DESIGN AND DEVELOPMENT OF COOPERATIVE CACHING FRAMEWORK FOR IMPROVING DATA ACCESSIBILITY IN WIRELESS AD HOC NETWORKS

A thesis submitted to the
Cochin University of Science and Technology
in partial fulfillment of the
requirements for the award of the Degree of
Doctor of Philosophy
By

PREETHA THERESA JOY
Reg. No. 4521
Under the guidance of
Dr. K. POULOSE JACOB

DEPARTMENT OF COMPUTER SCIENCE
COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY
KOCHI-682 022, INDIA

AUGUST 2015

DESIGN AND DEVELOPMENT OF COOPERATIVE CACHING FRAMEWORK FOR IMPROVING DATA ACCESSIBILITY IN WIRELESS AD HOC NETWORKS

Ph.D Thesis

Author:

Preetha Theresa Joy
Department of Computer Science
Cochin University of Science and Technology
Cochin-682 022, Kerala, India.
preetha@mec.ac.in

Supervisor:

Prof. (Dr.) K. Poulose Jacob
Pro-Vice-Chancellor
Professor in Computer Science
Cochin University of Science and Technology
Cochin-682 022, Kerala, India.
kpj@cusat.ac.in

AUGUST 2015

CERTIFICATE

This is to certify that the thesis entitled "**Design and Development of Cooperative Caching Framework for Improving Data Accessibility in Wireless Ad Hoc Networks**" *is a bonafide record of the research work carried out by* **Ms. Preetha Theresa Joy** *under my supervision in the Department of Computer Science, Cochin University of Science and Technology, Kochi-22. The results presented in this thesis or parts of it have not been presented for the award of any other degree. The relevant suggestions during the pre-synopsis seminar and recommended by the Doctoral Committee have been incorporated in the thesis.*

Kochi
August 2015

Prof. (Dr.) K. Poulose Jacob
(Supervising Guide)
Pro-Vice-Chancellor
Cochin University of Science
and Technology

DECLARATION

I hereby declare that the thesis entitled "**Design and Development of Cooperative Caching Framework for Improving Data Accessibility in Wireless Ad Hoc Networks**" *is the authentic record of research work carried out by me, for my Doctoral Degree under the supervision and guidance of* **Dr. K. Poulose Jacob**, *Pro-Vice-Chancellor, Cochin University of Science and Technology and that no part thereof has previously formed the basis for the award of any degree or diploma or any other similar titles or recognition.*

Kochi
August 2015.

Preetha Theresa Joy

Acknowledgemnets

I thank God almighty for the grace showered upon me without which this work would not have been possible.

I wish to express my deepest sense of gratitude to my research guide, Prof. (Dr.) K. Poulose Jacob, Pro-Vice-Chancellor of Cochin University of Science and Technology, for his valuable guidance, supervision, keen observations and encouragement throughout the course of this research work. I am indebted to him for the support he extended to me all through these years to carry out this research work to completion.

I take this opportunity to thank Dr. Sumam Mary Idicula, Professor and Head, Department of Computer Science, Cochin University of Science and Technology, for providing me with all the necessary facilities for the research.

I would like to thank Dr. G. Santhosh Kumar, Assistant Professor, Department of Computer Science, Cochin University of Science and Technology, for sharing his expertise and giving me valuable suggestions. I am also thankful to Mr. Muralidharan K. B., Assistant Professor, Department of Computer Science, for his support.

I wish to express my sincere thanks to all the technical and office staff of Department of Computer Science, Cochin University of Science and Technology, for the help they have rendered to me.

My sincere thanks to the Director, IHRD for giving me an opportunity to carry out this research work at Cochin University of Science And Technology.

Thanks a lot to all my friends in the research lab of Department of Computer Science, who provided a great friendly environment for the

years I have been studying here.

I owe my deep-felt thanks to my colleagues at Model Engineering College for their support and cooperation.

I record my sincere and utmost gratitude to my parents and family for the concern and support they extended at various stages of my study.

I also place on record, my sense of gratitude to one and all, who directly or indirectly, have lent their hand in this venture.

Preetha Theresa Joy

Abstract

Mobile Ad hoc Networks are wireless networks established by a group of mobile devices on a shared wireless channel without any infrastructure. The rapid deployment and self organizing nature make these types of networks completely different from any other type of networks. However, low bandwidth and frequent disconnections caused by node mobility makes high data accessibility a challenging problem in mobile ad hoc networks. Caching is a technique that has been deployed in networks to reduce network traffic and provide better response time for data accesses. The dynamic nature of ad hoc networks poses new problems for reliable cache management. As the nodes move arbitrarily, disconnection and network division occur frequently. In addition, the mobile devices have relatively limited computing resources and battery life. Therefore an effective caching technique should consider these issues to arrive at an optimal solution. A variety of cooperative caching protocols were proposed for MANETs to address the problem of data accessibility. Most of the available literature in cooperative caching concentrates on improving the user response time. However, in most of these studies, they have neglected the cache overhead and network scalability issues. An important requirement for the successful implementation of cooperative caching in ad hoc networks is to make a balanced use of computation and communication resources. In the light of this, the purpose of this research work is to focus on several aspects of cooperative caching and in particular to devise a cooperative caching scheme that achieves high data accessibility with reduced cache overhead. The proposed scheme addresses cache management and cache discovery problems of cooperative caching. For

cache management, a new cache replacement algorithm to improve cache hit ratio is presented and a cache placement policy to enhance the effective capacity of the cooperative cache is proposed. The main goal of the cache discovery algorithm introduced is to get the data correctly and efficiently with minimum overhead and bandwidth consumption. Another important criterion in cooperative caching is network scalability. The performance of various cooperative caching schemes degrades when the network size increases. The present work focuses on this issue and presents a scalable solution for cooperative caching. The performance of the proposed cooperative caching schemes was compared with the existing techniques.

Contents

List of Figures

List of Tables

Abbreviations

ABR	: Associativity Based Routing
AODV	: Ad hoc On demand Distance Vector
BAN	: Body Area Networks
CDS	: Connective Dominating Set
CEP	: Cache Eviction Period
CGSR	: Cluster head Gateway Switching Routing
CSMA	: Carrier Sense Multiple Access
CTS	: Clear to send
DAFN	: Dynamic Access Frequency and Neighborhood
DARPA	: Defense Advanced Research Projects Agency
DCC	: Distributed Contention Control
DCF	: Distributed Coordination Function
DCG	: Dynamic Connectivity based Group
DSDV	: Destination Sequenced Distance Vector
DSR	: Dynamic Source Routing
DSSS	: Direct Sequence Spread Spectrum
FHSS	: Frequency Hopping Spread Spectrum
FSR	: Fisheye State Routing
GSR	: Global Sate Routing
HSR	: Hierarchical State Routing
ICP	: Internet Cache Protocol
IETF	: Internet Engineering Task Force
IRT	: Inter Request arrival Time
ISO	: International Standards Organization
LAN	: Local Area Network
LFU	: Least Frequently Used
LRU	: Least Recently Used

LRU-MIN	: Least Recently Used Minimum
LR-WPAN	: Low-Rate Wireless Personal Area Networks
MAC	: Medium Access Control
MANET	: Mobile Ad Hoc Networks
MH	: Mobile Host
MIS	: Maximal Independent Set
NIC	: Network Interface Card
OSI	: Open Systems Interconnection
OSLR	: Optimized Link State Routing Protocol
P2P	: Peer to Peer
PCMCIA	: Personal Computer Memory Card International Association
PDA	: Personal Digital Assistant
PAN	: Personal Area Network
PCF	: Point Coordination Function
PN	: Personal networks
PRNET	: Packet Radio Network
QoS	: Quality of Service
RTS	: Request to send
SAF	: Static Access Frequency
SSR	: Signal Stability Routing
SURAN	: Survivable Radio Networks
TORA	: Temporally Ordered Routing Algorithm
TTL	: Time to Live
VANET	: Vehicular Ad hoc Networks
WLAN	: Wireless Local Area Network
WRP	: Wireless Routing Protocol
WSN	: Wireless Sensor Networks
WWW	: World Wide Web

Chapter 1

Introduction

Contents

1.1 Introduction

Recent technological advances have enabled a push towards wireless systems in every aspect of technology. The advances in network infrastructures, availability of wireless applications, and the emergence of mobile devices such as portable computers, PDAs and cell phones fuelled an unexpected growth in the wireless arena. The current wireless systems can be classified into two broad categories based on how the network is established: infrastructure based network and infrastructure-less networks. The infrastructure based networks include cellular networks and wireless LAN. The base stations in cellular networks and access points in wireless LAN act as the infrastructure for these types of networks. The connection is established in one hop between the mobile nodes and the local base station in cellular networks. In contrast, an infrastructure-less network is formed dynamically through the cooperation of an arbitrary set of independent nodes. These mobile nodes make their decisions independently, based on the network situation without using a preexisting infrastructure. The flexibility and distributed architecture of these networks made them applicable to areas in which a prior deployment of network infrastructure is impossible. Certain environments where ad hoc networks are used include military and rescue operations, natural disasters or places where deployment of wired networks is not possible. Furthermore, ad hoc networks play an important role in commercial applications such as automotive safety improvement, provide passengers with information and entertainment, smoothen traffic flow on the roads *etc.* The ad hoc network can be realized by different networks such as body area networks (BAN), vehicular ad hoc networks (VANET), wireless networks (varying from personal area network to wide area network), and wireless sensor

networks (WSN). However, the design of these networks poses various challenges due to the multi-hop nature and lack of fixed infrastructure.

1.2 A Review of Ad hoc Network Evolution

The concept of mobile ad hoc networking can be traced back to the Defense Advanced Research Projects Agency (DARPA) Packet Radio Network (PRNET) project in 1972 [Freebersyser, 2001]. PRNET had a distributed architecture consisting of a network of broadcast radios with minimal central control. A combination of Aloha and Carrier Sense Multiple Access (CSMA) channel access protocols are used to support the dynamic sharing of the broadcast radio channel. It also supports multi-hop store-and-forward routing techniques which remove the radio coverage limitation and enable multi-user communication within a very large geographic area.

The successful demonstrations of the PRNET proved the feasibility and efficiency of infrastructure-less networks and their applications for civilian and military purposes. DARPA extended the work on multi-hop wireless networks through the survivable radio networks (SURAN) project that aimed at providing ad hoc networking with small, low-cost, low-power devices with efficient protocols and improved scalability and survivability. However, interest in this area grew rapidly in the nineties due to the popularity of a large number of portable digital devices such as laptop and palmtop computers, and the common availability of wireless communication devices.

The rising popularity of Internet and the development of internet-working protocols for mobile ad hoc networks which operates in license-free radio bands, lead to the development of a working group within

Internet Engineering Task Force (IETF) termed the Mobile Ad Hoc Networking (MANET) working group to standardize the protocols and functional specifications of ad hoc wireless networks. The vision of IETF effort in MANET working group is to provide improved standardized routing functionality to support self organizing mobile networking infrastructure. Motivated by the growing interest in ad hoc networking, a number of commercial standards were developed in the late nineties. This includes the IEEE 802.11 physical layer and MAC protocol [Anastasi, 2003], which has since then evolved into the more updated versions. Today, one can build an ad hoc network by simply plugging in 802.11 Personal Computer Memory Card International Association (PCMCIA) cards into laptop computers. Bluetooth [Bisdikian, 2001] and Hiperlan [Mingozzi, 2002] are some other examples of related existing products.

Although ad hoc wireless networks are expected to work in the absence of any fixed infrastructure, recent advances in wireless network architectures reveal interesting solutions that enable the mobile nodes to function in the presence of infrastructure [Murthy,2004]. These hybrid architectures which combine the benefits of cellular and ad hoc networks improve the capacity of the system significantly.

The mobile nodes in an ad hoc network utilizes multi-hop radio relaying for information exchange where data packets are transmitted in a store and forward manner from any source to an arbitrary destination via intermediate nodes as shown in Fig. 1.1. Nodes that lie within each other's transmission range can communicate directly and are responsible for dynamically discovering each other. The transmission range is limited by the energy preserved in the mobile nodes. In order to enable communication between nodes that are not directly within each other's

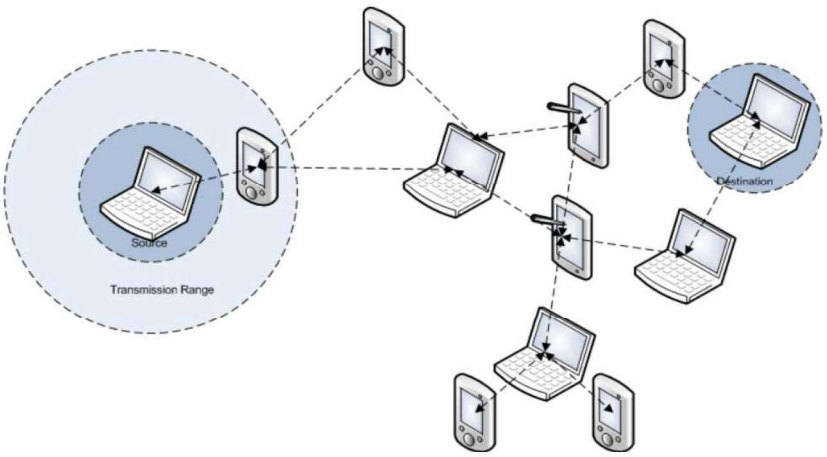

Figure 1.1: Model of an ad hoc network

send range, intermediate nodes act as routers to relay packets generated by other nodes to their destination. Furthermore, the topology of these networks may change dynamically as devices are free to join and leave the network at any time. In this energy-constrained, dynamic, distributed multi-hop environment, nodes need to be organized dynamically in order to provide the necessary network functionality. In addition, these networks are faced with the traditional problems inherent to wireless communications such as lower reliability than wired media, limited physical security, time varying channels, interference, *etc.*

1.3 MANET Architecture

The MANET IETF working group has been the major reference point for the research activities on pure general purpose MANET. The MANET

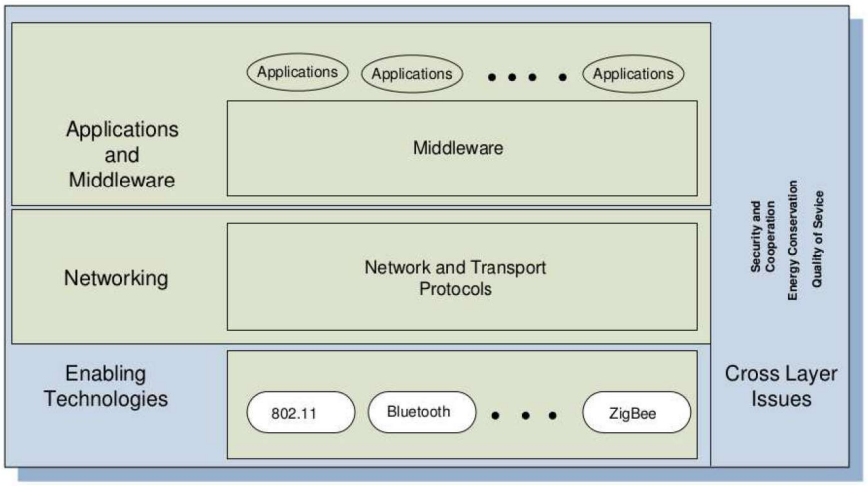

Figure 1.2: A MANET Architecture

IETF working group adopted an IP centric view of MANET that inherited the TCP/IP protocols stack layering with the aim of redesigning the network protocol stack to respond to the new characteristics, complexities and design constraints of MANET [Corson, 1999]. All layers of MANET's protocol stack were subjects of intensive research activities. Fig. 1.2 gives a layered view of the protocol stack.

As shown in the figure, the research activities in MANETS are grouped into three main areas:

- Enabling technologies;
- Networking;
- Middleware and applications.

In addition, issues like energy management, security and cooperation and

Quality of Service (QoS) span in all layers.

1.3.1 Enabling technologies

Enabling technologies of MANET guarantee direct single hop communications between user's devices. These include 802.11 families, Bluetooth and Zigbee. Bluetooth technology is a de-facto standard for low cost, short range radio links between mobile PCs, mobile phones and other portable devices. The IEEE 802.11 standard is currently the most mature technology for WLANs. It defines two operational modes—infrastructure based and ad hoc networks. Network interface cards can be set to work in either of these modes but not in both simultaneously. The IEEE 802.11 technology is a good platform to implement single hop ad hoc networks because of its extreme simplicity. It specifies the MAC layer and Physical layer for WLANs in a publicly available ISM bands. Initially, three different physical layer options were provided: Infrared, Frequency Hopping Spread Spectrum (FHSS) at 2.4 GHZ and Direct Sequence Spread Spectrum (DSSS) at 2.4 GHZ. The infrared physical layer is neglected gradually. For the MAC layer two channel access methods were defined. The first method, Distributed Coordination Function (DCF), is used in WLAN and Point Coordination Function (PCF) for ad hoc networks. Among the different IEEE 802.11 variants (Task Groups) like IEEE 802.11a, IEEE 802.11b, IEEE 802.11g, IEEE 802.11h, IEEE 802.11e, IEEE 802.11c, IEEE 802.11d, IEEE 802.11f and IEEE 802.11i, IEEE 802.11b (Wi-Fi) is the most popular one and is the one used for ad hoc networking.

ZigBee builds on IEEE 802.15.4 standard [Baronti, 2007] which defines the physical and MAC layers for low cost and low rate personal

area networks. ZigBee defines the network layer specifications for star, tree and peer-to-peer network topologies and provides a framework for application programming in the application layer. The IEEE 802.15.4 standard defines the characteristics of the physical and MAC layers for Low-Rate Wireless Personal Area Networks (LR-WPAN). The advantages of an LR-WPAN are ease of installation, reliable data transfer, short-range operation, extremely low cost, and a reasonable battery life while maintaining a simple and flexible protocol stack.

1.3.2 Networking Layer

In network layer, special attention is given to routing and forwarding. It includes the network and transport protocols.

Network Protocols. The ad hoc network routing protocols are generally divided in to two broad categories: proactive routing protocols and reactive on demand routing protocols [Royer 1999]. Proactive routing protocols maintain consistent, up to date routing information between every pair of nodes by propagating, proactively, route updates at fixed time intervals. As the routing information is usually maintained in tables, these protocols are sometimes referred to as Table Driven protocols. Representative protocols include Destination Sequenced Distance Vector (DSDV) protocol, Cluster head Gateway Switching Routing (CGSR) protocol, Wireless Routing Protocol (WRP), Global Sate Routing (GSR), Optimized Link State Routing Protocol (OSLR), Fisheye State Routing (FSR) protocol and Hierarchical State Routing (HSR).

Reactive on demand routing protocols, on the other hand, establish the route to a destination only when there is a demand for the route. The

source node initiates the route request. Once the route has been established, it is maintained either until the destination becomes in-accessible, or until the route is no longer used or expired. In reactive routing protocols a 'Just in Time' (JIT) approach is used to discover routes. The nodes discover routes to destination on demand, i.e., a node does not need a route to a destination until that destination is to be the sink of the data packets sent by the node. Reactive protocols often consume much less bandwidth than proactive protocols, but the delay to determine a route can be significantly higher. Representative reactive routing protocols include: Dynamic Source Routing (DSR), Ad hoc On demand Distance Vector (AODV), Temporally Ordered Routing Algorithm (TORA), Associativity Based Routing (ABR), and Signal Stability Routing (SSR).

Transport protocols. TCP was primarily designed to work for a wired network. The error rates in wired networks are quite low and therefore packet loss in these networks is taken as an indication of network congestion. In the case of mobile ad hoc networks, mobility may cause route failures which result in packet loss and increased delays. The TCP misinterprets these losses as congestion and invokes the congestion control mechanisms which will lead to unnecessary transmissions. To improve the performance of the TCP protocol in a MANET, several TCP optimization techniques have been presented over the past few years. For example, techniques have been proposed to minimize the impact of mobility and link disconnection on TCP performance by the use of explicit link failure notification (ELFN) [Holland, 2002], a technique to distinguish between packet losses due to network congestion and other factors such as buffer overflow, transmission errors and mobility by using link connection information [Gerla, 1999]. However, Anastasi, [2009] show

that TCP based solutions might not be the best approach when operating in MANET environments.

1.3.3 Middleware and Applications

Middleware supports development of applications for information access and sharing with considerable flexibility. However, middleware support for ad hoc networks is still in the infant stage. The middleware developed for MANETS are tightly coupled with applications and there is no single general middleware that resolves all problems. STEAM [Meier, 2002], EXpeerience [Bisignano, 2003], EMMA [Musolesi, 2005], LIME [Murphy, 2001], LIMONE [Fok, 2004], XMIDDLE [Mascolo, 2002], MATE [Hadim, 2006] are some of the middleware developed.

1.3.4 Cross Layer Research Issues

In cross layer research issues, special attention is given to energy efficiency, security and cooperation and QoS. These issues cannot be completely implemented inside a single layer, but they are implemented by combining and exploiting mechanisms implemented in several layers.

Energy Efficiency. The mobile node in an ad hoc network has limited power supply and has no capability to generate power. In addition, the mobile nodes must implement all the network basic functions like routing and forwarding. It is therefore vital to have energy efficient protocol design at all layers while designing algorithms for mobile devices. Energy conservation requires a coordinated effort from all related layers, including the physical layer transmissions, the operating system and the applications [Basagni, 2004].

Network security and Cooperation. The ad hoc nature of MANET brings new security issues that are not addressed by the services provided for infrastructure based networks. As the nodes in MANET communicate with each other using the shared broadcast wireless channels, they are more vulnerable to security attacks. Compared to wired networks which have dedicated routers, each mobile node in an ad hoc network may function as a router and forward packets for other peer nodes. As a result, there is no well defined place to implement traffic monitoring or access control mechanism. Furthermore, the distributed and infrastructure-less nature of ad hoc networks makes a centralized security control hard to implement. The existing routing and MAC protocols assume a trusted and cooperative environment. Thus it becomes easier for an attacker to disrupt network operations. In order to ensure a complete security solution for MANET, the mobile nodes must co-operate each other. The cooperation should be in detecting and isolating misbehaving nodes. Table 1.1 gives the possible attacks on MANET at each layer [Yang, 2004].

Quality of Service. QoS is generally defined as a set of service requirements that needs to be satisfied by the network while sending a packet stream from a source to its destination. The network is expected to deliver a set of measurable predetermined service attributes to users in terms of end-to-end performance, such as delay, bandwidth, probability of packet loss, and delay variance (jitter). QoS attributes that are more specific to MANETs include power consumption and service coverage area.The multi-hop nature of ad hoc networks introduces several issues and difficulties for QoS support. These issues include features and consequences [Mohapatra, 2003]. Examples of features include unpredictable link properties, node mobility, and limited battery life, whereas hidden

Layer	Security issues
Application layer	Detecting and preventing viruses, worms, malicious codes, and application abuses
Transport layer	Authenticating and securing end-to-end communications through data encryption
Network layer	Protecting the ad hoc routing and forwarding protocols
Link layer	Protecting the wireless MAC protocol and providing link-layer security support
Physical layer	Preventing signal jamming denial-of-service attacks

Table 1.1: Possible attacks on MANET

and exposed terminal problems, route maintenance, and security can be categorized as consequences. Important QoS components in MANETs include QoS MAC, QoS Routing, and resource reservation signaling. The various QoS requirements include throughput, delay, jitter, error rate, battery life *etc.*

1.4 MANET Characteristics

MANETs inherit the common characteristics found in wireless networks in general which include:

- Packet loss due to transmission errors
- Variable capacity links

- Frequent disconnections / Partitions
- Limited Communication Bandwidth
- Broadcast nature of the communications

The characteristics specific to ad hoc networks include:

- *Wireless*—Nodes communicate wirelessly and share the same media (radio, infrared).
- *Ad hoc based*—A temporary network formed dynamically in an arbitrary manner by a collection of nodes as need arises.
- *Autonomous and infrastructure-less*—MANET does not depend on any established infrastructure or centralized administration. Each node operates in distributed peer to peer modes, acts as an independent router and generates independent data.
- *Multi-hop routing*—No dedicated routers are necessary; every node act as a router and forward its packets to enable information sharing between mobile hosts.
- *Mobility*—Each node is free to move about while communicating with other nodes. The topology of such an ad hoc network is dynamic in nature due to the constant movement of participating nodes causing the inter communication patterns among nodes to change continuously.
- *Anytime, Anywhere*—Ad hoc wireless networks eliminate the constraints of infrastructure and enable devices to create and join networks on the fly- anytime, anywhere—for virtually an application.

1.5 Application

Mobile ad hoc networks have been employed in scenarios where an infrastructure is unavailable, or to deploy one is not cost effective, or when there is no time to set up a fixed infrastructure. It allows users to access and exchange information regardless of their geographic position or proximity to infrastructure. In contrast to other wireless networks, all nodes in MANETs are mobile and their connections are dynamic. This decentralized character of the MANET makes the networks more flexible and robust. Historically, MANETs were first used by the military as a part of tactical networks to improve battle field communication. Although ad hoc networks were initially used in military applications, a number of non military applications have emerged due to the availability and advances in mobile ad hoc research. Some examples of these applications include rescue operations, emergency services, communication between vehicles, sensor networks and personal area networks [Basagni, 2004].

MANETs are ideal for crisis management applications such as disaster recovery, where the entire communication infrastructure is destroyed and establishing communication quickly is not possible [Shibata, 2007]. One of the many possible uses of ad hoc network is in personal area networks. Personal area networks are created when a small number of nodes meet spontaneously to form a network for the purpose of teleconferencing, file sharing, or peer-to-peer communication. Sensor Networks [Pandey,2010] is an adhoc network consisting of a large number of distributed sensor devices, which is densely deployed in remote areas to detect the environmental conditions such as temperature, pressure, humidity, sound, vibration, motion, pollutants *etc.* Applications of these networks include health monitoring, military surveillance, monitoring household items, animal

tracking *etc.* Mobile ad hoc networks allow rapid network deployment in an emergency situation. Emergency networks can be set up in remote or hostile areas where there is no existing communication infrastructure, thereby assisting relief work and rescue missions. A vehicular ad hoc network (VANET) provides communication between vehicles, roadside equipment and vehicles travelling in close proximity. Data is exchanged between nearby vehicles to provide traffic information and early warnings for accidents and road works. The purpose of vehicular ad hoc networks is to provide a communication network of safety and information for users [Wang, 2009].

1.6 Motivation

The top challenges for ad hoc network design are limited node energy reserves and communication bandwidth. Due to the ad hoc nature of this type of network, network devices are often battery-operated, and thus limited in their energy supply. Bandwidth, on the other hand, is limited by the physical operating frequency of the wireless channel. Cooperative caching protocols for mobile ad hoc networks should try to minimize energy consumption. So far in the literature there is little attention paid to communication overhead occurring in cooperative caching. Most of the cooperative caching schemes developed for MANETs often focus on reducing access latency while ignoring the cost of retrieving the data from neighboring nodes. The major factor that causes message overhead in cooperative caching is cache discovery. Generally, in cooperative caching there are two main approaches for cache lookup: Search based and Directory based. In search based lookup schemes, flooding is the technique used to disseminate data request among the neighboring nodes. For each

local miss, the search message is propagated to all the neighboring nodes in the network. Although time latency is minimal in flooding, it generates a large number of messages which will cause excessive control message overhead as unintended nodes have to receive and process these packets. As a result, the energy consumption and bandwidth utilization of the network is increased. Moreover, there is a chance of increased collisions which could degrade the overall performance of the system. Directory based technique uses a centralized approach in which a coordinator node will maintain the status of the data present in the neighboring nodes and the data requester can get the data directly from the coordinator node. This approach greatly reduces the number of messages and latency. However, this approach also imposes several challenges. As the mobile hosts move freely, maintaining group information is difficult and the control node may get disconnected which causes excessive overhead.

Another interesting research issue associated with the exiting approaches is network scalability. As the number of mobile clients increases, the communication overhead between the mobile nodes increases and the performance is degraded. Against this backdrop, this research work aims to design an energy efficient and scalable cooperative caching framework.

Because of the limited cache memory available in a mobile device, it may be necessary to remove some old data in the cache when the cache is above a certain limit. A cache replacement algorithm that can yield high hit ratio is necessary for the successful implementation of cooperative caches. Since the replacement algorithm decides the data to be cached and removed, it affects the cache hits of future requests. Another factor that determines the performance of cooperative cache is cache placement. Among the cooperative caching protocols reported in literature, only a

few have examined cache placement. Uncoordinated cache placement policies can lead to duplication of data. The final motivation for this research is to look into new cache placement and replacement policies that will improve the performance of the cooperative cache.

1.7 Objectives

The main objective of this research work is to develop an efficient cooperative caching framework to improve data accessibility in mobile ad hoc networks. The existing cooperative caching schemes have been studied and an attempt has been made to develop a new cooperative caching technique towards improving data accessibility with minimum energy consumption.

Cache discovery overhead increases with an increase in network size. Design of cache discovery protocols for cooperative caching should consider power and resource limitations of the mobile nodes. Previous studies show that communication is the major source of energy consumption. This emphasizes the need to design a cache discovery scheme that conserves energy. Previous literature says the cache replacement policies play a vital role in improving the cache hit ratio. A cache placement scheme that exploits coordination and sharing of cache state among the neighboring nodes can effectively utilize the cache space available in the cooperative cache.

In order to achieve this, the research work has focused on the following objectives.

- Review the existing cooperative caching schemes proposed for MANETs in order to determine the current state of affairs.

- To propose a novel cache replacement algorithm to improve the cache hit ratio.
- To propose a cache placement policy to store more distinct data items to improve the effective cooperative cache size and thereby increase the cache performance.
- To develop an enhanced cache discovery and data dissemination scheme for cooperative caching which reduces the message overhead consequently leading to lower bandwidth utilization and cost.
- To propose a scalable cache lookup and data discovery framework for cooperative caching in mobile ad hoc networks.

1.8 Contributions of the Research Work

The major contributions of the research work are listed below:

A new improved algorithm is developed for cache replacement which improves performance in terms of cache hit ratio and access latency.

In this work, cache replacement issues for ad hoc networks have been explored and a new cache replacement policy for cooperative caching is presented. The proposed algorithm is an extension of the basic Least Recently Used (LRU) algorithm. The algorithm takes in to account the access history of last two references and calculates the inter arrival time of recent references for cache replacement. In LRU only the last time of reference is taken and the number of references is not considered. Since the inter arrival time of the recent reference is taken, more preference

is given to objects that have been accessed more than once. Hence, we are able to distinguish between data that are frequently referenced and the data that are occasionally referenced. The newly developed algorithm takes in to account the frequency of access of the data item. This improves the cache hit ratio as well as response time of data access.

A coordinated cache placement policy to enhance the effective cache size of cooperative cache is proposed.

In this placement policy, the key idea is to cache as many data items as possible avoiding duplications. Therefore, new coordinated cache data placement algorithm is proposed. The decision on caching an incoming data is done coordinately among the neighboring nodes that already have a copy of the data item. This scheme has been proposed for effective memory utilization for mobile clients with limited memory. Simulation results show that the proposed policy can significantly improve the performance compared to independent cache placement schemes especially for applications with limited cache.

A distributed cache discovery algorithm which reduces message overhead is presented.

The objective of the proposed cache discovery process was to minimize the number of messages flooded in to the network, which in turn reduces the communication cost and bandwidth utilization. The proposed data discovery process is based on the position of the neighboring nodes. More precisely, message broadcasting is reduced by dividing the transmission range into two zones. This work is extended further to reduce the power consumption by dividing the transmission range in to smaller regions with small set of nodes in each zone. Simulations have been carried out

to find the effectiveness of the proposed models.

A novel virtual backbone based cooperative caching framework is developed to address network scalability.

Existing cooperative caching algorithms for mobile ad hoc networks face serious challenges due to message overhead and scalability issues. To solve these issues, an adaptive virtual backbone based cooperative caching that uses a Connective Dominating Set (CDS) to find the desired location of cached data is proposed. The idea in this scheme is to reduce the number of nodes involved in cache look up process, by constructing a virtual backbone adaptive to the dynamic topology in mobile ad hoc networks. The proposed algorithm is decentralized and the nodes in the CDS perform data dissemination and discovery.

1.9 Outline of the Thesis

Chapter 1 introduces the area of mobile ad hoc networks and cooperative caching.

Chapter 2 is a systematic survey of the existing cooperative caching protocols proposed for mobile ad hoc networks. A brief review of various cache replacement policies available for wireless networks is also presented.

Chapter 3 formulates a scheme for cache replacement. The implementation and performance analysis are also illustrated.

Chapter 4 explains a coordinated cache placement policy. A review of literature on various cache placement techniques is also presented.

Chapter 5 describes the implementation of a distributed cache discovery technique.

Chapter 6 discusses the various power conservative issues for mobile ad hoc networks. A power conservative design for cache discovery is also presented.

Chapter 7 describes the implementation of an adaptive virtual backbone based cooperative caching framework.

Chapters 8 summarizes the research work, highlights the contributions, and discusses the potential for future research.

Chapter 2

Introduction to Cooperative Caching

Contents

2.1 Introduction

In the previous chapter various characteristics of MANETs were discussed. In the data-centric point of view, these characteristics pose new problems that need consideration. Due to the ad hoc nature of MANETs, changes in topology occur frequently and traditional data management schemes fall short in providing data availability. In addition, multi-hop communication leads to longer query latency and high power consumption. The mobile nodes are battery-operated, and thus limited in their energy supply. Bandwidth is also limited by communication through wireless channels in several ways. In order to confront these issues and to achieve high data access performance, data transfers must be minimized so that wireless environment's limitations do not result in a degraded service. Data management in MANET invariably involves caching or replication.

