search cost. Lastly, with the optimal paging strategy with the associated costs, an optimal update strategy is necessary to balance with the paging cost such that the overall cost is minimal. In this thesis, we have addressed all these issues.

There are many protocols proposed with the target at balancing the update cost with the paging cost, such as the time-based protocol and the distance-based protocol. To test the performance of these protocols, researchers have devised some mathematical models to evaluate the system costs. However, their models may not be capable of capturing real life characteristics of user movement that are important in the performance evaluation. In particular, for easy formulation, they assume that each mobile computer has a spatially uniform, symmetric movement pattern. Moreover, they do not address the issues that the location of a mobile computer depends on the time an incoming call arrives. As a result, their evaluation may not be reliable as compared to the performance of these protocols during actual deployment.

To cope with these deficiencies, we have proposed a novel, more general mathematical framework to include those important properties. This framework is based on a continuous-time Markov process whose states are the locations of a mobile computer. In this way, we can model the different movement patterns by determining the state transition rate matrix. Moreover, in order to capture the time-varying property of the location of a mobile computer, we employ the uniformization method to evaluate the transient location probabilities accurately and efficiently.

By using this framework, we can compare different update and paging protocols in a uniform manner. In this thesis, we have demonstrated the use of this framework to derive analytically the performance of the time-based and the distance-based protocols. We have also used simulation analysis to show that the results obtained in our analysis are very close to their actual performance.

In addition to these analyses, we have also take advantage of the prediction result to devise an optimal probability-based paging protocol. This protocol can obtain an optimal paging process such that the overall paging cost is minimized.

Lastly, another important contribution in this thesis is that we have proposed a new update protocol, the reverse-guessing protocol, that uses a better heuristic to help the mobile computer determine when to send an update message. This protocol can result in a lower system cost for mobile terminal tracking.

Although we have illustrated the abilities of our model to capture different user movement patterns and to address the time-varying property of user locations, our framework can still be extended to a more general model. First of all, the direction of user movement is also an important hint to predict user location. For example, in a two-way highway, a car often moves only along one direction. It seldom turns back and moves in opposite direction in the middle of the road. It is worth noting that our movement model can easily be extended to include the direction information. Instead of just recording the current location in the Markov process, we can also incorporate into the state the current direction of movement of the mobile computer. However, this extension will increase the transition rate matrix of the Markov process by another dimension. In other words, the complexity for the prediction will be increased.

Another possible extension is allow different arrival patterns for incoming calls to a mobile computer, rather than requiring it to conform to a Poisson arrival pattern. This can be done by determining the distribution function of the time a call arrives.

However, the mathematical analysis will become more convoluted.

Besides these modeling techniques, we can also explore better solutions to the mobile terminal tracking problem. In particular, we have analyzed the time-based and the distance-based protocols and shown that different optimal points may exist for these protocols. We have also illustrated the use of iteration to obtain those values. However, we have not yet investigated the feasibility of using direct methods to get the optimal settings. This can be an interesting problem to work on.

Bibliography

[AAFZ95] Swarup Acharya, Rafael Alonso, Michael Franklin, and Stanley Zdonik. Broadcast Disks: Data Management for Asymmetric Communication Environments. In *ACM SIGMOD International Conference on Management of Data*, 1995.

[AB94a] Arup Acharya and B. R. Badrinath. Checkpointing Distributed Applications on Mobile Computers. In *The 3rd IEEE International Conference on Parallel and Distributed Information Systems*, October 1994.

[AB94b] Arup Acharya and B. R. Badrinath. Recording Distributed Snapshot based on Causal Order of Message Delivering. Technical report, Department of Computer Science, Rutgers University, 1994.

[AB96] Arup Acharya and B. R. Badrinath. A Framework for Delivering Multicast Messages in Networks with Mobile Hosts. In *ACM-Baltzer Journal on Mobile Networks and Applications*, volume 1, 1996.

[AFZ95] Swarup Acharya, Michael Franklin, and Stanley Zdonik. Dissemiation-based Data Delivery Using Broadcast Disks. In *IEEE Personal Communications*, 1995.

[AFZ96] Swarup Acharya, Michael Franklin, and Stanley Zdonik. Prefetching from a Broadcast Disk. In *International Conference on Data Engineering*, February 1996.

[AGG96] Daniel O. Awduche, Aura Ganz, and Arthur Gaylord. An Optimal Search Strategy for Mobile Stations. In *Proceedings of IEEE International Conference on Universal Personal Communications*, 1996.

[AH95a] Ian F. Akyildiz and Joseph S. M. Ho. A Mobile User Location Update and Paging Mechanism Under Delay Constraints. In *Proceedings of ACM SIGCOMM*, 1995.

