Sapienza Università di Roma

Dottorato di Ricerca in Ingegneria Informatica

XX Ciclo – 2008

Distributed Search and Service Discovery in Wireless Ad Hoc Networks using Random Walk

Adnan Noor Mian

Sapienza Università di Roma

Dottorato di Ricerca in Ingegneria Informatica

XX Ciclo - 2008

Adnan Noor Mian

Distributed Search and Service Discovery in Wireless Ad Hoc Networks using Random Walk

Thesis Committee		Reviewers
Prof. Roberto Baldoni	(Advisor)	Prof. Dr. Miroslaw Malek
Dr. Roberto Beraldi	(Co-Advisor)	Prof. Dr. Michel Banâtre
Prof. Giacomo Cioffi		

Author's address:
Adnan Noor Mian
Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
"Sapienza" Università di Roma
Via Ariosto 25, I-00185 Roma, Italy.
e-mail: `adnan@dis.uniroma1.it`
www: `http://www.dis.uniroma1.it/~adnan/`

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
"Sapienza" Università di Roma
Roma, Italy

The undersigned hereby certify that they have read and recommend the thesis entitled **"Distributed Search and Service Discovery in Wireless Ad Hoc Networks using Random Walk"** by **Adnan Noor Mian** to the Faculty of Graduate Studies for the degree of Doctor of Philosophy in Computer Engineering.

Dated: February 2009

Advisor: ______________________________
Prof. Roberto Baldoni

Co-Advisor: ______________________________
Dr. Roberto Beraldi

Abstract

Wireless ad hoc networks are networks of devices that are autonomous and possibly mobile. The devices cooperate with each other by providing different types of services. These services must be discovered before they can be used. This work focuses on service searching and discovery protocols in wireless domain based on random walk method. There are many service searching and discovery protocols in wired networks that use random walk but due to the challenges associated with the wireless domain, the use of random walk method for searching in wireless domain has not been fully exploited. The thesis studies the use of random walk method in wireless ad hoc networks from three different aspects. Firstly, it proposes an energy efficient protocol that implements one step of random walk, secondly, it studies the mechanisms to reduce the search time and proposes a novel random walk biasing strategy in this regard and lastly, the thesis proposes a stopping rule for random walk with which the lifetime of the random walk adapts itself according to the network size without a priori knowledge of the network size.

Acknowledgements

I thank Allah Almighty, the creator of knowledge, the creator of every thing, for giving me the ability, perseverance and knowledge for this academic contribution. Whenever I was stuck, He created a way out. This work had not been possible without His help.

My sincere thanks to my advisor Professor Roberto Baldoni and co-advisor Dr. Roberto Beraldi. Baldoni taught me how to look at problems in an abstract way. His multitasking personality inspired me to handle different academic tasks in parallel. His encouragement in different aspects of the PhD work greatly contributed to my personal and professional development. He has a great sense of humor which I always enjoyed. My special thanks to Roberto Beraldi who taught me the details of doing research. We often had very long academic conversations and he never get bored of it. I always enjoyed discussing with him. His unusual way of looking at a problem often surprised me. I also thank Professor Giacomo Cioffi whose time to time remarks was a source of encouragement.

Many thanks to Professor Ravi Prakash and Professor Roy Friedman whose critical remarks helped me to think deeper into certain research problems of this PhD work. I would like to thank my thesis reviewers, Professor Dr. Miroslaw Malek and Professor Dr. Michel Banatre who gave me very useful detailed feedback and reviews of the thesis that greatly helped me to improve its overall presentation and the technical quality.

I wish to thank to all my friends and colleagues with which I worked during the tough days of PhD. Many thanks to Shah Rukh, Andrea, Marco, Luca, Vittorio, Sara, Sirio, Alessia, Geuseppe, Vishvas, Camil, Amir, Ejaz and Zulqarnain. I would specially like to mention Antonella and Leonardo who always helped me whenever I had difficulty due to my Italian language.

I would also like to thank the Italian Ministry of Foreign Affairs for providing me the financial support for the whole period of PhD.

Many thanks to my wife for always giving me the emotional support and my little princess whose innocent comments were always amusing for me. I wish to thank my sweet sisters and especially my brother, Rehan. He took care of many things that I was supposed to do thus relieving me to focus on my studies. Finally I would like to thank and express my gratitude to two very important personalities, my father and my mother. Without their encouragement and prayers for my success I would never have achieved this goal.

Adnan Noor Mian

Roma, Italy,
Februry 2009.

Contents

List of Figures

Chapter 1

Introduction

Wireless ad hoc networks consist of autonomous, and possibly mobile devices which communicate over radio without having any support from a fixed infrastructure. Compared to existing fixed infrastructure based networks, wireless ad hoc networks are challenging in many aspects. On one side the high degree of dynamism due to highly variable wireless channels conditions and mobility of devices and on the other side the small size and resource limited nature of wireless devices constituting the ad hoc network makes centralized approaches, like the use of dedicated servers, often unsuitable.

Mobile devices constituting the wireless ad hoc network due to their small size and mobility are inherently scarce in resources, which necessitates the need to cooperate among them for performing operations that cannot be done alone. At application level this cooperation can take the form of services following the paradigm of so called service oriented architecture. To get benefit from the services offered by other devices, they have to be searched. Search and Service Discovery Protocols (SDPs) are used for this purpose. This is an important area of research in mobile and pervasive computing and also the focus of our thesis.

In recent years due to the growth of numerous handheld and even smaller devices the size of the network is ever increasing. With the increase of number of devices and due to the deceasing constraints on their mobility the conventional ways of service discovery used in wired and small wireless networks are no more efficient and there is a dire need to investigate other methods for searching services.

1.1 Problem domain and description

In wired networks and for small ad hoc networks there are many SDPs. Often such approaches presuppose the presence of a reliable and constantly available devices having the directory, which is a repository to keep information about services of whole or part of the network and act as service brokers. Such directory-based SDPs are unsuitable for service discovery in mobile environments because the consistent reliable presence of devices with directory cannot be guaranteed.

For high mobility scenarios often directory-based approaches are not well suited. Instead, in such cases directory-less approaches are used for service discovery. Such directory-less approaches employ network wide query flooding to search for services. But the methods that use flooding are not scalable. These methods are primarily efficient for network wide information dissemination but as far as searching for a specific service is concerned any method that does network wide query flooding waste valuable resources in an already resource constraint ad hoc environments. For example a device with a service may be just a few hops away from the search initiating device. In such a case network wide flooding is clearly an inefficient method to search for a service.

In this dissertation we investigate for random walk as an efficient method of searching services in resource constraint large wireless ad hoc networks potentially consisting of thousands of devices. Due to the energy constraints and potentially being very small we assume that the devices have no hardware for identify its location and these devices can be stationary to moving with high speeds and also can join and leave the network at random.

We enquire four main problems that are associated with the use of random walk as a search mechanism in wireless ad hoc networks.

- What are the core problems associated with SDPs in mobile ad hoc networks?
- How random walk can be made robust in wireless scenario?
- How random walk can be used efficiently for searching and service discovery?
- When to stop a random walk if, for example, a service may not be available in the network?

1.2 Research Contribution

In this thesis we fist study some of the existing SDPs for MANETs and identify that search mechanism is the most crucial aspect of any SDP. We have seen that in search mechanisms often flooding or gossiping is used in wireless ad hoc networks, which have inefficiencies when used for searching. Different efficient approaches for flooding and gossiping have been proposed which have actually proved a nice solution for network wide information dissemination but as far as searching for a specific service in large resource constraint wireless ad hoc networks is concerned we argue for the use of RW as a better alternative search mechanism. RW is already being used in wired peer-to-peer (P2P) networks but it has not been use in wireless ad hoc networks for the purpose of service discovery. This is due to the inefficiencies associated with RW in wireless domain due to the following three important aspects.

- one step implementation of RW
- large mean number of steps required for searching a service
- termination of RW when, for example, service is not found

In this thesis we have discussed all these three aspects of RW and have proposed efficient solutions to address these issues. The research contribution of the thesis are summarized below.

First contribution of this thesis is to provide a comprehensive study of multihop MANETs Service Discovery Protocols (SDPs) and come up with a list of open problems. We chose MANETs as this is the most challenging of all the ad hoc networks. First we give a general framework and then discuss different building blocks of the MANET SDPs. We critically analyze different SDPs with respect to their building blocks like description, registration, routing, mobility support and discovery mechanisms. We classify these SDPs with respect to different architectural views and discuss suitability of different classes of SDPs with respect to mobility and network size. We then discuss some of the most challenging issues and open problems related to service discovery in mobile ad hoc environments. In this study we recognize that search mechanism is the most crucial aspect of a SDP. We found that RW, which is already being used in wired P2P networks for searching, is not being used in wireless ad hoc networks. This led us to dwell on the issues related to implementation of RW based search in wireless domain.

Second contribution of the thesis is to provide an energy efficient, robust and low latency protocol for implementing one step of RW. As RW is a sequential process, the most crucial aspect is to have an efficient mechanism that implements one step of the query in wireless ad hoc networks. This thesis proposes such an energy efficient and robust next hop selection distributed algorithm that is implemented at data link layer level which fully exploits the broadcast nature of wireless transmission. We show that we can greatly reduce the energy requirements and make RW robust to dynamics by exploiting the broadcast nature of the transmissions. We evaluate the algorithm and present a extensive simulation study performed with *ns*-2. We found that in the proposed algorithm energy is reduced to more than 4 times and the selection delay is reduced to more than 8 times as compared to a standard next hop selection implementation.

Third contribution of the thesis is that it proposes a biasing strategy to reduce the number of steps for searching. The pure random walk leads to large number of steps which makes searching inefficient. In order to have efficient search the number of steps the query makes before hitting the target node should be low. This can be achieved exploiting some form of bias, which forces the query to always explore new parts of the network. There are many biasing strategies some of which depend on some external information source to achieve efficient search. In this thesis we propose a novel biasing strategy for RW that relies only on the local information available to a node. We then compare our proposed strategy with some of the existing strategies, that also rely on local information. Through extensive simulations on square grid topology we show that the proposed biasing strategy is most cost effective in searching, most scalable, least effected by neighbor density, and most effective in replicated services scenarios among the compared strategies.