Caching is a fundamental concept used in various domains in computer systems. It underlies many of the techniques that are used today to alleviate the latency experienced by the end users. Caching is implemented across a variety of domains including address translations, memory locations, pages, file blocks, file names, network routes, authorizations for security systems, and so on. A cache is a temporary storage area that keeps data available for fast and easy access. Once data is stored in the cache, it can be used in the future by accessing the cached copy rather than re-fetching or re-computing the original data. By keeping additional copies, caching can speed up the access to frequently used data, at the cost of slowing down the access to infrequently used data.

In web, copies of remote data can be cached locally on the user's computer or on a server on the web to reduce data retrievals from the original server. Caching on the web is different from memory, disk and file sharing systems. The caching systems on the web have to handle requests from a very large number of users, with varying size of data. Another issue in web caching is the randomness of objects chosen by a user surfing the web, where the popularity of objects may change from one day to the next. Web caching is a widely studied area and has influenced future research on MANET. Web caching reduces the amount of information that has to be transmitted across the network which in turn reduces the bandwidth and processing requirements of the server. Data caching on the web has the following advantages: First, the total latency associated with fetching data is reduced. Latency is reduced further if the proxy cache is located closer to the client. Second, proxy caching reduces the servers' load since cache hits do not involve the server. The transmit costs of access providers are also lowered. The network traffic is reduced as documents are retrieved from the cache rather than from the network. The success in web caching prepared the way for more research in ad hoc network caching. There has been continuing research in the area of ad hoc network caching, although the field is not as mature as web caching. The following sections of this chapter detail the issues in data caching in ad hoc networks, architecture and taxonomy of cooperative caching architecture in ad hoc networks.

2.2 Factors influencing data caching design

Caching can improve data availability and performance in distributed systems. Data availability is improved by allowing access to the data even

when a mobile node is disconnected from the data server. Performance improvements include reduced latency which is achieved by letting users access nearby copies of data and avoiding remote network access. More than the availability and performance issues, data caching in ad hoc networks is influenced by additional issues arising from the constraints imposed by the ad hoc network environment which includes: Autonomous and infrastructure-less network, Multi-hop routing, Frequently changing network topologies and Variation in link and node capabilities[Gupta, 2005]. Detailed discussion about these issues is given below.

Autonomous and infrastructure-less: MANET does not depend on any established infrastructure or centralized administration. Each node operates in distributed peer-to-peer mode, acts as an independent router and generates independent data. As the network management is distributed across different nodes, fault detection and management is difficult. Mobile hosts often get disconnected from the network due to various factors like power failure and mobility. In the case of disconnection, servers which hold the data cannot provide services to these mobile hosts.

Multi-hop routing: In MANETs due to limited radio range of wireless devices, the route from one device to another may demand multiple hops. Every node acts as a router and forwards the packets to enable information sharing between mobile hosts. These characteristics raise several challenges for data access applications regarding data availability and access efficiency.

Variation in link and node capabilities: The next characteristic of ad hoc networks which affect data caching is the variation in link and node capabilities. Each node may be equipped with one or more radio

interfaces that have varying transmission/receiving capabilities and operate across different frequency bands. In addition, wireless links have substantially lower capacity compared to their hardwired counterparts [Corson, 1999]. Besides, mobile nodes may have to compete with several other mobile nodes to get access to an upstream channel. This consumes battery power since a mobile node may have to keep its network interface alive from the time it initiates the request to the time the response is received from the server. Exploiting the variation in link and node capabilities of wireless connectivity to ensure low data latency and resource consumption is a new problem posed by these characteristics of wireless networks.

Weak Connectivity: Each node in an ad hoc network is free to move randomly. This feature makes unpredictable changes in the network topology. Also, the wireless links may be of low bandwidth in comparison to wired links leading to weakly-connected mobile clients. Thus the mobile clients are often disconnected from their data servers. These disconnections can be either voluntary (e.g. when the user disables the wireless network interface card (NIC) to conserve battery power) or involuntary (e.g. when the user moves to an area where wireless service is not available) [Gupta, 2005]. Even though frequent disconnection occurs, the user should be able to allow applications and services to operate without disruption. Ensuring high availability of data in mobile computing environments in the case of frequent disconnections creates a new challenge for data management in ad hoc networks.

Low-Power and Resource-Limited Operation: Another factor influencing data accessibility in a MANET environment is energy constrained operation. The power source which supplies energy for the mo-

bile nodes consists of batteries with limited energy budget. Moreover, these batteries could not be recharged as the nodes are deployed in a hostile or unpractical environment. This will limit the services and applications that can be supported by each node. In ad hoc networks since each node is acting as both an end system and a router at the same time, additional energy is required to forward packets from other odes. Cache management in ad hoc networks has to take care of this problem i.e., how to reduce the energy and bandwidth consumption while ensuring desirable level of data accessibility.

Location and Time dependent data: In an ad hoc network environment data may be location and context dependent [Gupta, 2005]. This feature influences the cache design in ad hoc networks. A mobile client may retrieve data from various databases periodically which is both location-dependent and time-dependent information. The mobile clients may have similar sets of desired data while they are traveling in the same location. For example, people are more likely to ask for information about animals/birds when they are visiting a zoo, but people who are shopping at a downtown area hardly have the interests to know anything about a lion or a deer. For a traveller visiting a city may need to know the list of restaurants in its vicinity. Therefore, the type of information people access is related to their location. Caching can be an effective technique in the cases mentioned above, to reduce the impact of low bandwidth, longer query delay and higher energy consumption. "How to improve the existing cache management techniques for context-dependent data?" is another issue which needs to be addressed for mobile computing environments.

Due to the above challenges, the design goals of wireless caching are

not same as those of wired network caching [Chen, 2006]. The following are goals for designing good caching in wireless environment.

Increasing cache hit: For a caching system, if a requested document is found in the cache, the request can be satisfied immediately, which is called hit; otherwise, the document has to be fetched from the original server, which is termed as a miss. Improving cache hit ratio is one of major design goal in wireless caching.

Reducing communication cost: Since bandwidth in a wireless environment is limited, communication cost is a very important issue. An effective cache hit reduces communication cost by avoiding the fetching of the data from the server. In some caching schemes the server has to inform the clients when the data has been updated. This notification may cause a different cost. Furthermore, in cooperative caching cache discovery involves a different communication cost. A caching algorithm may aim to minimize this communication costs.

Reducing energy consumption: Since energy is an important constraint of mobile devices reducing energy consumption is another goal. A scheme may minimize the number of transmissions to reduce energy consumption.

Reducing latency: Many research works have been done towards reducing latency in ad hoc networks. Caching is identified as an important technique for reducing latency in these types of networks.

Some of the above goals are conflicting. Satisfying all those design goals at the same time is unlikely to be achieved. A good algorithm needs to be adjusted towards different goals based on different network conditions, mobile devices, application requirements, and users' requirements.

2.3 Data Caching and Replication

Replication is a technique used in distributed systems for storing the same data or service at multiple nodes [Derhab, 2009]. With replication, data generated one time is copied or replicated to multiple nodes. By replicating data at mobile clients, data availability can be improved as there are multiple copies of it in the network and therefore probability of finding copy of data is increased. It can also reduce the query delay, since mobile nodes can get the data from some nearby replicas. Below we discuss some important works related to replication in MANETs. A detailed survey on replication techniques can be found in [Padmanabhan, 2006].

Hara, [2001] proposed data replica allocation techniques adopted in mobile cooperative caching to improve data accessibility and to alleviate the network partitioning problem. Three data replica allocation schemes were proposed: SAF (Static Access Frequency), DAFN (Dynamic Access Frequency and Neighborhood) and DCG (Dynamic Connectivity based Group). In SAF each mobile host (MH) considers its own individual access probability to each data item for replication and arranges data items in the descending order of their access frequencies. In DAFN, SAF is extended to take the access probability of data item present in the MHs' connected neighbourhood. The DCG method shares replicas in larger groups of mobile nodes. The groups are formed by mobile nodes that are bi-connected components in the network. In DCG access probability to each data item of all MHs in the same group are taken into account. Performance evaluation shows that DCG gives the highest data accessibility, but subject to higher network traffic than the other two schemes. These three methods are extended and presented by Hara [2003] to handle peri-

odic and a periodic updates. Extended SAF (E-SAF), Extended DAFN (E-DAFN) and Extended DCG (E-DCG), consider periodic data update by allowing the extended allocation schemes to consider the remaining time until next update of each data item. In addition, to decrease the effect of network partitioning, Hara, [2003a] also considers the stability of radio links. The stability of a radio link is defined as the remaining time period that two MHs will still be connected to each other. If the time period is longer, higher is the stability of a radio link. Duplicate replicas of data items are eliminated between two nodes if the wireless link that connects them is stable. These methods assume that each node knows the speed and the direction of the movement through Global Positioning System (GPS), and thus the time at which two neighbouring nodes will be disconnected is estimated by their movements.

Huang [2003] addresses the problem of replica allocation in a MANET by exploring group mobility. They propose a scheme called DRAM to allocate replicas by considering group mobility. It is assumed hat some MHs tend to roam together and they share a common access range. To find the group mobility pattern among MHs, a decentralized clustering algorithm is proposed to cluster several MHs that possess similar mobility pattern into a group. The clustering algorithm is executed periodically to adapt to the changes in network topology. Then, the data replicas are allocated to each group member based on group access probability to the data items, and the remaining time until the next update on them. DRAM is found to perform better than E-DCG in terms of data accessibility and network traffic.

Although replication and caching do the same function, pre-fetching the needed data and storing it closer to the source, there are a number of

differences between them [Derhab, 2009]. First, for getting data closer to the source, replication uses a separate process to copy data items to the target nodes. Caching is a by-product of query execution. Replication copies data at servers even if no queries are processed by these servers, whereas data is not transferred to a client node if no queries have been processed by that client. As a result, caching decisions are made by the query process while replication decisions are made by a separate component that is established at every server and works independently of the query process.

Second, replicas are stored in the servers until they are explicitly removed whereas copies of data are kept in a cache until they are evicted by copies of other new incoming data using a replacement policy or until the data become invalid due to consistency issues. Third, replication uses propagation-based protocols to keep replicas of data consistent and accessible at servers at all times. For caching, consistency is maintained by using protocols that are based on invalidation. Finally, the difference comes in the amount of data cached or replicated. Replication copies all or the majority of data items and supports a large group of clients. Caching stores data items based on queries raised by individual clients and supports fairly small group of clients. In caching it is assumed that users will be only interested in a small fraction of the data store at the server.

Although many replication protocols have been proposed for ad hoc networks, focus is given to caching protocols as they can improve data access efficiency more than the replication due to the following reasons. Replication requires an earlier knowledge of the operational environment and is susceptible to node mobility. This makes data replication un-

suitable for highly dynamic network topologies. A common requirement when data is replicated is consistency which can vary in strength between applications. Increasing the number of replicas will increase the replication cost and update cost. Caching reduces memory usage and network bandwidth compared to replication. As the cache stores only used data, memory usage is reduced. Caching reduce resource usage through reduction in round trips. The advantage of caching over replication makes caching a better technique for improving data access efficiency in MANETs.

2.4 Cooperative Caching

The response time for retrieving data in ad hoc network varies greatly. This is because multiple factors affect the response time, like: the amount of bandwidth available at a given time, the amount of traffic and the number of requests the server is trying to handle. Caching eliminates this delay and unpredictability to a certain extend by storing frequently accessed data close to the user. In traditional caching when we need a particular piece of information, the request is first sent to the local cache. If the requested information is present in the cache it is directly sent back to the client, skipping an upstream journey to the data server. If the requested information is not present in the local cache, the client will retrieve the data from the data server. Once a copy of the object has been retrieved a copy of that information is made in the cache assuming that we will need it again soon. By doing so, we decrease the searching time of that particular information. This traditional way of resolving an application's data request is called non-cooperative caching scheme. This scheme works well for infrastructure based networks as they have a

reliable high speed connection to the server. But for MANETs, mobile nodes have only limited storage space and it is impossible to keep all data in one node. Due to resource constraints like low bandwidth and energy, if the data server handles all the requests, the wireless channels near the data server becomes congested and the odes closer to the server consume more energy as they have to relay data for others. This will affect the whole network. However, if the mobile nodes can share the data they have stored in the local cache, the whole network will be benefited. Cooperative caching is a powerful paradigm to improve cache effectiveness, where caches cooperate in serving each other's request and in making storage decisions. Cooperative caching increases cache utilization by coordinating the usage of distributed caches. Since the mobile nodes can make use of the data stored in another node's cache the effective cache size is increased. Such cooperation is particularly attractive in ad hoc networks [Yin, 2006]. In cooperative caching, if a mobile node experiences a local miss the query is forwarded to the neighboring nodes. This enables some of the local misses to be handled by the neighboring nodes, offloading the server and improving the performance of the system.

The operation of cooperative caching scheme is depicted in Figure 2.1. In case the mobile node A needs a document, it first checks in its own local cache for a valid copy of data, if it finds one the node reads the data from the local cache, otherwise node A sends a data request to its first hop neighbors B, C and D and they perform a cache lookup. If the data is not present in the neighboring nodes, the source node A forwards the request to the data source S using appropriate routing protocol. The data source node S will respond to the request and finally node A will store the document in its local cache. If node B requests the same

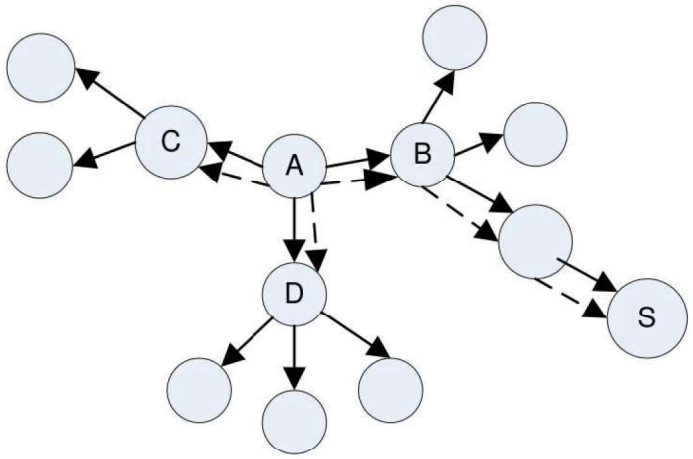

Figure 2.1: Cooperative Caching in MANET

document which is not present in its local cache, node A will respond to the data request with the document. This reduces the number of hops from B to S, which consequently reduces the latency perceived by each node. Fig 2.2 illustrates the framework for cooperative caching.

Cooperative caching in ad hoc networks is effective because of data locality and commonality in user's interest. Users around the same location tend to have similar interests. For example, a group of researchers who want to share their research findings or presentation materials during a conference or a lecturer distributing notes to a class have a common interest.

Two examples, one taken from an academic scenario [Taniar, 2007] and the other from an emergency service, are used to illustrate how cooperative caching can help to improve data access efficiency in MANETs.

Consider an academic environment. The main entities include all

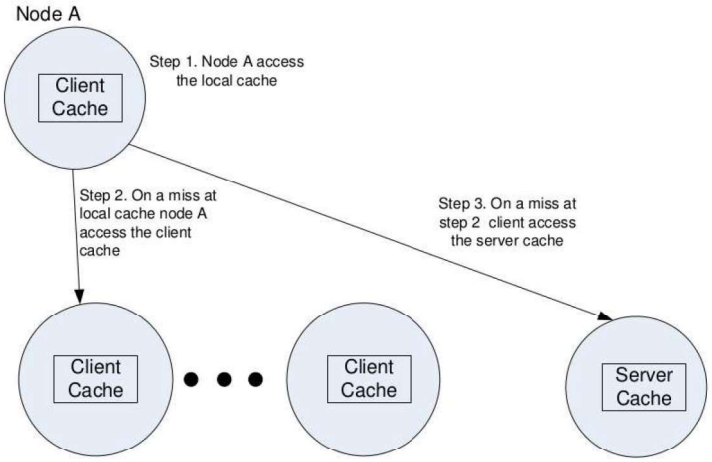

Figure 2.2: Framework of cooperative caching

different types of academics existing in university environment like students, instructors and lecturers, researchers and professors as well as visitors to the campus. A student can make enquires for local information (maps, buildings), monitor the information updates (announcements, urgent notices, deadlines, events), meet with a colleague or other student mates, exchange data with others (docs, mp3, video clipsetc), communicate with professor/tutor/technical supervisor, respond to emergency situation within the campus area and so on. Personal networks (PN) can be configured in an ad hoc fashion establishing any possible peer to peer (P2P) connection among users belonging to the same local Personal Area Network (PAN) and other remote PANs as well. PN consist of communicating clusters of personal digital devices, possibly shared with others and are connected. The students can be informed about course announcements, important notice from the student office or from any

other local online stores related to his or her.

Another area in which MANETs can be very useful is emergency search and rescue sites, where the conventional infrastructure-based communication facilities are destroyed due to natural calamities such as earthquakes, or simply do not exist. Deployment of MANETs in these scenarios can be used for rapid activity coordination. For instance, police squad vehicles and fire brigades can remain connected and exchange information about the rescue arrangements more quickly if they can cooperate to form ad hoc networks. The mobile nodes that have interfaces to external networks serve as gateways to allow other mobile nodes to communicate with external data servers. From these examples, we can see that MANETs are formed for data transfers of similar interests, where cooperative caching which utilizes the sharing and coordination of cache state among neighboring nodes can help to save time and energy costs.

The P2P architecture of MANET makes cooperative caching a better model to improve cache performance. Cooperative caching provides more cache space and faster speed in these networks. Since the nodes can link to cooperate and serve each other's misses by sharing and coordination of cache state among the neighboring nodes network traffic and resource consumption is further reduced. Apart from this, the benefits of cooperative caching include:

- Bandwidth savings in the whole network can be higher than the single cache approach.

- Multiple caches increases cache hit resulting in efficient utilization of cache resources. Most importantly, this also improves user response time.

- While single cache can often be a single point of failure, by having multiple caches system fault tolerance is largely improved.

Correspondingly, a cooperative caching scheme needs to specify [Du, 2004]:

- How users' requests are processed using the cooperative caching system? That is, once a caching node receives a data request, how does it proceed to resolve the request?
- How does a caching node manage the cached data on behalf of the cooperative caching system?
- How to reduce client side latency?
- How to maintain cache consistency between various caches and the servers?
- How to ensure high data availability in presence of frequent disconnections?
- How to achieve high energy/bandwidth efficiency?
- How to determine the cost of cache miss and how to incorporate this cost in cache management scheme?
- How to enable cooperation between multiple peer caches?

The next section presents the cooperative caching design and architecture proposed to address these issues.

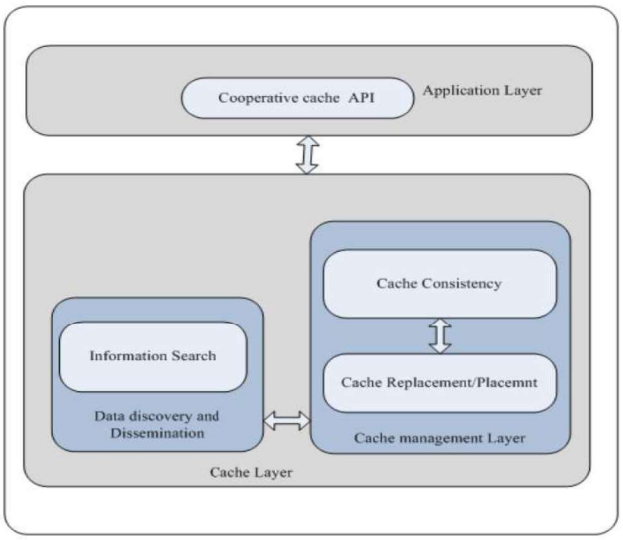

Figure 2.3: Cooperative Caching Architecture

2.5 Cooperative Caching Design and Architecture

In cooperative caching mobile nodes cache the data or path to the data to enhance data availability. The data is stored in a distributed way and is shared in the entire network. The design of a cooperative caching system mainly focuses on cooperative data dissemination and cooperative cache management [Chin, 2006]. Cooperative data dissemination mainly focuses on designing protocols for data discovery and information search. Discovery refers to how a mobile client locates cached objects. The research work on cooperative cache management mainly focuses on developing algorithms for mobile nodes to manage their cache space. A

good cooperative cache management technique should have proper cache admission control, cache replacement and cache consistency maintenance schemes. The cache admission control module decides whether a received data item is cacheable or not. Since the network node has limited memory, it can cache multiple data items subject to its memory capacity. The objective of cache admission control module is to store more distinct data items in the given cache space. Consistency strategies ensure that each node caching the data item is aware of the data update at the source. The cache replacement policy decides the items to be removed from the existing cache to make way for new ones. Replacement algorithms should keep track of the access pattern to improve the cache hit ratio. The consideration for data availability and energy efficiency is embedded in the research of cache management and data discovery techniques developed for mobile ad hoc networks. Fig. 2.3 illustrates the basic building blocks of a cooperative caching architecture.

2.6 Classification of Cooperative Caching Techniques

All cooperative caching techniques implicitly assume that nodes will cooperate each other in delivering the desired data. Based on how the mobile nodes coordinate each other for cooperative caching, the proposed caching techniques can be categorized in to decentralized group based architecture and centralized group based architecture. In decentralized group based architecture, mobile nodes are arranged in to different groups based on various parameters. Each node will belong to only one group. There is no control node in the group and the clients can access data from

any other node in the group. In centralized group based architecture, a mobile node belongs to different groups based on various criteria. Here each group is associated with group control node to handle query and update operations.

Depending on functionality, the different modules in cooperative cache management can be further divided into different groups. The data discovery process can be either push based or pull based. In push based approach, the data requesting node can directly find the node which has the cached copy of data. In pull based approach when a local cache miss occurs, the request is broadcasted to its neighboring nodes and a node which has the data will respond. Flooding is the technique used for broadcasting.

Cache admission can be done in two different ways. In the general scheme all the incoming data are cached. In function based cache admission control, a cost function will decide whether to cache the incoming data. The value function is formed by combining different parameters like distance and access frequency. The cache replacement policies were classified into two groups, local and coordinated. In local replacement the data item to be evicted is determined independently by each node based on its local access information. In coordinated replacement policy the mobile nodes which forms cooperative cache collectively takes the replacement decision. There are two kinds of consistencies: strong and weak. A cache access mechanism can either use strong or weak consistency depending on applications. Cooperative caching techniques mainly uses weak consistency model based on time to live parameter. A detailed description of various techniques is given in the subsequent chapters. Fig. 2.4 shows the classification of different cooperative caching techniques.

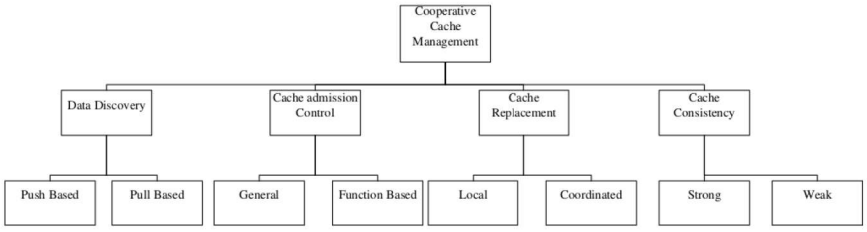

Figure 2.4: Taxonomy of different cooperative cache management schemes

2.7 Conclusion

Data availability in MANET can be increased by caching and replication. However, different characteristics of mobile computing environment pose new challenges for cache management. These schemes have to work efficiently in the presence of frequent disconnection with minimum energy consumption. In this chapter, several design issues concerning the development of a caching protocol for MANETs have been discussed. Through the study of different data replication protocols, it is found that each replication method has advantages and disadvantages and in certain situations it exhibits poor performance. Cooperative caching can be instrumental in solving problems of data availability in ad hoc networks. Effective cooperative caching involves proper data dissemination and cache management. Existing cooperative caching protocols have been classified based on how they addressed the identified issues.

Chapter 3

Related Works

Contents

3.1 Introduction

Cooperative caching is a powerful paradigm to improve cache effectiveness where caches cooperate in serving each others' request. The idea of cooperative caching has been introduced in web caching to improve web performance since early 90's. Cooperative caching on web uses proxy servers to cache documents that can be retrieved by the same group of clients. A lot of research work has been done in the area of web caching. This chapter gives a brief summary of previous works on web caching. Then surveys of the state of the art techniques which have been used in cooperative caching in ad hoc networks are given. Cache replacement algorithm plays a vital role in improving the performance of cooperative caching strategies. Finally, a brief summary of various cache replacement policies proposed for wireless and ad hoc networks are also presented in this chapter.

3.2 Cooperative Caching Strategies

Cooperative caching has long been employed in Network File Systems and in the World Wide Web (WWW) as an effective technique to reduce client latencies and to save network bandwidth. Traditional network file systems rely on a central server, where all the data is stored and satisfies all client requests. A distributed File system [Anderson,1995], on the other hand, distribute processing control and storage to all the machines with the help of fast local area networks, to provide better performance and scalability. High data availability is also achieved due to redundant

data storage. In World Wide Web, Internet proxy caching has become potentially beneficial in improving the performance of Web browsers.

Typically, the proxy is placed in front of an entire company or organization. By caching requests for a group of users, a proxy can quickly return documents previously accessed by other clients. Cooperation among proxies increase total client population, improve hit ratios, and reduce document-access latency. The performance of such cooperative proxy caching depends on a number of factors. These include the sharing patterns of documents across organizations, the ratio of inter-proxy communication time to server fetch time, and the scale at which cooperation is undertaken [Wolman,1999].

Several cooperative caching protocols have been proposed in literature for web caching to reduce access latency and bandwidth consumption. These proposals include hierarchical schemes like Harvest and Squid [Chankhunthod,1995, Wessels,1998], hash-based schemes [Karger,1999, Valloppilli,1998], directory-based schemes [Fan,2000, Menaud,1998] and multicast-based schemes [Michel,1998, Touch,1998]. In hierarchical schemes [Chankhunthod,1995], when a miss occurs in a cache, a query is send to all its siblings using Internet Cache Protocol (ICP). If the data is not found, it forwards the request to its parent. The hierarchies are manually configured and ICP is not a very scalable protocol. The directory based systems [Fan,2000] proposed a Bloom filter to reduce the amount of information exchanged between proxies. In hash based systems [Karger,1999] inter-cache communication is eliminated upon the occurrence of a miss, by letting clients decide for themselves which cache has the required data. On a cache miss, instead of contacting a primary cache for locating the desired data in another cache, a user's browser directly contacts

the cache that contains the required resource. Web browsers make their decision with help of a hash function that maps resources (or URLs) to a dynamically changing set of available caches. Multicast based systems tries to locate the requested resource in other cooperating caches by using multicasting. However, these schemes designed for web cannot be directly applied to ad hoc networks. In web, cooperation occurs between dedicated cache servers with high speed network connections. Since they have a fixed infrastructure, these structures on the network of cooperative nodes, such as hierarchical, hash-table based, and directory-based etc. can be introduced to search the desired data. In the case of MANETs, its infrastructure is not fixed and multi-hop routing makes it difficult to manage cooperation among the nodes. As seen in section 2.2 data caching in mobile environments are characterized by different challenges.To address these challenges a number of researchers have proposed different cooperative caching techniques for mobile ad hoc networks.

3.3 Cooperative Caching Strategies for MANET

To increase the data accessibility in ad hoc networks, several data caching schemes have been proposed in the literature.These schemes allocate every mobile node a certain amount of cache space and the cache space is shared among the neighboring nodes. Here we compare some representative cooperative caching strategies for mobile ad hoc networks which combine the different aspects of the cooperative caching mentioned in the previous chapter.

Yin [2006] has proposed three caching schemes, Cache Data, Cache Path and Hybrid Cache, to provide efficient data access in MANETs. In Cache Data scheme, if the intermediate nodes in the routing path

between the source and the destination have enough free cache space, it caches the data item which is popular. There is no cooperative caching among the mobile hosts. Each mobile host independently performs the caching tasks such as placement and replacement. Cache Path works in a similar manner to Cache Data, but the forwarding nodes cache the data path information for future use, instead of storing the data. Hybrid Cache takes advantage of the above two schemes for further improving the caching performance. Here a mobile node caches the data or the path based on some criteria. These criteria include the data item size and the Time to Live (TTL). If data item size is small, Cache Data is used since the data item needs only a small portion of the available cache; otherwise, Cache Path is used. If TTL is small, Cache Data is used because the data item might soon be invalid, using Cache Path can result in changing the path frequently. Here the cache consistency is ensured by TTL mechanism. A routing node considers a cached copy valid if its TTL hasn't expired. If the TTL has expired, the node removes the cached data item.

The disadvantage of Cache Data algorithm is the consumption of caching space. A lot of caching space is consumed for the forwarding nodes. Also the data could not be used when a node moves from one network to another network. In Cache Path algorithm the path becomes obsolete when a client moves away from the network. Another disadvantage of this caching policy is that the forwarding nodes will become exhausted if different nodes access the data items continuously over certain period of time.

A caching scheme for Internet based MANETs was presented by Lim [2003]. In this scheme, an Aggregate Caching mechanism is pro-

posed. Here each mobile node stores data item in its local cache and these local caches are aggregated to unified large cache. The data items can be received from local caches of the mobile nodes or through an access point or a data centre connected to the Internet. When a node needs a data item, it broadcasts a request to all of the adjacent nodes. The node which has the data item in its local cache will send a reply to the requester; otherwise, it will forward the request to its neighbors. Thus a request is broadcasted to the other connected nodes and eventually acknowledged by an access point or some nodes with cached copies of the requested data item. A broadcast based approach, called Simple Search (SS) algorithm, is implemented on the top of existing routing protocols, to locate the requested data items.

COOP, a caching scheme proposed by Du [2005], addressed two basic problems in co operative caching i.e., cache resolution and cache management. In cache resolution COOP tries to discover a data source which includes less communication cost by utilizing historical profiles of forwarding nodes. For this a mobile node uses three schemes; adaptive flooding, profile-based resolution and road side resolution. In adaptive flooding, a node uses constrained flooding to search for items within the neighborhood. In profile-based resolution, a node uses the past history of received requests. In road side resolution, forwarding nodes caching the requested item, reply to the requests instead of forwarding them to the remote data source. In cache management, in order to minimize the caching duplication between neighboring nodes it uses inter-category and intra-category rules. Cache consistency is maintained by TTL scheme.

Chand [2006] proposed a zone based caching scheme for efficient data retrieval in MANETs. In this scheme, the set of one-hop neighbours form

a cooperation zone. The mobile nodes (MN) within the zone form the cooperative cache and shares data. The data source is the node where the data is originated. Each MN stores the frequently accessed data items in its cache. When there is a miss in the local cache, the node broadcasts the request to its zone members. If any of the mobile nodes has the data item, it responds with an acknowledgement. If the zone members do not contain the data item, the request is forwarded to the server along the routing path. If data is not found, each mobile node in the routing path search for the data item in its local cache or zone cache before forwarding the request to the next node. Once a mobile node receives the requested data, it cache the data based on server's location. If the server of the data is in the same zone of the requesting MN, then the data is not cached. Hence, mobile nodes in the same zone can store different data items. However, the delay may become longer if the neighbors of intermediate nodes do not have a copy of the requested data item for the request, because requested node wait for the reply before it resend the request to server. A cache replacement policy based on Least utility Value (LUV) has been used. For computing the LUV of a data item the access probability (A_i), size of the data item (S_i), coherency which can be known by TTLi field and distance (δ) between the mobile client and data source were considered for *utility* function for a data item (d_i)

$$\text{utility } i = \frac{A_i \cdot \delta_i \cdot TTL_i}{s_i}.$$

The concept of *Neighbor Caching* (NC) [Cho,2003] is to deploy the cache space of inactive neighbors for caching tasks. In this method a node uses its neighbor temporarily as follows. When a node takes data from a remote node, it puts the data in its local cache for reuse. A least recently

used replacement algorithm is used to remove the data from the cache. The data evicted from the cache is stored in the idle neighbor node's storage. If the node needs the data again, it will retrieve it from the near neighbor that keeps the copy of the data. This scheme exploits the available cache space of neighbor to improve the caching performance. In order to find a neighbor node with the highest possibility to keep the data for a long time a ranking based prediction algorithm is used.

Group caching [Ting,2007] uses a table based approach for cache discovery and data dissemination. Each node maintains two tables, a group table and self table to maintain the status of neighboring caching nodes. In order to utilize the cache space of each node in a group, the nodes periodically send their caching status in a group. By checking the cache status of the group, each mobile host knows the remaining available cache space of other mobile host in a group, the IDs and time stamps of their cached data object. Thus, the mobile host selects the proper group member for cache placement and replacement. For cache discovery, the local cache table is searched first and if data is not found, the group table is searched to find the location of the cached data. The group table contains cached data id and the node id which contains the data. The cache replacement is based on LRU policy and a cache consistency is based on TTL attribute. The drawback of this approach is the number of control messages exchanged when the node density is high. Also individual nodes have to process these messages which increase the computational overhead.

In *Cluster-based Cooperative Caching* (COCA) [Denko,2008] a cross-layer approach to improve caching performance is proposed. COCA is a middleware which implements cross-layer optimization for cooperative

caching. It consists of a stack profile, clustering, information search, cache management and prefetching modules. The stack profile module provides cross-layer information exchange, the clustering module for cluster formation and maintenance, the information search module for locating and fetching the data item requested by the client. The basic task of prefetching module is to anticipate user's request and to fetch and cache highly likely data items in advance. The cache management module includes a cache admission control, cache replacement and cache consistency. For cache admission control, a node caches all received data items until its cache space is full. When the cache is full the data items are not cached if the cluster has a copy of it.For cache consistency this scheme uses TTL strategy and uses Least Recently Used Minimum (LRU-MIN), for cache replacement.