[AH95b] Ian F. Akyildiz and Joseph S. M. Ho. Dynamic Mobile User Location Update for Wireless PCS Networks. In *ACM Journal of Wireless Networks*, volume 1. Baltzer Publishers, 1995.

[BB94] Ajay Bakre and B.R. Badrinath. I-TCP: Indirect TCP for Mobile Hosts. Technical Report DCS-TR-314, Department of Computer Science, Rutgers University, 1994.

[BHG87] Philip A. Bernstein, Vassos Hanzilacos, and Nathan Goodman. *Concurrency Control and Recovery in Database Systems.* Addison-Wesley Publishing Company, 1987.

[BI94] Daniel Barbara and Tomasz Imielinski. Sleepers and Workaholics: Caching Strategies in Mobile Environments. In *ACM SIGMOD International Conference on Management of Data*, 1994.

[BN95] Yitzhak Birk and Yaron Nachman. Using Direction and Elapsed-Time Information to Reduce the Wireless Cost of Locating Mobile Units in Cellular Networks. In *ACM Journal of Wireless Networks*, volume 1. Baltzer Publishers, 1995.

[BNK93] Amotz Bar-Noy and Ilan Kessler. Tracking Mobile Users in Wireless Communications Networks. In *Proceedings of IEEE INFOCOM*, 1993.

[BNKS94] Amotz Bar-Noy, Ilan Kessler, and Moshe Sidi. Mobile users: To Update or not to Update? In *Proceedings of IEEE INFOCOM*, 1994.

[Cox62] D. R. Cox. *Renewal Theory.* Methuen (London), 1962.

[DN95] Antonio DeSimone and Sanjiv Nanda. Wireless Data: Systems, Standards, Services. In *ACM Journal of Wireless Networks*, volume 1. Baltzer Publishers, 1995.

[dSeSG90] Edmundo de Souza e Silva and H. Richard Gail. Analyzing Scheduled Maintainence Policies for Repairable Computer Systems. In *IEEE Transactions on Computers*, November 1990.

[dSeSG92] Edmundo de Souza e Silva and H. Richard Gail. Performability Analysis of Computer Systems: From Model Specification to Solution. In *Performance Evaluation*, 1992.

[dSeSG96] Edmundo de Souza e Silva and H. Richard Gail. The Uniformization Method in Performability Analysis. In *IBM Research Report RC 20383 (90082)*, 1996.

[FC92] Michael J. Franklin and Michael J. Carey. Client-Server Caching Revisited. Technical Report CS-TR-92-1089, Department of Computer Science, University of Wisconsin-Madison, May 1992.

[FZ94] George H. Forman and John Zahorjan. The Challenges of Mobile Computing. In *IEEE Computer*, April 1994.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, 1979.

[HA95a] Joseph S. M. Ho and Ian F. Akyildiz. Local Anchor Scheme for Reducing Location Tracking Cost in PCNs. In *Proceedings of ACM MOBICOM*, 1995.

[HA95b] Joseph S. M. Ho and Ian F. Akyildiz. Mobile User Location Update and Paging Under Delay Constraints. In *ACM Journal of Wireless Networks*, volume 1. Baltzer Publishers, 1995.

[HS96] Anna Hać and Chengdi Sheng. User Mobility Management in PCS network: Hierarchical Databases and their Placement. In *Proceedings of IEEE International Conference on Universal Personal Communications*, 1996.

[HSW94] Yixiu Huang, Prasad Sista, and Ouri Wolfson. Data Replication for Mobile Computers. In *ACM SIGMOD International Conference on Management of Data*, 1994.

[HW94] Yixiu Huang and Ouri Wolfson. Object Allocation in Distributed Databases and Mobile Computers. In *Proceedings of the 10th International Conference on Data Engineering*, 1994.

[IA94] Tomasz Imielinski and Arup Acharya. The Network as a Database Machine. Technical report, WINLAB and Department of Computer Science, Rutgers University, 1994.

[IDM95] John Ioannidis, Dan Duchamp, and Gerald Q. Maguire, Jr. Communications and Consistency in Mobile File Systems. In *IEEE Personal Communications*, December 1995.

[JBEA95] Jin Jing, Omran Bukhres, Ahmed Elmagarmid, and Rafael Alonso. Bit-Sequences: A New Cache Invalidation Method in Mobile Environments. Technical Report CSD-TR-94-074, Department of Computer Sciences, Purdue University, May 1995.

[Jen53] A. Jenson. Markov Chains as an Aid in the Study of Markov Processes. In *Skand. Aktuarietidskr*, 1953.

[JLLM94] Ravi Jain, Yi-Bing Lin, Charles Lo, and Seshadri Mohan. A Caching Strategy to Reduce Network Impacts of PCS. In *IEEE Journal of Selected Areas in Communication*, volume 12, October 1994.