Fourth contribution of the thesis is that it proposes a stopping rule for RW that the mean coverage of at least some fraction of the network is very less dependent of the network size and the RW with the proposed stopping rule adapts its lifetime according to the network size. A RW terminates when the required service is found but if the service is not available or we are just interested in searching a specific portion of the network, the RW should not continue for ever. A stopping rule gives the conditions for the termination of RW. In this thesis we propose a stopping rule for RW which depends on the two metric, number of steps and number of visited nodes, both of which can be calculated online with out any need of global information. We have studied the proposed stopping rule analytically and through simulations and found that with this stopping rule the RW coverage adapts its lifetime with the network size and the fraction of network coverage very less depends on the netwrok size.

Part of the content of this thesis has been published in the following papers [11] [68] [69] [70] [66] [67].

1.3 Organization of the thesis

In Chapter 2 we critically analyze different search and Service Discovery Protocols (SDPs) in mobile ad hoc environments. We discuss some of the differences why SDPs used in wired networks cannot be used in ad hoc and especially mobile ad hoc environments. We then present a general framework of a SDPs for Mobile Ad Hoc Network (MANETs) and discuss its different building blocks. We conclude the chapter by giving some of the issues and open problems for service discovery in mobile ad hoc environments.

In Chapter 3 we introduce and give an account of work done on flooding and gossiping, the basic distributed search mechanisms used in wireless ad hoc networks. We then introduce random walk and also present the work done in applications of random walk and specially its application as a search tool. We compare random walk with flooding and gossiping and we give the advantages of using random walk for search and point out some of the reasons why random walk has not been used for searching in wireless ad hoc networks although random walk based search algorithms are widely used in wired peer-to-peer networks. We then summarize some of the important definitions and results regarding random walk that introduces the topic formally and give background for further understanding of the thesis.

In Chapter 4 we propose a protocol for implementing one step of the RW in wireless ad hoc networks. We show that the proposed distributed algorithm is more energy efficient, incur less delay in one step and more robust in mobility as compared to a standard protocol. Splitting of RW can occur in such kind of distributed algorithms. We show that the proposed algorithm non-splitting property.

In Chapter 5 we propose a biasing mechanism that only rely on the local information. We compare different biasing strategies that also rely on local information and show that the proposed biasing strategy is not only best among the compared biasing strategies with respect to estimated cost of search but also is more cost effective than flooding and gossiping.

In Chapter 6 we discuss another important issue related to RWs, which is when to stop the RW. We suggest a stopping rule that we have studied analytically and through simulations. The proposed stopping rule has a nice property that the RW adapts its lifetime with the network size and coverage is very less dependent on the network size.

In Chapter 7 we draw some conclusions and give directions for future work in the field of using random walk as a search mechanism in wireless ad hoc networks.

Chapter 2

Service Discovery Protocols in Multihop Mobile Ad hoc Networks

Mobile Ad doc Networks (MANETs) may include variety of devices like mobile phones, palmtop, laptops, or other relatively big devices. These devices can move with high speeds to very low speeds or even be stationary and enter or leave the system by switching them on or off. Such variety of devices has variety of services to offer. A service can be any tangible or intangible facility provided by a device that can be useful for any other device. Services can be software services or hardware services. A software service can be a simple file, for example an MP3 file or it can be, for example, a software implementation of an algorithm like, converting one audio file format to another. A service can be a hardware service like a printer that can be used by a mobile device to print a file. To get benefit from these services a device must be able to locate them in the network and also has the ability to invoke these services. Service Discovery Protocols (SDPs) provide such capabilities to devices.

SDPs have been built for wired networks, single-hop wireless networks and multihop MANETs. In wired networks there are many industry standard SDPs, for example, Jini [71] by Sun, Universal Plug and Play (UPnP) [40] by Microsoft, Salutation [32] by IBM and Service Location Protocol (SLP) [44] by IETF. In single-hop wireless we have, for example, protocols like DEAPSpace [73] and an industry standard protocol Bluetooth SDP [93]. In the wired networks devices do not move whereas in single hop wireless ad hoc networks, the network can have very restricted mobility with very low rate of joins and leaves of devices. We do not discuss SDPs for wired or for single hop networks in this survey.

This study focuses only multihop MANET SDPs . Such networks are one of many ways of realizing pervasive computing environments. In these networks devices are allowed to have unrestricted mobility, with no constraints on joining or leaving the network. This makes the problem of service discovery very challenging. Most of the SDPs in wired networks are made to work at application level. In theory any such application level SDP can work for MANETs also. But in practice mobility and scarcity of resources introduce two new dimensions that are not taken into account in SDPs for wired or single hop wireless networks. In the SDPs built for such networks location awareness and physical proximity between the service provider and the user is not taken into account and hence the physical proximity remains undetectable. In multihop MANETs the physical proximity between the service provider and the user does matter. In these networks we are interested in utilizing services that are physically close and are at less number of hops. This is needed to efficiently utilize the bandwidth and to reduce the chances of link breakages. The OSI architecture layering approach works well in wired networks where there are no resource constraints, but it introduces of inefficiencies in resource constraint MANETs. Cross layering can improve the efficiency, for example a specially designed SDP that integrates service discovery with routing can improve the efficiency. Many SDPs have been proposed but still these are not as developed as we can find in wired networks or single hop ad hoc networks.

There are some good comparative studies of SDPs that also examine few SDPs for MANETs, for example, the surveys done by Cho and Lee [29], by Zhu and Mutka [99], and by Marin-Perianu, Hartel and Scholten [65]. These studies survey the SDPs for all of the three types of networks and none of them goes deep into analyzing SDPs for only the multihop MANETs. In this chapter we focus only on the SDPs for multihop MANETs. We selected twelve multihop MANET SDPs [27]-[22]. We identified basic building blocks of any SDP and then analyzed the selected SDPs in context of these building blocks.

The chapter is organized as follows. In Section 2.1 we discuss the basic building blocks of a SDP. In Section 2.2 we describe service description mechanisms in MANETs. In Section 2.3 and Section 2.4, we discuss the service registration mechanism and service discovery mechanisms. We devote Section 2.5 for describing routing mechanisms and Section 2.6 for dealing with mobility. In Section 2.7, after discussing the existing categorizations, we propose another classification of service discovery architectures of multihop MANETs and then in Section 2.8 we discuss suitability of different SDPs in different conditions. In Section 2.9 we present some of the challenging issues and open problem.

2.1 Framework of a Service Discovery Protocol

The Figure 2.1 shows the basic building blocks of framework of a SDP in MANETs. The basic mechanisms that make a complete SDP are the description mechanism, registration mechanism, discovery mechanism consisting of search methods and selection mechanism, routing mechanism, and mobility support mechanism. A user, which can be an application software that acts on the behalf of the end user, interacts with the SDP. The arrows in Figure 2.1 show that the user can initiate any of the pointed mechanisms. The user interacts with description mechanism, which describes a service in a format used to register or to search a service. A proper description facilitates searching of a service. Section 2.2 provides the details of description mechanism. After a service is described the user may want to search for the service or to register it. If it is a new service then it must be first registered so that it can be discovered later. This is done by the registration mechanism. Registration means to store the description of the service at some places. There are many different ways to register a service. The services may be registered only in the local cache of the nodes or in all nodes of the network or in few specially selected directory nodes or in a subset of nodes, which can be either n-hop neighbor nodes or a multicast group. The registration mechanism is explained in detailed in Section 2.3. Once the services are registered, they can be found by the discovery mechanism. The user interacts with the discovery mechanism, explained in section 2.4, specifying the service through the description mechanism and selection criteria. The search mechanism (Section 2.4.1) searches for the service information in all nodes of the network or just the local cache or specially selected directory nodes or subset of nodes, which may be n-hop neighbor nodes or a multicast group. When all instances of a service are found, the selection mechanism (Section 2.4.2), selects the most suitable service provider. The separation between search and selection mechanisms is often logical. It can also happen that a service is searched based on selection criteria and the first service provider fulfilling the criteria is selected.

SDP can also form overlay networks. An overlay network is build on top of the existing network. It is a virtual network consisting of subset of nodes from the existing network and logical links between these nodes. The nodes are selected to form a certain topology with some special properties. A mulithop protocol is used to achieve links between these nodes. An overlay network direct traffic to specific nodes and hence greatly reduces the network traffic. Registration and discovery mechanisms can use overlay networks to help route the registration or search messages to specific selected nodes. The specific routing mechanisms are explained in Section 2.5.

In MANETs the topology of the network changes due to the mobility of nodes. If a SDP depends on some specific overlay topology then it must

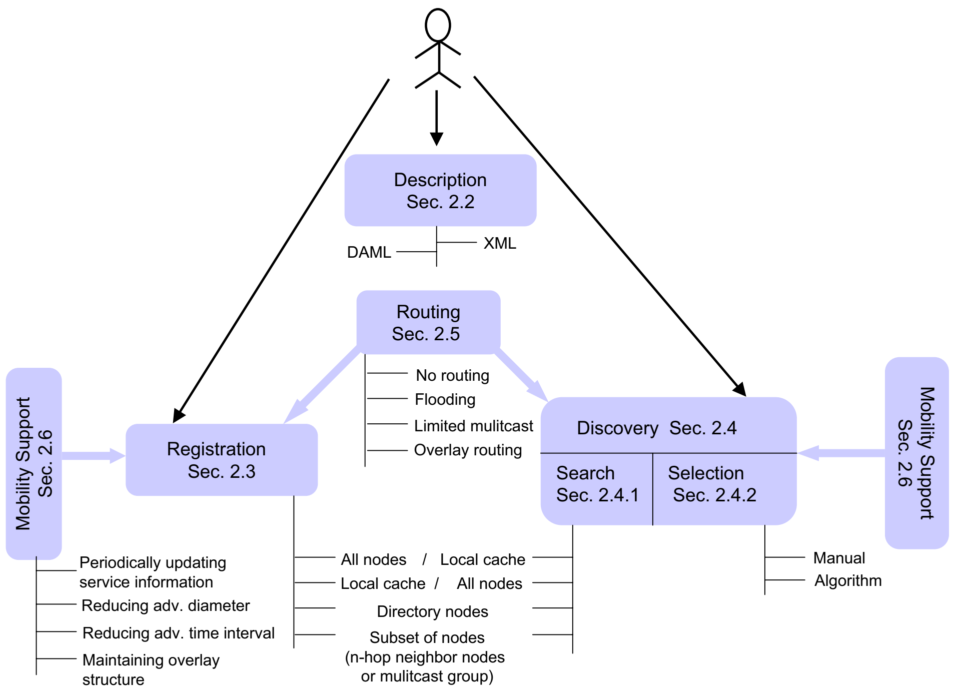


Figure 2.1: Framework of SDP

maintain the overlay, which may become faulty due to the mobility of nodes. To compensate for mobility, there is a mobility support mechanism, explained in Section 2.6.