Resource Efficient Adaptive Caching [Hirsch,2010] uses an adaptive cache distribution and replication. The adaptive cache replication scheme, known as Tidal Replication, considers a global demand for data for making replication decision. Here replication is done by caching the data items that consume significant portion of bandwidth for a control node towards a Dominant Request Path i.e., the network ath responsible for majority of request for a particular data item at the given node. When the demand wanes, the replicas are de-allocated and the burden for the data item returns to the control node. When replication request is received the available storage space for the node (RSA) is given by,

$$\text{RSA} = \frac{s - m - c}{s - m},$$

where s represents the total storage capacity of the node, m represents the total storage utilization of its non replica data items and c repre-

sents the total storage utilization of any cached replica it is holding. For resource efficiency an adaptive de-allocation methodology based current utilization and age has been included.
The current utilization is given by

$$U = \frac{x_i}{z_i},$$

where x_i is the windowed cumulative number of requests, and z_i is the age of the replica, for the given data item i. The age based de-allocation threshold Tag e is represented in Eq. where y_i is equal to the age threshold time in minutes, and z_i is the age of the replication.

$$\text{Tag } e = \lfloor \frac{y_i}{z_i} \rfloor.$$

The *Magnetic Distribution scheme* redistributes data items according to their demand, balancing the demand in a place of equilibrium, in order to reduce hop count and response time.

COCAS [Artail,2008] is a distributed caching scheme designed for finding the requested data from cached nodes. In this scheme node can take two roles: CN (Cache Nodes) and QD (Query Directories). The submitted queries are cached in query directories (QD) and these queries are used as an index to find the previously cached data. The CN's task is to cache responses to queries. The QDs are the central component of the system and nodes that are expected to stay the longest in the network and with sufficient resources are selected as QDs. Nodes have to calculate and store a special score that summarizes their resource capabilities, including the expected time during which the device is in the MANET, the battery life, the available bandwidth, and the available memory for caching. To

be considered a candidate QD, the device must meet a minimum criterion in each category. Whenever a data item is retrieved, cache nodes (CN) cache the data and the nearest QD to the cache node will cache the query along with the address of the CNs containing the corresponding data. When a node requests data that is not cached in the system (a miss), the database is accessed to retrieve this information. Upon receiving the response, the node that requested the data will act as a CN by caching this data. The nearest QD to the CN will cache the query and make an entry in its hash table to link the query to its response. The assignment of QDs and CNs are done by a service manager. The limitations of this scheme include, broadcasting of requests for searching QDs, and the single point of failure of the service manager.

CLIR (Cross Layer Interception and Redirection) proposed by [González-Cañete,2012] is an information and direct request caching scheme that implements a local cache in every mobile in the wireless network. In this scheme the local cache can directly serve the documents that are requested by the mobile nodes as well as the requests that are forwarded by the other nodes in the MANET (request interception). Thus, a mobile node can reply to a node requesting a certain document using the copy stored in its local cache instead of forwarding the request to the data servers. The mobile nodes manage information about the location of the documents disseminated in the network by analyzing the messages they forward. Using this information, the mobile nodes redirect the requests to other nodes that are closer to the requesting node than the DSs. In addition, a cross layer interception caching, in which the routing algorithm is manipulated for the process of looking for the documents in the MANET. A Redirection Caching procedure is implemented in all nodes

to provide up to date information about the location of the documents. LRU is the replacement policy used.

Fiore [2011] introduced a cooperative caching scheme (Hamlet) for mobile ad hoc networks where information is exchanged in a peer-to-peer fashion among the network nodes. The nodes in the network contain large and small-sized caches. For large-sized caches, a caching strategy where nodes, independent of each other, decide whether to cache some content and for how long is proposed. In the case of small-sized caches, a content replacement strategy is devised for the nodes to store newly received information. In both schemes each node takes this decision according to its perception of what nearby users may be storing in their caches and with the aim to differentiate its own cache content from the others'. The result is the creation of a content diversity within the nodes neighborhood, so that a requesting user can find the data easily. The decision of what information to keep, and for how long, is based on a probabilistic estimate of what is cached in the neighborhood. In this process each user is allowed to estimate for how long they should store in their cache an information item they requested and, thus, act as providers for that content. This quantity is taken as the information cache drop time; it is computed separately for each information item and applies to all chunks belonging to that item. The information drop time equal to zero means that the user does not store the content in its cache. The process to estimate the cache drop time is based on the node's observation of the information presence in its proximity by observing the queries and information messages that are sent on the wireless channel as part of the content sharing application.

A cooperative data dissemination scheme called energy-efficient coop-

erative caching with optimal radius (ECOR), [Shen,2004] was designed for power conservation in a mobile environment. In ECOR, an optimal radius (in number of hops) is estimated by an analytical model that considers the MH's location, data access probability and network density for each data item. The MHs exchange the cache content and the optimal radius of each cached data item among themselves. When an MH encounters a local cache miss, if it finds that any peers cache its desired data item and the distance between the MH and the peer is within the optimal radius based on its local state, the MH sends a request message to the peer that is the closest to the selected holder of the data item. Otherwise, the MH obtains the data items from the MSS.

Two group-based mobile cooperative caching schemes, namely, GroCoca [Chow,2004] and DGCoca [Chow, 2005], make use of a concept of a tightly-coupled group (TCG) that is defined as a group of MHs that are geographically and operationally close, i.e., sharing common mobility and data access patterns. Two MHs are considered to be geographically and operationally close based upon their locations and the set of data items they access respectively. GroCoca (GROup-based COoperative Caching) is a centralized group-based mobile cooperative caching scheme, in which the mobile nodes uses an incremental clustering algorithm to discover TCGs based on the weighted average distance and data access similarity of any two MHs. In GroCoca, when a MH encounters a local cache miss it can get the data from its peer. The data is cached only if its local cache has not been fully occupied or the peer is not belonging to its TCG. In other words, the MHs do not cache the data items that are provided by their TCG members, on the belief that the data items can be readily available from the peer if needed. On the contrary, DG-

Coca (Distributed Group based Cooperative Caching) is a distributed group-based mobile cooperative caching scheme, in which a stable neighbor discovery algorithm is proposed for the MHs to discover their own TCG's members dynamically without any help of the MSS. The MHs adopting DGCoca use cooperative cache replacement to further improve data accessibility. The proposed cooperative cache replacement protocol possesses three important properties. First, the most valuable data items are always retained in the local cache. Second, in a local cache, a data item which has not been accessed for a long period is replaced in the end. Third, in a TCG, a data item which "spawns" replica is first replaced in order to increase the effective cache size.

PCache [Miranda,2005] is an algorithm for efficiently replicating and retrieving data items in MANET. PCache uses an epidemic approach to perform the data dissemination. In order to achieve an even distribution of replicated data, the data dissemination is done based on network topology and the data replicated on each mode. As a result PCache is able to satisfy requests for data items using a small number of messages. In favorable conditions like number of nodes in the MANET, the size of their caches and the total number of items present in the network, PCache is capable of placing a copy of most data items within the one hop neighborhood of any node.

Global Cluster Cooperative Caching (GCC) proposed by Chauhan [2010], partitions the whole network into equal size $(r/\sqrt{8})$ clusters based on the geographical network proximity. The mobile nodes in the network interact with each other and form a cluster. In each cluster area a "super" node is selected to act as Cache State Node (CSN), which is responsible for maintaining the Global Cache State (GCS) information of different

clusters in the network domain. GCS for a network is the list of data items along with their TTL stored in its cache. When a node caches or replaces a data item, its GCS is updated at the CSN. In GCC, when there is a local cache miss, the mobile client will look up the required data item from the cluster members. If the client cannot find the data item in the cluster member's caches, it will request the CSN which keeps the global cache state and maintains the information about the MN in the network which has copy of desired data item. If a cluster other than requesting MN's cluster has the requested data (remote cache hit), then the request is served from there. Otherwise, the request will be satisfied by the data server. The GCC caching scheme reduces the message overheads and enhances the data accessibility. Once a MN receives the requested data, it triggers the admission control based on the server's location to determine whether data need to be cached. If the server of the data resides in the same cluster of the requesting MN, then the data is not cached. The same data items are cached in different clusters without replication to reduce the delay, bandwidth and energy. The cooperative caching scheme uses simple weak consistency than strong consistency model based on TTL. The MN removes the cached data when the TTL expires.

Ke [2010] proposed a cooperative caching scheme based on grouping mobile nodes. The network topology is partitioned into groups based on the physical network proximity and each group is controlled by a master node. The distance between the master node and the member node is one hop. The nodes select data items to be cached according to the data access frequency, the distance from the data source node, the data validation time and the node topology updating time. Based on these

parameters a CP value is created with the formula:

$$\mathrm{CP}_j = F_j D_j \mathrm{Data}\, v_j \mathrm{Top}\, v_i$$

where, CP_j represents the possibility of caching a data item j for node i. F_j represents the times which the data item j are processed on node i from the queries of local group and other groups. D_j is the distance from the node i to the data source node where the data item j locates. Data v_j is the validation time of data item j. Top v_i is the topology validation time of node i. After collecting the information, the master nodes compute the group CP value of each data item and sort the information record of data items with descending order according to the CP value. For cache replacement, if all the caches on the member nodes in a group are full, the data item which has the least CP value would be deleted. Table 3.1 summarizes different cooperative caching techniques proposed for mobile ad hoc networks.

3.4 Cache Replacement Policies

Cache replacement algorithm plays a central role in response time reduction by selecting suitable subset of data for caching. When the cache is full, an object has to be removed from the cache to make room for the data that has to be brought in. While it would be possible to pick a random object to replace when cache is full, system performance will be better if we choose an object that is not frequently used. If frequently used data item is removed it will probably have to be brought back quickly, resulting in extra overhead. A lot of research has been done in the area of cache replacement.

Table 3.1: Summary of cooperative caching techniques

Caching Scheme	**Centralized / Decentralized**	**Cache admission control**	**Data discovery**	**Cache replacement**	**Cache consistency**	**Performance metric used**	**Advantages**	**Disadvantages**
Cache Data /Path Hybrid cache	Decentralized	Based on no. of hops	By message exchange	Based on distance and access frequency	TTL based	Average delay	Increased data accessibility	Cache path become invalid when topology changes, space needed is more.
Aggregate Cache	Decentralized	Based on no. of hops	Broadcast based simple search	Based on distance and access frequency	Not present	Throughput Avg. no of hops	Distinct is stored increases throughput	Multiple copies of data returned to server extra communication cost
COOP	Decentralized	Based on no of hops	Adaptive flooding	LRU	TTL based	Request success ratio, Avg response delay	Limited flooding	Overhead in maintaining cache resolution
Zone Cooperative	Decentralized	Based on distance no of hops	Broadcast request to neighbors	Value based	TTL based	Avg no of hops, Avg. query latency	Simple to implement, Value based replacement	Latency is more if a neighbor does not have a cached copy
Neighbor caching	Decentralized	Not present	Neighbor nodes address stored in local node	LRU	Not present	Avg. ending time, traffic	Simple and Costless, utilizes the cache space of inactive neighbors	Data thrown out without reusing lacks efficient cooperation among nodes
Group Caching	Centralized	Based on group member status	Search in local table and group table	LRU	TTL based	Cache hit ratio, Avg. latency	Power efficient protocol, redundancy is reduced	Extra cost to maintain local and group tables

Table 3.1: Summary of cooperative caching techniques (Contd.)

Caching Scheme	Centralized / Decentralized	Cache admission control	Data discovery	Cache replacement	Cache consistency	Performance metric used	Advantages	Disadvantages
COCA	Centralized	Cluster based	Hierarchical	LRU Min	TTL based	Data accessibility ratio, Avg. query delay	Cross layer design with pre-fetching	Difficult to maintain when the number of nodes increases
Adaptive Cache	Decentralized	Data is replicated	Shared common index	Based on TTL	TTL based	Mean Query response time, Hop Count, Energy saving	Reactive caching, resource efficient	Duplication of data is high if the replicas are closer
COCAS	Decentralized	Not present	Distributed index system	Not present	Not present	Avg. Delay, Network traffic, Bandwidth consumption	Special nodes for caching queries which act as index to data	Separation of nodes into two sets
CLIR [9]	Decentralized	Not present	Request interception and cross layer interception	LRU	TTL	Document delay, % of timeouts, cache hit	Cross Layer implementation	Too many parameters to manage the redirection cache
Hamlet	Decentralized	Base on probabilistic estimate of what is cached in neighbours	Mitigated flooding	Not present	TTL	solved queries ratio, average cache occupancy, query traffic	Low overhead and reduced traffic	Overhearing of information sent from one node to another
ECOR	Decentralized	Not present	Cache hint table	LRU	TTL	Hit ratio, power per query, access delay, throughput	Best size for the cooperate zone is defined	Exchange of additional control message

Table 3.1: Summary of cooperative caching techniques (Contd.)

Caching Scheme	Centralized / Decentralized	Cache admission control	Data discovery	Cache replacement	Cache consistency	Performance metric used	Advantages	Disadvantages
GroCoca	Centralized	Data from peers are not cached	Based on message exchange between TCG members	Least valuable data item based on time stamp	TTL	Access latency, serves request ratio, global cache hit ratio, power consumption	Formation of groups with similar mobility pattern and similar data affinity	Centralized solution, not scalable, higher power consumption
DGCoca	Decentralized	Data from same TCG are not cached	Cash signature exchanged	Key based replacement	TTL	Access latency power consumption	Groups of MHs sharing common mobility and access pattern are dynamically formed	Overhead of maintaining and identifying cached items
PCache	Decentralized	Probabilistic approach	Broadcasting	Flag based. Popularity of data is taken	TTL	No. of messages, Avg. distance of replies, Avg. nodes without an item in one hop	Even distribution of data in one hop	Assumed large cache size and low mobility. Intermediate nodes have to process messages
GCC	Centralized	Based on location	Through cluster head	When the TTL expires	TTL	Avg. query latency, No. of messages	Exploits clustering for caching	Data about all the clusters are stored in one node
Grouping nodes	Centralized	Based on a Value Function	Through the master node	Data items with minimum function Value	Validation Period	Cache hit Rate, Energy consumption, Query Processing time	Low message overhead	Central point of failure

Caching in wireless environment has unique constraints like scarce bandwidth, limited power supply, high mobility and limited cache space. Due to the space limitation, the mobile nodes can store only a subset of the frequently accessed data. The availability of the data in local cache can significantly improve the performance since it overcomes the constraints in wireless environment. A good replacement mechanism is needed to distinguish between the items to be kept in cache and that is to be removed when the cache is full. The extensive research on caching for wired networks can be adapted for the wireless environment with modifications to account for mobile nodes limitations and the dynamics of the wireless channel. These limitations include the mobile node's limited battery life and its small cache size.

The following section provides a general comparison of the cache replacement policies in wireless mobile networks based on the criteria used for evicting documents. The various replacement policies for wireless networks are reviewed with more focus on function based and location based policies. The different policies used in ad hoc networks are also reviewed. The topic of caching in ad hoc networks is rather new, and not much work has been done in this area. The replacement policies for MANETs can be classified in to two groups, local and coordinated. In coordinated replacement policy the mobile nodes which forms cooperative cache collectively takes the replacement decision. In the later case the data item to be evicted is determined independently by each node based on its local access information. Alternative techniques for cache replacement are also proposed.

3.5 Cache Replacement Policies in Wireless Networks

Efficient replacement schemes for wireless mobile environments should consider different parameters like data access pattern, access costs, mobility pattern, connectivity, bandwidth, update rates and location dependence of the data. Most of the replacement algorithms form a value function by combining these parameters and evicts the data with minimum value. This section discusses some of the function based replacement policies in wireless environment.

Yin [2003] proposed a generalized cache replacement policy for mobile environment. The value function they proposed can be used for different performance metrics and they considered minimum query delay and minimum download traffic as the target. The value function was based on parameters like probability of reference, cost of fetching data item, cost of validation, probability of invalidating cached data item and cost of getting updated data item to the cache. Based on these parameters the algorithm replaces a data item with min Value $(i)/S_i$, where S_i is the size of the data item. Here a strong consistency model is assumed. Xu [2000] proposed a gain based replacement policy SAIU, for on demand broadcasts. The gain function for each data item is calculated as gain (i) $= L_i.\ A_i/S_i.\ U_i$, where L_i is data retrieval delay, A_i is the access rate, S_i is the size of the data item and U_i is the update frequency.

Another algorithm proposed by Zeitunlian [2010] uses a least unified value cache replacement for SACCS, scalable asynchronous cache constituency scheme. Here the replacement is based on the reference information of the object, fetch cost and size. They considered the complete

Table 3.2: Summary of function based cache replacement policies

Algorithm	Parameters Considered	Eviction	Performance measure	Advantage	Disadvantage
Target Based	Reference Probability, cost of fetching data validation cost, probability of invalidating cached data item, cost of getting updated data	Value is calculated using the parameters considered and replaces data with min value by size	Average delay, Average downlink traffic	Can be used for multiple targets. Considered data updations	Too many parameters to consider. How to select the target is not specified
SAIU	Data retrieval delay, access probability, size, update frequency	Low access rate, low delay and maximum sized data	Cache hit ratio strech	Uses a new performance metric	Parameters considered are not easily available
LUV-SACCS	Access frequency, recency, fetch cost, size	Smaller size, low access frequency, low cost	Cache hit ratio, total delay	Relates cache replacement with consistency	Book keeping is high, usage of a fixed parameter
On Bound Selection	Access frequency, update frequency	Low access frequency, high update frequency	Cache hit ratio, communication cost	Stale documents are evicted increase hit ratio	Not useful for short term access

reference history for finding the probability of reference in the future. The book keeping involved in this method is too high. Chen [2007] presented a cache replacement policy called On Bound selection which used both data access and update information for replacement decision. The above mentioned schemes use a function based policy. The drawback of this approach is that the relative importance of these parameters may vary with different application some policies are needed to adjust the weights dynamically to achieve the best performance. Table 3.2 gives the summary of function based replacement policies.

3.6 Location Based Cache Replacement Policies

In Location Dependent Information Services (LDIS) the value of the data item depends on the location and varies as the user changes his location. The factors that are considered in a location aware replacement policy are the valid scope area, distance and direction of client movement. The area under which the data item is valid is the valid scope area. Distance is the distance between mobile node's current location and the valid scope area. When the data is distant from the valid scope area, it will have a lower chance to become useful. Direction indicates the direction of data movement from the valid scope area. The data that are moving in the opposite direction of the valid scope area will be irrelevant after sometime.

The cache replacement policy that supports location dependent services was early proposed by Dar [1996] (Manhattan). Here the replacement was based on the Manhattan distance, which is the distance between the location of each cached data item's origin location and a mobile client's current location. The data items having the highest Manhattan distance are replaced. The only parameter considered for replacement is the distance.

The FAR (Farther Away Replacement) [Ren,2000] policy considers the current location and direction of the mobile client to make the replacement decision. The replacement strategy is based on the fact that the data which are not in the moving direction and farthest away from the user won't be visited in the near future. Based on the direction of movement, the data is arranged as two sets, In-Direction and Out-Direction. Whenever we want to replace data the Out- Direction set is considered first, when it is empty the furthest segment in the In- Direc-

tion set will be replaced. FAR considers only the spatial properties for cache replacement and the temporal properties are not taken.

Zheng [2002] has proposed two cache replacement policies PA and PAID. In this replacement policy a cost function is formed by considering the parameters access probability, valid scope area and data distance. Valid scope area refers to the geometric area of the valid scope of a data value. When this area is broad there is a higher chance that the client will request the data. In PA the cost function is formed as the product of access probability and valid scope. In PAID in addition to the above mentioned parameters data distance is also considered. The data with low access probability, a small valid scope area, and a long distance is evicted first.

Lai [2004] designed and implemented Mobility Aware Replacement Scheme (MARS) which uses a cost function which consists of a client location, movement of direction and access probability. The data item with lowest value for cost function is removed first. They also proposed an extension to this, the MARS+. The MARS+ tries to keep the clients movement patterns and from this history the future location of the client can be predicted. This is incorporated in to the replacement cost function and more accurate replacement decisions are made.

A network distance based cache replacement policy (ND-CRP) introduced by Jane [2008] considers the network distance which is the shortest path from current location of the mobile client (P) to a point of interest Pi for data eviction. Access probability and network density are the other factors considered in the replacement policy. This algorithm assumes that when the network density is high there is more chance to remain in that area for a long time. Dijkstra's algorithm is used to find

the shortest path from the single source to single destination. The policy would choose the data with less access probability, less network density and greater network distance for eviction.

Table 3.3: Summary of Location based Cache Replacement Policies

Algorithm	Parameters considered	Eviction	Performance measure	Advantage	Disadvantage
Manhattan	Manhattan distance	Lowest distance	Response time, network traffic	Supports location dependent queries	Single parameter. Difficult to find estimated weights
FAR	Distance and movement direction of clients	Data in the out direction set is evicted first then the farthest in the indirection	Average response time	Considers the direction of client motion and future movements	Not taken temporal properties. Ineffective when client changes its direction frequently
PA	Access probability and valid scope area	Low access probability, minimum valid scope area	Cache hit ratio	Considers temporal property	Objects close to the client are often replaced as their valid scope area is smaller
PAID	Distance between the current location and valid scope, Access Probability, valid scope area	Low access probability, minimum valid scope area, maximum distance	Cache hit ratio	Considers temporal and spatial property	Considers only the clients current movement direction
MARS	Client location, movement direction, access probability, update and query rate	Low temporal score and spatial score	Cache hit ratio	Temporal and spatial properties are taken along with update frequency	Fails to recognize regular client movement patterns

Prioritized Predicted Region based Replacement Policy (PPRRP) [Kumar,2010] tried to get the benefit of both temporal and spatial property in one unified scheme. In their scheme the distance is calculated based on a predicted region, where the client can be in the near future. In this policy instead of taking the direction of client's movement they

predict an area in which the client will be in the near future. The data item cost is calculated based on the access probability, valid scope area, data size in cache and distance of data based on the predicted region. Table 3.3 summarizes the various location based replacement policies.

3.7 Cache replacement policies in Ad hoc networks

The available cache replacement mechanisms for ad hoc network can be categorized in to local and coordinated replacement mechanisms depending on how replacement decision is made.In local replacement the data item to be evicted is determined independently by each node based on its local access information. In coordinated replacement policy the mobile nodes which forms cooperative cache collectively takes the replacement decision. Another feature of coordinated replacement is that the evicted data may be stored in neighboring nodes which have free space.

In the following section we discuss various uncoordinated cache replacement policies for mobile ad hoc networks.

LRU. LRU (Least Recently Used) is based on the observation that data that have been heavily used recently will probably be heavily used again in the future. Conversely, data that have not been used for ages will probably remain unused for a long time. In LRU when cache is full the data item that has been unused for the longest time has been thrown out. It is a widely used algorithm in cache replacement. Logically, the cache consists of a list with most recently referenced data being in the front of the list. When a data item is referenced it is moved from its existing position to the front of the list. When a new data comes in it is placed

on the top of the list and the data at the back end is removed. LRU doesn't take in to account the non uniformity in the size of data, which is an important factor in mobile communication as the cost to fetch the data depends on size.

LRU Min. LRU Min [Denko,2008] is a variant of LRU that tries to minimize the number of documents replaced. It is similar to LRU in implementation but will consider size of the data during replacement. In this scheme the data is arranged on the basis of access time and if a data item of size S needs to be cached it will search for items least recently accessed with size greater than S. If there isn't any data in cache with size S, we start removing the items with size greater than S/2 and then objects of size S/4 until enough cache space is created. LRU Min policy will increase the hit ratio of smaller sized data items.

SXO. This is a local replacement policy [Yin,2006] which considers the parameters data size and access frequency for replacement. Here larger sized data items are removed first as they occupy more cache space. More cache space can be made available by replacing bigger objects. The second parameter considered is order (d_i) which gives the frequency of access of data. Here replacement is done by combining the two parameters as value (d_i) = S^* order (d_i). The advantage of this scheme is that the parameters used are easily available. But recently accessed data are not given any privilege.

LUV. A cache replacement policy based on Least Utility Value (LUV) has been used by Chand [2006]. For computing the LUV of a data item the access probability (A_i), size of the data item (S_i), coherency which can be known by TTLi field and distance (δ) between the mobile client and data source were considered. Eq. for utility$_i$ function for a data item

(d_i) is:

$$\text{utility } i = A_i \,.\, TT \; L_i \; \delta_i / S_i$$

3.7.1 Coordinated cache replacement policies

TDS. The cache replacement [Lim,2003] is based on two parameters distance (D) which is measured as the number of hops and access frequency. As the network is mobile the value of distance (d) may become obsolete. So the value is chosen based on the time at which it is last updated. The T value is obtained by the formula $1/t_{\text{cur}} - t_{\text{update}}$. Distance is updated by looking at the value of T. Based on how the distance and time is selected three different schemes are proposed TDS_D, TDS_T and TDS_N. TDS_D considers distance as the replacement criteria. If two data items have the same distance least value of $(D + T)$ is replaced. In TDS_T the replacement decision is made by selecting the data with lowest T value. In the third scheme product of distance and access frequency is considered. In these algorithms TDS_D has the lower success rate and TDS_T has the higher hit ratio.

LUV M_i**.** This replacement scheme [Chand,2007] has two parts replacement and migration. The replacement decision is based on a utility value formed by combining the parameters access probability, distance, size and coherency. In the migration part the replaced data is stored in the neighboring nodes which have sufficient space. For migration the data with highest utility value is given preference. Here, even though the replacement decision is made locally migration is a coordinated operation. In order to save the cache space the data item is cached based on the location of the data source. If it is from the same cluster the data is not cached. The limitation of this scheme is that no checking is done

whether the data is already present in the migrating node.

ECORP. Energy Efficient Cooperative Cache Replacement Problem (ECORP) [Li,2007] is an energy efficient cache replacement policy used in ad hoc networks. They considered the energy cost for each data access. For this, they considered the energy for in zone communication, energy for sending the object, energy for receiving and energy cost for forwarding the object. Based on this they proposed a dynamic ECORP DP and ECOPR_greedy algorithms to replace data. The neighboring nodes will not cache the same data item in its local cache which reduces the redundancy and increases hit ratio.

Count Vector. In this scheme [Zheng,2002], each data item maintains a count which gives the number of nodes having the same data. Whenever the cache is full data item with maximum count is removed first as this will be available in the neighboring nodes. Whenever a data item is removed from the cache the access count will be decremented by one. Initially when the data is brought in to cache the count is set to zero.

3.8 Conclusion

As ad hoc networks are becoming popular, high data availability is a major issue. Great efforts have been made to improve the data availability in ad hoc networks. Cooperative caching is recognized as one of the effective techniques to improve data availability in ad hoc networks. This chapter presented an overview of recent cooperative caching schemes. By surveying earlier works on cooperative caching, it is noticed that there are still some open problems in developing effective, robust, scalable and adaptive cooperative caching techniques. A general comparison of the

major replacement policies in wireless networks is also made. Numerous replacement policies are proposed for wireless networks, but a few for cooperative caching in ad hoc networks. The strengths and drawbacks of these algorithms were also summarized.

Chapter 4

E-LRU: A Cache Replacement Algorithm for Cooperative Caching

Contents

4.1 Introduction

Cache replacement policy is a major design parameter of any caching policy. This is particularly true for MANETs since they are characterized by memory and resource constraints. The efficiency of the replacement policy affects both the hit rate and access latency of the system [Zahran, 2007]. The higher the memory constraints of the cache, the more vital the replacement policy becomes. As seen in the previous chapter, a lot of research work is geared towards finding the best cache replacement policy. But almost all the replacement policies proposed for ad hoc networks deal with LRU replacement policy. This chapter addresses the problem of cache replacement in cooperative caching for ad hoc networks and proposes a new algorithm for cache replacement.

Another issue in caching is consistency. The data supplied from the cache may not be the same when compared to the original copies on the data server, where the data is originated. In order to avoid giving stale data, caches have to update their contents within them so that they are not stale. Therefore attention is paid to the problem of maintaining cached data up to date with the server.

4.2 Formulation of Cache Replacement Problem

The objective of cache replacement is to maximize hit rate and minimize access delay of all nodes in ad hoc networks by appropriately replacing cached copies of data. A poor replacement policy can increase the number

of misses in the cache and cause a lot of traffic out of the cache and increase the miss penalty [Zahran, 2007].

Cache replacement becomes necessary when a cache miss occurs and no free slots exist in the local cache. In principle, the cache memory handler could select a data item randomly for replacement. However, another cache miss would result when the replaced data item is referenced again, so it is important to replace data items that are not likely to be referenced in the immediate future. Replacement strategies are concerned with deciding which data item or document to be displaced to make room for an incoming data when the cache storage is already full. Due to the space limitation, the mobile nodes can store only a subset of the frequently accessed data. A good replacement mechanism is needed to distinguish between the items to be kept in cache and that is to be removed when the cache is full.

Almost all of the replacement algorithms proposed for cooperative caching are LRU based. LRU is the simplest replacement policy suggested from early days for data caching. The disadvantage of LRU is, it considers only too little information for replacement. Another category for cache replacement is function based policies which involve the calculation of a value function based on different parameters, which use complex data structures that make the implementation difficult. The replacement policy introduced in this chapter, Extended-LRU (E-LRU), aims at replacing the data objects based on the time difference between the recent references. The novelty of this approach lies in the key based replacement where we first consider the time interval between the recent references for each data item and the data with longest reference interval time is dropped first. The advantage of this scheme is that the pages with

shortest access time interval, which have more probability of reference in the future, will be kept in the cache.

The basic idea of E-LRU is to keep track of the last two references to the cached data item and using this information the inter arrival time between requests are calculated for each data item. Since LRU drops the data from cache that has not been accessed for the longest time, it is unable to differentiate between data that have relatively frequent references. The basic frequency based strategy for replacement is Least Frequently Used (LFU) strategy. LFU removes the least frequently requested data. LFU based algorithms take more history information in to consideration but have no means to discriminate between recent and past reference frequency of a data item and is unable to cope with the evolving access patterns. E-LRU effectively addresses the limits of these algorithms by making use of inter reference time for making replacement decision. The major objective of the proposed approach is to effectively address the limits of LRU by retaining the low overhead merit of LRU.

4.3 Temporal Locality

Generally data availability is enhanced by different mechanisms like caching, replication and prefetching protocols. All these protocols exploit the property of locality of reference for its implementation. These protocols can be generally classified as server based, client based and network based systems [Jin, 2001]. The important characteristic that classifies these systems are web reference patterns. In each of these categories the request streams generated are different. In client based systems, the request streams are given to a limited and homogeneous community of users. On the other hand the request stream is limited to the objects offered by the

server in a server based systems. In network based systems the existence of these protocols are transparent to both server and client and therefore the above mentioned limitations does not exist. In each of the above mentioned category the underlying determinant of temporal locality is different and an effective cache replacement policy should characterize the degree of locality present in typical web request streams.

Ad hoc networks can be considered as client based systems and previous studies have concluded that temporal locality is weakening for client based systems [Barford, 1999]. This is because the data requests generated for client based systems are generally the set of requests that missed in the client cache. Such a request stream is likely to exhibit weak temporal locality of reference in particular, a recently accessed object is unlikely to be accessed again in the future [Barford, 1999].

Using an independent reference model [Coffman, 1973, Breslau, 1999] showed that the Zipf-like popularity distribution of objects in web request streams can asymptotically explain other properties (namely, cache efficiency and temporal locality). In particular, they showed that the probability of referencing object t units of time after it has been last referenced is roughly proportional to $1/t$. Thus, the probability distribution of reference inter-arrival times (or inter-request time) could be used to model temporal locality.

4.4 Cache Consistency

As discussed in chapter 2, there are mainly two techniques for ensuring cache consistency: strong cache consistency and weak cache consistency. The main difference between strong cache consistency and weak cache consistency is that in strong cache consistency the cache ensures

the freshness of content by cross checking with the original server every time the content is asked for. On the other hand, in weak consistency model the cache may employ heuristics for ensuring freshness and need not contact the server every time it serves the content. Hence, strong consistency could turn out to be more expensive for ad hoc networks than weak consistency.

The adaptive TTL consistency model which has its origin in Alex protocols [Cate, 1992] represents a weak consistency model. In this model, the TTL of a document is regulated by monitoring its life time. It uses a statistical property of file life time distribution. This distribution tends to be bimodal if a file has not been altered for a comparatively long time. The TTL dimension of an object is computed as a fraction of the document's current age. The current age of a document is calculated by subtracting the earlier time of modification of the document from the prevailing time. Earlier studies have shown that the probability of stale documents occurring in adaptive TTL is about 5% [Cate, 1992].

Previous studies show that TTL-based caching consistency strategy is suitable for ad hoc networks [Fan, 2011]. In a TTL-based strategy, location of the cache nodes may not be tracked by the data server. Therefore it is not necessary for the data source to keep track of the locations of cache nodes. This makes it adaptable to the dynamic nature of ad hoc networks, where the mobile nodes join or leave the set of cache nodes at any time. The validity of the cached data item is determined autonomously and the validation of stale data can be performed either by data server or by the neighboring nodes having valid copies. The TTL based consistency also ensures flexibility. Different TTL values can be used for different consistency requirements that may vary with the nature

of shared contents and user tolerance.