[Joh94] David B. Johnson. Routing in Ad Hoc Networks of Mobile Hosts. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, December 1994.

[KKT93] Henning Koch, Lars Krombholz, and Oliver Theel. A Brief Introduction into the World of 'Mobile Computing'. Technical Report THD-BS-1993-03, Department of Computer Science, University of Darmstadt, 1993.

[Kle75] Leonard Kleinrock. *Queueing Systems, Volume I: Theory.* Wiley-Interscience, 1975.

[KS91] Hank Korth and Abraham Silberschatz. *Database System Concepts.* McGraw-Hill Publishing Company, 1991.

[Lam78] Leslie Lamport. Time, Clocks and Ordering of Events in Distributed Systems. In *Communications of ACM*, volume 21, 1978.

[Lit61] J. D. C. Little. A Proof of the Queueing Formula $l = \lambda w$. In *Operations Research*, volume 9, 1961.

[LL96] Kin K. Leung and Yonatan Levy. Use of Centralized and Replicated Databases for Global Mobility Management in Personal Communication Networks. In *Proceedings of IEEE International Conference on Universal Personal Communications*, 1996.

[LQ95] Victor O. K. Li and Xiaoxin Qiu. Personal Communication Systems (PCS). In *Proceedings of IEEE*, volume 83, September 1995.

[LS95] Qi Lu and Mahadev Satyanarayanan. Improving Data Consistency in Mobile Computing Using Isolation-Only Transactions. In *The 5th Workshop on Hot Topics in Operating Systems*, May 1995.

[LW95] Ying Kit Lee and Wing Shing Wong. Location Update by Binary Cutting of ID's. In *Proceedings of International Conference on Universal Personal Communications*, 1995.

[MES95] Lily B. Mummert, Maria R. Ebling, and Mahadev Satyanarayanan. Exploiting Weak Connectivity for Mobile File Access. In *Proceedings of the 15th ACM Symposium on Operating Systems Principles*, December 1995.

[MGTH95] Teresa H. Meng, Benjamin M. Gordon, Ely K. Tsern, and Andy C. Hung. Portable Video-on-Demand in Wireless Communication. In *Proceedings of IEEE*, 1995.

[MHS94] Upamanyu Madhow, Michael L. Honig, and Ken Steiglitz. Optimization of Wireless Resources for Personal Communications Mobility Tracking. In *Proceedings of IEEE Infocom*, 1994.

[MJSL96] José M. F. Moura, Radu S. Jasinschi, Hirohisa Shiojiri, and Jyh-Cherng Lin. Retrieving Quality Video across Heterogeneous Networks—Video over Wireless. In *IEEE Personal Communications*, February 1996.

[Nel95] Randolph Nelson. *Probability, Stochastic Processes, and Queueing Theory.* Springer-Verlag, 1995.

[PA96] Sudeep K. Palat and Steinar Andresen. User Profiles and their Replication for Reduction of HLR Accesses and Signalling Load. In *Proceedings of IEEE International Conference on Universal Personal Communications*, 1996.

[PB94] Charles E. Perkins and Pravin Bhagwat. A Mobile Networking System based on Internet Protocol. In *IEEE Personal Communication Magazine*, volume 1, February 1994.

[Pol96] Gregory P. Pollini. Trends in Handover Design. In *IEEE Communications Magazine*, March 1996.

[PSS95] Ravi Prakash, Niranjan G. Shivaratri, and Mukesh Singhal. Distributed Dynamic Channel Allocation for Mobile Computing. In *Principles of Distributed Computing*, 1995.

[Ros96] C. Rose. Minimizing the Average Cost of Paging and Registration: A Timer-Based Method. In *ACM Journal of Wireless Networks*, volume 2. Baltzer Publishers, 1996.

[RY95] C. Rose and R. Yates. Minimizing the Average Cost of Paging under Delay Constraints. In *ACM Journal of Wireless Networks*, volume 1. Baltzer Publishers, 1995.

[SKM$^+$93] Mahadev Satyanarayanan, James J. Kistler, Lily B. Mummert, Maria R. Ebling, Puneet Kumar, and Qi Lu. Experience with Disconnected Operation in a Mobile Computing Environment. In *1993 USENIX Symposium on Mobile and Location-Independent Computing*, 1993.

[Ste94] William J. Stewart. *Introduction to Numerical Solution of Markov Chains.* Princeton University Press, 1994.

[Sto79] M. Stonebraker. Concurrency Control and Consistency of Multiple Copies of Data in Distributed INGRES. In *IEEE Transaction on Software Engineering*, volume 3. Institute of Electrical and Electronic Engineers, 1979.