In the following sections we now describe these building blocks in detail.

2.2 Description Mechanism

Service description is an abstraction of the facilities and characteristics of a service. The description of a service is necessary if it is to be utilized by other devices or services. The nodes search for services only by looking at the descriptions of the services advertised by the service provider. A service may remain completely unknown to other devices in the network if it is not properly described. Description of a service includes service properties like ID of a service, server properties like load on a service provider, and network properties like maximum number of hops from the current host to a service provider.

Most commonly used method for service description is by using eXtensible

Markup Language (XML) and its extensions like DAML (DARPA Markup Language) and Web Ontology Language (OWL) [21][48][22]. The description file is a plain text file that has all the information about the service. XML service description enables hardware and software interoperability between different hardware and software platforms. On the other hand, XML requires more resources for parsing. SDPs can also be independent of any description language. In this case any language or description method can be used [79][54][53][94]. To have this independence some times constraints are imposed on the specific language [54].

2.3 Registration Mechanism

The service description must be stored somewhere so that other nodes can contact this node to discover a particular service. Registration mechanism include aspects like where to store service description, time duration for which the description will be stored, distance in number of hops the information will travel as advertised by the service provider.

Different SDPs manage service descriptions in different ways. Some SDPs stores service descriptions only in the local cache of nodes, in which all nodes have to be searched for finding the service. Some SDPs store service descriptions on every node of the network, that is, the service advertisements are flooded to the whole network. In such a case only the local cache is searched for a service. Another approach is to store service descriptions to a subset of nodes by flooding advertisements to n hops [21][59], which is the advertisement diameter, or to flood advertisements to multicast groups of nodes [48]. By restricting the flow of advertisements these protocols reduce memory requirements and also increase the probability of discovering a service in vicinity of the service provider. The advertisement diameter can be just one hop in which the service descriptions are stored just in immediate neighbors [79][22].

Some SDPs store service descriptions in special selected nodes, called directory nodes. A directory or a directory node is an entity that stores information about services available in the network. This helps in the service discovery process [29]. The selection of such nodes depends on specific protocols. In some SDPs these are pre-assigned nodes [98], whereas in some protocols these nodes are selected at run time and may be deselected if the network conditions change due to mobility. For example [54][92] form groups of nodes that are physically close to each other. Each group selects a node, which keeps record of the services present in that group. The SDP in [56] forms virtual backbone from some of the mobile nodes and service descriptions are stored in these virtual backbone nodes.

2.4 Discovery Mechanism

The discovery mechanism is responsible for the discovery of suitable services. This mechanism consists of search and selection mechanisms. These two mechanisms can be independent or they can be integrated. For example a SDP may first search for all instances of a service provider and then select a suitable service or the searching and selection can be done simultaneously. We shall explain both mechanisms in the following subsections.

2.4.1 Search Mechanism

The search mechanism is responsible for searching services, which already have been registered by the registration mechanism. In networks that do not have special directory nodes to keep service descriptions, search for a service depends on how the services were registered. SDPs that register services in local cache search all nodes. This is achieved by flooding the search message [27]. Such type of discovery is called active search [94]. SDPs can also register their services in all nodes. In such cases the discovery mechanism just searches the local cache. Such type of discovery is called passive search [79][94]. Passive search has the advantage of being efficient in terms of latency in detecting the changes in the environment. When a device receives an advertisement from a new device, it automatically knows that there is a new node in the vicinity offering services [79]. In active search whenever a service is required, a search message is sent which results in increased latency in search as compared to passive search. Active search, on the other hand, has the advantage of generating less traffic in the network as the search message is only sent when a node is in need of a service, whereas in passive search more network traffic is generated due to the periodic dissemination of advertisement messages. A SDP can support both active and passive approaches [48][94]. Another approach is selective forwarding of search requests. In this method the local cache of a node is first searched and if service is not found, the request is selectively forwarded to other nodes based on the ontology descriptions [22]. When a node does not have enough information to selectively forward a request it simply broadcasts the request to its neighboring nodes. The SDPs with selectively forwarding of service requests scale well [22] as compared to SDPs using simple flooding but under high mobility these protocols may suffer because of the forwarding of messages in regions where the service is no longer available due to mobility of nodes [22].

In networks that have directories for storing service descriptions, the service provider nodes advertise services only to directory nodes. These directory nodes may be pre-assigned the task of storing service descriptions [98] or they may be selected at run time [54], depending on the protocol. The clients only search these directory nodes. For example in Service Rings [54] the protocol only searches the Service Access Points (SAPs) nodes of the service rings. In [56] and [92] the client queries only the Virtual Backbone Nodes (VBNs) and the gateway of each cell respectively. In Lanes [53] SDP nodes are grouped together to form an overlay network forming lanes of nodes. Each group is called a lane. Nodes in the same lane have the same directory replicated in each node cache. So the search is done using unicast to any of these directory nodes in a lane overlay instead of one specific node.

2.4.2 Selection Mechanism

The query request from a client node to the network can result in many responses of matching services. Selection from among the discovered services is then essential to invoke one of the services. Selection can be done manually by the user or can be automated using some algorithm based on some criteria. The criteria can be the lowest hop count or current load of a service provider or bandwidth available of the communication channel between the service provider and the client or the velocity of the service provider [92].

Often the selection mechanism is integrated with the search mechanism. In [94] authors have demonstrated that proper service selection integrated with search mechanism improves the overall network performance, by localizing the network communication, thus reducing interference and allowing multiple concurrent transmissions in different parts of the network. In [59] client selects services using two metrics, one is the number of hops from the service provider and other is the capacity of service. The algorithm for service selection is distributed and does not involve direct interaction with the client.

There are many MANETs SDPs that just ignore the selection issue [27]-[56][22].

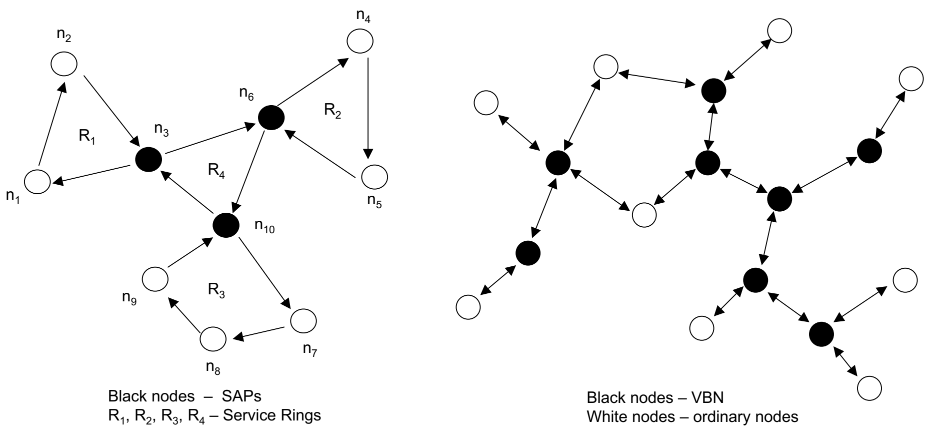
2.5 Routing Mechanism

The SDPs that register [27] or search in all nodes of the network use flooding of query or registration messages. A SDP can scale well if the network traffic it generates is limited. To limit the network traffic SDPs employ different routing mechanisms. For example SDPs limit the flooding to n-hops [21] or use multicast routing to specially formed groups of service providers [48] or use selective forwarding based on ontology [21][22] or selective forwarding based on the notion of potential [59]. SDPs can also form overlay networks and have specific routing mechanisms within these overlay networks. For example Service Rings [54] protocol, shown in Figure 2.2(a) forms an overlay structure, called service ring, by grouping of nodes that are physically close and offer similar services. Each service ring has a designated service access point (SAP), which keeps the directory, having information about available services with the service rings. The SAPs are connected with SAPs of other service rings. When a node wants to search for a service, the query is routed through the ring structure, passing through SAPs of other rings and reaching only to those subrings that can possibly offer the service. SDP in [56] form virtual backbone from specially selected nodes called Virtual Backbones Nodes (VBNs), shown in Figure 2.2(b). These VBNs constitute a dominating set. A dominating set is one in which all the nodes in the network are either in this set or only one-hop away from at least one member of the set. The VBNs keep the directory, which stores the advertised information about other services in the network. The client forwards the service request to VBNs, which in turn multicast the query message to all the other VBNs. Thus multicasting the backbone nodes, instead of flooding all nodes in the network reduces the overhead of broadcasting a query. The arrows in Figures 2.2 denote the routing path of query or registration messages. SDP in [92] divides the environment in hexagon cells, with each cell having a gateway node. The gateway node of each cell provides the directory services. When a client wants to search for a service, the query is routed only to the gateway nodes. In Lanes [53] the nodes are grouped together to form an overlay network in which each node in the group has the same copy of the directory. The registration or search messages are sent through different lanes by anycast routing.