4.5 The Proposed Scheme

The replacement algorithm illustrated here considers the latest reference times for a data item and gives more importance to data items that are referenced more than once. The requested data items are divided into two categories: data items requested only once and data items referenced more than once. If we have data items referenced only once then that set is given priority for replacement. For this LRU policy is used. If an item is referenced more than once, the inter request arrival time (IRT) between the recent two references is considered for eviction. For each data item, IRT is taken as the recorded history information. IRT refers to the time interval between the last and penultimate (second to last) reference of a data item. It is assumed that if the IRT of a data item is large, the data is not likely to be referred in the near future.

Let t_c be the current reference time and t_r be the penultimate reference time then IRT, the inter request arrival time is given by,

$$\text{IRT} = t_c - t_r.$$

For items with one reference t_r is taken as ∞.

If $t_c - t_r = \infty$, an item with least t_c is replaced.

If IRT $\neq \infty$, then the replacement decision is made on the value of $K(i)$ which is given as,

$$K(i) = \text{Max} \sum_{i=1}^{n} tc - tr.$$

The data item with maximum IRT is considered for replacement.

4.6 Cache Consistency and Cache Invalidation

In cooperative caching, nodes share the cache contents of neighboring nodes to utilize the full advantage of caching. In order to improve the content diversity in the cooperative cache, the cache admission control scheme does not cache any data coming from the neighboring nodes. This increases the availability of information for the user, as more data items are cached and also avoids additional request to the server. Weak consistency model is used to maintain cache consistency. Each data item is associated with a TTL field which contains the allowed caching time and a time stamp. Data is considered as valid if the sum of TTL and time stamp is greater than the current time.

4.7 Simulation Scenarios and Metrics

In this work, to evaluate the performance of E-LRU the following network model is used. The nodes are randomly placed in the simulation area. Each node is identified by a node id and a host name. The data server is implemented as a fixed node in the simulation area. The data server contains all the data items requested by the mobile nodes. The nodes in the network move randomly based on a random path. The nodes within a transmission range of 100m are taken as the neighboring nodes. The nodes that generate data request are selected randomly and uniformly. Each mobile node generates a single stream of read only queries. After a query is sent out, the client does not generate new query until the pending query is served. The data access pattern follows a Zipf distribution

[Breslau, 1999] with a skewness parameter θ as 0.8. The simulation is implemented in JAVA.

4.7.1 Performance Metrics

Two metrics are taken for performance evaluation: cache hit ratio and average number of requests served. The evaluation of these parameters is done by varying the cache size of the mobile nodes. The hit ratio is defined as the percentage of requests that can be served from previously cached data. Since the replacement algorithm decides whether to cache the data or not, it affects the cache hits of future requests. The number of requests served for a particular period of time is taken. Average number of request is the average number of requests served for a particular period of time.

4.8 Performance Evaluation

The performance of the proposed E-LRU is compared with LRU for different cache sizes. Different cache sizes were used, ranging from 10% to 60% of the total size of the database. Fig 4.1 and Fig 4.2 show the result with a set of 6 cache sizes which are 10%, 20%, 30%, 40%, 50% and 60% of the total size of the database, respectively. Fig 4.1 depicts the comparison of cache hit ratio for different cache sizes. This figure shows that cache hit ratio of E-LRU is more for all cache sizes than LRU. At small cache sizes E-LRU shows significant improvement in cache hit ratio compared to LRU. For large cache sizes the difference in hit ratio of the two policies became less significant. Fig. 4.2 is the comparison of average number of requests served for different cache sizes and it shows that the

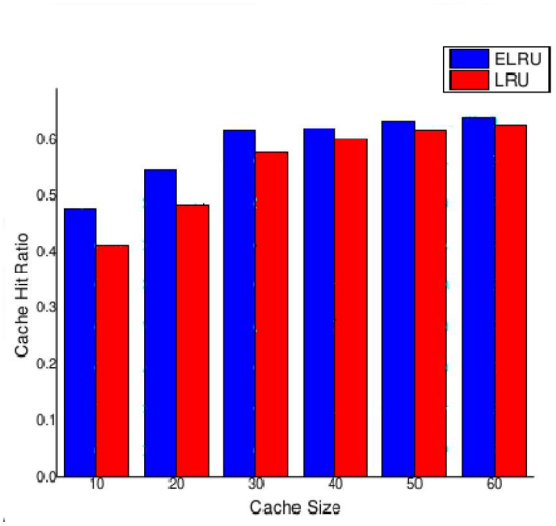

Figure 4.1: Cache Hit Ratio

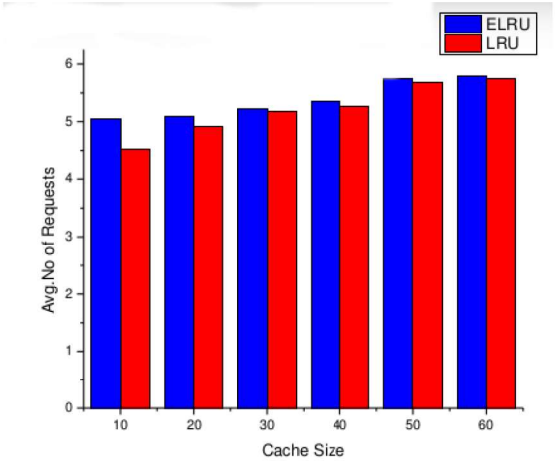

Figure 4.2: Average number of requests

average number of request is higher for E-LRU than LRU. This is due to the fact that E-LRU uses the cache space more effectively and the number of data requests send to the server is reduced.

4.9 Conclusion

This chapter explored the cache replacement issues for ad hoc networks and presented a new cache replacement policy for cooperative caching. The proposed algorithm takes in to account the inter arrival time of recent requests for data item for replacement. In LRU only the last time of reference is taken and the numbers of references are not considered. Since the inter arrival time of the recent request is taken more preference is given to the data items that have been accessed more than once. Hence we are able to distinguish between data that are frequently referenced with that of occasionally referenced. Since the algorithm is based on a key based approach it is simple to implement. Experimental results show that the proposed replacement algorithm can significantly improve the cache hit ratio and lower the data access delay when compared to LRU.

Chapter 5

A Coordinated Cache Placement Scheme for Cooperative Caching

Contents

5.1 Introduction

In mobile ad hoc networks, network connectivity, energy constraints, low bandwidth and frequent disconnections pose new problems for efficient data access. The idea of sharing caches among neighboring nodes is an important technique to improve data access efficiency in mobile ad hoc networks. Cache buffer management mainly consists of two components, cache replacement and cache placement. The role of cache replacement was described in the previous chapter. Cache placement component determines where to place the incoming data in order to minimize the average access cost. The effectiveness of a given amount of cache space can be improved by efficient data placement schemes. This chapter focuses on cache coordination issue and discusses the need for coordinated storage decision. Coordinated storage helps for two reasons [Korupolu, 2002]. First, it allows a busy cache to utilize a nearby idle cache. Second, coordination improves the hit rate by having more unique data in the shared nodes. A new cache data placement scheme combined with cache replacement ideally suited for mobile host with limited memory is introduced in this chapter.

Although several cooperative caching protocols have been proposed in literature, only a few studies have examined the coordination of caches from data placement point of view. Many of the studies focus on number of hops for making data placement decision. The remaining works on cache placement mainly focus on selecting cache nodes based on the information about data access frequency and network topology. The main drawback of this approach is that it does not consider the cooperation of caches for data placement. As a result, data may be cached in several caches in the cache group which may lead to inefficient utilization

of available cache space. This may cause significant reduction in hit rate especially for ad hoc networks since the nodes have lower cache storage than other type of networks. This chapter introduces a coordinated data placement scheme that eliminates redundant data stored in multiple neighboring nodes by sharing and coordination of cache state among neighbouring nodes for efficient data storage and retrieval using the concept of cached object's eviction period. The cache data placement scheme based on eviction period reduces replication of data by assuring that a copy of the data will be present in the cooperate cache, without any extra communication overhead. The simulation results show that the proposed scheme yields better performance in terms of cache hit ratio and average latency compared to independent cache data placement.

5.2 Need for Coordination

In traditional caching scheme, each data request is satisfied by the cache associated with the requesting node. The storage decision made by one cache is independent of those made by other caches in the system. Cooperative caching on the other hand, cooperates in serving one another's request as well as in storage decisions. By sharing the cache contents, the clients get more chance to fetch data from geographically close nodes instead of remote data server. The placement scheme determines how to coordinate the contents of local cache. Without coordination, the requested object is fetched from the neighbouring node and the local node stores it. This scheme limits the advantage of cooperative caching due to indiscrete replication of objects. In the worst case all the neighbouring nodes can have the same contents and get nothing from cache contents share.

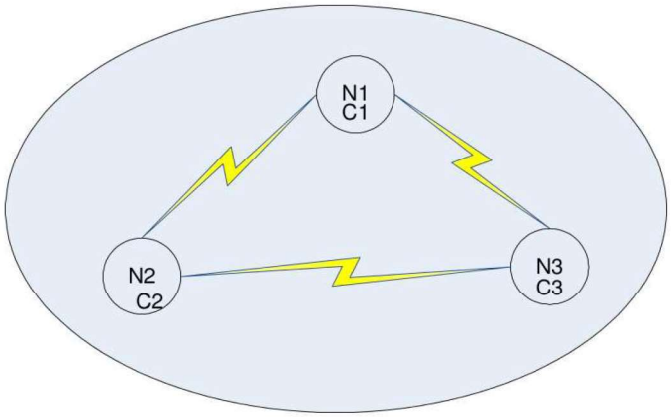

Figure 5.1: Example to show cache data placement

The following example illustrates the above mentioned issues. Consider the above network given in Fig. 5.1 with three nodes N_1, N_2 and N_3 with local cache C_1, C_2 and C_3, which coordinate with each other to share data. When node N_1 experiences a local miss, it sends cache requests to nodes N_2 and N_3. In case of non availability of data in either of the nodes, N_1 fetches the data from the server, stores a local copy and serves the client. If after some time, node N_2 needs the same data, the data request is satisfied from N_1 and N_2 stores it locally. A similar case occurs when N_3 needs the same data. This uncontrolled placement of data in the cache can result in several performance limitations. As the storage space is limited, each additional copy of data causes the eviction of other data items from the cooperative caches. Hence the number of unique documents available within the cache shrinks. Further the average amount of time the data items remain in the cache before they are evicted decreases. The cumulative effect of these two is the decrease in

the cumulative hit rate of the cooperative cache.

Therefore an independent object placement scheme is not optimal for cooperative caching. Another extreme is to prohibit replication of data items in the neighbouring nodes. This scheme can decrease cache miss ratio by making the number of distinct objects maximum in the group. But the access latency may increase due to a lot of remote fetching data from the neighbor nodes and not from the local cache. Between both extremes there were efforts to find out efficient placement scheme that achieves minimum access latency of objects.

The decision of whether to cache data in each node is made collectively among the neighboring nodes that cache the same data. The coordinated placement of data is based on the cache life time of individual nodes. We use the concept of cached object's eviction period to measure the contention of caches. The proposed scheme tries to reduce duplicate data in the neighboring nodes by storing the data items only in the cache with lowest eviction rate.

5.3 Related Works

Cache placement schemes assigns data to caches without violating the cache size constraints. As discussed in chapter 3 several cooperative caching protocols have been proposed for MANETs to improve cache hit ratio and to reduce data access delay. The following section presents an overview of different proposals made for cache placement strategies related to the work presented in this chapter. Compared to the amount of work done on cache replacement policies there exists only a few proposals for cache placement. Most of the cache data placement schemes devised so far performs cache placement independently.

Korupolu [2002] defined the cost of cache placement as the sum over all nodes 'u' and all objects ' ψ ' of access frequency for object 'ψ ' at node 'u' times the distance from node 'u' to the closest copy of that object. The objective of a cooperative placement algorithm is to compute placement with minimum cost. By optimally placing the cache objects the network load and server load is not explicitly minimized but these would be low when the access cost is minimized. This is because the objects will be stored closer to the clients, thereby reducing the load on both the network and the server. In their proposed work, a cost function for document placement was formulated to measure the cost incurred by a cache to obtain the document from its nearest available copy. Access frequency of each object present in the cache is taken as the main parameter to calculate the cost function. The document placement schemes aim at minimizing the sum of the cost functions of all documents over all caches in the group. The arrangements of caches in the cache group are based on clusters and these clusters themselves are arranged in a tree structure. They also proposed a greedy placement algorithm which involves a bottom-up pass of the cluster tree structure to determine the cache where the document has to be stored.

A benefit-based greedy cache placement strategy has been presented in [Tang, 2008], where the cache placement is done based on the total data access cost of all the nodes in the network.The benefit score for a data item D_j in node i is the product of access frequency of (t_{ij}) of data item D_j in node i and least distance (δ_j) to the neighboring node containing the data item.

$$\text{Benefit Score } B_{ij} = t_{ij} * \delta_j.$$

When a node caches a data item it broadcasts information about the availability of the corresponding data item to the broker. Similarly, when a data item is deleted it broadcasts the non-availability of the corresponding data item to the broker. The broker periodically broadcasts the meta data of the cache updates to the whole network. When the cache is full, data item with lowest benefit score is replaced. The primary objective of this approach is to cache data at available memory location of all nodes available in the network so that all the available memory are utilized over a period of time.

Du [2005] proposed two schemes for cache placement. In order to cache more data in the cooperative cache the cached data copies are categorized into two groups based on whether they are already present in the neighbouring nodes or not. If the incoming data item is not from any one of the nodes present in the cooperation zone it is taken as a primary copy. The data taken from the neighbouring nodes is taken as secondary. The priority of primary and secondary data is made by the inter- and inter-category rules. The intercategory rule put primary items at a higher priority level and the secondary items are purged to accommodate primary items, but not vice versa. In order to determine whether the data is primary or secondary, a labeling mechanism is used.

When a node receives a data item, it labels the item as primary if the item comes from a node beyond the zone radius and if the data item comes from within the zone radius, then check for the label, to determine whether the data item is primary or secondary. If the data item is already labeled as primary, the new copy would be secondary since there will not be duplicated primary copies in the same cooperation zone. On the other hand, if the data item is tagged as secondary, the provider needs to attach

the information of the primary copyholder. If the primary copy holder is beyond the zone radius, the new copy is primary copy; otherwise, the new copy is a secondary copy. The intra-category rule is used to evaluate the data items within the same category. The LRU algorithm is used to remove the data items present in the same category.

Fan [2013] presented a Gossip based Cooperative Caching (GosCC), which considers the sequential relation among the data items. In this scheme, each mobile node stores the IDs of the data items cached locally and the ID of the data item in use in to its progress report. Each mobile node makes use of this progress report to determine whether a data item should be cached locally. The progress reports are propagated within the network in a gossip based way. The progress report contains the progress of each node consuming data items and the contents in cache at each node. Cache digest describes its own cache content to other nodes such that other mobile nodes can send their data request to the closest cache node for data item. In this scheme , the mobile node investigate the progress digest report and send the cache request for the most popular data item in future. The disadvantage of this scheme is the message overhead due to the periodical transfer of cache digests and progress reports. Another disadvantage is that this work mainly focuses on applications with correlated data.

Yin [2006] proposed a hybrid cache scheme in which the incoming data is cached according to certain criteria. The selected criteria include data item size, TTL parameter and number of hops. Based on the value of these parameters, data itself or path to the nearest cache is stored in the cache. Hara [2006] proposed a cache placement scheme based on the access frequency for a particular data item. A mobile host selects data

items to cache based on the access frequency. Nuggehalli [2003] considered the trade-off between query delay and overall energy consumption for cache placement.

From the survey it is found that most of the work related to cache placement involves independent cache placement. Function based cache placement policies require some parameters to calculate the cost function. The disadvantage of this approach is that the relative importance of these parameters can vary from one type of request to another and these policies need to adjust the weights dynamically to achieve the best performance. Function based policies are also limited since they do not consider the redundancy of data. Another category for cache placement scheme is cache placement based on number of hops. The drawback of this approach is, it considers only the number of hops for cache object placement. But the data item present in the neighboring nodes may sometimes get evicted soon from the cache. In order to resolve these issues, a new cache data placement policy based on the eviction period of the data item stored in the cache is introduced.

5.4 The Proposed Scheme

This section presents an overview of cache data placement scheme proposed for cooperative caching in ad hoc networks. In this placement algorithm, each node makes informed and intelligent decisions on whether to cache a particular data. The decision to cache a particular data item is based on two factors: whether the data is already available in the neighboring nodes and if available how long it is likely to remain in the caches where they are currently stored. In order to find the likely time a data item will remain in the cache, the concept of Cache Eviction Period

(CEP) is used [Psounis, 2002].

The eviction period of individual cache can be calculated from the last access time of the data item and the time at which it is removed from the cache. The last access time of each data item can be taken from the replacement policy. The additional parameter required to calculate the eviction period is the time at which the data item is removed from the cache since its last access. The eviction period of a data item is defined as the time duration between the time it was evicted from the cache and the time it was last accessed. This can be denoted as $T_e - T_a$, where T_e is time at which the data item is evicted from the cache and T_a is the time of last access. This period implies how long a data item is placed in the cache after its last access. The average eviction period of a cache in a finite time period is given by the average of the eviction period of the data items that were removed from this cache in this period.

For a given finite time duration $T_i - T_j$, let $S(C, T_i, T_j)$ denotes the set of data items evicted from the cache.

$$S(C, T_i, T_j) = \{D | \forall\ T < T_i, P(D, C, T) \wedge \forall T' > T_j, R(D, C, T')\}$$

where $P(D, C, T)$ is the set of data items that remains in the cache C at time T and the set $R(D, C, T')$ denotes the data items already evicted from the cache C at time T'.

The cardinality of the set $S(C, T_i, T_j)$ indicates the total number of documents evicted from a cache C during the duration (T_i, T_j). The Cache Eviction Period (CEP) of the cache C for the duration (T_i, T_j) is

denoted by

$$\text{CEP}\,(C, T_i, T_j) = \frac{\sum_{D \in S(C,T_i,T_j)} \text{Eviction period}\,(D, C)}{|S(C, T_i, Y_j)|}$$

The CEP period denotes the average time a data item is expected to live in the cache after the last hit. If the CEP is higher, the data item is expected to stay longer in the cache.

The cache data placement scheme is combined with cache replacement policy E-LRU proposed in the previous chapter. In E-LRU, the time interval between last two references of each cached data item is stored. Using this information the IRT of data items is estimated. Whenever the cache space of the mobile host becomes full and a new data is to be added, the data item with maximum inter reference time is evicted from the cache. If there are data items that are referenced only once, Least Recently Used (LRU) policy is used to select the replacement victim. In LRU policy, the data item that has not been accessed or the longest time is dropped first. In both cases the time of last hit for each data item is maintained. In such a situation it is straight forward for any node to support the proposed cache placement policy.

5.5 Algorithm

The design rationale of the proposed cache placement algorithm is to optimize the cache usage, for applications with limited memory. The proposed algorithm tries to reduce duplicated data across neighboring nodes. The core idea of this approach is to use the eviction period of a data item as an indicator of the cache space contention at individual

nodes. In the following section we explain the proposed cache data placement algorithm and how each node makes decision on whether to cache the data obtained from another node in the neighbor group.

Whenever there is a local miss, the query node will send a cache control message to the neighboring nodes. The cache control message contains the requested data id and the CEP. If any one of the neighboring node has the requested data, it is sent back to the query node along with its CEP. Upon receiving the data, the query node checks its CEP with the CEP of the responder node. If the CEP of the query node is greater than the CEP of the responder node, it will store a copy of the data item in its local cache. If the CEP of the query node is less than the CEP of the responder, the query node will not store a copy of the data item. This is because the copy of the data item at the responder node is likely to remain in the cache for a longer period. Therefore, under the proposed scheme the query node will not store a copy of the requested data in its local cache, when the CEP is less than the CEP of the responder. At the responder node, if the CEP of the query node is higher, the last access time is not modified, so that the referenced data item will be evicted soon. When the query node receives a negative reply from the neighboring nodes, data is fetched from the server.

The proposed cache placement algorithm involves the following message types.
REQ (i, d_j): the data request message from node 'i' to the neighbouring nodes for data item d_j.
REPLY (j, i): reply message from node 'j' to node 'i' to inform that it has the data item d_j.
ACK (i, j, CEP_i): acknowledgement from node 'i' to node 'j' along with

CEP of node 'i'.
REPDAT $(j, i, d_j, \text{CEP}_j)$: reply message from node 'j' to node 'i' ,along with CEP of node 'j'.

Algorithm 1 Cache Placement

```
for every data request in local cache LC_i of node i do
   // search for data in the local cache
   if data d_i is present in LC_i then then
      Get d_i
   else
      // Search for data in the neighboring nodes
      Send msg REQ (i, d_j) to all neighbours
   end if
end for
if a positive REPLY (j, i) from N_j arrives then
   // Find the CEP
   Send msg ACK (i, j, CEP_i) to node N_j
   On receiving ACK (i, j, CEP_i) from N_j
   Send msg REPDAT (j, i, d_j, CEP_j)
   if (CEP_i >= CEP_j) then
      Cache d_i;
   else
      Serve the data request
   end if
else
   // Request is redirected to server
   Sendmsg REQ (i, d_j) to server
   Get d_i;
   Cache d_i;
   Serve the request;
end if
```

A node N_i that experiences a local miss sends out a REQ (i, d_j) to all its neighbouring nodes. If any one of the neighbouring node N_j has the requested data it sends a REPLY(j, i) message to node N_i. Node N_i on receiving REPLY (j, i) message from node N_j, send an ACK (i, j, CEP_i) back to node N_j. Upon receiving ACK (i, j, CEP_i) Node N_j sends the

Algorithm 2 Cache Placement at the responding node

```
// At the responding node N_j
Upon receiving a msg REQ (i, d_j)
begin
Check for the availability of data
if d_i is available then
   Send REPLY (j, i) to N_i
   On receiving msg ACK (i, j, CEP_i) from node N_i
   // Check for CEP
   if (CEP_i >= CEP_j) then
      Send msg REPDAT (j, i, d_j, CEP_j)
   else
      IRT d_i < −IRT_new
      Update LC_j
      Send msg REPDAT (j, i, d_j, CEP_j)
   end if
else
   Send msg REQ (i, d_j) to server
end if
```

REPDAT $(j, i, d_j, \text{CEP}_j)$ message back to node N_i. Node N_i compares its own CEP_i with CEP_j. If CEP_i is greater than CEP_j, N_i does not store the data locally, it just serves the user with the data. However, if CEP_j is greater than CEP_i, the node N_i stores a copy of the data locally. On the other hand, node N_j also compares its own CEP with the CEP of node N_i. If CEP_j is greater than CEP_i, IRT is changed and the data position is updated. Otherwise, IRT is not changed and the data position remains un-altered. By this, the placement scheme ensures that the data is cached only if the new copy has a reasonable chance to survive longer than the original copy, which obviously reduces redundant copies in the cooperative cache. The complete sequence of messages transmitted during cache placement is shown in Fig. 5.2.

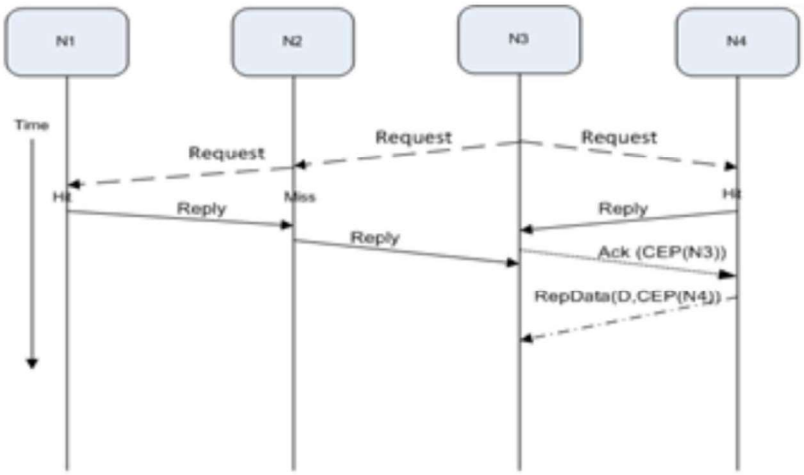

Figure 5.2: Message exchange for cache placement

5.6 Simulation and Performance Evaluation

The performance of the proposed coordinated cache data placement scheme using CEP is compared with independent cache data placement scheme based on number of hops. Following sections explain the simulation model and the performance metrics used for performance evaluation.

A mobile ad hoc network is abstracted as a graph $G(V, E)$, where V is the set of nodes and $E \subseteq V^2$ is the set of links which gives the available communication. An edge (u, v) belongs to E means that there is direct communication between two nodes u and v. The elements of E depend on the position and the communication range of nodes. All links in the graph are bidirectional i.e., if u is in the transmission range of v, v is also in the transmission range of u. The mobile ad hoc network environment consists of a number of mobile nodes and a fixed data server. Each mobile node

is identified by a node id and a host name. The transmission radius R determines the maximum communication range of each node and is equal for all nodes in the network. Two nodes in the network are neighbors if the Euclidean distance between their coordinates in the network is at most R. The Euclidean distance between the nodes are estimated based on the relative position of nodes. It is assumed that each node knows its current location precisely with the availability of Global Positioning System (GPS). For the simulation model it is assumed that the position of each node is given by the x and y coordinates. The data server contains all the data items requested by the mobile nodes. The database in the data server contains the data items, with each item identified using a data id. The nodes that generate data request are selected randomly and uniformly. Each node maintains a list which stores the cached data item. The list contains the following fields: cached data id, cached data item, TTL, time difference between the recent two data access and the time at which data item is evicted from the cache. This table is updated whenever an event occurs in the cache. The cache space for each node is limited and when it is full, the replacement strategy evicts the unwanted data. The contents of the local cache are shared by its neighboring nodes. When a node fails to find data in neighboring nodes, data is retrieved from the data server. When a node receives fresh data directly from the server, it caches a copy of it in the local cache and becomes a provider for that cached content for the neighboring nodes.

The mobile client generates read only queries and the time interval between two consecutive follows an exponential distribution. The queries generated follow a Zipf distribution which is frequently used to model non uniform distribution. The data request is processed in FCFS manner at

the server. An infinite queue is used to buffer the request when the data server is busy. Each miss in the cooperative cache will incur a delay of 4 ms to retrieve data from the data server. Initially, the mobile nodes are randomly distributed in the simulation area. After that each node randomly chooses its destination with a speed s which is uniformly distributed $U(V_{\min}, V_{\max})$ and travels with that constant speed s. When the node reaches destination, it pauses for 200 seconds. After that it moves to the new destination with speed s'. The simulator was implemented in JAVA.

5.6.1 Performance Metrics

The following performance metrics are used to evaluate the proposed cache data placement scheme: cache hit ratio and the average latency. The evaluations of these parameters are done by varying the cache size with fixed number of nodes. Cache hit ratio is defined as the percentage of requests that can be served from previously cached data. If a data item is requested by a mobile node at time T_R and the data is served at time T_C, latency for data item is defined as $(T_C - T_R)$. Average latency is the average of the latencies experienced for all the data items served.

5.7 Simulation Results

To evaluate the performance of the proposed cache data placement scheme, CEP it is compared with an independent hop based scheme (HB scheme) in which the data items from the neighboring nodes are not cached. This scheme is chosen for comparison because most of the existing cooperative caching techniques use this criterion for cache data

placement. The cache hit ratio and average latency for different cache sizes is measured. Fig. 5.3 shows the comparison of cache hit ratio for different cache sizes. From the figure we can see that the difference between cache hit ratio is higher when the cache size is small. This is due to the fact that when the size of the cache memory is small, effective memory utilization can make noticeable improvements in cache hit rates. Fig. 5.4 depicts the comparison average latency of proposed cache data placement for different cache sizes. Since the cache miss rate is relatively low at small cache size the average latency becomes insignificant at large cache sizes.

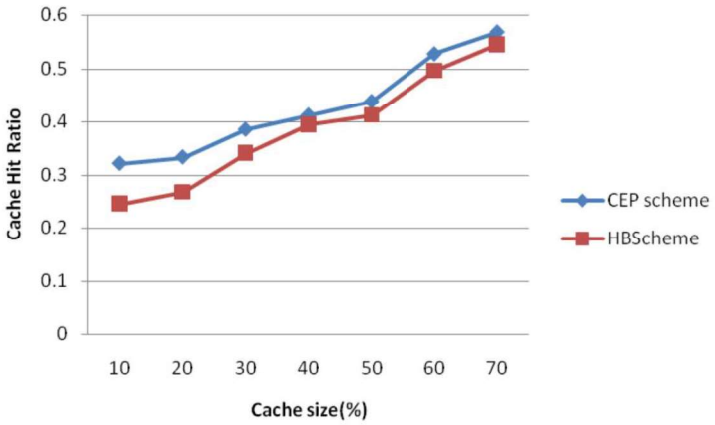

Figure 5.3: Cache Hit Ratio

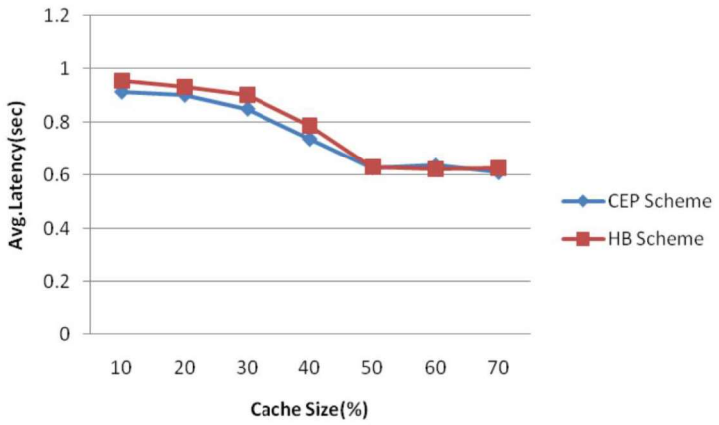

Figure 5.4: Average Query Delay

5.8 Conclusion

The performance of cooperative caching can be improved by proper cache management. This chapter discusses the role of coordinated cache placement for cooperative caching in ad hoc networks. A coordinated cache data placement algorithm based on CEP is presented. The CEP is taken as an indicator of the cache space contention at individual nodes. The decision on caching an incoming data is done coordinately among the neighboring nodes that already have a copy of the data item. The proposed scheme effectively reduces the number of redundant data. This scheme has been proposed for effective memory utilization for mobile clients with limited memory. Simulation results show that the proposed policy can significantly improve the performance compared to independent cache placement schemes especially for applications with limited cache.

Chapter 6

Cache Discovery in Cooperative Caching

Contents

6.1 Introduction

Cache discovery is an important service in cooperative caching. As discussed in the earlier chapters designing a cooperative caching protocol includes effective cache management and proper cache discovery. Previous chapters explained cache management schemes for cooperative caching. This chapter deals with cache discovery. Cache discovery refers to the mechanism for finding the nearest cache copy of data present in other nodes. In ad hoc networks cache discovery is more challenging because of the mobility and instability of the network.

Existing cooperative caching techniques pay less attention to the message overhead due to cooperative caching. These messages increase the load on the network and therefore reduce the efficiency of the protocol. The nodes in a mobile ad hoc network operate with limited battery energy. The number of messages processed by the mobile nodes must be controlled to increase the life time of the network. The simplest way for cache discovery is flooding. But flooding is inefficient in terms of the number of messages generated. This chapter presents a new cache discovery protocol, which reduces the caching overhead and delay by reducing the number of control messages flooded in to the network. A location aided cache discovery process based on location of neighboring nodes is developed for this.

6.2 Cache Discovery for mobile Ad hoc Networks

Generally, in cooperative caching there are two main approaches for cache discovery: passive discovery and active discovery [Wu, 2012]. In passive

discovery the data requests are always destined to source [Yin, 2006, Tyan, 2005]. In this scheme, to forward a request message to the data server, the cache layer wraps the original request message with a new destination address, which is the next hop, to reach the server. The cache layer accesses the routing table to find the next hop address. In this way, the packet is received and processed hop by hop by all nodes on the path from the requester to the data server. During this process the intermediate nodes checks if it has the requested data item. If data is present in any one of the intermediate nodes it is given to the source node without forwarding the request to the data server. In active discovery, the existence of the cached copy is known to the requesting node and a request is sent to the neighbouring nodes rather than to the data server. Active cache discovery can be further divided into two categories: proactive approach and reactive approach. In proactive approach, a cache node informs other nodes that it has a cache copy [Ting, 2007, Tang, 2008]. Whenever a cache event occurs, it is informed to all other neighbouring nodes. However, this generates a significant overhead not only to the network, but also on other nodes due to message processing. In addition, to locate the cache contents, a node has to maintain information about other node's cache contents. This can consume significant amount of resources with the increase in number of nodes. Obviously, proactive approach is not effective to discover cache copies in ad hoc networks with dynamic topologies because a large portion of bandwidth is used to keep the cache data information up-to-date. In the case of fast node mobility, the cache updates may be more frequent than requests, thus wasting bandwidth as much of the caching information will never be used.