[TA96] Leandros Tassiulas and Farooq M. Anjum. A Hierarchical Multiresolution Registration Structure for Mobility Track. In *Proceedings of IEEE International Conference on Universal Personal Communications*, 1996.

[Tho79] R.H. Thomas. A Majority Approach to Concurrency Control for Multiple Copy Databases. In *ACM Transaction on Database Systems*, volume 4. Association for Computing Machinery, 1979.

[Ver96] Markku Verkama. Optimal Paging — A Search Theory Approach. In *Proceedings of IEEE International Conference on Universal Personal Communications*, 1996.

[WL95] Man Hon Wong and Wing Man Leung. A Caching Policy to Support Read-Only Transactions in a Mobile Computing Environment. Technical Report CS-TR-95-07, Department of Computer Science & Engineering, The Chinese University of Hong Kong, 1995.

[Wol82] R. W. Wolff. Poisson Arrival See Time Averages (PASTA). In *Journal of Operations Research*, volume 30, 1982.

Appendix A

Proof of NP-hardness of the Broadcast Set Assignment Problem

In this section, we present a proof to show that the problem of assigning data objects into the broadcast set is NP-hard. We begin by defining a restricted version of the caching problem. Then we transform an instance of the set packing problem [GJ79], which is NP-complete, into an instance of our problem.

Proof: We restrict the broadcast set assignment problem into the following one:

> INSTANCE: A cell has M mobile computers, m_1, m_2, …, m_M, each with cache memory of size 1. There is a set $\mathcal{D}$ of N data objects, o_1, o_2, …, o_N, each has a constant size of 1. The read overhead $(b_{query} + b_j)$ is a constant value of B_{query}. The write overhead (b_{update}) is a constant value of B_{update}. Moreover, the overhead of broadcasting a data object is a constant value of $B_{broadcast}$. Each data object has o_j a write rate of λ_j^w. And each mobile computer m_i accesses each object o_j with an individual read rate of λ_{ij}^r. Also given a constant $P \in \Re^+$. Lastly, the write rate to each object o_j is given by λ_j^w.
>
> QUESTION: Is there a subset $\mathcal{S} \subseteq \mathcal{D}$ such that the total of the reduction in bandwidth as defined by $G(\mathcal{S}, o_j)$ for each object o_j in $\mathcal{S}$ is at least P?

We begin the transformation with the following set packing problem:

> INSTANCE: Collection $\mathcal{C}$ of finite sets, and a positive integer $K \leq |\mathcal{C}|$.
>
> QUESTION: Does $\mathcal{C}$ contain at least K mutually disjoint sets?

The transformation is as follows.

- Let $\mathcal{C}$ be a collection of N sets, each set $\mathcal{J}$ corresponds a distinct data object o_j in the database $\mathcal{D}$ in the broadcast set assignment problem.
- Let $\mathcal{I}$ be the set of all elements in $\mathcal{J}$ such that $\mathcal{J} \in \mathcal{C}$, *i.e.* $\mathcal{I} = \cup_{\mathcal{J} \in \mathcal{C}} \mathcal{J}$.
- Let $|\mathcal{I}|$ be M; and for each element $i \in \mathcal{I}$, we correspond it to a distinct mobile computer m_i.
- Let $K = P$.
- Let $B_{broadcast} = B_{read} = B_{update} = 1$ for all data objects.
- $1/\mathcal{T} = 1$
- For each element $\mathcal{J} \in \mathcal{C}$ which corresponds to data object o_j, let $\lambda^r_{ij} = 2$ for all mobile computers and lastly, let $\lambda^w_j = 2|\mathcal{J}| - 2$.

It is easy to see that the above transformation takes polynomial time.

Intuitively, a set $\mathcal{J} \in \mathcal{C}$ consists of mobile computers that will cache o_j in a given broadcast period. The gain of broadcasting the object is then given by

$$\sum_{i \in \mathcal{J}} \lambda^r_{ij} B_{query} - B_{update} - B_{broadcast} = 1 \qquad \text{for each data object}$$

From the above formula, we can see that the gain of 1 is only possible when all the mobile computers in $\mathcal{J}$ want to cache the object. Otherwise, the gain is negative.

Given the above gain formulation, we see that if the total gain in communication overhead is larger than or equal to P, *i.e.* it is a YES instance to the broadcast set assignment problem, then the number of objects S included in the broadcast set should also be larger than or equal to P. In the corresponding set packing instance, the number of sets S is then larger than $P = K$. Moreover, as each mobile computer has only a memory size of 1, it can only cache one object. This means that the corresponding element i should only appear in the S set by at most once. As a result, the S sets in the set packing instance form a partition. The resulting problem instance is also a YES instance to the set packing problem. This completes the proof. ■