2.6 Mobility Support Mechanism

In MANETs nodes move frequently and change their position with respect to each other. SDPs in which there are no directory nodes, all nodes keep information of only their own services. In such a case mobility is automatically handled by flooding or multicasting query messages to all or few nodes [27] respectively. On the other hand mobility is a challenging issue in SDPs with directory nodes. We now discuss some of the methods to handle the adverse effects of mobility in such SDPs.

Periodical updating of service information [48][98][59] is one method to handle mobility. When a service provider advertises its services, it also announces the timeout after which the service information will expire. The mobile nodes cache this information. The service provider has to announce the service information again before it gets expired. Rediscovery and reselection of services are also used to handle the mobility. In rediscovery the network is probed for up-to-date information about the available service providers and in reselection the services are selected based only on the current entries in the service table [94]. Another method to handle mobility is to reduce the advertisement diameter and advertisement time interval [21][79][22] to compensate



(a) SDP forming Service Rings

(b) SDP forming VBN

Figure 2.2: Routing mechanisms in two SDPs

for the effects of rapidly changing vicinity.

In SDPs that form overlay networks there are special algorithms for maintaining the structure of overlay, which gets faulty due to the mobility and logging in and logging out of nodes from the network. Service rings [54] have special algorithms to maintain the consistency of the ring overlay. Each ring member knows only its successor and it predecessor. RingCheck messages are initiated periodically by the appropriate Service Access Points. The messages circle through each ring to check its consistency. If a node does not receive such a message in one of its rings for a certain time it checks for a link breakage or a partition in the network. If any of these cases is detected then an appropriate algorithm is initiated to repair the ring. Similarly in Lanes [53] and SDP in [56] different algorithms are used to maintain the lane structure and the dominating set feature of the virtual backbone respectively. The SDP in [92] has two mechanisms to support mobility. First when a gateway node moves to another cell, it broadcast the service information to nodes in its previous cell. These nodes elect another gateway node. The second mechanism is by specifying time to live parameter, which is the clock time after which a service has to refresh its advertisement.

2.7 Architectural Views of Service Discovery Protocols

After having discussed the major components of a SDP, let us have a look on different architectural views of SDPs. Broadly speaking architecture specifies the layout of any structure and how major components of that structure are connected with each other. An architectural view would thus be a specific perspective of looking at the SDPs.

2.7.1 Directory-based Vs. Directory-less

One way of looking at the architecture of SDPs in multihop MANET is with respect to directory. In mulithop MANET SDPs, the architecture can be directory-based or directory-less. In directory-based architecture certain nodes in a MANET are chosen to be directory nodes. The registration and discovery mechanisms are specially tailored to access these nodes as explained in the Sections 2.3 and 2.4. For example the SDPs Lanes [53], by Kozart et al [56], by Tyan et al [92], Splendor [98] are directory based. In directory-less architecture there are no directory nodes as in SDPs by Cheng et al [27] and Varshavsky et al [94], GSD [21], Allia [79], Konark [48], Service Rings [54], Field Theoretic Approach [59] and DSD [22]. The registration and discovery mechanisms approach all or subset of nodes of the network. Directory-less approach is often used in MANETs. The reason is that it is difficult to host directory in mobile nodes that are already scarce in resources like memory and power and also cannot guarantee their availability in the network. A directory node is supposed to have large resources and remain available in the network as different nodes in the network will be frequently contacting it.

2.7.2 Overlay-based Vs. Overlay-less

Another way of looking at the architecture of SDPs is with respect to overlay. The overlay network has the advantage of controlled multicast of service query or advertisement message. This controlled multicast restricts and greatly reduces the network traffic as in SDPs by Kozart et al [56] and Tyan et al [92], Allia [79], Service Rings [54], Lanes [53]. There are many SDPs that do not form overlay network, for example, the SDPs by Cheng et al [27], Varshavsky et al [94], Splendor [98], GSD [21], Konark [48], DSD [22] and Field Theoretic Approach [59]. The reason for having more SDPs that do not form overlay network is that in mobile environments forming and maintaining overlay networks require the resource constraints nodes to be vigilant about the topological changes and also to run algorithms, which are resource consuming tasks.

2.7.3 Architectural views based on directory and overlay

Instead of looking at SDPs with respect to directory or overlay, if we look at a SDP w.r.t directory and overlay, we propose to classify SDP architectures in four categories, as stated below.

(i) directory-based with overlay support [54][53][56][92]

(ii) directory-based without overlay support [98]

(iii) directory-less with overlay support [79]

(iv) directory-less without overlay support [27][21][48][94][59][22]

Each category of architecture identified above is suited for different environmental conditions. There can also be other factors that can influence the use of a SDP. In the next section we discuss the suitability of different protocols under different conditions.

2.8 Suitability of protocols under different conditions

There are different dimensions for understanding the behavior of SDPs, for example network size, node mobility and behavior of a SDP in a network with multiple service instances. Figure 2.3 shows the suitability of different types of SDPs with respect to different network sizes and mobility conditions. The magnitudes of different network sizes can be as follows.

- small network: up to 10 nodes
- medium networks: from 10 to 100 nodes
- large network : greater than 100 nodes

Similarly the mobility conditions can be expressed as follows.

- low mobility : up to 5km/h (e.g., scenario in a building)
- medium mobility: from 5km/h to 50 km/h (e.g., traffic scenario in a city)
- high mobility: greater than 50km/h (e.g., highway traffic speeds)

The shaded region in Figure 2.3 shows the region in which all SDPs will work well. There are some specific regions, which are not shaded, are exclusive for only a particular kind of protocol. In these regions only special types of SDPs are expected to work well. We shall first explain the suitability of different protocols in these specific regions and then discuss the behavior of SDPs in a network with multiple service instances

Protocols suitable for small networks and high mobility

Protocols that are directory-less and do not form overlay networks are suitable for small networks with high mobility. Such protocols use flooding for registration and discovery mechanisms. Flooding has the negative side of generating large overhead packets and hence scalability is an issue. SDPs that use flooding thus cannot be used for large networks. On the other hand such protocols do not form overlay and do not have directory nodes so there is no overhead in maintaining the overlay structure and also in selecting the directory nodes. Hence such protocols perform well under high mobility, for example, the SDP by Cheng et al [27]. Figure 2.3 shows the upper smaller region, which is well suited and exclusive for such type of SDPs.

Protocols suitable for medium size networks and medium mobility

Protocols that use limited flooding, for example flooding limited by TTL, or multicasting to a group of nodes as in GSD [21] or flooding on top of overlay network as in SDP by Kozart and Tassiulas forming VBN [56] are well suited for medium size network with medium speeds of nodes. The limited flooding improves mobility handling and the use of overlay network improves scalability. Figure 2.3 shows that for medium size networks and medium mobility, those SDPs are well suited which are directory-based without overlay support or directory-less with overlay support.

Protocols suitable for large networks and low mobility

Overlay networks provide an efficient routing mechanism for searching. In low mobility there is no issue of overlay maintenance, so the SDPs like Service Rings [54] and Lanes [53], that form overlay networks, perform well in large networks. On the other hand overlay maintenance becomes an issue under high mobility and such protocols cannot be used in high mobility scenarios. Figure 2.3 shows that the SDPs, which are directory-based with overlay support are exclusively suited for this region.

Protocols suitable for networks with multiple service instances

If there are many instances of the same service in a network then SDPs that integrate searching and selection with routing protocol perform better than having a separated search, selection and routing mechanisms. In such cases the search success increases with the increase of service instances. The

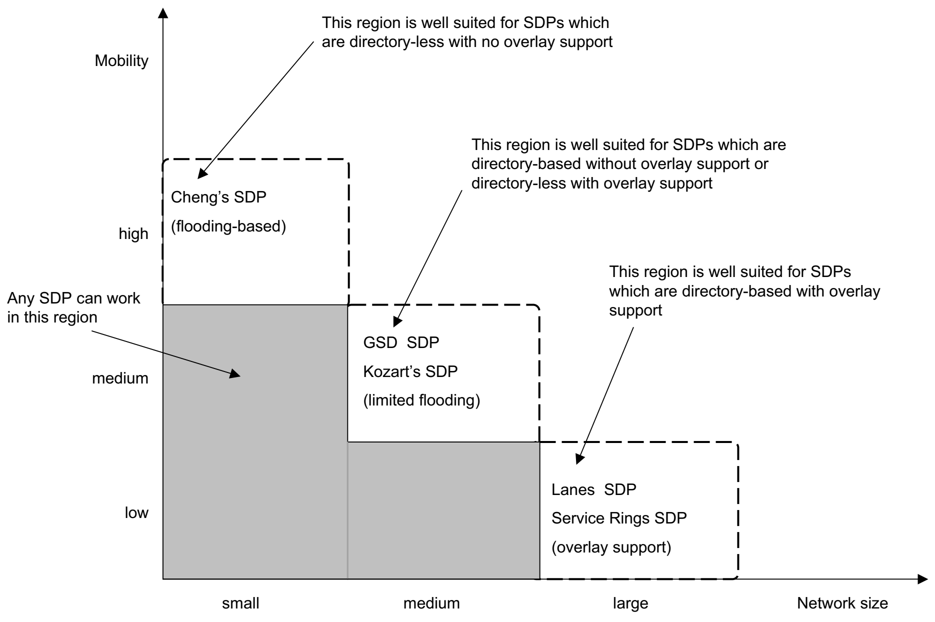

Mobility
high
medium
low
small
medium
large
Network size
This region is well suited for SDPs which are directory-less with no overlay support
Cheng's SDP
(flooding-based)
This region is well suited for SDPs which are directory-based without overlay support or directory-less with overlay support
GSD SDP
Kozart's SDP
(limited flooding)
This region is well suited for SDPs which are directory-based with overlay support
Lanes SDP
Service Rings SDP
(overlay support)
Any SDP can work in this region

Figure 2.3: Suitability of SDPs

other advantage is the reduced message overhead and less latency in searching the suitable services. For example the Field Theoretic SDP [59] is very suited for an environment with multiple service instances.