In reactive approach, a node needs to locate the cache node by query

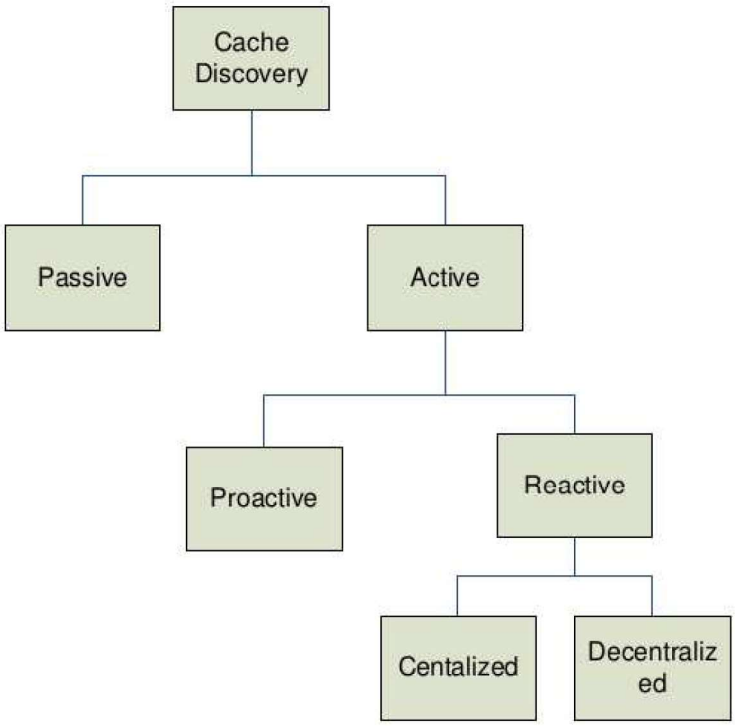

Figure 6.1: Taxonomy of Cache Discovery Protocols

before sending out its request. The task of locating data in reactive approach can be either centralized or distributed. In centralized cache control, one node is selected as coordinator which maintains the information about other nodes cache contents [Chand, 2007]. Due to the nature of the ad hoc environment, solutions that concentrate the data in a single entity must be discarded; the node hosting such a centralized entity would require significantly more resources than other nodes. Usually, ad hoc networks are formed by peer nodes, with limited capabilities, so it is not practical to elect one node to act as a repository for the data needed by the other nodes. Moreover, this approach introduces a single point of failure; this is unacceptable given that node failures are an integral part of ad hoc networks due to voluntary departures, crashes, or simply due to medium impairments [Ting, 2007].

In distributed reactive approach, [Lim, 2006] the source node will broadcast a data request message to the neighboring nodes. Although this method is simple, it relays on flooding to broadcast the data request. Flooding causes network contention and overhead when the network density is higher. Flooding is a very expensive process with respect to the bandwidth and energy [Tseng, 2002]. With resource constrained environments like MANETs employing flooding for cache discovery will be very costly. It also introduces lot of redundancy in the packet retransmission process. In flooding if there are N neighboring nodes in the network, then the data request is transmitted to all nodes except the target node which causes $N-1$ transmissions. If there are, on an average 'a' neighbors per node, where a is the average node degree, we get $a \cdot (N-1)$ receptions per flooding which causes significant communication overhead. Fig.6.1 gives the taxonomy of cache discovery protocols.

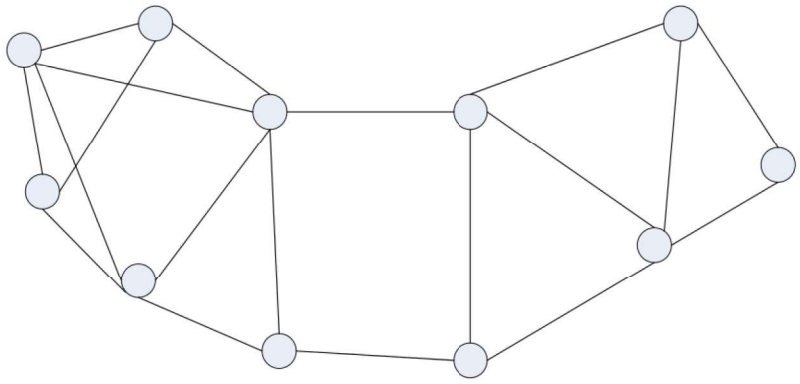

Figure 6.2: Graph representation of MANET

To overcome the limitations of above mentioned schemes, a distributed cache discovery protocol that reduces the message overhead is proposed in this chapter. The basic idea is to minimize the cache overhead and delay by reducing the number of control messages flooded in to the network. In the proposed work, the one hop neighbor list is divided into two zones based on transmission range and allocation based data discovery is adopted. This eliminates not only the flooding of data request into the network, but also reduces the overhead in processing tables.

6.3 Models and Assumptions

6.3.1 Network Model

A mobile ad hoc network is abstracted as a graph $G(V, E)$, where V is the set of nodes and $E \subseteq V^2$ is the set of links which gives the available communication. An edge (u, v) belongs to E means that there is direct

communication between two nodes u and v. The elements of E depend on the position and the communication range of nodes. All links in the graph are bidirectional i.e., if u is in the transmission range of v, v is also in the transmission range of u. The maximum communication range is assumed to be same for all nodes and is represented as R, which is given by the Euclidean distance $d(u, v)$ between nodes u and v. The set of neighborhood nodes in the range R_1 are represented as $N_{R_1}(U)$ and the set of neighborhood nodes in the range R_2 are represented as $N_{R_2}(U)$. $N_{R_2}(U)$ $N_{R_1}(U)$ represents the neighbor node set in the secondary zone.

Definition 6.3.1. Given a mobile ad hoc network with maximum communication range R, and $d(u, v)$ is the Euclidean distance between nodes u and v, the set of links E is defined as:

$$E = \{(u, v) \in V^2 | d(u, v) \leq R\}.$$

Definition 6.3.2. The neighborhood set $N(U)$ of the node u is the set of neighboring nodes that can be reached in one hop and is defined as:

$$N(U) = \{v | (u, v) \in E\}.$$

Definition 6.3.3. The neighborhood set $N_{R_1}(U)$ of node u is the set of neighboring nodes in the primary zone which are closer to the source node that comes in the range R_1, where $R_1 = R/2$ and is defined as:

$$N_{R_1}(U) = \{v | d(u, v) \leq R_1\}.$$

Definition 6.3.4. The neighborhood set $N_{R_2}(U)$ of node u are those nodes present in the secondary zone, which comes under the transmission

range R_2, where $R_1 <= R_2 <= R$.

$$N_{R_2}(U) = \{v | d(u,v) \leq R_2, R_2 > R_1\}.$$

6.4 System Architecture

The mobile computing environment considered in this work consists of mobile clients and a data base server connected to a static network. Each mobile client runs an application program that communicates with the data base server through messages. In order to minimize the number of data server request, the client node caches a portion of database in its local memory. For that each node maintains a list with the following fields: cached data id, cached data item, TTL, time difference between the current access and previous access. The contents of the local cache are shared by its neighboring nodes. Nodes in the network retrieve data items either from the local cache or from the neighboring cache if there is a local miss. When a node fails to find data in neighboring nodes, data is retrieved from the data server. A node that receives fresh data from the server caches a copy of it in the local cache and becomes a provider for that cached content for the neighboring nodes. Cache consistency is maintained through a weak consistency model using the parameter TTL.

The transmission radius R determines the maximum communication range of each node and is equal for all nodes in the network. Two nodes in the network are neighbors if the Euclidean distance between their co-ordinates in the network is at most R. The Euclidean distance between the nodes are estimated based on the relative position of nodes. The location based approach becomes practical due to the availability of low power, inexpensive GPS receivers for determining absolute or relative

position of nodes in an ad hoc network [Borriello, 2001]. Each node has a unique identifier and location, which it wishes to communicate to all its neighbors. Initially, to find the neighbor node set in the transmission range R for node A $N_R(A)$, a short neighbor request control message is disseminated in to the network. The request control message contains the following fields: *the source id*, *current location* and *a request id*. The *request id* is used to identify the neighbor request control message. When a node receives the request control message, it sends back a reply control message which includes the *node id* and current *location coordinates*. As the nodes are mobile, the particular position of any node changes in time, as do the connectivity between them. Thus, each node's set of neighbouring nodes defined within the transmission range will continuously change. In order to maintain the neighbour node set accurately, each node periodically sends a request control message to its neighbours. To reduce further message overhead the frequency at which update messages are disseminated is determined by the mobility rate of the nodes.

6.5 Cache Discovery Algorithm

Cache discovery refers to how a mobile node locates cached objects present in the neighboring nodes. Efficiency of a cache discovery protocol is measured using different metrics which include, communication overhead, energy consumption, delivery success rate and response delay. The nodes in an ad hoc network are, in general, battery powered and hence energy constrained. One of the major source for energy consumption is communication, both transmission and reception of packets. The design rationale of our cache discovery protocol is to optimize network traffic, the fundamental factor that reduces the communication cost. The

following section describes the cache discovery process proposed for the cooperative caching protocol.

6.5.1 Primary Zone and Secondary Zone

Cache discovery refers to how a mobile node locates cached objects present in the neighboring nodes. Two zones are defined, primary zone and secondary zone from the perspective of a wireless node as the average number of neighbors it can communicate directly and it depends on the transmission range of a node. The transmission range of a node can be changed in a number of ways (Chun, 2008). To increase the transmission range a node may transmit at a low rate, allowing more nodes to decode its transmission or may transmit with higher power. However, longer transmission range increases the number of nodes that locally compete on a shared channel, effectively increasing the access delay and reducing the network capacity (Rodoplu and Meng, 2009). Conversely, for short transmission range the chance of network being partitioned increases. Here our aim is to reduce the search space without affecting the transmission range. The primary zone is the smallest region that contains nodes which are closer to the source node that comes under transmission range R_1, where R_1 is equal to half the transmission radius R. The secondary zone contains the nodes that come between the transmission range R_1 and R. The neighbor node list is updated every 't' seconds and the value of 't' depends on the mobility rate of the nodes.

The steps involved in cache discovery process are given below.

1. For each node 'i', broadcast a neighbor request control message to find the neighboring nodes within the transmission range R.

2. For every node 'j', which is a neighbor of node 'i', send a reply control message which contains the node id and current location coordinates.
3. Calculate the Euclidean distance between the source node 'i' and the neighboring nodes.
4. Arrange the neighboring node list in ascending order based on Euclidean distance.
5. Divide the list in to two sets based on transmission radius of primary zone and secondary zone.
6. For each data request if there is a local miss:
7. Send the request to the nodes present in the primary zone.
8. If there is miss in the primary zone send the request to nodes in the secondary zone.
9. If there is a miss in the secondary zone fetch data from the data server.
10. Repeat steps 1 to 5 in every 't' seconds.

The process of the proposed cache discovery is illustrated in Fig. 6.3. Let us suppose that node A needs a document. First, the source node A looks up its own local cache for the document. If data is not found in the local cache, or if the received data does not result in a positive response due to invalid time to live parameter, then the source node queries its neighboring nodes present in the primary zone. If the node A is not able to find a copy of the data, it requests a look up from the nodes present in the secondary zone. If there is a cache miss in secondary zone the request is given to the source node S, where it replies to the request.

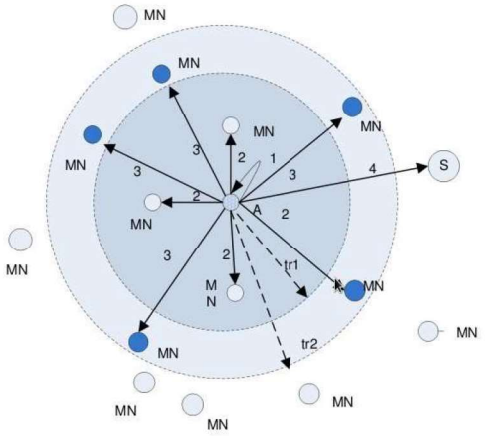

Figure 6.3: Cache Discovery Process

6.6 Analytical study

An analytical study has been carried out to assess the performance improvement for cooperative caching in terms of average user response time. This section develops an expression for average user response time for cooperative caching. The increase in speed, 'S' is the ratio of time needed for a task without cooperative caching, T_{wo} to the time for a task with cooperative caching T_w.

$$S = \frac{T_{wo}}{T_w} \tag{6.1}$$

If cooperative caching is not implemented, all local cache misses are sent to the origin data centre. If h_2 is the local hit rate, T_{local} is the average time to retrieve a document from local cache and T_s is the average time

to retrieve a document from the server, the average response time, T_{wo}, is given by,

$$T_{wo} = h_2 T_{\text{local}} + (1 - h_2) Ts. \tag{6.2}$$

With cooperative caching, the fraction of local misses satisfied by cooperative caches is h_3 and the fraction sent to the data server is $1 - h_3$. The overhead costs for cooperative caches for hit and miss is $T_{o,\text{hit}}$ and $T_{o,\text{miss}}$ respectively. The expression for average response time with cooperative cache is given by,

$$T_w - h_2 T_{\text{local}} + (1 - h_2) \times [h_3(T_c + T_{o,\text{hit}}) + (1 - h_3)(T_s + T_{o,\text{miss}})]. \tag{6.3}$$

Since the average time to retrieve an object from local cache is significantly less than the average time required to retrieve the data from remote data centre, $h_2 T_{local}$, can be neglected from the eqn (6.3).

With this assumption, the response time and speed up equations for proxy cooperation simplify to:

$$T_w = (1 - h_2) \times [h_3(T_c + T_{o,\text{hit}}) + (1 - h_3)(T_s + T_{o,\text{miss}})] \tag{6.4}$$

$$S = \frac{1}{h_3 \left(\frac{T_C + T_{o,\text{hit}}}{T_s}\right) + (1 - h_3)\left(1 + \frac{T_{o,\text{miss}}}{T_s}\right)} \tag{6.5}$$

The upper bound on average cooperation speed up when discovery overhead is negligible, $T_{o,\text{hit}} = T_{o,\text{miss}} = 0$ and discovery is perfect ($h_3 = H_3$) where H_3 is the maximum cooperative hit ratio, is given by

$$S_{\max} = \frac{1}{1 - H_3} \tag{6.6}$$

where $S_{\max}$ is the maximum speed up that can be achieved through

cooperative caching.

6.7 Simulation Study

The simulation model of the proposed cooperative caching protocol, COPN consists of nodes that caches part of the requested items temporarily. Each node is identified by a node id and a host name. The position of each node is given by the x and y coordinates. The transmission radius is assumed to be the same for all nodes and ranges from 200–500m. The transmission range of the primary zone is taken as half of maximum transmission radius. The nodes move in the given area according to the random waypoint mobility model [Broch, 1998]. Each node generates read only queries periodically. The queries generated follows a Zipf distribution [Breslau, 1999], which is frequently used to model non uniform distribution. There is a single data server available in the system. It is implemented in a fixed position in the simulation area. The data server contains all the data items requested by the mobile nodes with each item identified using a data id. The details of the simulation parameters are given in table 6.1.

6.8 Simulation Parameters

Parameter	Value
Simulation Time	3600 sec
Database	1000 items
Cache size	20–70 % of total database size
Number of Nodes	10 50
Mobility model	Random waypoint
Pause time	200sec
Skewness parameter (θ)	0.8
TTL	1000sec
Mean query generation time	10s

Table 6.1: Simulation Parameters

6.9 Simulation Results

The impact of different parameters like cache size, node density and transmission range on the performance of the proposed cooperative caching scheme (COPN) explained in the following sections. For performance comparison simulation of Global Cooperative Caching (GCC) scheme [Chauhan, 2011] and Neighbor Caching (NC) scheme [Joonho, 2003] is performed. Neighbor caching uses broadcasting for data discovery and LRU for cache replacement. GCC uses a cluster based approach for cache discovery. For the evaluation the same data access pattern and mobility model are applied to both the schemes.

Message Overhead. Figures 6.4 to 6.6 shows the results of comparison between COPN, GCC and NC in terms of message overhead. In

Fig. 6.4, the number of nodes is varied while keeping the cache size and transmission range fixed. The cache size is taken as 30% of the total database size and transmission range is kept as 250m. The message overhead for flooding algorithm increases much more rapidly than location aided caching when the number of nodes is increased. It is observed that COPN outperforms NC and GCC for all node densities. The difference is more significant for NC in higher node densities, because it uses broadcasting for cache discovery. Fig. 6.5 shows the comparison of message overhead for different cache sizes. The results show that COPN has better performance for all the cache sizes. This is because as the number of messages needed for cache discovery in COPN is very less compared to other schemes. Fig. 6.6 shows the effect of varying transmission range on message overhead. GCC and NC experience an increase in message overhead compared to COPN because as the transmission range is increased more number of nodes will come under neighboring node set and the number of nodes processing the broadcast message also increases.

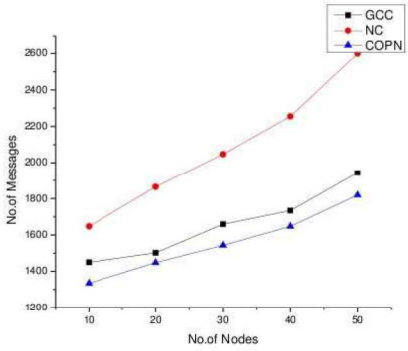

Figure 6.4: Message overhead vs node density

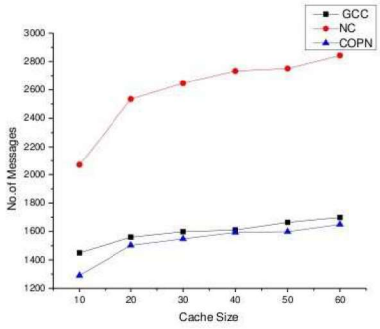

Figure 6.5: Message overhead vs cache size

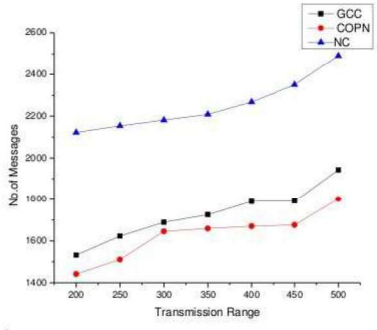

Figure 6.6: Transmission range vs message overhead

Cache Hit Ratio. Figures 6.7 and 6.8 show the result of comparison between COPN, GCC and NC in terms of cache hit ratio. Cache hit ratio includes local hit and neighbor hit. In Fig. 6.7 the cache size is varied from 10–60% of the total data base size, while keeping the number of nodes and transmission range constant. We choose the number of nodes as 50 and transmission range as 250m. COPN outperforms GCC and NC at all cache sizes. At small cache sizes COPN shows significant improvement in cache hit ratio compared to other schemes. For large cache sizes the difference in hit ratio between the schemes compared became less significant. This is because the replacement strategy plays a significant role at smaller cache sizes as the number of evictions from the cache is higher due to small cache size. Fig. 6.8 shows the effect of transmission range on cache hit ratio. In fig. 6.8 the transmission is varied ranges from 200–500m while keeping the cache size as 30% of the total database size and number of nodes as 50. COPN out performs

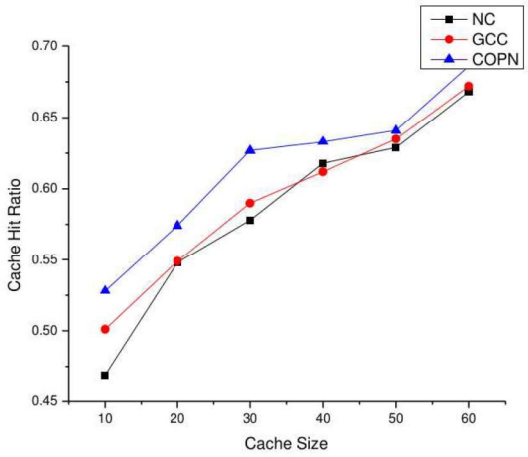

Figure 6.7: Cache hit ratio (%) for different cache sizes

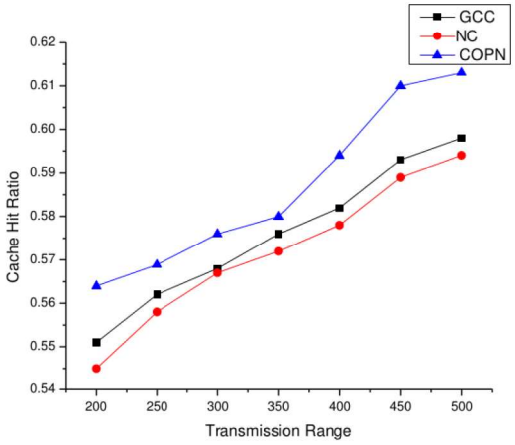

Figure 6.8: Transmission range vs hit ratio (%)

GCC and NC at all transmission ranges.

Query Delay. Figures 6.9 and 6.10 are presented to show the performance comparison between COPN, GCC and NC in terms of average query delay. In fig 6.9, the cache size is varied while keeping the number of nodes constant. It is observed that the average query delay for COPN is less for all cache sizes compared to GCC and NC. The difference becomes more significant at lower cache sizes because the cache hit ratio is low for other schemes at smaller cache sizes. Fig. 6.10 observed the result of varying the transmission range in terms of average query delay. Query delay is increased for higher transmission ranges. This is because as the transmission range is increased the number of neighboring nodes is also increased, which results in high cache lookup delay. The results show that COPN has better performance compared to other schemes at all transmission ranges.

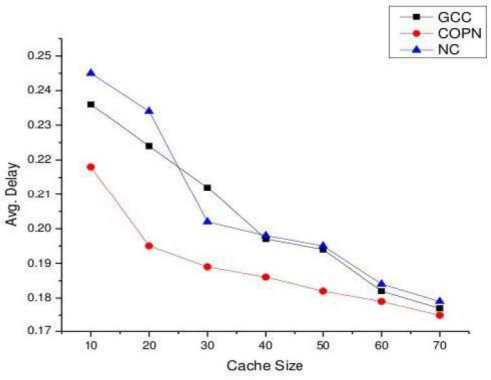

Figure 6.9: Avg. Query delay vs cache size

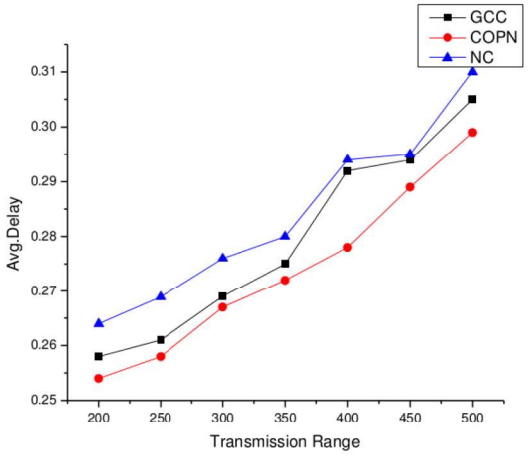

Figure 6.10: Transmission range vs Avg. Query Delay

6.10 Conclusion

Cache discovery is an important service in cooperative caching which has to be concentrated to improve the efficiency of cooperative caching. Many strategies were suggested by different researchers for cache discovery. Taxonomy and limitations of different cache discovery protocols are presented in this chapter. A new cache discovery protocol to deal with the limitations of earlier techniques was suggested in this chapter. The proposed protocol utilizes position information for efficient cache discovery. More precisely, message broadcasting is reduced by dividing the transmission range in to two zones. For comparison, two other algorithms were simulated. Simulation results show that this approach obtains good tradeoff between cache hit ratio, average query delay and message overhead.

Chapter 7

Energy Efficient Cache Discovery

Contents

7.1 Introduction

Mobile hosts are powered by battery while they are on move. Thus, to ensure good continuity of system operations over time, several approaches are taken to enhance the battery life. One such approach is to design power aware transactions, which make efficient use of the overall energy resources of the network. Conserving power prolongs the life time of a node and also the life time of the network as a whole.

The topology of an ad hoc network is very dynamic; it can change at any moment as disconnections are frequent due to mobility of nodes. The mobile clients may suffer from long access delay or access failure, when the nodes holding the data items are far away or unreachable. Data transfers must be reduced and mechanisms must be deployed to address the frequent disconnections and low bandwidth constraints. As discussed in the previous chapters, cooperative caching is a technique used in MANET to improve the data access efficiency. In cooperative caching the major amount of data transfer occur for cache discovery.

Efficiency of a cache discovery protocol may be measured using different metrics. Some of them include communication overhead, energy consumption, delivery success rate and response delay. This chapter addresses the problem of an energy efficient cache discovery protocol towards the goal of improving the performance in cooperative cache. The main design objective is to minimize the energy cost by dynamically adjusting the transmission range to reach its neighbouring nodes, thus saving power whenever possible. This chapter also discusses the basic concept behind energy conservation as well the importance and classification of energy conservative mechanisms.

7.2 Energy Model

Energy constraints are not inherent to all ad hoc wireless networks [Goldsmith, 2002]. In some cases devices in an ad hoc network may not be moving and is attached to a large energy source. The devices may be part of a large vehicle that can generate significant amount of power over the long term. However, most of the nodes in an ad hoc network are powered by batteries with limited lifetime. Some of the most exciting applications for ad hoc wireless networks follow this paradigm. For example an ad hoc network set up for a military application contains a distributed collection of mobile devices powered by batteries. Since these devices have limited battery, it could drain its battery power when communication is high. Thus, these networks must conserve battery power in all communication functions, and devise techniques that use the available power at each node in a fair and efficient manner. Devices with rechargeable batteries must conserve energy to maximize time between recharging.

There are three major causes of energy consumption in a node [Ephremides, 2002]. Energy consumed while transmitting a message, energy consumed while receiving a message and when a node is 'on' and is not actively receiving. In the transmitting mode, energy is consumed in two ways: For message processing and transmission. In the receiving mode, energy is consumed only for processing. Finally, in the 'on' mode energy consumption is for processing, but it is quite low compared to the transmitting and receiving modes. Thus, to reduce energy depletion in a node the number of transmissions should be reduced. While transmitting a message, a node spends part of its energy and there are a few energy models used to compute this consumption. In the most commonly used one, the measurement of energy consumption when transmitting a unit

message depends on the range of the emitter n:

$$E(n) = r(n)^{\alpha}$$

where α is the real constant greater or equal than 2 and $r(n)$ is the range of the transmitting node.

7.3 Power Conservative Protocols

The power conservative protocols for ad hoc networks are generally divided into two categories.

1. Transmitter power control mechanism
2. Power management algorithms.

The latter can be further classified into MAC layer, network layer and higher layer implementations.

7.3.1 Transmitter power control Mechanism

In wired networks, the network topologies are fixed and can easily be determined by wire-line deployments. In contrast, the topology of a wireless network will change even if no host mobility is present. The topology of a wireless network depends on uncontrollable factors and controllable factors [Ilyas, 2014]. The uncontrollable factors include node mobility, weather interference and noise. The controllable parameters include transmit power and antenna direction. By adjusting antenna directions and tuning transmission powers, one can change the network topology. Power control refers to the technique of tuning hosts transmission powers to proper range. Power control has two advantages.

- It can conserve battery energy.
- It can reduce radio interference and thus increase spatial use of wireless bandwidth.

Several topology control algorithms are proposed for ad hoc networks. Ramanathan [2005], considered the problem of adjusting the transmit power of nodes in an ad hoc network to create desired network topology. For static networks two centralized algorithms were proposed to construct a connected and biconnected network. In biconnected network the loss of any single node or link will not partition the network. The proposed algorithm assumes that the location of nodes is known. A greedy, minimum cost spanning tree approach is used for constructing the network. For a mobile multi-hop network, two distributed heuristics were presented, where hosts are allowed to constantly adjust their transmission power to maintain desired topology. Whenever the degree of desired node is below certain threshold, a host is allowed to increase its transmission power. Thus a connected topology is maintained using minimum power.

7.3.2 MAC layer Power Management

In OSI reference model, the MAC layer is a sub layer of the data link layer. The MAC layer is mainly responsible for access scheduling of shared medium. MAC layer power control mechanism mainly involves reduction in retransmission of data, use separate channel for data and control packets and to efficiently manage the IEEE 802.11 Power-Saving mode. [Bononi, 2001] proposed a protocol called Distributed Contention Control (DCC) to reduce the retransmission of data. In DCC, the probability of successful transmission is calculated before sending the actual

frame. If the calculated probability of success is low, the transmission is deferred to minimize the potential retransmission overhead. The Power Aware Multi access protocol (PAMS) proposed by [Raghavendra, 1997] uses a separate channel for data and control packets. This protocol uses two signaling packets: RTS (request to send) and CTS (clear to send). The RTS and CTS packets carry the expected duration of the data packet transmission. By doing this the host can determine when and for how long to power its antenna off. The protocols to effectively manage the power saving mode of IEEE 802.11 is proposed by [Hsu, 2002].

7.3.3 Network Layer Power Management

The Network layer is responsible for routing packets towards destination. In wireless networks, mobile host are free to move, resulting in frequently changed routing paths. Traditional ad hoc routing protocols adopt metrics such as hop count, link quality and location stability for route selection. These metrics, however do not take battery power conditions into account, which may lead to improper energy expenditure and thus reduced network life time. In power aware routing protocols, routing is based on shortest cost path instead of shortest hop metrics, where "cost" is related to energy efficiency. Several power aware routing protocols are proposed for multicast [Singh, 1998], [Chang, 2000] and broadcast ad hoc networks [Wieselthier, 1998], [Shah, 2002]. The basic design principle of power aware routing protocol is to balance energy usage among mobile nodes to prolong network lifetime, while at the same time conserving overall power consumption.

For power aware routing in broadcast networks, energy efficient multicast/broadcast trees are constructed. This is usually accomplished by

transmitter power control [Wieselthier, 1998]. When the power level is high, a transmitter can reach more receivers within one hop. On the other hand, when the power level is low, fewer receivers can be reached at a time, but energy expense is conserved. The power aware multicast/broadcast trees should balance between these two total power consumption and increases hosts / networks life time, while maintain acceptable throughput and delays.

7.3.4 Higher layer Power Management

Higher layer power management consists of designing energy efficient transport layer and application layer protocols. Energy efficient transport protocols attempt to reduce delays and the number of unnecessary retransmissions, both of which are energy consuming. Kravets, [1998] and Kravets[1999] address power efficient transport protocol based on TCP to reduce unnecessary battery energy expenditure.

For application layer, several techniques have been proposed to include energy efficiency in software design considerations. [Tan, 2001] focuses on organizing the invalidation report for cache invalidation to conserve energy of mobile hosts. [Agrawal, 1998], the authors address a power sensitive video processing system. The basic idea is to reduce the number of transmitted bits. Other application level energy efficient strategies are provided in [Flinn, 1999] and [Naik, 2001].

In conclusion the power saving mechanisms can be classified as:

1. Scheduling the wireless nodes to alternate between active and sleep mode such that the redundant coverage is minimized.

2. Power control by adjusting the communication/sensing range of wire-

less nodes.

3. Energy efficient routing, data gathering.

4. Reducing the amount of data transmission and avoiding redundancy.

7.4 Problem Formulation

The aim is to minimize the amount of power (or energy per bit) required to transmit a packet from source to destination. More precisely the problem is stated as

$$\text{Minmize} \quad \sum_{i \in \text{path}} P(i, i+1)$$

where $P(i, i+1)$ denotes the power expended for transmitting and receiving between two consecutive nodes, 'i' and $i+1$ in the route P.
The link cost can be defined for two cases:

When the transmit power is fixed and when the transmit power is varied dynamically as a function of distance between the transmitter and the intended receiver. For the first case, energy for each operation: receive, transmit, broadcast, discard etc. on a packet given by

$$E(\text{packet}) = b * \text{packet size } + c$$

where 'b' and 'c' are the appropriate coefficient for each operation. Coefficient 'b' denotes the packet size dependent energy consumption where as 'c' is a fixed cost that accounts for acquiring the channel and for MAC layer control negotiation. For the second case the power needed for transmission and reception is a linear function of distance between

the two neighboring nodes. This model make use of the GPS position information to transmit packets with minimum required transmit energy. The following section explains this model in more detail.

7.5 Proposed Cache Discovery Algorithm

The main objective of the cache discovery protocol proposed in this chapter is to minimize the power or energy per bit required to transmit a packet from source to destination. The goal of this scheme is to reduce the average number of messages among the cooperative caches while maintaining high cache hit ratio. As explained in the earlier section the link cost for the transmission can be defined for two cases. Here the second case is considered where the power is varied dynamically as a function of distance between transmitter and receiver. The power consumed $P(d)$ by a node in transmitting a request for a distance d is given by

$$P(d) = d^{\alpha} + c$$

for some constants α and c. If we ignore the constant we can see that the power consumption is directly proportional to distance. If the nodes can adjust their transmission power for each node based on distance, the power consumption can be reduced considerably, which leads to increased battery life. So by making use of the position coordinates we can transmit packets with minimum required transmit energy. The basic requirement of this scheme is that each node should know its relative position. This approach is limited to ad hoc networks of relatively low mobility patterns. If the nodes are highly mobile, the power management algorithm might fail to cope with the fast and sudden changes due

to fading and interference conditions [ElBatt, 2000].

In the proposed algorithm, the decision to forward the data request is based solely on the location of itself and its neighboring nodes. The maximum transmission range R of a node is divided into different zones with transmission radius $R/2$, $R/4$ and $R/6$ and find the nodes present in this transmission radius excluding the one already present in the lower range. This can be found from the neighbor node list formed when a node is active. When the requested data is not found in the local cache the request is forwarded to the nodes in the lowest transmission radius zone. After sending the request the node waits for the reply. If the node doesn't receive a positive reply after a period of time t_1, which is a predefined threshold, the node search for the data in the next zone and this process continues until it gets the needed data or when it reaches the maximum transmission range. If we are not able to find the required data within the transmission radius R the request is directly given to the server. The time out interval set for each zone is different to minimize the waiting time. The power needed for each transmission is assumed to be different. Initially, the transmission power is kept at the minimum level and the node will search for the data in the lowest transmission range. If we could not find the desired data, the transmission power is increased to search for the data in the next zone. Currently in the proposed algorithm, power is increased only by a fixed amount. The process of cache discovery is fully distributed and runs in all the nodes in the network.