2.9 Challenges and Open Problems

In this chapter we surveyed SDPs for multihop MANETs. The summary of analysis is shown in tables of Figures 2.4 and 2.5. We now summarize some of the most challenging issues and open problems.

- Often the existing SDPs are built at application level or some are proposed at network level. These SDPs do not exploit the broadcast nature of the wireless medium. We need to look into new search techniques that exploit the broadcast to become aware of services in the network by just listening (eavesdropping) to the traffic generated by the other nodes.

- SDPs using overlay networks are scalable and can work for large networks as compared to SDPs that do not form overlay. But the constraint of overlay based SDPs are that the searching efficiency decreased with increasing mobility. This is due to the overlay maintenance overheads. It is still open an open problem to see which type of overlay suits MANET environments under high mobility.

- There are no standard performance metrics which can be used to compare different SDPs. Different researchers often define there own metrics and measure performance of their proposed SDPs with respect to these metrics. It becomes impossible to have a realistic comparison of different SDPs.

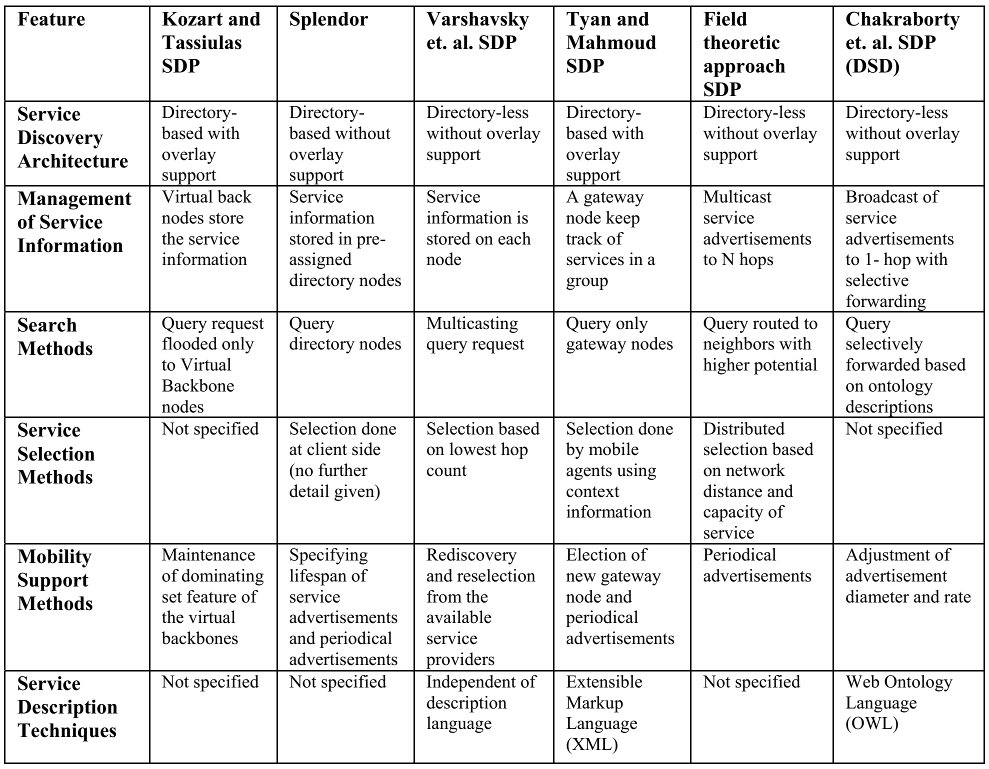
- In MANETs SDPs one of the challenging issue is to limit the network traffic. For this purpose often limited flooding, multicasting or overlays are used. Another less explored method is the replication of services. We need to see how replication of services can help to limit the network traffic. In this regard there are issues like consistency of services, number of replications and placement of replicated services in the network.

- In peer-to-peer networks, random walk has been used for searching whereas in MANET environment random walk has not been explored much for searching and service discovery. One critical aspect is to reduce the number of hops. To reduce number of hops, biasing mechanisms can be used. It is still an open problem to find efficient biasing mechanism, in terms of number of hops, specially suited to wireless mobile environments.

Service discovery in mobile and pervasive environments is an important and an active field of research. Still there are many challenges that need to be resolved before SDPs for multihop MANETs can be made practical.

Feature	**Cheng and Marsic SDP**	**GSD**	**Allia**	**Konark**	**Service Rings**	**Lanes**
Service Discovery Architecture	Directory-less without overlay support	Directory-less without overlay support	Directory-less with overlay support	Directory-less without overlay support	Directory-based with overlay support	Directory-based with overlay support
Management of Service Information	Stores information of local services and multicast service advertisements	Multicast of service advertisements to N hops	Broadcast of service advertisements to 1- hop (immediate neighbors)	Multicast service advertisements	SAP keep information of services in the Service Rings	Nodes in a lane have replicated directory
Search Methods	Multicast query request to a multicast group	Query request selectively forwarded based on semantic information	Query request first multicasted to alliance member and then to other alliances	Multicast query request	Service Access Points	Anycast query request to lanes
Service Selection Methods	Not specified	Not specified	Not specified	Not specified	Not specified	Not specified
Mobility Support Methods	Multicast service advertisements	Adjustment of service advertisement diameter and rate	Adjustment of advertisement rate and alliance diameter	Specifying lifespan of service advertisements and then periodical advertisements	Repair service rings damaged due to mobility	Maintenance of lane structure
Service Description Techniques	Not specified	DARPA Markup Language and DAML	Independent of description language	Extensible Markup Language (XML)	Any language that has a description and summarize functions	Independent of description language

Figure 2.4: Analysis of SDPs with respect to different feature



Feature	**Kozart and Tassiulas SDP**	**Splendor**	**Varshavsky et. al. SDP**	**Tyan and Mahmoud SDP**	**Field theoretic approach SDP**	**Chakraborty et. al. SDP (DSD)**
Service Discovery Architecture	Directory-based with overlay support	Directory-based without overlay support	Directory-less without overlay support	Directory-based with overlay support	Directory-less without overlay support	Directory-less without overlay support
Management of Service Information	Virtual back nodes store the service information	Service information stored in pre-assigned directory nodes	Service information is stored on each node	A gateway node keep track of services in a group	Multicast service advertisements to N hops	Broadcast of service advertisements to 1- hop with selective forwarding
Search Methods	Query request flooded only to Virtual Backbone nodes	Query directory nodes	Multicasting query request	Query only gateway nodes	Query routed to neighbors with higher potential	Query selectively forwarded based on ontology descriptions
Service Selection Methods	Not specified	Selection done at client side (no further detail given)	Selection based on lowest hop count	Selection done by mobile agents using context information	Distributed selection based on network distance and capacity of service	Not specified
Mobility Support Methods	Maintenance of dominating set feature of the virtual backbones	Specifying lifespan of service advertisements and periodical advertisements	Rediscovery and reselection from the available service providers	Election of new gateway node and periodical advertisements	Periodical advertisements	Adjustment of advertisement diameter and rate
Service Description Techniques	Not specified	Not specified	Independent of description language	Extensible Markup Language (XML)	Not specified	Web Ontology Language (OWL)

Figure 2.5: Analysis of SDPs with respect to different feature

Chapter 3

Distributed Search in Wireless Networks

In the previous chapter we surveyed and discussed different service discovery protocols in MANETs. We gave a general frame work for the such protocols. All the protocols discussed employ non-probabilistic methods for searching and related aspects, like registration of services. These protocols in theory guarantee that if a service is present in the network it will be found with probability 1. Such deterministic approaches are less scalable especially in case of high mobility of nodes in the network. In this chapter we shall discuss probabilistic approaches to search. Such protocols have to deal with two main issues. The first one is the efficient diffusion of query message in the network and second is controlling the degree of diffusion and terminating search. The later mentioned issue is often neglected but it importance is even more in resource scarce networks. A service can be just a few hops from the search initiating node then there is no need to search the whole network and the search should terminate, for example, when the first instance of the service is found instead of continuing to search the whole network.

We shall discuss three main approaches, namely flooding, gossiping and Random Walk (RW) used as a search tool and how one approach improves over the other with respect to efficient diffusion and terminating search. Flooding is often used, as we have also seen in the chapter 2 for search. Flooding suffers from two main problems. The first one is broadcast storm problem and second is the efficient control of the diameter of flooding. There are many approaches to address these issues which give fairly reasonable reliability while taking care of these two issues. One of the methods is to use gossiping. In one type of gossip protocol it has been shown that it requires 60 % of number of messages as compared with simple flooding [75]. Gossiping although improves on flooding as far as number of messages are concerned but the problem of selecting gossip probability, reliability under high mobility and terminating the gossip process in case of service is found remain issues. RW methods have been suggested to solve many problems in wireless networks. The advantage of RW is that it requires no topology information which suits very well for high mobility scenarios. On the other hand search using pure or unbiased RW is very inefficient due to loop formation. To reduce loop formation many biasing mechanisms have been suggested. It has been seen that the with proper biasing mechanism and/or replicating the target, the estimated number of transmissions in case of RW can be even less than that of flooding and gossiping with the added advantage of having more control over terminating the search. We shall study these in the forth coming chapters.

Let us now discuss different distributed approaches used for search in wireless ad hoc networks. Our purpose is not to survey the field, instead we shall highlight problems in each approach and how these problems have been tackled in some of the important works. We shall first have a brief survey of flooding and gossiping approaches used for search and then dwell on RW search methods.

3.1 Search by flooding

Flooding is the mechanism by which a node, receiving flooded message for the first time, rebroadcasts the message once to all of its neighbors [78]. The neighbors in turn relay to their neighbors and so on until the message has been propagated to the entire network. Note that flooding is different for broadcast in which the transmission is received by all nodes within transmission range of the broadcasting node. The algorithm for simple flooding is shown in the algorithm 1.

Algorithm 1: Simple flooding algorithm on a node *i*

```
1 Upon reception of query packet
2 if query is received for the first time and node i $\neq$ destination then
3     broadcast query
```

3.1.1 Broadcast storm problem

Every node receives the message from each neighbor within transmission range, except when messages are lost due to contention and collisions. This problem is known as the broadcast storm problem [72], which leads to a large number of redundant messages. Especially in CSMA/CA based networks, it has been shown that the performance of simple flooding depends on the average number of neighbors (neighbor degree) [60]. As the neighbor degree gets higher [72], the simple flooding suffers from the increases of:

1. redundant and superfluous packets
2. the probability of collision
3. congestion of wireless medium

The broadcast storm problem results in wastage of scarce resources such as bandwidth and power and cause contention, collisions and thus additional packet loss. Most of the research in flooding is focused on addressing this problem.

Reliability generally involves sending a greater number of redundant messages. These are needed so that all nodes received the message and to recover from packet loss but on the other hand it incurs a higher message overhead. Reducing the number of redundant broadcasts results in a lower degree of reliability. Thus in any flooding mechanism, one must balance reliability against message overhead.

Broadly there are two classes of approaches to deal with the broadcast storm problem in flooding: overlay based approaches and probabilistic approaches, which is also know as gossiping. We shall discuss these approaches in detail in the next sections.

3.1.2 Overlay based approaches to flooding

In the overlay based approaches the topology or neighborhood information is exploited. In this case only a subset of nodes broadcast, instead of all, to guarantee that the query message reaches to all nodes of the network.

An overlay based approach superimposes a routing structure onto the ad hoc network in support of flooding. A node decides either to rebroadcast a flooded packet, or to only process and then drop it depending on the position of the node in the overlay topology. While overlays networks reduce the message overhead of flooding and increase reliability, they need to be reconfigured when connectivity changes due to node mobility. Restructuring the overlay networks not only increases overhead but also the likelihood that messages will be lost, and thus may decrease coverage of the flooding protocol. Following are some of the important variations in this overlay-based approaches.

- Ni et al. [72] propose an overlay structure that form clusters of nodes. Each cluster has a cluster head. Only the cluster head rebroadcast and remaining nodes drop the query messages.

- Gandi et al. in [41] provide a mechanism for low-latency flooding by minimizing the collisions and interference. They construct a multicast tree and compute a rebroadcasting schedule such that the expected rate of collisions will be low.

- The approaches given in [35] [7] are based on the approximation of (minimal) connected dominating sets (MCDS). As explain previously in chapter 2 a dominating set (DS) contains a subset of all nodes such that every node not in the DS is adjacent to one in the DS. Thus, a DS creates a virtual backbone that can be used to efficiently flood messages. It has been shown that the creation of an MCDS is NP-complete. Thus, most approaches attempt to find a sufficiently good approximation to a MCDS. A number of approaches for forming MCDS rely on two-hop neighbor information to select nodes that rebroadcast the message. These approaches require that hello messages containing neighbor information are exchanged between the nodes. For instance, in the Double-Covered Broadcast (DCB) [61], a node collects information about the two-hop neighbor set. Among its one-hop neighbors it then picks nodes that rebroadcast the message (called forward node) such that (1) the rebroadcast by the forward node covers the two-hop neighbors, and (2) the one-hop neighbors that are no forward nodes are within range of at least two rebroadcasts by forward nodes. The reception of the message by the forward node is implicitly acknowledged when n overhears the rebroadcast.

- The scalable broadcast algorithm (SBA) [75] employ a simple idea that a node does not need rebroadcast a message if all its neighbors have been covered by previous transmissions. It is shown in [75] that about 60% duplicate messages can be saved compared with that of flooding. This algorithm uses two-hop neighbor knowledge. With node mobility, the two-hop neighbor sets need to be updated frequently. Otherwise, the neighbor sets become outdated and reliability drops (as observed in [61]).

There is a lot of flooding protocols in literature. We have given here a couple of interesting approaches to reduce broadcast storm problem using overlay formation. The interested reader can see the references for details.

3.1.3 Flooding control

Flooding usually covers all the nodes in a network even if the flooded query packet reaches the target. This waste scarce resources like bandwidth and energy. Also often we are interested in searching in limited portions of the network, instead of flooding the whole network. For example we are interested in a service that should be near to the querying node or it is present in a specific region of the network. Following are different methods to control flooding.

- Flooding can be controlled by setting the hop limit called the Time-To-Live(TTL) of the query packet, which guarantees that the query will not go beyond a certain number of hops.
- Another approach to control flooding is to gradually increase TTL. Such a scheme is called a expansion ring scheme [95]. Several researchers have studied such schemes [57] [23] [24] [95] [28]. There are mainly two variants of this scheme, one used in DSR [50] and other in AODV [76]. In both of these protocols, nodes search for a target gradually in order to avoid flooding the entire network. In DSR the procedure is relatively simple. A node searches its one-hop neighbors first, and if the target is not found, the node then searches the entire network. In AODV a node increases its searching radius linearly from an initial value until it reaches a predefined threshold. After that, a network-wide search has to be performed. Both of these expansion ring schemes assume that there are route caches residing in the nodes. When the route caching condition is weak and node mobility is high, these schemes will not reduce the search overhead compared with flooding the entire network once as effectively as expected. Instead, an improper utilization of the expansion ring scheme will lead to even more overhead [95].
- Flooding can also be limited to a set of nodes defined by a geographical area. Such type of flooding is also called geocast flooding [55]. A node that receives the flooded message only rebroadcasts it if it is within the specified area. Each node is assumed to be location-aware, for example, using GPS. The cost and energy requirements of such devices still confine the use of such protocols.

We have seen that none of the above methods give a complete control over flooding. We just get a loose control with which it is not possible stop the flooding process as soon as the desired service is found. This control often comes at the cost of device complexity and its more energy requirements.

3.1.4 Critical remarks on using flooding for search

We have seen that overlay networks are used to mitigate the broadcast storm problem. Although these approaches reduce redundant broadcasts but introduce complexity in algorithms for overlay maintenance. The reliability comes at the cost of collecting topological information and maintaining the overlay structure The problem here is that if nodes have low quality connections to neighbors and/or are in motion, the overlay structure must be adapted. As a consequence, a high rate of management messages may be required, and if a flooded message is propagated while the overlay is out of date, that message may experience a high loss rate. In the worst case, the system might end up in a state of churn, constantly adapting the overlay but never managing to achieve the high quality of flooding that the overlay is intended to support.

We also see that although there are some mechanisms to terminate the search, like using TTL or using expansion ring schemes, but these are not very efficient especially in case of high mobility.

3.2 Search by Gossiping

Simple flooding, as we have seen in the previous section, suffers from the problem of redundant message reception, once per neighbor. Even in a reasonably connected network, the same message is received multiple times by every node, which is inefficient, wastes valuable resources, and can create contention in the transmission medium.

Gossiping is a probabilistic approach to flooding [78]. A simple gossip protocol can be as follows. A source sends the query with probability 1. When a node first receives a query packet, it broadcasts the query to its neighbors with probability p and discards the query with probability $1 - p$. If the node receives the same query again, it is discarded. The algorithm 2 shows the process of gossiping. Note that $p = 1$ leads to simple flooding.

Algorithm 2: Simple Gossiping algorithm on a node i

```
1 Upon reception of query packet
2 if query is received for the first time and node $i \neq$ destination then
3     broadcast query with probability $p$
```

Gossip reduces the number of broadcasts but when applied naively the reliability of the protocol also decrease. The nodes with fewer neighbors may fail to receive a flooded message. Most commonly, gossip protocols use information from received broadcasts to adaptively determine the forwarding probability. These protocols are naturally suitable for mobile ad hoc scenarios, as they do not need to maintain any kind of overlay structure that might change and need to be adapted in the event of mobility or other topology changes.

There are a number of variants of simple gossip protocol. For example in [72] it is shown that the additional coverage gained by rebroadcasting decreases with the number of overheard rebroadcasts and decreasing distance to neighboring rebroadcasting nodes. This leads to approaches in which the rebroadcast decision is either based on the number of already overheard rebroadcasts, or on the distance or location of the overheard rebroadcast's sender. Such approaches may introduce high latency due to time taken to collect the statistics which delay the rebroadcast decision. In [91], Tseng et al. extend earlier approaches in [72] to allow nodes to dynamically adapt threshold values such as the rebroadcast counter.

Zhang and Agrawal in [96] propose an approach in which they adjust p according to the information collected by the counters, instead of using a static rebroadcast probability p. While this makes p adaptable, it becomes dependent upon other fixed parameters that need to be carefully selected (e.g., timeouts).

Dynamic Gossip [82] assumes uniformly distributed neighbor nodes. This approach uses one-hop neighbor density information to adjusts the rebroadcast probability p of a node. Density information is collected using a relay-ping method.

3.2.1 Bimodal behavior of gossip protocols

Krishnamachari et al. [58] have studied the behavior of gossip in random geometric graph, a widely accepted model for MANETs. They showed the existence of a critical probability p_c such that for a gossiping probability $p < p_c$ the gossip process quickly dies out, while for p higher than p_c the whole network is covered. This transition phase phenomenon was first observed in percolation theory by Broadbent and Hammersley [16].

Sasson et al. [81] theoretical and experimental results indicate that there does not exist a sharp threshold if collision are taken into account [81]; moreover, with an optimal gossiping probability maximum efficiency of the search is guaranteed. Hence, increasing p much beyond p_c is not very useful.

Haas et al. [45] studied the phase transition phenomenon in ad hoc network and showed the same bimodal delivery distribution. Their simulations reveal that either almost every node receives the message, or virtually none. To reduce the likelihood of the latter case, they explore a variety of approaches, such as adapting the rebroadcast probability to the density or the distance to the flooding source. They showed that by using appropriate heuristics up to 35% message overhead can be saved as compared to flooding.