7.6 System Design

7.6.1 Network Model

A mobile ad hoc network is abstracted as a graph $G(V, E)$, where V is the set of nodes and $E \subseteq V^2$ is the set of links which gives the available communication. An edge (u, v) belongs to E means that there is direct communication between two nodes u and v. The elements of E depend on the position and the communication range of nodes. All links in the graph are bidirectional i.e., if u is in the transmission range of v, v is also in the transmission range of u. The maximum communication range is assumed to be same for all nodes and is represented as R, which is given by the Euclidean distance $d(u, v)$ between nodes u and v. The set of neighborhood nodes in the range R are represented as NR (U) and the set of neighborhood nodes in the range R_i, is given as $\mathrm{NR}_i(U)$.

7.7 System Model

The system model consists of a mobile ad hoc network with a set of nodes which are able to communicate with each other. The transmission radius R determines the maximum communication range of each node and is equal for all nodes in the network. Two nodes in the network are neighbors if the Euclidean distance between their coordinates in the network is at most R. The Euclidean distance between the nodes are estimated based on the relative position of nodes. It is assumed that each node knows its current location precisely. For power adjustment path loss model is used. In this model, the path loss depends on the height s of the antennas as well as the transmitter-receiver separation [Rappaport, 1996].

Initially, to find the neighbor node set in the transmission range R for node A NR(A), a short neighbor request control message is disseminated in to the network. The request control message contains the following fields: the source id, current location and a request id. The request id is used to identify the neighbor request control message. When a node receives the request control message, it sends back a reply control message which includes the node id and current location coordinates. Upon receiving the location coordinates of neighboring nodes, the source node measures the Euclidean distance D_{ij} between the source node n_i and neighbor n_j. When a node is farther away from the source the Euclidean distance will be large. Each node will arrange the neighboring node list in the ascending order of their distance to determine the order of neighboring nodes to receive the data request. To reduce the number of messages, the neighboring node set is divided into different zones, based on the transmission range. To maintain the neighbor node set accurately, each node periodically sends a request control message to its neighbors. Each node maintains a list which stores the cached data item. The list contains the following fields: cached data id, cached data item, TTL, time difference between the current access and previous access. This table is updated when a new data is placed in to the cache. The cache space for each node is limited and when it is full, a replacement strategy evicts the unwanted data. The contents of the local cache are shared by its neighboring nodes.

The data server is assumed to be a fixed location. The data server maintains a set of data items uniquely identified by means of data item id D_i for $1 \leq i \leq n$ where 'n' is the size of the data base. Each node has a local cache, with certain data items. Each mobile nodes is identified by

a distinct ⟨Host id, Name⟩ for $1 \leq i \leq N$, where N is the density of the network. Nodes in the network retrieve data items either from the local cache or from the neighboring cache if there is a local miss. When a node fails to find data in neighboring nodes, data is retrieved from the data server. When a node receives data from the server, it caches a copy of it in the local cache and becomes a provider for that cached content for the neighboring nodes. When a node wants to access data, it checks in its own local cache. If the requested data is not cached, the node checks whether the data is present in the neighboring nodes. If we are not able to find the data from the neighbor list the request is given to the data server.

7.8 Implementation

The proposed scheme is a fully distributed scheme, with each node runs an application to retrieve and cache data items from other neighboring nodes. Each node caches part of the requested items temporarily. Initially, the mobile nodes are randomly distributed in the simulation area and the mobility of the nodes is assumed to be low. Each node is identified by a node id and a host name. The position of each node is given by the x and y coordinates. The data server is implemented in a fixed position in the simulation area. The data server contains all the data items requested by the mobile nodes. The nodes that generate data request are selected randomly and uniformly. The time interval between two consecutive queries generated from each node follows an exponential distribution with mean of 10sec. Each mobile node generates a single stream of read only queries. The queries generated follow a Zipf distribution, which is frequently used to model non uniform distribution. The

data request is processed in First Come First Served (FCFS) manner at the server. Each miss in the cooperative cache will incur a delay of 8 ms to retrieve data from the data server. The simulation model is implemented in JAVA.

7.8.1 Metrics

The performance metrics evaluated includes cache hit ratio, message overhead and power savings ratio. The evaluation of these parameters are done by varying the number of cache locations with respect to number of nodes and the behavior of cache hit ratio for different cache sizes. The percentage of power saved is calculated as power usage of cooperative caching scheme compared—power usage of the proposed scheme / power usage of the proposed scheme. Message overhead is the overhead messages needed to manage the cache discovery process in cooperative cache.

7.9 Performance Evaluation

The performance of the proposed cache discovery scheme Cooperative Cache New (CCN) is evaluated by comparing with Neighbor Caching (NC), a caching scheme which uses broadcasting for cache discovery. Fig. 7.1 shows the performance comparison of two schemes, as a function of message overhead for different node densities. The figure shows that CCN outperforms NC at all node densities. As the node density increases, the difference become more significant, this implies that CCN can benefit from larger node densities. Fig. 7.2 depicts the power savings ratio of the proposed cache discovery scheme compared to neighbor caching.

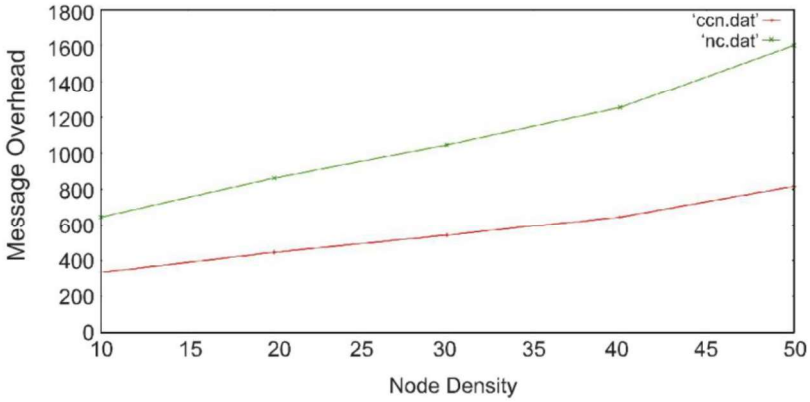

Figure 7.1: Message overhead for different node densities

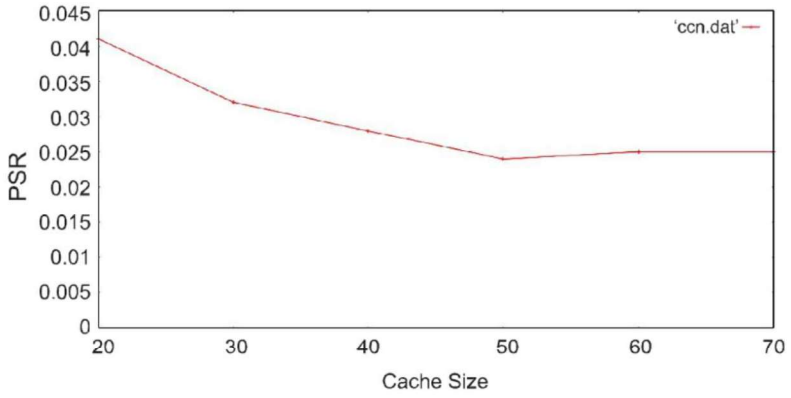

Figure 7.2: Power savings ratio for different node densities

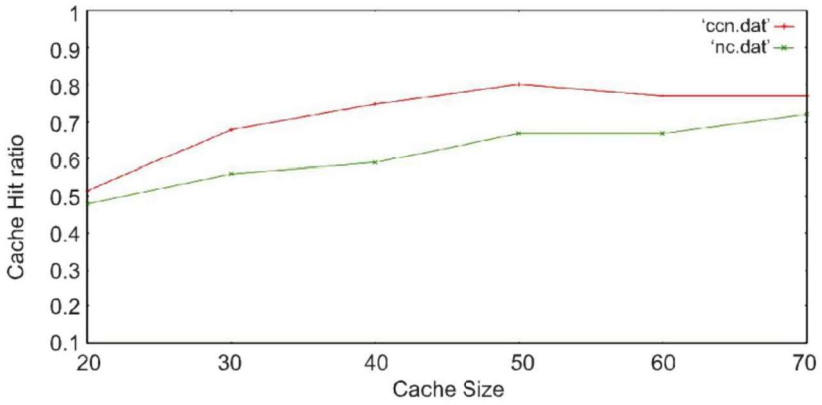

Figure 7.3: Cache hit ratio for different cache sizes

From figure 7.3 we can see that the cache hit ratio for CCN is greater than NC for different cache sizes. The relative performance of cache hit ratio remains relatively stable for higher cache sizes.

7.10 Conclusion

In this chapter, a cache discovery algorithm for cooperative caching for optimizing the power usage and decreasing energy consumption by transmission range adjustment is proposed. In this algorithm the data discovery process is based on the position of the neighboring nodes and according to the position of neighboring nodes the transmission range is adjusted for data retrieval. The proposed algorithm is compared with other cooperative caching protocol in terms of message overhead and cache hit ratio. The power savings ratio of the proposed approach is also calculated.

Chapter 8

Adaptive Backbone Based Cooperative Caching in Mobile Ad hoc Networks

Contents

8.1 Introduction

Existing cooperative caching algorithms for mobile ad hoc networks face serious challenges due to message overhead and scalability issues. In the last two chapters, cache discovery protocols which reduce message overhead and power consumption are explained. However, the performance of these protocols is limited by the network size. Since most of the cache discovery protocols proposed for ad hoc network do not consider the problem of scalability, new approaches are needed to achieve better performance in large scale networks.

This chapter proposes an adaptive virtual backbone based cache discovery protocol that can be used for large scale networks. A dynamic virtual backbone structure can be formed among the mobile nodes by Connected Dominated Set (CDS). Message overhead in cooperative caching is mainly due to cache lookup process used in cooperative caching. The idea in this scheme is to reduce the number of nodes involved in cache look up process, by constructing a virtual backbone adaptive to the dynamic topology in mobile ad hoc networks. The proposed algorithm is decentralized and the nodes in the CDS perform data dissemination and discovery.

Due to dominating behavior of virtual backbone, every node in a connected network is a neighbor of at least one backbone member. Since MANETs are infrastructure less networks, they do not have a physical backbone infrastructure. The aim of this work is to develop a cache look up protocol that takes the advantage of CDS. The idea in this scheme is to reduce the number of nodes involved in cache lookup process, by constructing a CDS that contains a small number of nodes, still having full coverage of the network while only relaying on local exchange of

information and knowledge.

Cooperative caching based on virtual backbone reduces the search space for cached data to the hosts in a smaller subnetwork generated from the CDS. The proposed cooperative caching architecture consists of two phases: Backbone formation phase and a distributed cache look up phase. In phase 1, a dynamic virtual backbone is formed such that each node in the network is either a part of the backbone or one hop away from at least one of the backbone nodes. The second phase is the distributed cache look up phase, in which data request message is forwarded to nodes inside the dominating set for data discovery. Simulation results show that the message overhead created by the proposed cooperative caching technique is very less compared to other approaches. Moreover, due to the CDS based cache discovery we applied in this work, the proposed cooperative caching has the potential to increase the cache hit ratio and reduce average delay.

8.2 System Model

The proposed framework considers an ad hoc network environment with a number of mobile nodes and a database server. It is assumed that all the nodes in the network have the same transmission range, R. Two nodes in the network are neighbors if the Euclidean distance between their coordinates in the network is at most R. The communication channel is shared between the nodes in the network, hence transmitting and receiving messages is not allowed at the same time. An abstract entity called virtual backbone is formed, which includes a collection of nodes responsible for cache lookup. After the formation of a virtual backbone, the nodes in the network will be either a backbone node or an ordinary

node. The backbone nodes are referred as Dominant Nodes (DN) and the ordinary nodes which are not in the set are referred as End Nodes (EN). The rest of the notations and assumptions are as follows.

In order to minimize the number of data server request, every node caches a portion of database in its local memory. The contents of the local cache are shared by its neighboring nodes. Each node has a unique identifier and location, which it wishes to communicate to all its neighbors. A node x is a neighbor of another node y if x is located within the transmission range of y. Each node maintains two lists, one to store the ID of the neighboring nodes and the second list to store the neighbor node list of each neighboring node. Initially, to find the neighbor node set of node x in the transmission range R, a short neighbor request control message is disseminated into the network. The request control message contains the following fields: the source id, current location and request id. The request id is used to identify the control message as a neighbor request control message. When a node receives the request control message, it sends back a reply control message which includes the node id and the neighbor list of its neighbors. This is to find the nodes that are two hops away from each node. As the nodes are mobile, the location coordinates of a node may change in time, as do the connectivity between them. Thus each node's set of neighboring nodes defined within the transmission range will continuously change. In order to maintain the neighbor node set accurately, each node periodically sends a neighbor request control message to its neighbors.

8.3 The Main Components of the Framework

For efficient cache discovery and data access, the cooperative caching architecture uses a virtual backbone based design in which data items are easily located through the indexing performed by the backbone nodes constructed from the connective dominating set. The proposed solution for cooperative caching works in two steps: In the first step, virtual backbone setup is performed with a subset of nodes from the CDS. The second step is the cache lookup phase in which the desired data item is searched by sending the request to the nodes in CDS. The role of EN is to cache data items in its local cache that are returned by the server or by other EN. Data request is originated either at DN or EN, and is served only by the DN. For a local miss at the EN, the DN makes decisions in serving request based on the needs of the EN.

Furthermore, a DN can also play the role of EN by requesting data items or as a caching node by caching data items in the local cache. To cope with the limitations of mobile ad hoc networks and to reduce message overhead, the DNs are utilized to store indexing information of the cached data items. Each DN maintains index information of the data item present in its neighboring nodes. An index table is maintained for this. Each entry in the table consists of a pair of index information $\langle D_i, n_i \rangle$. The parameter D_i is the data ID and the second parameter is the associated node ID that contains the data. The size of the index table is equal to the number of data items present in the neighboring caches and is sorted based on the key value.

The proposed architecture minimizes the message overhead in two ways. First, for each local miss, EN directly contacts its neighboring DN node for the requested data item. Secondly, the DN node maintains

an index of the cached data items during the neighborhood formation. The number of messages required for data lookup is reduced as the DN takes the responsibility of finding the requested data item. Each DN maintains an index table to record the information about its neighboring nodes cached data item. More details of this are addressed in the coming subsections.

8.4 Virtual Backbone Construction

An ad hoc network topology can be modelled as a unit disk graph $G = (V, E)$ in which there is an edge between two nodes if and only if their distance is at most one [Clark, 1990]. Virtual backbone in an ad hoc network can be formed by CDS of the corresponding unit disk graph [Das, 1997].

Definition 8.4.1. For a graph $G(V, E)$, a dominating Set S of G is defined as a subset of V such that each node in $V \backslash S$ is adjacent to at least one node in S.

Definition 8.4.2. A connected dominating set (CDS) for a graph $G(V, E)$ is a subset V' of V, such that each node in V-V' is adjacent to some node in V', and V' induces a connected subgraph of G.

The virtual backbone has an important role in ad hoc networking for routing, broadcasting and mobility [Das, 1997]. The problem of finding minimum dominating set has been proven to be NP-Hard [Clark, 1990]. The heuristics to find minimum CDS falls into two categories: Prune based and Maximal Independent Set (MIS) based. In prune based approach [Dai, 2003, Wu, 1999] the redundant nodes are pruned from the

CDS set according to certain rules. In maximal independent based algorithms [Ding, 2003, Wan, 2002] first MIS is formed and then connectors are found to form a CDS. In this paper, we implement a prune based algorithm proposed by [Wu, 1999] since it is simple and easily maintainable CDS construction algorithm.

8.4.1 CDS Construction

Wu [1999] proposed a localized algorithm for the formation of CDS based on a marking process. In this algorithm, every node $v \in V$, in the given graph $G = (V, E)$ will be either in marked (T) or in unmarked state (F) after the marking process. A node is in marked state if it has two unconnected neighbors, otherwise it is unmarked. A set of marked nodes form the CDS. The size of the CDS is further reduced by applying two rules for pruning which is explained below. The open neighbor set of each node v is represented as $N(v) = \{u | \{u, v\} \in E\}$.

Marking Process

1. Initially, all nodes are unmarked, in F state.

2. Every node v exchanges its $N(v)$ with all its neighbors.

3. Mark v to T if it has at least two unconnected neighbors.

Figure 8.1, where a connective dominating set is computed illustrates this algorithm. In this example, node 0 is not marked because all the neighboring nodes are connected, while node 1 is in the marked state since it has two unconnected neighbors node 8 and node 6. After the marking process assume that V' is the set of vertices that are marked

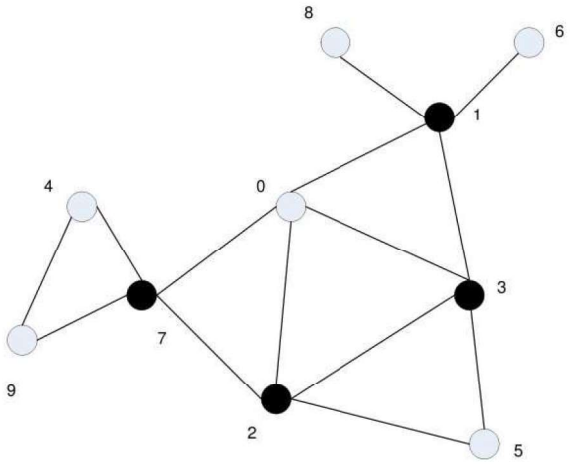

Figure 8.1: A connective Dominating Set

T in V i.e., $V' = \{v|v \in V, m(v) = T\}$. The induced graph G' is the subgraph of G induced by V, i.e., $G' = G[V']$, forms the CDS. To remove redundant nodes from the obtained CDS set, two pruning rules based on node ID are applied. A distinct ID, id (v) is assigned to each vertex in G. $N[v] = N(v) \cup \{v\}$ is the closed neighbor set of v.

Rule 1. Consider two vertices v and u in the marked state. If $N[v] \subseteq N[u]$ in G and if id $(v) <$ id (u), the marker of v is changed to F.

According to Rule 1, a marked node can unmark itself if ts neighbor set is covered by another marked node with higher node ID.

Rule 2. Assume that u and w are two neighbors of marked vertex v in G'. If $N(v) \subseteq N(u) \cup N(w)$ in G and $\mathrm{id}(v) = \min\{\mathrm{id}(v), \mathrm{id}(u), \mathrm{id}(w)\}$, then the marker of v is changed to F.

According to rule 2, a marked node can unmark itself if its neighborhood is covered by two other directly connected marked nodes with higher node IDs. The combination of the two rules reduces the size of a CDS. The message complexity of this algorithm is $O(n)$, where 'n' is the number of nodes in the network.

8.5 Cache Lookup Phase

The virtual backbone constructed using the Dominating Set algorithm. Some mechanisms are needed to locate data present in other mobile clients. The DN is the central component of the system and plays a major role in finding the location of the cache data. All the entries in the index table are sorted in the increasing order of the data ID. In our proposed system the data search process can be initiated either by the DN or by the EN. To retrieve data, each node initially searches in its local cache. If there is a cache miss, it tries to locate the requested data in other cooperate caches. In order to minimize redundant messages, DN performs the data lookup process for primary cache miss. Every DN keeps a list of data items present in their neighboring nodes. When there is a primary cache miss, the query node sends Request Packet (RP) to the nearest DN. Upon receiving the RP packet, the DN node extracts the data ID from the RP packet; performs a binary search using the data ID as the key value. If there is match in data ID, the corresponding node ID is taken from the index table. If the DN does not have a matching data, it adds its address to the RP and sends the modified RP to the nearest DN that has not been checked yet. This continues until a hit occurs or until all the DNs are checked, in which an attempt is made to contact the data source. For a cache hit, the DN nearest to the query node stores the

Table 8.1: Notations used in the algorithm formulation

Notation	Description
D_i	Set of data items 1.N
d_i	Search data item
d_x	Data content
LC_i	Local cache of node i
IT	Index table
DS	Data server
REN_i	Requesting end node i
EN [1] ... EN [n]	Set of end nodes
DN [1] ... DN [n]	Set of dominating nodes

data id and query node id in the index table. If there is no reply from the DN or when there is a time out, the data request is forwarded to data server. Typically, the time complexity of the search process is $O(n \log n)$ where 'n' is the number of neighboring nodes, making it very efficient. The detailed algorithm given below shows how the data search process is carried out in a mobile node. The notations used in the algorithm are summerized in table 8.1.

Algorithm 3 depicts the search process for a data request originating in a mobile node. There are three possible sources for serving a request: the source that caches the data item at its local cache, the EN and the data server. These three possible sources are accessed in an order based on the availability of data in each data source. The procedure *Data Search* is called when a node needs some data (line 1). The requested data item is checked in the local cache first to see if it is available (line 5). If the requested data is present in the local cache the request is served locally and the process is ended (line 10). Furthermore, if there is a miss

Algorithm 3 Data Search Process

```
1: SearchData (d_i):
2: //SearchData returns the data item to the requesting node
3: Begin
4: // Search local cache for data item
5: if (Search_LC (d_i)=True) then
6:    // Returns the content
7:    return d_x;
8: else
9:    Forward Req_packet (d_i) to DN_i
10: end if
11: for every mode ∈ DN[i] do
12:    Search in the index table of DN
13:    if (Search_IT (d_i) = DN[i].item[j]) then
14:       Begin
15:       // Send Acknowledgement with node id that caches d_x
16:       Send ack_packet(n_i) to DN
17:       // DN redirects the req_packet to EN_i who caches d_x
18:       Forward Req_packet (d_i) to EN_i
19:       // EN_i sends reply_packet that contains d_x
20:       return d_x;
21:    end if
22: end for;
23: while (REN_i doesn't receiving a reply packet that contains d_x) do
24:    // DN redirects the req_packet to DS
25:    SendReq(d_i) to DS
26:    return d_x;
27: end while
```

in the local cache, lines 11 and 13 perform the data discovery process from the index table of DN. The index table stores the information of the data items cached by the neighboring nodes. Lines 14–21 deal with the retrieval of data from the EN. A fast response is achieved as the DN forwards request directly to the EN, instead of checking node by node. If the requested data is not found at the index table entries at the DN the request is forwarded to the data server (Lines 25 and 26).

8.5.1 Index Table Management

The DN acts as an index node storing the address of the neighboring nodes which stores the data. Upon receiving a data request from the EN, if the data item is present in any one of the neighboring nodes the table is not updated. Otherwise, the corresponding EN ID and data ID is stored in the index table. This is because after some time the node will cache the requested data. When an item is removed from an EN cache when it is full, it has to be notified to the corresponding DN. To enable the notification of items being removed from the cache, a Cache Item Remove (CIR) packet is send to the DN. The CIR packet contains $\langle D_i, n_i \rangle$ signifying the ID of the node and data that is going to removed. Upon receiving the CIR packet, the DN will remove the selected entry from the index table to prevent misdirected requests. Whenever an EN leaves the network, the node ID (n_i) field is searched to find corresponding entry related to that node and is removed from the DN index table.

8.5.2 Cache Management

Cache management involves cache replacement and cache placement. As in any caching scheme, the capacity of cache in each node is limited

and appropriate cache replacement mechanism is needed to evict unwanted data from the cache. The proposed cooperative caching make use of replacement policy proposed in chapter 4 for replacement [5], called Extended-Least Recently Used (E-LRU) replacement. Since cache placement is performed through DN duplicate data placement is reduced. Cache consistency is maintained through a weak consistency model using the parameter TTL. When the TTL time expires, the cached data item is not valid and we have to fetch the updated data from the server.

8.5.3 Effect of Node Mobility

Due to mobility of nodes there may be several new link connections and disconnections to the nodes within the transmission range. When the topology changes the role of nodes may change. For maintaining the CDS in the case of node mobility the following steps are taken.

Each mobile node updates its neighbor list every t seconds. A mobile node x compares its new neighbor set $N'(x)$ with its original neighbor set $N(x)$. If they are same, it simply remains in its state. Otherwise, it performs the following steps. The new open neighbor set of node x is exchanged with its neighbors. Each node in the set will change its state to DN, if it has two unconnected neighbors. If a new DN is formed, the newly formed DN and its DN neighbors apply Rule 1 and Rule 2 to reduce the number of nodes in the CDS. Whenever DN node recognizes a broken link to its neighboring nodes, DN maintains its status as dominating node if it has two unconnected neighbors; otherwise it changes its state to EN. Due to the reconstruction of virtual backbone the issue of load balancing can be solved because the role of each node changes between DN and EN. However, at higher node densities nodes with higher node ID have

more chance to become DN due to the pruning rules used.

8.6 Performance Evaluations

8.6.1 Simulation Model and Environment

To evaluate and compare the proposed scheme with other cooperative caching schemes, a simulation model was developed a using JAVA. The simulated model consists of mobile nodes which can move freely. The position of each node is given by the x and y coordinates. The transmission radius is assumed to be the same for all nodes. The nodes move in the given area according to the random waypoint mobility model [Broach, 1998]. Initially, the mobile nodes are randomly distributed in the simulation area. After that each node randomly chooses its destination with a speed s which is uniformly distributed $U(V_{\min}, V_{\max})$ and travels with that constant speed s. When the node reaches destination, it pauses for 200s. After that it moves to the new destination with speed s'. We considered a low mobility network where the virtual backbone is reconstructed in every 250s. Each node periodically generates read only queries. Both the EN and DN can generate queries. The querying nodes are selected randomly from those nodes that move locally in the defined simulation area. The time interval between two consecutive queries generated from each node follows an exponential distribution with mean of 10s. The queries generated follow a Zipf distribution [Breslau, 1999].

The data server available in the system is implemented in a fixed position in the simulation area. The data server is supposed to have direct access to the Internet using the wired medium. The data server contains all the data items requested by the mobile nodes. Each data

item is associated with a TTL parameter that determines when the data item becomes obsolete. The data request is processed in First Come First Served (FCFS) manner at the server. An infinite queue is used to buffer the request when the data server is busy. Finally, each node (EN and DN) maintains a local cache that uses the E-LRU replacement policy. The cache size of the all the nodes are assumed to be the same. The capacity of the cache is varied during simulation to study the influence of different cache size.

8.7 Results and Discussion

The performance of the algorithms were evaluated,

1) The proposed Virtual backbone based Cooperative caching (VC),

2) Group Caching (GC) [Ting,2007], that uses a table based approach for cache discovery,

3) Neighbour Caching (NC), [Cho,2003], which uses broadcasting for data discovery.

In particular, the performance parameters evaluated are: Cache hit ratio, Average delay, Message overhead and Power Savings Ratio.

8.7.1 Effect of Cache Size

In this set of simulation, the cache size is varied to study how different caching schemes behave when the cache size varies. Ad hoc networks are formed for users with similar interest. With the presence of a cooperate cache the number of data requests sent to the data server reduces

considerably. Cache size is an important parameter to take into account in cooperative caching. When the cache size is increased the number of document included into the cache is also increased. Thus the percentage of cache hit ratio will be higher for large cache sizes. Figure 8.2–8.4 shows the performance of VC compared to NC and GC in terms of cache hit ratio, average delay and message overhead for different cache sizes. Here, the number of nodes is set to be constant and the simulation time is set to 1800s.

Figure 8.2 shows the effect of cache hit ratio as a function of cache size. As it could be expected, all schemes exhibit better cache hit ratio with increasing cache size. This is because the required data items are likely to be found in the local cache due to increased cache size. As can be seen from figure the cache hit ratio of the NC scheme is lowest. For a cache size of 400 KB the percentage of cache hit ratio is about 60% for VC, 28% for NC and about 38% for GC. This is because cooperate hit will be higher for VC based scheme since the coverage is higher in this scheme.

Figure 8.3 compares the average delay for different schemes under varying cache sizes. As it could be expected, the average delay is high for all the three schemes at lower cache sizes. The cache capacity of each node has direct impact on average delay. Another factor which reduces delay is the sharing of cached data item among neighboring nodes. These two factors play an important role in serving requests with less average delay. VC has the lowest delay although GC obtains a similar delay for bigger cache sizes. The proposed VC scheme shows an average delay of 0.25 s while GC and NC show an average delay of 0.35 and 0.45 s respectively. As the cache size increases the difference between the delay

decreases. The reason behind this is as the cache size increases, the nodes can access most of its required data from the local and cooperate cache. When the cache size is increased beyond 40 the query delay does not drop significantly.

Figure 8.4 represents the message overhead as a function of cache size. From the figure we can see that VC performs much better than NC and GC in terms of message overhead. VC and GC generate lower traffic than NC. This is because the number of messages needed for cache discovery in VC and GC is very less compared to NC which uses broadcasting for neighbor cache discovery. The number of messages processed by each node is increased gradually and remains almost constant for higher cache sizes. This is because as the cache size increases the query delay decreases and more number of requests are handled in the given time interval. Due to cache cooperation the cache hit ratio is increased for higher cache sizes and the mobile nodes gets the data from nearby nodes instead of far away data source. Therefore, the mobile nodes needs to process only small number of messages. However, NC caching has the worst message overhead due to message broadcast.

8.7.2 Effect of Number of Nodes

Figure 8.5–8.7 illustrates the effect of node density on various parameters. In this set of simulations, we study the behavior of the caching system by varying the number of nodes to measure the scalability of the algorithms. Figure 8.5 depicts the cache hit ratio as a function of the number of nodes in the network. In this simulation the cache size is kept constant and is taken as 40% of the total database size. Simulation time is set to 1800s. It is observed that the hit ratio of both the approaches, NC and GC is

lower compared to the hit ratio of the proposed VC approach. This is because as the node density increases the cache look up procedure used in both schemes are not able to resolve data request due to higher message overhead. The effect of access latency on the number of nodes is shown in Fig 8.6. The access latency is affected by the cache hit ratio as well as the data lookup process during a data request. As can be seen from the figure the average delay is low for all the approaches when the node density is lower. This is because when the number of nodes is low the cache resolution process is able to retrieve the data from the cache more efficiently. The difference in delay between the proposed VC scheme and other schemes become more significant when the node density is higher. Figure 8.7 shows the effect of node density on the message overhead for all the above mentioned approaches. The proposed VC based approach has significant reduction in number of messages for larger node density. This is because as the VC based scheme uses CDS based approach for cache look up, the number of messages processed at each node is less. For NC scheme the message overhead is increased significantly for higher node density. This is due to the broadcast based approach used to find data from the neighboring nodes. The message overhead using GC based approach also increases as the number of nodes increase. The increase is lower compared to NC based approach because they perform a limited flooding.

8.7.3 Effect of Mean Time between Requests

This set of simulation is conducted in order to evaluate the performance of different cooperative caching schemes when the query interval is changed. Figure 8.8 represents the cache hit ratio of as a function of varying time

between request queries. More queries are processed for lower time interval. For all the query intervals VC based approach have higher hit ratio compared to the other two schemes. This is because when the query interval is small more requests are generated and the cache replacement plays an important role in increasing the hit ratio. When the query interval is increased the hit ratio decreases because the data items stored in cache may become invalidate due to their expired TTL value. Figure 8.9 shows the average delay obtained for all the schemes for different query interval. VC based approach performs better than NC and GC schemes for all values of query interval. For small intervals, more number of requests is generated and the average delay is high. When the query delay is increased less number of requests is generated and the average delay drops due to the low message overhead. Figure 8.10 illustrates the message overhead as a function of mean time between requests. As the mean time between the requests is increased the message overhead is obviously decreased for all the three schemes. It can be seen that the proposed VC achieved low message overhead compared to NC and GC.