Beraldi in [14] has presented a polarized gossip protocol for path discovery in MANETs. The protocol has a variable gossip probability, which is high enough to sustain the spreading process in the direction of the target. The gossiping probability of a node is determined by the difference between its proximity to the destination and the proximity to the destination of the node from which the message was received. The proximity is estimated with out any GPS mechanism and just using periodic beacons for determining the time

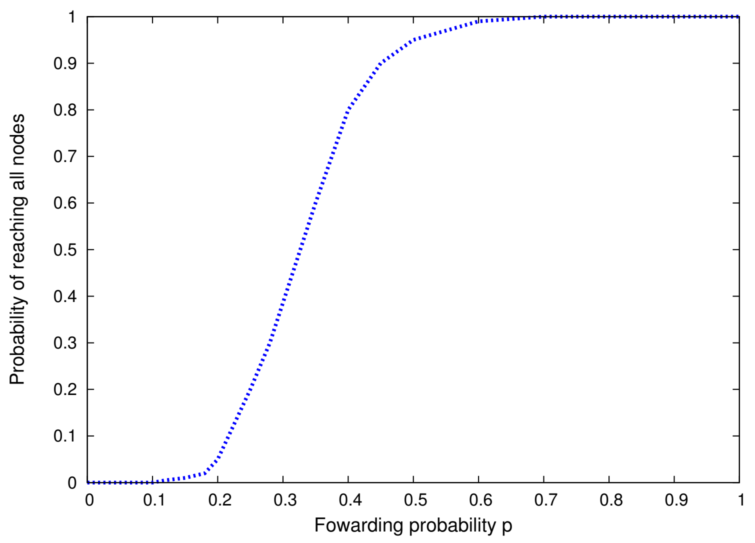


Figure 3.1: Phase transition in gossiping

elapsed since a node met the destination and the dwell time of a node with the destination.

Any approach that bases rebroadcast decision on observation of neighbors and on overheard broadcasts is at risk of using stale information if nodes might move before the information is used. MANETs, of course, can have a high degree of mobility, hence neither of these approaches is ideal [78].

3.2.2 Critical remarks on using gossip for search

After having a brief overview of different approaches to gossip we see that although it is efficient than simple flooding due to less number of packets generation but there are two main disadvantages when gossip is used for search in wireless ad hoc networks. The first disadvantage is that it is not possible to find an optimal value of gossip probability p without knowing the network topology and node density. Also as the topology changes over time in case of mobile network, a value of p which is suitable at one time may become unsuitable over the network lifetime [96], leading to either more redundant massages or dying out of the gossip process.

The second disadvantage is that if the gossip probability $p > p_c$ then the query is transmitted to whole of the network and there is no mechanism to terminate the gossip, for example, when a service is found. It can happen that a service is just a few hops from the source node. Even in such a case the whole network transmits and receives the query, leading to redundant transmissions and thus wastage of scarce resources like energy and bandwidth. There is no mechanism to terminate the gossip process when a specific portion of the network is searched or when a service is found.

In the next section we shall discuss about how we can use RW for search and how it addresses the above mentioned problems in flooding and gossiping.

3.3 Search by random walk

Given a graph and a starting point, we select a neighbor of it at random and move to this neighbor. We then select a neighbor of this point at random and move to it and so on. The random sequence of points selected this way is a RW on the graph [83] [8] [80] [15].

When random walk is used for search a node, which is interested in a particular service sends a query message. This query message plays the role of the walker that randomly moves on the graph formed by ad hoc networks and eventually approaches the service. This is expressed by algorithm 3.

Algorithm 3: Simple random walk algorithm on a node i

```
1 Upon reception of query packet
2 if node i = destination then
3     terminate algorithm
4 else
5     select neighbor at random
6     forward query to the selected neighbor
```

RW has been studied theoretically extensively. The interested reader is referred to [62] in which the theory of RWs on a graph has been explained in detail.

There is also a significant body of theoretical literature on RWs as querying mechanisms [6][62][18]. Regarding search methods in unstructured networks like P2P networks, RWs approved to be a promising alternative in general and especially in large scale networks [10] [34] [43] [46] [64] [87] [97], where the standard method by flooding may be prohibitive. Following are some of the important protocols that use RW for searching.

- Avin and Brito [8] have shown that simple RWs can be used for efficient searching due its inherently load-balancing property and also because RW partial cover show good scaling properties. In particular they have shown that RW is more efficient than cluster head on grids and robust that the spanning tree under dynamics.

- The AQUIRE protocol [80] combine RW with look-ahead with tunable controlled flooding. AQUIRE adopts a look ahead mechanism in which at each step, information is collected from every node at distance at most d hops away from the current one, and then the walk jumps to a node at distance d. The goal of such a mechanism is to reduce the impact of the random steps. Their analysis, however, only investigates how long it takes for a RW to reach a node with specific information, assuming a uniform distribution of data values over the nodes in the network. They show that in the presence of replicated data such a hybrid approach can perform better than flooding and even expanding-ring based approaches.

- The rumor routing algorithm [15] is a hybrid push-pull mechanism that uses multiple RWs from the sources as well as the sinks. Both the base station and a node that observes some event of interest start RWs that leave traces on the nodes that they visit. Whenever the traces intersect, a route is discovered and the information can be transferred between the two points. Shakkottai [84] has analyzed different variants of random-walk-based query mechanisms and concludes that source and sink-driven sticky searches (similar to rumor routing) provide a rapid increase of query success probability with the number of steps.
- One of the drawbacks of RWs is the significant delay that they encounter. In [100] exploit heterogeneity to reduce the required number of steps which results not only in the reduction of cost but also the delay. They showed that heterogeneity allows random-walk-based queries to enhance their performance.
- Servetto and Barrenechea [83] proposed and analyzed the use of constrained RWs on a grid for performing load-balanced routing between two known nodes.

3.3.1 Other applications of RW

RW approaches are now being widely used in distributed systems. A very brief account of RW applications other than searching is given below.

- RW have been used in unstructured P2P Networks [31] [4] [26], for hybrid application overlays [90], for distributed model checking [86], and for index quality determination for the world-wide web [49].
- Most recently, Alanyali et al. [5] have proposed the use of RWs in energy-constrained networks to perform efficient distributed computation of a class of decomposable functions which are useful in computing certain kinds of aggregates.

- In mobile agent based routing, the mobile agents perform a RW or a variant of the RW while searching for the destination. In Ant-Net, loop-erased RWs are used by the mobile agents [36]. Mobile agents using RW have been proposed for providing membership services for ad-hoc networks by Dolev et al. [38].

- As a sampling technique, RWs have been used for providing membership services in ad hoc networks [38] [12] [38] that provide the nodes in the network with a view of the other nodes and that are used by various applications such as location services, peer sampling services and random overlay constructions[12].

- Bar-Yossef et al. [12] develop a membership service for ad hoc networks based on RW using highest degree. They show that the performance of such membership service is superior to other existing membership services based on gossiping or flooding[12].

3.3.2 Comparison with flooding and gossiping

After having discussed all the three main distributed approaches for searching, we compare RW with the other two approaches as follows.

Scalability aspect: RW approaches are scalable as compared to flooding. In RW there is no broadcast storm problem whereas in flooding we have seen that to mitigate the broadcast problem and make flooding scalable overlays are formed. These overlay require overhead messages for overlay maintenance. In RW approaches there are no such kind of overlay formation which makes RW based search algorithm more scalable.

Suitability for high mobility: Due to overlay formation and maintenance in flooding and due to the precise setting of the forwarding probability, which depends on the topology of the network, in gossip, the reliability of search based on these approaches is decreased with the increase of mobility of nodes in the network. The RW based search do not require any topology information and hence most suitable for high mobility scenarios.

Reliability of continuation of RW based search: As we know RW is a sequential process and nodes are searched one after the other at each step. If due to any reason the query is not forwarded to another node, the RW walk will stop before the service is found or termination condition is reached. Thus the reliability of one step of RW become very critical. To implement a highly reliable one step of RW distributed selection mechanism [70] can be adopted, which is also part of the work done in this thesis. We see that there is no such problem is flooding and gossiping.

Latency aspect: One of the inherent problems of RW approach is it high latency as it is a sequential process whereas flooding and gossip based search have low latency. The latency in RW is proportional to the number of steps required to accomplish the task. In many cases this will be of the order of the size of the network. One possible idea to mitigate this problem is to perform RWs in parallel [8]. The other approach to deal with latency is to reduce the number of steps required to search by using a biased selection mechanism [55] [66]. We shall discuss this issue of biasing random selection in detail later.

Control over search termination: We need to terminate the search process as soon as the query finds the required destination. We have seen that flooding has a limited control over termination and gossip has no control. The RW based search approaches, on the other hand, can not only be terminated but also can have a variety of different termination rules depending on the particular needs. For example terminating RW when first or second etc instance of a service is found or when a specific portion of the network is searched whereas such type terminations are not possible in flooding or gossiping.

Cost of search: We have discussed that we can approximate the cost of search with number of query transmissions. On this criteria RW based search leads to more cost as compared to flooding and gossiping [66]. It has been discussed in [66] how to reduce the number of query transmissions using different biasing methods. It is shown that cost of biased RW combined with service replication leads to less cost in RW based search as compared with flooding.

3.4 Formal aspects of RW

Random walks are modeled using Markov chains. We shall, first briefly describe Markov chain and related concepts that are used in RW and then discuss different aspects of RW that are important to understand the thesis.

3.4.1 Metrics of random walk

There are many nice properties of RW that make it useful for many applications. An important property of RW exploited for searching is that it eventually visits all nodes in a network. If a query packet starts RW in search of a node having a certain property and if such a node is in the network, it will be eventually found. This will be explained later. Let us now define metrics of a RW that are most important with respect to the thesis. For details see the reference [62]. We assume that the RW is on a connected graph $G = (V, E)$ with n nodes and m edges and for any pair of vertices $i, j \in V(G)$.

Definition:(Hitting time) Hitting time h_{ij} is defined as the number of steps taken by a RW to reach j for the first time starting from i. The mean hitting time $E[h_{ij}]$ is the expected value of h_{ij} [62].

Definition:(Commute time) Commute time k_{ij} is defined as the number of steps taken by a RW to return to i for the first time starting from i via j. It is thus,

$$k_{ij} = h_{ij} + h_{ji}$$

The mean commute time $E[k_{ij}]$ is the expected value of k_{ij}.