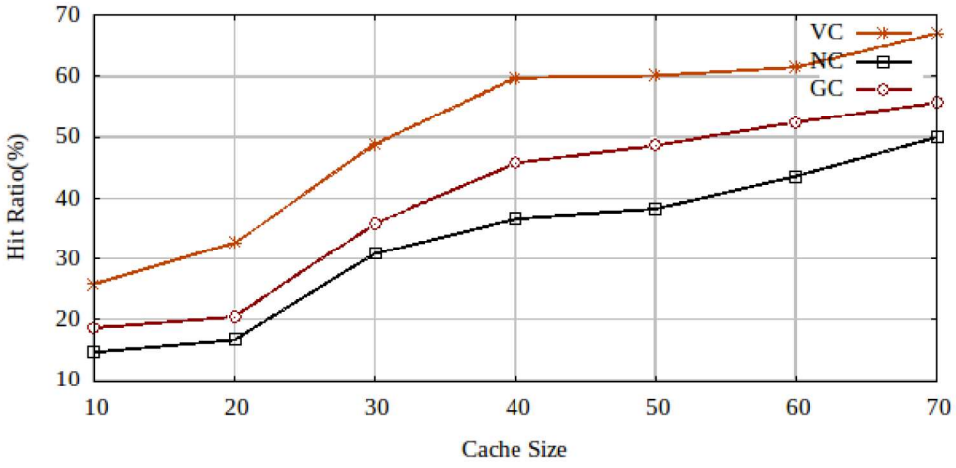

Figure 8.2: Effect of cache size on cache hit ratio

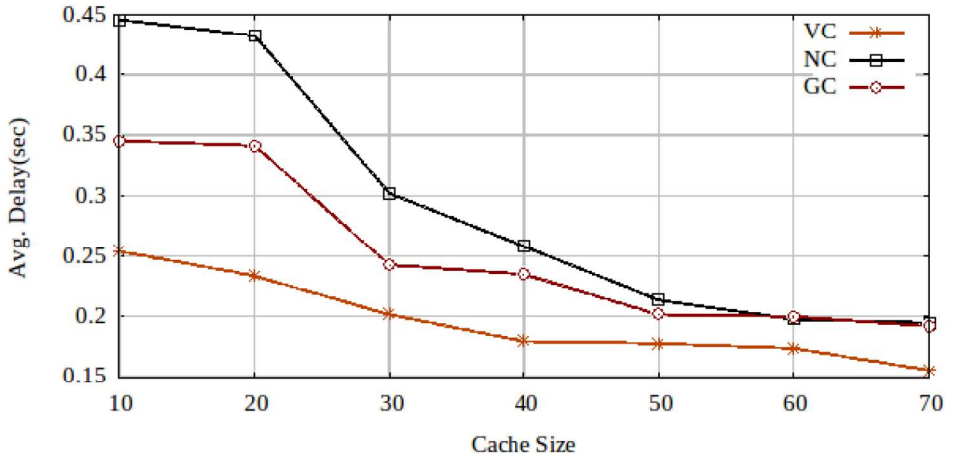

Figure 8.3: Effect of cache size on average delay

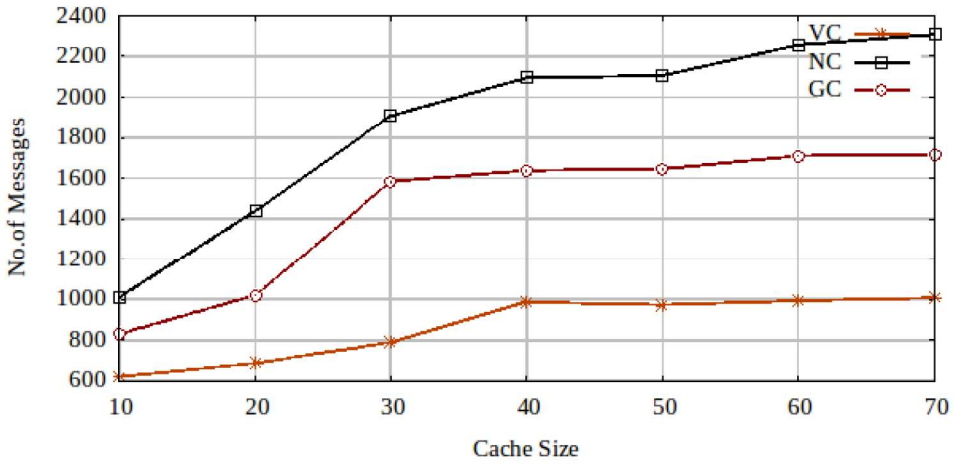

Figure 8.4: Effect of cache size on message overhead

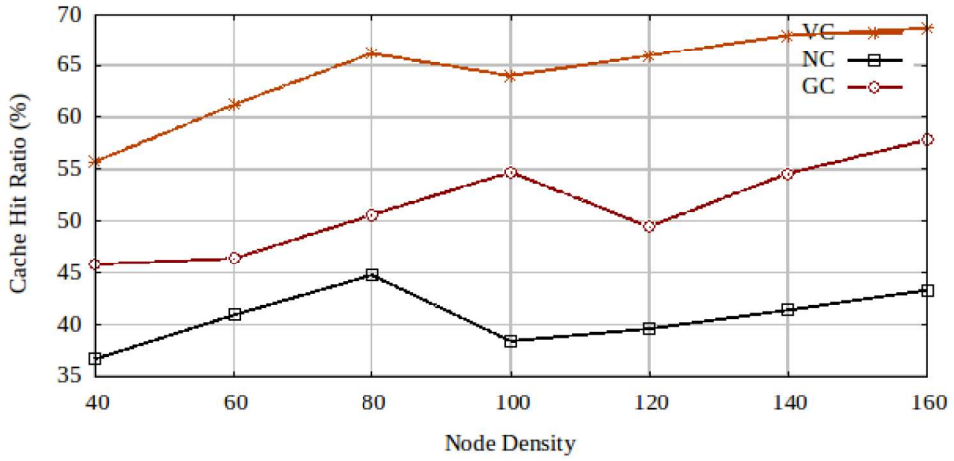

Figure 8.5: Effect of Node density on cache hit ratio

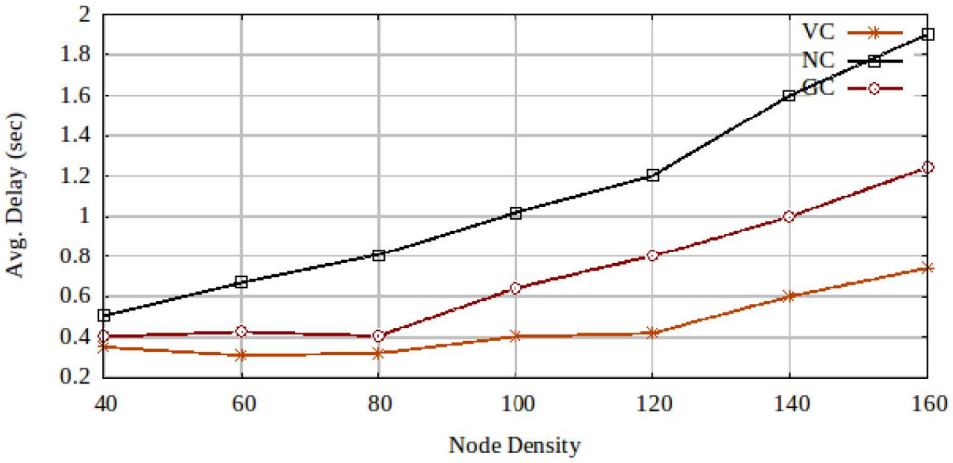

Figure 8.6: Effect of node density on average delay

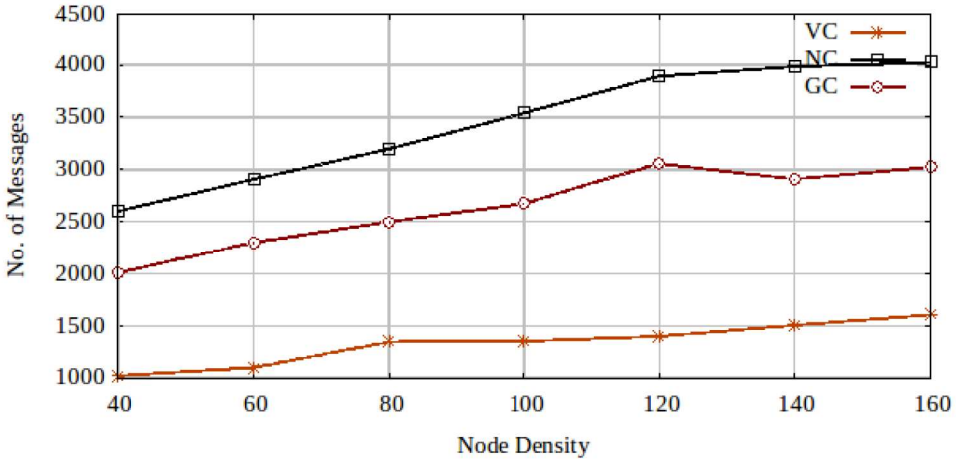

Figure 8.7: Effect of node density on number of messages

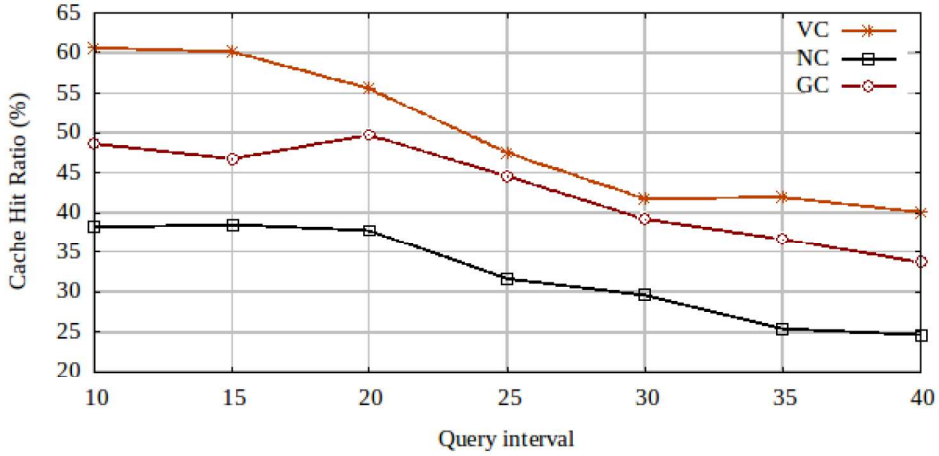

Figure 8.8: Effect of query interval on cache hit ratio

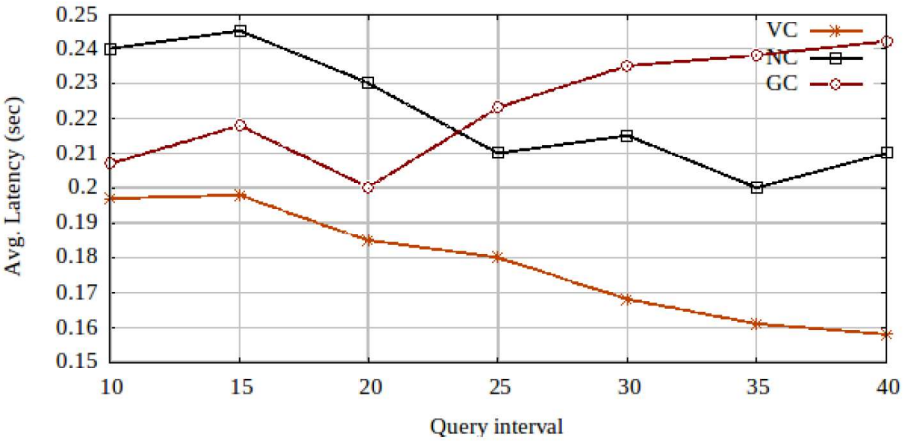

Figure 8.9: Effect of query interval on Avg. Latency

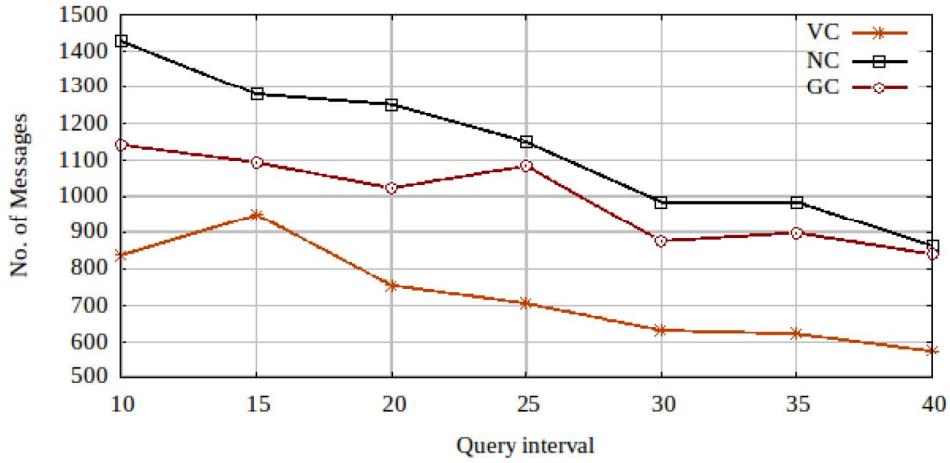

Figure 8.10: Effect of query interval on message overhead

8.8 Conclusion

This chapter introduced a new approach for cooperative caching, in mobile ad hoc networks that resolves data request based on CDS. The main advantage that distinguishes the proposed cooperative caching compared to other caching schemes is its ability to reduce the message overhead and delay in large scale networks. The new cooperative caching scheme forms a connective dominating set and the cache lookup process is reduced to nodes inside the CDS. The results obtained through simulations show that the proposed Virtual Backbone based cooperative caching significantly outperforms NC and GC in terms of message overhead, hit ratio and average delay.

Chapter 9

Conclusion and Future Directions

Contents

9.1 Conclusion

Mobile ad hoc networks have received significant attention among researchers in recent years. Innovations and technology advancements in mobile devices and wireless technology have enabled ad hoc networks to emerge as a preferred alternative for information sharing. MANETs offer great potential for a variety of new applications and better services for mobile users without any prior infrastructure. However, the unpredictable nature of the dynamically changing environment makes

high data accessibility a challenging problem in MANET. To meet this challenge, a number of data replication and caching protocols have been proposed in previous literature. Among the different methods proposed for improving data accessibility in MANETs, cooperative caching offered a better solution. This research is primarily focused on the design and development of novel cooperative caching techniques that can significantly reduce caching overhead and access delay. This thesis has covered three important issues in cooperative caching: Cache replacement, Cache placement and Cache discovery.

An exhaustive study of existing cooperative caching approaches revealed that there are several subtle issues on developing effective, robust, scalable and adaptive cooperative caching techniques which need to be studied and solved. The main conclusion from this study was that existing cooperative caching techniques pay less attention to the message overhead encountered during cooperative caching. Limited resources of wireless networks necessitates that the message overhead be kept low. An important resource in MANET is the individual battery levels of each node that limits the useful lifetime of the network. In ad hoc networks, there is a direct tradeoff between the number of messages transmitted and the amount of energy consumed by transmitting data. Therefore message overhead will result in reduced battery life and can significantly limit the lifetime of the network. Scalability is another important issue that has not been taken in to consideration in the design of existing cooperative caching protocols. When the number of mobile nodes increases, the communication overhead between the mobile nodes increases and the performance gets degraded.

The present thesis therefore addressed the above challenges in the

following way. The various cooperative caching techniques were evaluated using several metrics including the message overhead. The first part of the thesis concentrated on cache management and developed new algorithms for effective cache replacement and cache placement. It then focused on the message overhead question further and developed two techniques to reduce the message overheads, viz., location aided and energy efficient cache discovery. Finally, an adaptive virtual backbone based cache discovery protocol to address the issue of network scalability and message overhead was developed using the concept of a connected dominating set. The simulation results verified the effectiveness of all the above proposed methods. Summarized below are the major contributions made in this research work.

First, a new approach for cache replacement, referred to as E-LRU that can adapt dynamically to the changing access patterns by taking into consideration the short-term temporal correlation of references has been developed. After a careful study of the techniques reported in literature for cache replacement, it was found that most of the reported techniques for cooperative caching in MANET used LRU based replacement, which is based on the property of locality of reference. LRU replaces the cached data that have not been requested for the longest time. Thus it uses only the time of the last request to replace the data. Once a data is accessed it was given more cache life, even if it had been never referenced before. Thus LRU is not able to differentiate between data that had relatively frequent reference with that of infrequent reference. Hence, a variation of LRU that can adapt dynamically to the real time access patterns was introduced. The main idea was to keep a record of access history of recent two references and calculate the inter request arrival

time for each data item. The data items that were referenced only once were given more priority for replacement. For this LRU policy is used. If an item is referenced more than once, the inter request arrival time (IRT) between the recent two references is considered for eviction. The proposed approach outperforms LRU from the perspectives of cache hit ratio and average number of requests served.

The second area in which the research work concentrated was on cache placement. A cache placement scheme has been introduced to reduce duplication of data in neighboring nodes, by sharing and coordinating the cache state among neighboring nodes. A coordinated cache data placement algorithm based on CEP was presented. The CEP is taken as an indicator of the cache space contention at individual nodes. This scheme reduced replication of data by assuring that a copy of the data will be present in the cooperative cache, without any extra communication overhead. The simulation results show that the proposed scheme yields better performance in terms of cache hit ratio and average latency compared to independent cache data placement especially for applications with limited cache.

Third part of this research work concentrated on developing cache discovery protocols aimed at lowering message overhead and improving energy efficiency on mobile devices. Cache discovery is an essential part of cooperative caching. The major communication overhead occurs in this part of cooperative caching. Existing cache discovery algorithms has potential problems of high message overhead and energy consumption. Most of the reported techniques in cache discovery use either broadcasting or centralized approach for cache discovery. The major drawback of broadcast based approach is increased network contention with redun-

dant messages. On the other hand centralized approach in cache discovery is not suitable for ad hoc networks due to the mobility of nodes.

To cope with the limitations of above mentioned schemes, a distributed cache discovery protocol that reduces the message overhead was presented. A location based data discovery where the one hop neighbor list is divided in to two zones based on transmission range is adopted in this cache discovery scheme. This eliminates not only the flooding of data request into the network, but also reduces the overhead in processing tables. The simulation findings give a promising result based on the metrics of studies. To further improve the performance of the system an energy efficient cache discovery protocol was presented. The energy cost was minimized by dynamically adjusting the transmission range to reach its neighboring nodes, thereby saving power whenever possible. This approach helps in reducing the power consumption and hence prolongs the network life time. Simulations show that the scheme achieves lower power consumption compared to a scheme that uses broadcasting.

Finally, a scalable hierarchical approach which uses a virtual backbone using Connected Dominating Set for cache discovery was introduced. Once a virtual back bone is formed, all cache discovery information is limited among the virtual backbone member set thereby reducing the message overhead dramatically. The two phases in this scheme were: Backbone formation phase and a distributed cache lookup phase. In first phase, a dynamic virtual backbone is formed using a Connective Dominating Set and each node in the network is either a part of the backbone or one hop away from at least one of the backbone nodes. The second phase was the distributed cache look up phase, in which data request message is forwarded to nodes inside the dominating set for data dis-

covery. The cache lookup protocol based on virtual backbone attempts to achieve complete network coverage by ensuring that every node receives the request. The simulation results indicate that this approach can greatly reduce the number of messages and the cache overhead while maintaining high cache hit ratio and reduced delay.

9.2 Future Directions

The present work was an attempt to develop cooperative caching techniques for improving data availability in mobile ad hoc networks. The work focused on the design and development of scalable and energy efficient cooperative caching schemes. Following are some aspects that can be taken up for future work.

The simulation model developed assumes mobile nodes of equal cache size and processing capabilities. The developed model could be further toned to its perfection if heterogeneous set of mobile nodes were included.

In the proposed work the source of data is data server. This model could be extended by considering more sophisticated models where data could be generated by all the mobile nodes and could be accessed by any of these nodes.

The cache consistency model developed in this work is based on weak consistency model. Another extension to the current research work is to develop new algorithms to improve cache consistency.

The cache discovery schemes proposed in this work could be further improved by predicting the location of mobile users using accurate prediction models.

The proposed scalable cache look up framework exploits existing method to find the Connective Dominating Set. Better approximation algorithms for finding Connective Dominating Set could be further investigated.

The idea of cooperative caching could be explored to other application areas of mobile computing like vehicular networks, sensor networks etc.

Bibliography

1. [Adjih et al., 2002] Adjih, C., Jacquet, P., and Viennot, L. (2002). Computing connected dominated sets with multipoint relay.

2. [Agbaria et al., 2011a] Agbaria, A., Hugerat, M., and Friedman, R. (2011a). Efficient and reliable dissemination in mobile Ad Hoc networks by location extrapolation. Journal of Computer Networks and Communications.

3. [Agbaria et al., 2011b] Agbaria, A., Hugerat, M., and Friedman, R. (2011b). Efficient and reliable dissemination in mobile Ad Hoc networks by location extrapolation. Journal of Computer Networks and Communications.

4. [Agrawal et al., 1998] Agrawal, P. (1998). Battery power sensitive video processing in wireless networks. Proceedings IEEE PIMRC98.

5. [Ahmed and Shirmohammadi, 2007] Ahmed, D. T. and Shirmohammadi, S. (2007). Design issues of peer-to-peer systems for wireless ad hoc networks. Networking, 2007. ICN'07. Sixth International Conference on. IEEE.

6. [Alzoubi et al., 2002] Alzoubi, K. M., Wan, P.-J., and Frieder, O. (2002). Message-optimal connected dominating sets in mobile ad hoc networks. Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing. ACM.

7. [Anastasi et al., 2004] Anastasi, G., Conti, M., and Gregori, E. (2004). IEEE 802.11 ad hoc networks: protocols, performance and open issues. *Mobile Ad hoc networking* 69-116.

8. [Anderson et al., 1995] Anderson, T. E. (1995). Serverless network file systems. ACM SIGOPS Operating Systems Review Vol. 29. No. 5. ACM.

9. [Artail et al., 2008] Artail, H. (2008). COACS: a cooperative and adaptive caching system for MANETs. Mobile Computing, IEEE Transactions on 7.8 : 961-977.

10. [Atsan et al., 2009] Atsan, E., Altinbuken, D., and Ozkasap, O. (2009). SCALAR data replication performance in mobile ad hoc applications. Computer and Information Sciences, 2009. ISCIS 2009. 24th International Symposium on. IEEE.

11. [Bae and Olariu, 2010] Bae, I.-H. and Olariu, S. (2010). Design and evaluation of a fuzzy cooperative caching scheme for MANETs. Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference on. IEEE.

12. [Barbará, 1999] Barbará, D. (1999). Mobile computing and databases-a survey. Knowledge and Data Engineering, IEEE Transactions on 11.1: 108-117.

13. [Baronti et al., 2007] Baronti, P. (2007). Wireless sensor networks: A survey on the state of the art and the 802.15 4 and ZigBee standards. *Computer communications* 30.7 : 1655-1695.

14. [Basagni et al., 2004] Basagni, S. (2004). eds. *Mobile ad hoc networking.* John Wiley & Sons.

15. [Bisdikian, 2001] Bisdikian, C. (2001). An overview of the Bluetooth wireless technology. *IEEE Commun Mag* 39.12 : 86-94.

16. [Bisignano et al., 2003] Bisignano, M. (2003). Experience: a JXTA middleware for mobile ad-hoc networks. Peer-to-Peer Computing, 2003.(P2P 2003). Proceedings. Third International Conference on. IEEE.

17. [Bononi et al., 1999] Bononi, L., Conti, M., and Donatiello, L. (1999). A distributed contention control mechanism for power saving in random-access ad-hoc wireless local area networks. Mobile Multimedia Communications, 1999.(MoMuC'99) 1999 IEEE International Workshop on. IEEE.

18. [Bononi et al., 2001] Bononi, L., Conti, M., and Donatiello, L. (2001). A distributed mechanism for power saving in IEEE 802.11 wireless LANs. Mobile Networks and Applications 6.3: 211-222.

19. [Borst et al., 2010] Borst, S., Gupta, V., and Walid, A. (2010). Distributed caching algorithms for content distribution networks. INFOCOM, 2010 Proceedings IEEE.

20. [Bose and Morin, 1999] Bose, P. and Morin, P. (1999). Online routing in triangulations. Algorithms and Computation. Springer Berlin Heidelberg, 113-122.

21. [Broch et al., 1998] Broch, J. (1998). A performance comparison of multi-hop wireless ad hoc network routing protocols. Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking. ACM.

22. [Butenko et al., 2004] Butenko, S. (2004). A new heuristic for the minimum connected dominating set problem on ad hoc wireless net-

works. Recent developments in cooperative control and optimization. Springer US, 61-73.

23. [Carle and Simplot-Ryl, 2004a] Carle, J. and Simplot-Ryl, D. (2004a). Energy-efficient area monitoring for sensor networks. Computer 37.2: 40-46.

24. [Carle and Simplot-Ryl, 2004b] Carle, J. and Simplot-Ryl, D. (2004b). Energy-efficient area monitoring for sensor networks. Computer 37.2: 40-46.

25. [Cecilia et al., 2002] Cecilia, M. (2002). XMIDDLE: a data-sharing middleware for mobile computing. Wireless Personal Communications 21.1: 77-103.

26. [Chand et al., 2006a] Chand, N., Joshi, R. C., and Misra, M. (2006a). Efficient cooperative caching in ad hoc networks. Communication System Software and Middleware, 2006. Comsware 2006. First International Conference on. IEEE.

27. [Chand et al., 2006b] Chand, N., Joshi, R. C., and Misra, M. (2006b). Efficient cache replacement in mobile environment using data profit. Parallel and Distributed Systems, 2006. ICPADS 2006. 12th International Conference on. Vol. 1. IEEE.

28. [Chand et al., 2007] Chand, N., Joshi, R. C., and Misra, M. (2007). Cooperative caching strategy in mobile ad hoc networks based on clusters. Wireless Personal Communications 43.1: 41-63.

29. [Chang and Tassiulas, 2000] Chang, J.-H. and Tassiulas, L. (2000). Energy conserving routing in wireless ad-hoc networks. INFOCOM

2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE. Vol. 1. IEEE.

30. [Chankhunthod et al., 1995] Chankhunthod, A. (1995). A hierarchical internet object cache. No. CU-CS-766-95. COLORADO UNIV AT BOULDER DEPT OF COMPUTER SCIENCE.

31. [Chauhan et al., 2011] Chauhan, N., Awasthi, L. K., and Chand, N. (2011). Global cooperative caching for Wireless Sensor Networks. Information and Communication Technologies (WICT), 2011 World Congress on. IEEE.

32. [Chen and Xiao, 2006] Chen, H. and Xiao, Y. (2006). Cache access and replacement for future wireless Internet. Communications Magazine, IEEE 44.5 : 113-123.

33. [Chen and Xiao, 2007] Chen, H. and Xiao, Y. (2007). On-bound selection cache replacement policy for wireless data access. Computers, IEEE Transactions on 56.12:1597-1611.

34. [Chidambaram et al., 2012] Chidambaram, L. M. (2012). Meloc: Memory and location optimized caching model for small mobile ad hoc networks. Mobile Data Management (MDM), 2012 IEEE 13th International Conference on. IEEE.

35. [Chiu and Young, 2009] Chiu, G.-M. and Young, C.-R. (2009). Exploiting in-zone broadcasts for cache sharing in mobile ad hoc networks. Mobile Computing, IEEE Transactions on 8.3: 384-397.

36. [Chlamtac et al., 2003] Chlamtac, I., Conti, M., , and Liu, J. J.-N. (2003). Mobile ad hoc networking: imperatives and challenges. *Ad*

hoc networks 1.1, 13-64.

37. [Cho, 2011] Cho, H. (2011). Path Conscious Cooperative Caching for Mobile Ad Hoc Networks.

38. [Cho et al., 2003] Cho, J. (2003a). Neighbor caching in multi-hop wireless ad hoc networks. Communications Letters, IEEE 7.11: 525-527.

39. [Chow et al., 2004] Chow, C.-Y., Leong, H. V., and Chan, A. T. (2004a). Group-based cooperative cache management for mobile clients in a mobile environment. Parallel Processing, 2004. ICPP 2004. International Conference on. IEEE.

40. [Chow et al., 2005] Chow, C.-Y., Leong, H. V., and Chan, A. T. (2005). Distributed group-based cooperative caching in a mobile broadcast environment. Proceedings of the 6th international conference on Mobile data management. ACM.

41. [Chow et al., 2007] Chow, C.-Y., Leong, H. V., and Chan, A. T. (2007). GroCoca: Group-based peer-to-peer cooperative caching in mobile environment. Selected Areas in Communications, IEEE Journal on 25.1: 179-191.

42. [Chow et al., 2004b] Chow, C.-Y., Leong, H. V., and Chan, A. T. S. (2004b). Group-based cooperative cache management for mobile clients in a mobile environment. Parallel Processing, 2004. ICPP 2004. International Conference on. IEEE.

43. [Clark et al., 1990] Clark, C., Colburn, J., and Johnson, D. (1990). Unit Disk Graphs, Discrete Mathematics 86(1-3) 165-177.

44. [Corson et al., 1999a] Corson, M. S., Macker, J. P., and Cirincione, G. H. (1999a). Internet-based mobile ad hoc networking. *Internet Computing, IEEE 3.4* : 63-70.

45. [Corson et al., 1999b] Corson, M. S., Macker, J. P., and Cirincione, G. H. (1999b). Internet-based mobile ad hoc networking. Internet Computing, IEEE 3.4:63-70.

46. [Dai and Wu, 2003] Dai, F. and Wu, J. (2003). Distributed dominate pruning in ad hoc wireless networks. Proc. ICC.

47. [Dai and Wu, 2004a] Dai, F. and Wu, J. (2004a). An extended localized algorithm for connected dominating set formation in ad hoc wireless networks. Parallel and Distributed Systems, IEEE Transactions on 15.10: 908-920.

48. [Dai and Wu, 2004b] Dai, F. and Wu, J. (2004b). An extended localized algorithm for connected dominating set formation in ad hoc wireless networks. Parallel and Distributed Systems, IEEE Transactions on 15.10: 908-920.

49. [Dai et al., 2012] Dai, J. (2012). Collaborative hierarchical caching with dynamic request routing for massive content distribution. INFOCOM, 2012 Proceedings IEEE. IEEE.

50. [Dar et al., 1996] Dar, S. (1996). Semantic data caching and replacement. VLDB. Vol. 96.

51. [Das and Bharghavan, 1997] Das, B. and Bharghavan, V. (1997). Routing in ad-hoc networks using minimum connected dominating

sets. Communications, 1997. ICC'97 Montreal, Towards the Knowledge Millennium. 1997 IEEE International Conference on. Vol. 1. IEEE.

52. [Denko et al., 2009] Denko, M. K. (2009). Cross-layer based data management in mobile ad hoc networks. Global Telecommunications Conference, 2009, GLOBECOM 2009. IEEE.

53. [Denko and Tian, 2006] Denko, M. K. and Tian, J. (2006). Cooperative caching with adaptive prefetching in mobile ad hoc networks. Wireless and Mobile Computing, Networking and Communications, 2006.(WiMob'2006). IEEE International Conference on. IEEE.

54. [Denko and Tian, 2008] Denko, M. K. and Tian, J. (2008). Cross-layer design for cooperative caching in mobile ad hoc networks. Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE. IEEE.

55. [Derhab and Badache, 2009] Derhab, A. and Badache, N. (2009). Data replication protocols for mobile ad-hoc networks: a survey and taxonomy. Communications Surveys & Tutorials, IEEE 11.2 : 33-51.

56. [Dimokas et al., 2011] Dimokas, N. (2011). High performance, low complexity cooperative caching for wireless sensor networks. Wireless Networks 17.3:717-737.

57. [Dimokas et al., 2008] Dimokas, N., Katsaros, D., and Manolopoulos, Y. (2008). Cooperative caching in wireless multimedia sensor networks. Mobile Networks and Applications 13.3-4: 337-356.

58. [Ding et al., 2003] Ding, M., Cheng, X., and Xue, G. (2003). Aggregation tree construction in sensor networks. Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th. Vol. 4. IEEE.

59. [Dodero and Gianuzzi, 2006] Dodero, G. and Gianuzzi, V. (2006). Saving Energy and Reducing Latency in MANET File Access. ICDCS Workshops.

60. [Du and Gupta, 2004] Du, Y. and Gupta, S. (2004). Handbook of mobile computing. CRC Press.

61. [Du and Gupta, 2005] Du, Y. and Gupta, S. K. S. (2005). COOP-A cooperative caching service in MANETs. Autonomic and Autonomous Systems and International Conference on Networking and Services, 2005. ICAS-ICNS 2005, Joint International Conference on. IEEE.

62. [ElBatt et al., 2000] ElBatt, T. A. (2000). Power management for throughput enhancement in wireless ad-hoc networks. Communications, 2000. ICC 2000, 2000 IEEE International Conference on. Vol. 3. IEEE.

63. [Fan et al., 2000] Fan, L. (2000). Summary cache: a scalable wide-area web cache sharing protocol. IEEE/ACM Transactions on Networking (TON) 8.3: 281-293.

64. [Fan et al., 2013] Fan, X. (2013). Gossip-based cooperative caching for mobile applications in mobile wireless networks. Journal of Parallel and Distributed Computing 73.5: 653-663.

65. [Finn, 1987] Finn, G. G. (1987). Routing and Addressing Problems in Large Metropolitan-Scale Internetworks. ISI Research Report.

66. [Fiore et al., 2009] Fiore, M. (2009). To cache or not to cache?. INFOCOM 2009, IEEE.

67. [Fiore et al., 2011a] Fiore, M., Casetti, C., and Chiasserini, C. (2011a). Caching strategies based on information density estimation in wireless ad hoc networks. Vehicular Technology, IEEE Transactions on 60.5: 21942208.

68. [Fiore et al., 2011b] Fiore, M., Casetti, C., and Chiasserini, C. (2011b). Caching strategies based on information density estimation in wireless ad hoc networks. Vehicular Technology, IEEE Transactions on 60.5: 2194-2208.

69. [Flinn and Satyanarayanan, 1999] Flinn, J. and Satyanarayanan, M. (1999). Powerscope: A tool for profiling the energy usage of mobile applications. Mobile Computing Systems and Applications, 1999. Proceedings. WMCSA'99. Second IEEE Workshop on. IEEE.

70. [Fok et al., 2004] Fok, C.-L., Roman, G.-C., and Hackmann, G. (2004). A lightweight coordination middleware for mobile computing. Coordination Models and Languages. Springer Berlin Heidelberg.

71. [Freebersyser et al., 2001] Freebersyser, J. A., , and Leiner, B. (2001). A DoD perspective on mobile ad hoc networks. *Ad hoc networking*. Addison-Wesley Longman Publishing Co., Inc.

72. [Gao et al., 2006] Gao, Q. (2006). Radio range adjustment for energy efficient wireless sensor networks. Ad hoc networks 4.1: 75-82.

73. [Gerla et al., 1999] Gerla, M., Tang, K., and Bagrodia, R. (1999). TCP performance in wireless multi-hop networks. Mobile Computing

Systems and Applications, 1999. Proceedings. WMCSA'99 Second IEEE Workshop on. IEEE.

74. [Giuseppe et al., 2009] Giuseppe, A. (2009). Design and performance evaluation of a transport protocol for ad hoc networks. The Computer Journal 52.2 : 186-209.

75. [Goel, 2002] Goel, S. K. (2002). et al., Efficient peer-to-peer data dissemination in mobile ad-hoc networks. Parallel Processing Workshops, 2002, Proceedings. International Conference on. IEEE.

76. [Goldsmith and Wicker, 2002] Goldsmith, A. J. and Wicker, S. B. (2002). Design challenges for energy-constrained ad hoc wireless networks. Wireless Communications, IEEE 9.4: 8-27.

77. [Gupta et al., 2005] Gupta, K. S., Richard, G., and Schwiebert, L. (2005). Fundamentals of mobile and pervasive computing. New York: McGraw-Hill.

78. [Haas and Pearlman, 2001] Haas, Z. J. and Pearlman, M. R. (2001). The performance of query control schemes for the zone routing protocol. IEEE/ACM Transactions on Networking (TON) 9.4: 427-438.

79. [Hadim et al., 2006] Hadim, S., Al-Jaroodi, J., and Mohamed, N. (2006). Middleware issues and approaches for mobile ad hoc networks. *The IEEE Consumer Communications and Networking Conf.* (CCNC 2006).

80. [Hadim and Mohamed, 2006] Hadim, S. and Mohamed, N. (2006). Middleware for wireless sensor networks: A survey. Communication

System Software and Middleware, 2006. Comsware 2006. First International Conference on. IEEE.

81. [Hara, 2001] Hara, T. (2001). Effective replica allocation in ad hoc networks for improving data accessibility. In Proc. IEEE Infocom 2001, volume 3, pages 1568–1576, April.

82. [Hara, 2003a] Hara, T. (2003). Replica allocation methods in ad hoc networks with data update. Mobile Networks and Applications, 8(4):343–354.

83. [Hara, 2005] Hara, T. (2005). Strategies for data location management in mobile ad hoc networks. Parallel and Distributed Systems, 2005. Proceedings 11th International Conference on. Vol. 1. IEEE.

84. [Hara et al., 2003] Hara, T., Loh, Y.-H., and Nishio, S. (2003). Data replication methods based on the stability of radio links in ad hoc networks. In Proc. 14th International Workshop on Database and Expert Systems Applications (DEXA '03), pages 969–973.

85. [Hara and Madria, 2006] Hara, T. and Madria, S. K. (2006). Data replication for improving data accessibility in ad hoc networks. Mobile Computing, IEEE Transactions on 5.11: 1515-1532.

86. [Hefeeda and Noorizadeh, 2008] Hefeeda, M. and Noorizadeh, B. (2008). Cooperative caching: The case for P2P traffic. Local Computer Networks, 2008. LCN 2008, 33rd IEEE Conference on. IEEE.

87. [Hightower and Borriello, 2001] Hightower, J. and Borriello, G. (2001). Location systems for ubiquitous computing. Computer 34.8: 57-66.

88. [Hirsch and Madria, 2010] Hirsch, D. and Madria, S. K. (2010). A resource-efficient adaptive caching scheme for mobile ad hoc networks. Reliable Distributed Systems, 2010 29th IEEE Symposium on. IEEE.

89. [Holland and Vaidya, 2002] Holland, G. and Vaidya, N. (2002). Analysis of TCP performance over mobile ad hoc networks. Wireless Networks 8.2/3 : 275-288.

90. [Hsu, 2002] Hsu, T. H. (2002). Power-Saving Protocols for IEEE 802.11- Based Multihop Ad hoc Networks, IEEE INFOCOM.

91. [Huang et al., 2003] Huang, J.-L., Chen, M.-S., and Peng, W.-C. (2003). Exploring group mobility for replica data allocation in a mobile environment. Proceedings of the twelfth international conference on Information and knowledge management. ACM.

92. [Ilyas, 2014] Ilyas, M. (2014). ed. The handbook of ad hoc wireless networks. CRC press.

93. [Jaikaeo and Shen, 2002] Jaikaeo, C. and Shen, C.-C. (2002). Adaptive backbone-based multicast for ad hoc networks. Communications, 2002. ICC 2002, IEEE International Conference on. Vol. 5. IEEE.

94. [Jones et al., 2001] Jones, C. E. (2001). A survey of energy efficient network protocols for wireless networks. Wireless networks 7.4: 343-358.

95. [Karger, 1999] Karger, D. (1999). Web caching with consistent hashing. Computer Networks 31.11 : 12031213.

96. [Ke, 2010] Ke, H. (2010). Cooperative caching algorithm based on grouping nodes in mobile ad hoc networks. Information and Automation (ICIA), 2010 IEEE International Conference on. IEEE.

97. [Khabbazian et al., 2012] Khabbazian, M., Blake, I. F., and Bhargava, V. K. (2012). Local broadcast algorithms in wireless ad hoc networks: Reducing the number of transmissions. Mobile Computing, IEEE Transactions on 11.3: 402-413.

98. [Klemm et al., 2003] Klemm, A., Lindemann, C., and Waldhorst, O. P. (2003). A special-purpose peer-to-peer file sharing system for mobile ad hoc networks. Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th. Vol. 4. IEEE.

99. [Korupolu and Dahlin, 2002] Korupolu, M. R. and Dahlin, M. (2002). Coordinated placement and replacement for large-scale distributed caches. Knowledge and Data Engineering, IEEE Transactions on 14.6: 1317-1329.

100. [Korupolu et al., 2001] Korupolu, M. R., Plaxton, C. G., and Rajaraman, R. (2001). Placement algorithms for hierarchical cooperative caching. Journal of Algorithms 38.1: 260-302.

101. [Kravets and Krishnan, 1998] Kravets, R. and Krishnan, P. (1998). Power management techniques for mobile communication. Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking. ACM.

102. [Kravets et al., 1999] Kravets, R., Schwan, K., and Calvert, K. (1999). Power-aware communication for mobile computers. Mobile Multi-

media Communications, 1999.(MoMuC'99) 1999 IEEE International Workshop on. IEEE.

103. [Krishnan et al., 2000] Krishnan, P., Raz, D., and Shavitt, Y. (2000). The cache location problem. IEEE/ACM Transactions on Networking (TON) 8.5: 568-582.

104. [Ku et al., 2002] Ku, W.-S., Zimmermann, R., and Wan, C.-N. (2002). Nearest Neighbor Queries with Data Sharing in Mobile Environments. Stojmenovic, Ivan, Position-based routing in ad hoc networks. Communications Magazine, IEEE 40.7: 128-134.

105. [Kumar et al., 2010] Kumar, A., Sarje, A. K., and Misra, M. (2010). Prioritised Predicted Region based Cache Replacement Policy for location dependent data in mobile environment. International Journal of Ad Hoc and Ubiquitous Computing 5.1: 56-67.

106. [Lai et al., 2004] Lai, K., Tari, Z., and Bertok, P. (2004). Mobility aware cache replacement for location dependent information services. Technical Report T R-04-04 (R MI T School of CS & IT).

107. [Lau et al., 2002] Lau, W. H., Kumar, O. M., and Venkatesh, S. (2002). A cooperative cache architecture in support of caching multimedia objects in MANETs. Proceedings of the 5th ACM international workshop on Wireless mobile multimedia. ACM.

108. [Li et al., 2003] Li, M., Wan, P.-J., and Frieder, O. (2003). Coverage in wireless ad hoc sensor networks. Computers, IEEE Transactions on 52.6: 753-763.

109. [Li et al., 2007] Li, W., Chan, E., and Chen, D. (2007). Energy-efficient cache replacement policies for cooperative caching in mobile ad hoc network. Wireless Communications and Networking Conference, 2007. WCNC 2007. IEEE. IEEE.

110. [Lim et al., 2003] Lim, S. (2003). A novel caching scheme for internet based mobile ad hoc networks. Computer Communications and Networks, 2003. ICCCN 2003. Proceedings. The 12th International Conference on IEEE.

111. [Lim et al., 2006] Lim, S. (2006). A novel caching scheme for improving internet-based mobile ad hoc networks performance. Ad Hoc Networks 4.2: 225-239.

112. [Liu and Feng, 2009] Liu, W.-J. and Feng, K.-T. (2009). Greedy routing with anti-void traversal for wireless sensor networks. Mobile Computing, IEEE Transactions on 8.7: 910-922.

113. [Ma and Jamalipour, 2010] Ma, Y. and Jamalipour, A. (2010). A cooperative cache-based content delivery framework for intermittently connected mobile ad hoc networks. Wireless Communications, IEEE Transactions on 9.1: 366-373.

114. [Macker et al., 2007] Macker, J. (2007). Evaluation of distributed cover set algorithms in mobile ad hoc network for simplified multicast forwarding. ACM SIGMOBILE Mobile Computing and Communications Review 11.3: 1-11.

115. [Madria et al., 2000] Madria, S. K. (2000). Data organization issues for location-dependent queries in mobile computing. Springer Berlin Heidelberg.

116. [Magdalene et al., 2008] Magdalene, M. J. (2008). Network Distance Based Cache Replacement Policy for Location Dependent Data in Mobile Environment. Mobile Data Management Workshops, 2008. MDMW.

117. [Meguerdichianet et al., 2001] Meguerdichianet, S. (2001). Localized algorithms in wireless ad-hoc networks: location discovery and sensor exposure. Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing. ACM.

118. [Meier and Cahill, 2002] Meier, R. and Cahill, V. (2002). Steam: Event-based middleware for wireless ad hoc networks. Distributed Computing Systems Workshops, 2002. Proceedings. 22nd International Conference on. IEEE.

119. [Menaud et al., 1998] Menaud, J.-M., Issarny, V., and Banâtre, M. (1998). A new protocol for efficient cooperative transversal web caching. Springer Berlin Heidelberg.

120. [Ming et al., 2012] Ming, Z., Xu, M., and Wang, D. (2012). Age-based cooperative caching in information-centric networks. Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on. IEEE.

121. [Mingozzi, 2002] Mingozzi, E. (2002). QoS support by the HiperLAN/2 MAC protocol: a performance evaluation. *Cluster Computing* 5.2: 145-155.

122. [Minh and Bich, 2011] Minh, T. N. T. and Bich, T. D. T. (2011). An efficient model for cooperative caching in mobile information systems.

Advanced Information Networking and Applications (WAINA), 2011 IEEE Workshops of International Conference on. IEEE.

123. [Miranda et al., 2005] Miranda, H. (2005). A Stateless Neighbour-Aware Cooperative Caching Protocol for Ad-Hoc Networks.

124. [Mohammad and Mahgoub, 2004] Mohammad, I. and Mahgoub, I. (2004). eds. Mobile computing handbook. CRC Press.

125. [Mohapatra et al., 2003] Mohapatra, P., Li, J., and Gui, C. (2003). QoS in mobile ad hoc networks. *IEEE Wireless Communications* 10.3: 44-53.

126. [Murphy et al., 2001] Murphy, A. L., Picco, G. P., and Roman, G. R. U. I. A. C. A. T. A. L. I. N. (2001). Lime: A middleware for physical and logical mobility. Distributed Computing Systems, 2001. 21st International Conference on.. IEEE.

127. [Murthy and Manoj, 2004] Murthy, C. S. R. and Manoj, B. S. (2004). Ad hoc wireless networks: Architectures and protocols, Pearson education.

128. [Musolesi et al., 2005] Musolesi, M., Mascolo, C., and Hailes, S. (2005). Emma: Epidemic messaging middleware for ad hoc networks. *Personal and Ubiquitous Computing* 10.1 : 28-36.

129. [Naderan et al., 2013] Naderan, M., Dehghan, M., and Pedram, H. (2013). Primal and dual-based algorithms for sensing range adjustment in WSNs. The Journal of Supercomputing 64.2: 310-330.

130. [Naik and Wei, 2001] Naik, K. and Wei, D. S. (2001). Software implementation strategies for power-conscious systems. Mobile Networks and Applications 6.3: 291-305.

131. [nete et al., 2009] nete, F. J. G.-C., Casilari, E., and no Cabrera, A. T. (2009). Proposal and evaluation of a caching scheme for ad hoc networks. Ad-Hoc, Mobile and Wireless Networks. Springer Berlin Heidelberg, 366-372.

132. [nete anf Eduardo Casilari and no Cabrera, 2012] nete anf Eduardo Casilari, F. J. G.-C. and no Cabrera, A. T. (2012). A cross layer interception and redirection cooperative caching scheme for MANETs. EURASIP Journal on Wireless Communications and Networking 2012.1:1-21.

133. [Nuggehalli et al., 2006] Nuggehalli, P. (2006). Efficient cache placement in multi-hop wireless networks. Networking, IEEE/ACM Transactions on 14.5: 1045-1055.

134. [Nuggehalli et al., 2003] Nuggehalli, P., Srinivasan, V., and Chiasserini, C.-F. (2003). Energy-efficient caching strategies in ad hoc wireless networks. Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing. ACM.

135. [Padmanabhan et al., 2006] Padmanabhan, P. ., Gruenwald, L., Vallur, A., and Atiquzzaman, M. (2006). A Survey Of Data Replication Techniques For Mobile Ad-hoc Network Databases, Journal of Very Large Data Bases.

136. [Padmanabhan and Qiu, 2000] Padmanabhan, V. N. and Qiu, L. (2000). The content and access dynamics of a busy website: Find-

ings and implications. ACM SIGCOMM Computer Communication Review 30.4: 111-123.

137. [Pandey, 2010] Pandey, V. (2010). A review on data aggregation techniques in wireless sensor network. *Journal of Electronic and Electrical Engineering 1.2.*

138. [Papadopoulos et al., 2010] Papadopoulos, F. (2010). Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces. INFOCOM, 2010 Proceedings IEEE. IEEE.

139. [Psounis and Prabhakar, 2002a] Psounis, K. and Prabhakar, B. (2002). Efficient randomized web-cache replacement schemes using samples from past eviction times. Networking, IEEE/ACM Transactions on 10.4: 441-454.

140. [Psounis and Prabhakar, 2002b] Psounis, K. and Prabhakar, B. (2002). Efficient randomized web-cache replacement schemes using samples from past eviction times. Networking, IEEE/ACM Transactions on 10.4: 441-454, ContentPlace: Social-aware Data Dissemination in Opportunistic Networks.

141. [Qadri and Fleury, 2012] Qadri, N. N. and Fleury, M. (2012). Overview of Mobile Ad Hoc Networks and their Modeling. Streaming Media with Peer- to-Peer Networks: Wireless Perspectives: Wireless Perspectives: 96.333-1344.

142. [Raghavendra and Singh, 1997] Raghavendra, C. S. and Singh, S. (1997). PAMAS-Power Aware Multi-Access protocol with Signalling for.

143. [Ramaswamy and Liu, 2004] Ramaswamy, L. and Liu, L. (2004). An expiration age-based document placement scheme for cooperative web caching. Knowledge and Data Engineering, IEEE Transactions on 16.5: 585-600.

144. [Rao et al., 2012] Rao, A., Kumar, P., and Chauhan, N. (2012). EDGC: Efficient Dynamic Group Caching Technique for Mobile Ad hoc Networks. International Journal of Computer Applications 41.13: 25-30.

145. [Rappaport, 1996] Rappaport, T. S. (1996). Wireless communications: principles and practice. Vol. 2. New Jersey: Prentice hall PTR.

146. [Ren and Dunham, 2000] Ren, Q. and Dunham, M. H. (2000). Using semantic caching to manage location dependent data in mobile computing. Proceedings of the 6th annual international conference on Mobile computing and networking. ACM.

147. [Repantis and Kalogeraki, 2010] Repantis, T. and Kalogeraki, V. (2010). Data dissemination and query routing in mobile peer-to-peer networks. Networking and Telecommunications: Concepts, Methodologies, Tools, and Applications: Concepts, Methodologies, Tools, and Applications: 371.

148. [Rodoplu and Meng, 1999] Rodoplu, V. and Meng, T. H. (1999). Minimum energy mobile wireless networks. Selected Areas in Communications, IEEE Journal on 17.8: 1.

149. [Salem et al., 2013] Salem, A., Abu, O., Alhmiedat, T., and Samara, G. (2013). Cache discovery policies of MANET. arXiv preprint arXiv:1310.1552.

150. [Sanchez et al., 1999] Sanchez, M., Manzoni, P., and Haas, Z. J. (1999). Determination of critical transmission range in ad-hoc networks. Multiaccess, Mobility and Teletraffic in Wireless Communications: Volume 4. Springer US, 293-304.

151. [Scott et al., 1998] Scott, M. (1998). Adaptive web caching: towards a new global caching architecture. Computer Networks and ISDN systems 30.22: 2169-2177.

152. [Shah and Rabaey, 2002] Shah, R. C. and Rabaey, J. M. (2002). Energy aware routing for low energy ad hoc sensor networks. Wireless Communications and Networking Conference, 2002. WCNC2002. 2002 IEEE. Vol. 1 IEEE.

153. [Shen et al., 2004] Shen, H. (2004). Cooperative caching with optimal radius in hybrid wireless networks. Networking 2004. Springer Berlin Heidelberg.

154. [Shi et al., 2008] Shi, Q. Q. (2008). A Modified Greedy Distance Routing Algorithm for Wireless Sensor Networks. Microwave Conference, 2008 China-Japan Joint. IEEE.

155. [Singh et al., 1998] Singh, S., Woo, M., and Raghavendra, C. S. (1998). Power-aware routing in mobile ad hoc networks. Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking. ACM.

156. [Stann and Heidemann, 2003] Stann, F. and Heidemann, J. (2003). RMST: Reliable data transport in sensor networks. Sensor Network Protocols and Applications, 2003, Proceedings of the First IEEE. 2003 IEEE International Workshop on. IEEE.

157. [Stoica et al., 2001] Stoica, I. (2001). Chord: A scalable peer-to-peer lookup service for internet applications. ACM SIGCOMM Computer Communication Review 31.4: 149-160.

158. [Stojmenovic and Wu, 2004] Stojmenovic, I. and Wu, J. (2004). Broadcasting and activity scheduling in ad hoc networks. Mobile Ad Hoc Networking: 205-229.

159. [Sucec and Marsic, 2000] Sucec, J. and Marsic, I. (2000). An efficient distributed network-wide broadcast algorithm for mobile ad hoc networks. CAIP Technical Report 248, Rutgers University.

160. [Taghizadeh et al., 2010] Taghizadeh, M. (2010). Towards optimal cooperative caching in social wireless networks. Global Telecommunications Conference (GLOBECOM 2010), IEEE.

161. [Taghizadeh and Biswas, 2011] Taghizadeh, M. and Biswas, S. (2011). Mobility-Aware Cooperative Content Caching in Social Wireless Networks. Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on. IEEE.

162. [Tan, 2001] Tan, K.-L. (2001). Organization of invalidation reports for energy-efficient cache invalidation in mobile environments. Mobile Networks and Applications 6.3: 279-290.

163. [Tang et al., 2008a] Tang, B., Gupta, H., and Das, S. R. (2008a). Benefit-based data caching in ad hoc networks. Mobile Computing, IEEE Transactions on 7.3: 289-304.

164. [Tang et al., 2008b] Tang, B., Gupta, H., and Das, S. R. (2008b). Benefit-based data caching in ad hoc networks. Mobile Computing,

IEEE Transactions on 7.3: 289-304.

165. [Tanga and Guptab, 2009] Tanga, B. and Guptab, H. (2009). Data Caching in Networks with Reading, Writing and Storage Costs.

166. [Taniar, 2007] Taniar, D. (2007). eds. Encyclopedia of mobile computing and commerce. IGI Global.

167. [Thai et al., 2007] Thai, M. T. (2007). Connected dominating sets in wireless networks with different transmission ranges. Mobile Computing, IEEE Transactions on 6.7: 721-730.

168. [Ting and Chang, 2007a] Ting, Y.-W. and Chang, Y.-K. (2007a). A novel cooperative caching scheme for wireless ad hoc networks: Groupcaching. Networking, Architecture and Storage, 2007. NAS 2007. International Conference on IEEE.

169. [Ting and Chang, 2007b] Ting, Y.-W. and Chang, Y.-K. (2007b). A novel cooperative caching scheme for wireless ad hoc networks: Groupcaching. Networking, Architecture and Storage, 2007. NAS 2007. International Conference on IEEE.

170. [Touch and Hughes, 1998] Touch, J. and Hughes, A. S. (1998). LSAM proxy cache: a multicast distributed virtual cache. Computer Networks and ISDN Systems 30.22: 2245-2252.

171. [Tseng et al., 2009] Tseng, V. S., Lin, K. W., and Hsieh, M.-H. (2009). Mining Temporal Region-Based Service Patterns for Cooperative Caching in Wireless Multimedia Sensor Networks. Intelligent Information Hiding and Multimedia Signal Processing, 2009. IIH-MSP'09. Fifth International Conference on. IEEE.

172. [Tseng et al., 2002] Tseng, Y.-C. (2002). The broadcast storm problem in a mobile ad hoc network. Wireless networks 8.2-3: 153-167.

173. [Tung et al., 2008] Tung, H. Y. (2008). QoS for mobile WiMAX networks: call admission control and bandwidth allocation. Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE. IEEE.

174. [Tyan and Mahmoud, 2005] Tyan, J. and Mahmoud, Q. H. (2005). A comprehensive service discovery solution for mobile ad hoc networks. Mobile Networks and Applications 10.4: 423-434.

175. [Valloppillil and Ross, 1998] Valloppillil, V. and Ross, K. W. (1998). Cache array routing protocol v1. : 03.

176. [Vasudevan et al., 2005] Vasudevan, S., Kurose, J., and Towsley, D. (2005). On neighbor discovery in wireless networks with directional antennas. INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE. Vol. 4. IEEE.

177. [Venkataramani et al., 2001] Venkataramani, A., Weidmann, P., and Dahlin, M. (2001). Bandwidth constrained placement in a WAN. Proceedings of the twentieth annual ACM symposium on Principles of distributed computing. ACM.

178. [Waharte et al., 2005] Waharte, S., Ritzenthaler, K., and Boutaba, R. (2005). Selective active scanning for fast handoff in WLAN using sensor networks. Mobile and Wireless Communication Networks. Springer US, 59-70.

179. [Wan et al., 2002] Wan, P.-J., Alzoubi, K. M., and Frieder, O. (2002). Distributed construction of connected dominating set in wireless ad hoc networks. INFOCOM 2002. Twenty-First annual joint conference of the IEEE computer and communications societies. Proceedings. IEEE. Vol. 3. IEEE.

180. [Wang and Bhulawala, 2005] Wang, J. Z. and Bhulawala, V. (2005). A P2P cooperative proxy cache system for wireless base stations. Wireless Networks, Communications and Mobile Computing, 2005 International Conference on. Vol. 2. IEEE.

181. [Wang and Li, 2009] Wang, Y. and Li, F. (2009). Vehicular ad hoc networks. Guide to wireless ad hoc networks, Springer London, 503-525.

182. [Wang et al., 2005] Wang, Y.-H. (2005). A dynamic caching mechanism for mobile ad hoc networks. Parallel and Distributed Systems, 2005. Proceedings 11th International Conference on. Vol. 2. IEEE.

183. [Wang, 2006] Wang, Y.-H. (2006). A distributed data caching framework for mobile ad hoc networks. Proceedings of the 2006 international conference on Wireless communications and mobile computing. ACM.

184. [Web-Age Information Management, 2008] Web-Age Information Management (2008). Ninth International Conference on. IEEE.

185. [Wessels, 1998] Wessels, D. (1998). Squid internet object cache. 337-343.

186. [Wieselthier et al., 1998] Wieselthier, J. E., Nguyen, G. D., and Ephremides, A. (1998). Multicasting in energy-limited ad-hoc wireless networks. Military Communications Conference, 1998. MILCOM 98. Proceedings., IEEE. Vol. 3 IEEE.

187. [Wolman et al., 1999] Wolman, A. (1999). On the scale and performance of cooperative web proxy caching. ACM SIGOPS Operating Systems Review. Vol. 33. No. 5. ACM.

188. [Wong et al., 2012] Wong, W., Wang, L., and Kangasharju, J. (2012). Neighborhood search and admission control in cooperative caching networks. Global Communications Conference (GLOBECOM), 2012 IEEE. IEEE.

189. [Wu and Dai, 2003a] Wu, J. and Dai, F. (2003a). Broadcasting in ad hoc networks based on self-pruning. International Journal of Foundations of Computer Science 14.02: 201-221.

190. [Wu and Dai, 2003b] Wu, J. and Dai, F. (2003b). Broadcasting in ad hoc networks based on self-pruning. International Journal of Foundations of Computer Science 14.02: 201-221.

191. [Wu and Dai, 2003c] Wu, J. and Dai, F. (2003c). Broadcasting in ad hoc networks based on self-pruning. International Journal of Foundations of Computer Science 14.02: 201-221.

192. [Wu and Dai, 2004a] Wu, J. and Dai, F. (2004a). A distributed formation of a virtual backbone in MANETs using adjustable transmission ranges. Distributed Computing Systems, 2004. Proceedings. 24th International Conference on. IEEE.

193. [Wu and Dai, 2004b] Wu, J. and Dai, F. (2004b). A distributed formation of a virtual backbone in MANETs using adjustable transmission ranges. Distributed Computing Systems, 2004. Proceedings. 24th International Conference on. IEEE.

194. [Wu and Li, 1999a] Wu, J. and Li, H. (1999a). On calculating connected dominating set for efficient routing in ad hoc wireless networks. Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications. ACM.

195. [Wu and Li, 1999b] Wu, J. and Li, H. (1999b). On calculating connected dominating set for efficient routing in ad hoc wireless networks. Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications. ACM.

196. [Wu and Wu, 2003] Wu, J. and Wu, B. (2003). A transmission range reduction scheme for power-aware broadcasting in ad hoc networks using connected dominating sets. Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th. Vol. 5. IEEE.

197. [Wu et al., 2003] Wu, J., Wu, B., and Stojmenovic, I. (2003). Power-aware broadcasting and activity scheduling in ad hoc wireless networks using connected dominating sets. Wireless Communications and Mobile Computing 3.4: 425-438.

198. [Wu and Cao, 2012a] Wu, W. and Cao, J. (2012a). Efficient Cache Discovery for Cooperative Caching in Wireless Ad Hoc Networks. ICPADS.

199. [Wu and Cao, 2012b] Wu, W. and Cao, J. (2012b). Efficient Cache

Discovery for Cooperative Caching in Wireless Ad Hoc Networks. ICPADS.

200. [Wu et al., 2009] Wu, W., Cao, J., and Fan, X. (2009). Overhearing-aided data caching in wireless ad hoc networks. Distributed Computing Systems Workshops, 2009. ICDCS Workshops' 09. 29th IEEE International Conference on. IEEE.

201. [Wu and Huang, 2010] Wu, W. and Huang, Y. (2010). Hierarchical Cooperative Data Caching for Wireless Mesh Networks. Embedded and Ubiquitous Computing (EUC), 2010 IEEE/IFIP 8th International Conference on. IEEE.

202. [Xu et al., 2001] Xu, D., Nahrstedt, K., and Wichadakul, D. (2001). QoS-aware discovery of wide-area distributed services. Cluster Computing and the Grid, 2001. Proceedings. First IEEE/ACM International Symposium on. IEEE.

203. [Xu et al., 2000] Xu, J. (2000). SAIU: An efficient cache replacement policy for wireless on-demand broadcasts. Proceedings of the ninth international conference on Information and knowledge management. ACM.

204. [Yang et al., 2004] Yang, H. (2004). Security in mobile ad hoc networks: challenges and solutions. *Wireless Communications, IEEE 11.1* : 38-47.

205. [Yang and Chuah, 2009] Yang, P. and Chuah, M. (2009). Performance evaluations of data-centric information retrieval schemes for DTNs. Computer Networks 53.4: 541-555.

206. [Yin and Cao, 2006] Yin, L. and Cao, G. (2006). Supporting cooperative caching in ad hoc networks. Mobile Computing, IEEE Transactions on 5.1 : 77-89.

207. [Yin et al., 2003] Yin, L., Cao, G., and Cai, Y. (2003). A generalized target-driven cache replacement policy for mobile environments. Applications and the Internet, 2003. Proceedings. 2003 Symposium on. IEEE.

208. [Yoshitaka et al., 2007] Yoshitaka, S. (2007). Large Scale Distributed disaster information system based on MANET and overlay network. Distributed Computing Systems Workshops, 2007. ICDCSW'07 27th International Conference on. IEEE.

209. [Zeitunlian and Haraty, 2010] Zeitunlian, A. and Haraty, R. A. (2010). An Efficient Cache Replacement Strategy for the Hybrid Cache Consistency Approach. World Academy of Science, Engineering and Technology 63, 268-273.

210. [Zhao et al., 2010] Zhao, J. (2010). Cooperative caching in wireless p2p networks: Design, implementation, and evaluation. Parallel and Distributed Systems, IEEE Transactions on 21.2: 229-241.

211. [Zheng et al., 2002a] Zheng, B., Xu, J., and Lee, D. L. (2002a). Cache invalidation and replacement strategies for location-dependent data in mobile environments. Computers, IEEE Transactions on 51.10: 11411153.

212. [Zheng et al., 2002b] Zheng, B., Xu, J., and Lee, D. L. (2002b). Cache invalidation and replacement strategies for location-dependent data

in mobile environments. Computers, IEEE Transactions on 51.10 : 11411153.

213. [Zhuo et al., 2011] Zhuo, X. (2011). Social-based cooperative caching in DTNs: A contact duration aware approach. Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on. IEEE.

List of Publications

Papers in International Journals

1. Preetha Theresa Joy and K. Poulose Jacob, "A review of Data Caching Techniques for Mobile Ad hoc Networks", International Journal of Mobile and Ad hoc networks, Vol. 1, Issue 3, pp. 361–365.

2. Preetha Theresa Joy and K. Poulose Jacob, "Cache Replacement Policies for Cooperative Caching in Mobile Ad hoc Networks", International Journal of Computer Science Issues, Vol. 9, Issue 3, No. 2, May 2012, pp. 500–505.

3. Preetha Theresa Joy and K. Poulose Jacob , "Cache Replacement Strategies for Mobile Data Caching", International Journal of Ad hoc, Sensor Ubiquitous Computing (IJASUC) Vol.3, No.4, August 2012, pp. 99 107.

4. Preetha Theresa Joy and K. Poulose Jacob, "Peer To Peer Cache Resolution Mechanism for Mobile Ad Hoc Networks", International Journal of Wireless & Mobile Networks, Vol.5, Issue 5, October 2013, pp. 192–196.

5. Preetha Theresa Joy and K. Poulose Jacob, "Cooperative Caching Framework for Mobile Cloud Computing", Global Journal of Computer Science and Technology – Network, Web & Security Volume 13, Issue 8, 2013, pp. 45– 52.

6. Preetha Theresa Joy and K. Poulose Jacob, "Cache Discovery over a Multihop Wireless Ad Hoc Networks", International Journal of

Information Processing, Vol. 8, Issue 2, 2014, pp. 71–80.

7. Preetha Theresa Joy and K. Poulose Jacob, " Adaptive Backbone Based Cooperative Caching in Mobile Ad hoc Networks", International Journal of Wireless Information Networks, **Springer**, Volume 21, Issue 4 (2014), pp. 306–316.

8. Preetha Theresa Joy and K. Poulose Jacob, (in press) "Cooperative caching architecture for mobile ad hoc networks", International Journal of Information and Communication Technology, **Inderscience**.

Papers in International Conferences

1. Preetha Theresa Joy, K. Poulose Jacob , "Cooperative Caching Techniques for Mobile Ad hoc Networks", Proc. of the International Conference on Data Science & Engineering (ICDSE) 2012, Kochi, India, pp. 175–180.

2. Preetha Theresa Joy and K. Poulose Jacob, "A Comparative Study of Cache Replacement Policies in Wireless Mobile Networks", Proceedings of the Second International Conference on Advances in Computing and Information Technology, July, 2012, Chennai, India, published in Lecture Notes in Computer Science, Vol. 5793 (Springer, Heidelberg), pp. 609–619.

3. Preetha Theresa Joy and K. Poulose Jacob, "A Key Based Cache Replacement Policy for Cooperative Caching in Mobile Ad hoc Networks", 3rd IEEE International Advance Computing Conference (IACC), Delhi, India, 2013, pp. 383–387.

4. Preetha Theresa Joy and K. Poulose Jacob, "A Novel Cache Resolution Technique for Cooperative Caching in Wireless MobileNetworks", Proc. of the Third International Conference on Computer Science, Engineering & Applications, May, 2013, Delhi, India, CS&IT, Vol. 3, No. 5, 2013, pp. 203–209.

5. Preetha Theresa Joy and K. Poulose Jacob, "A location Aided Cooperative Caching Protocol for Mobile Ad hoc Networks", 6th IEEE International Conference on Advanced Infocomm Technology, Taiwan, July 2013.

6. Preetha Theresa Joy and K. Poulose Jacob, "Energy Efficient Cache Discovery in Wireless Mobile Ad Hoc Networks", Proc. of Seventh International Conference on Communication Networks (ICCN-2013), Bangalore, India, August 2013, Elsevier Science and Technology, Vol. 3, 2013, pp. 233–242 .

7. Preetha Theresa Joy and K. Poulose Jacob, "A Coordinated Cache Data Placement Scheme for Mobile Ad hoc Networks", Fifth International Conference on Advanced Computing, Chennai, India, Dec 2013.

8. Preetha Theresa Joy and K. Poulose Jacob, "A Virtual Backbone Based Approach for Cooperative Caching in Mobile Ad hoc Networks", 16th IEEE International Conference on Advanced Communication Technology, Pyeongchang, Korea (South), Feb 2014.