Definition:(Cover time) Cover time $\mathcal{C}$ is defined as the number of steps taken by RW to visit all n nodes. The expected cover time $E[\mathcal{C}]$ is the mean of $\mathcal{C}$. If starting node is not specified then we mean starting from that node from which the expected cover time $E[\mathcal{C}]$ is maximum [62].

Matthews' theorem Matthews' theorem given below relates cover time with the hitting time by providing upper and lower bounds on the cover time C in terms of maximum hitting time [8]. Suppose a RW is performed on a graph G with n vertices. Let h_{max} be the maximum hitting time given by the maximum of h_{ij} and h_{min} be the minimum hitting time given by minimum of h_{ij} over all ordered pairs of nodes i and j

Theorem 1 (Matthews' Theorem)**.** *For any graph G,*

$$h_{min}.H_n \leq \mathcal{C} \leq h_{max}.H_n$$

where

$$H_n = \sum_{i=1}^{n} 1/i = ln(n) + \Theta(1) \tag{3.1}$$

gives the $n-th$ harmonic number. It follows from the definition of cover time and and the upper bound in the Matthews' theorem that [51]

$$h_{max} \leq \mathcal{C} \leq h_{max}.H_n$$

Thus the h_{max} approximates the cover time $\mathcal{C}$ within a factor of $\ln(n)$. The bounds given by Matthews' theorem are not always tight as in case of nodes placed in a line $\mathcal{C} = h_{max}$. The cover time for specific graphs vary from the best case of $\mathcal{O}(nln(n))$ to the worst case of $\mathcal{O}(n^3)$. The best cases are dense and highly connected graphs like complete graph and d-regular graphs with $d > n/2$. As the connectivity decreases the cover time increases, for example in a line the cover time is $\mathcal{O}(n^2)$ [8].

Definition:(Cost of search) Let there be a connected graph $G = (V, E)$ with n nodes and m edges. Search in a graph is defined as finding a path from a start node to a destination node. The cost of a search κ is the total number of edges traversed in locating the destination node. In terms of energy the cost of search is the total energy consumed by all the nodes of the system in searching for a destination, with the assumption that destination node is accessible. If we assume that each transmission last for a specific constant time cost of search translates to the total number of transmissions in search a target node j starting from node i. For the case of RW, the estimated cost of search $E[\kappa]$ in finding the target j when starting from i is the estimated number of steps to reach the target node.This is given by

$$E[\kappa_{ij}] = \sum_{k} kPr\{E[h_{ij}] = k\}$$

where $E[h_{ij}]$ is mean hitting time defined before. Let K be the cost of search starting from a node i to a target that can be on any node in the network, which has a size of N nodes, then

$$E[K_i] = \frac{1}{N}\sum_{j=1}^{N} E[\kappa_{ij}]$$

Property:(Steady state probability distribution) Consider a RW over a connected graph $G = (V, E)$ with n nodes and m edges. Then, for any node $q \in V$, the steady state probability of observing the RW to be in state q, $\pi(q)$, is given by

$$\pi(q) = \frac{d(q)}{2m}$$

If the graph G is d-regular, i.e., $m = dn/2$, the stationary distribution is a uniform distribution [62].

$$\pi(q) = \frac{d(q)}{2m} = \frac{1}{n}$$

3.4.2 Markov chain

Definition:(Markov chain) A Markov chain is a random process $\{X_k\}_k \geq 0$, such that the probability distribution of X_{k+1} is determined only by X_k, the present step. Formally,

$$Pr(X_{k+1} = x_{k+1}|X_k = x_k, ..., X_1 = x_1, X_0 = x_0) = Pr(X_{k+1} = x_{k+1}|X_k = x_k)$$

The definition states that only the present state gives any information of the future behavior of the process. Knowledge of the history of the process does not add any new information.

Markov chain is usually described by the probability transition matrix P, which is defined as follows.
Definition:(Transition probability) If at time k the Markov chain is in state is i, then at step $k+1$ the probability of moving to state j, p_{ij} is

$$p_{ij} = Pr(X_{k+1} = j | X_k = i)$$

The probability transition matrix P satisfies

$$\begin{aligned} 0 \leq p_{ij} \leq 1, \quad & i,j = 0,1,...,n-1 \\ \sum_{j=0}^{n-1} p_{ij} = 1, \quad & i = 0,1,...,n-1 \end{aligned}$$

where n is the number of states

The k step transition probabilities are given by the following theorem.

Theorem 2. *Let P be a $n \times n$ probability transition matrix of a Markov chain and let P^k be the k-fold matrix product. The ij-th entry $p_{ij}^{(k)}$ of the matrix P^k gives the probability that the Markov chain is in state j starting from state i after k steps.*

Let $\pi^{(k)}$ represent the probability distribution after k steps, which is a vector describing probabilities of all states of the Markov chain after k steps, then the probability distribution after $k+1$ steps is given by

$$\pi^{(k+1)} = \pi^{(k)} P$$

Theorem 3. *Let $\pi^{(0)}$ represent the starting distribution, then the probability that the chain is in state i after k steps is the i-th entry of the vector*

$$\pi^{(k)} = \pi^{(0)} P^k$$

Definition:(Stationary distribution) A probability distribution π is said to stationary or steady state if π satisfies

$$\pi = \pi P$$

Definition:(Irreducibility) A Markov chain is called irreducible chain if it is possible to go from every state to every state, not necessarily in one step, that is,

$$\forall i, j \qquad \exists k : p_{ij}^{(k)} > 0$$

We shall see later that a RW on a connected graph can be represented by an irreducible Markov chain, which means that a RW can move from every state to every state and there is a non-zero probability of RW being in any state after a certain number of steps. This leads to the conclusion that RW will eventually visit all nodes.

Definition:(Recurrent state) A state i is said to be recurrent if the probability of return to i having started from i is 1, that is,

$$Pr(X_n = i \text{ for some } k \geq 1 | X_0 = i) = 1$$

If a state is not recurrent then it is called transient state.

Definition:(Mean first passage) For an irreducible Markov chain that started from state i, the first passage time is the number of steps to reach state j for the first time. Formally we have

$$T_{ij} = min\{k \geq 0 : X_k = j | X_0 = i\}$$

The mean first passage is given by

$$\mu_{ij} = E[T_{ij}]$$

Definition:(Mean recurrence time) For an irreducible Markov chain that started from state i, the recurrence time is the number of steps to return to state i for the first time. Formally we have

$$T_i = min\{k > 0 : X_k = i | X_0 = i\}$$

The mean first passage is given by

$$\mu_i = E[T_i]$$

Definition:(Recurrent state) A recurrent state i is called

$$\mu_i = \begin{cases} \text{null recurrent} & \text{if } \mu_i = \infty \\ \text{non-null (or positive) recurrent} & \text{if } \mu_i < \infty \end{cases}$$

Definition:(Periodicity) A state i is said to be periodic and have period T, if $p_{ii}^{(k)} = 0$ whenever k is not divisible by T, and T is the largest integer having such a property. Formally the period of a state i is defined as

$$T = gcd\{k \geq 1 : p_{ii}^{(k)} > 0\}$$

A chain is periodic if some state has periodicity greater than 1. A state having period 1 is called aperiodic. If all the entries of the a transition probability matrix are positive then the Markov chain is aperiodic. However aperiodicity does not necessarily require all entries to be positive. Note that periodicity of a chain can be eliminated by adding self loops.

Theorem 4. *An irreducible chain has a stationary distribution π if and only if all its states are positive recurrent and aperiodic. In this case as $k \rightarrow \infty$, $p_{ij}^{(k)}$ converges a unique stationary distribution π having π_j entries, as given by*

$$\lim_{k\rightarrow\infty} p_{ij}^{(k)} = \pi_j = \frac{1}{\mu_j}$$

where i, j are any states and μ_j is the mean recurrence time of state j.

Note that there is no assumption on the starting distribution; the chain converges to the stationary distribution regardless of where it begins. This means that the Markov chain after a long time forget where it was started from. This property is widely used for sampling from a distribution.

Definition:(Mixing time) Mixing time is defined as the number of steps for a probability distribution to be close to the stationary distribution π and mixing rate is defines how fast the RW converges to its stationary distribution π.

The mixing time and mixing rate determines the efficiency with which a RW may be used to sample over the state space [10].

Definition:(Absorbing state) A state i is said to be an absorbing state if it is impossible to leave it, that is, $p_{ii} = 1$. A Markov chain is absorbing if it has at least one absorbing state and it is possible to reach an absorbing state from every other state [2].

Theorem 5. *In an absorbing state Markov chain, the probability that the process will be absorbing is 1.*

Before we proceed to explain RW, let us briefly state that graphs can aid in visualization of Markov chains. For example the states of a Markov chain may be represented by vertices of a graph, and one step transitions may be described by directed edges with weights. An irreducible Markov chain can be represented by a corresponding connected graph. In periodic Markov chain with the period $T > 1$ all cycles of the graph are multiples of T. If there is no such T then the Markov chain is aperiodic.

3.4.3 Random walk as a Markov chain

Suppose we have a connected graph $G = (V, E)$ with n nodes, m edges and $i, j \in V$. Let $d(i)$ be the degree of node i. We start from a node i and pick a random node from it neighbors with probability $1/d(i)$ and move to it. We say that a node is visited when we move to it. We continue moving successively from one node to another node selected at random. The transition to the next node does not depend on the previously visited nodes. Clearly this sequence of visiting nodes is a RW which can be modeled by Markov chain. There is not much difference between the theory of random walks and the theory of Markov chain. Every Markov chain can be viewed as random walk on directed graph, if we allow weighted edges.

Let p_{ij} represent the probability of transition from node i to a node j and $P = (p_{ij})_{i,j \in V}$ be the matrix of transition probabilities of this Markov chain, then the ij-th element of this matrix is given by

$$p_{ij} = \begin{cases} 1/d(i) & \text{if } i, j \in E \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

Calculation of mean hitting time:

We can calculate mean hitting time using either steady state analysis or transient analysis. We shall briefly discuss both of these methods as follows.

Steady state analysis

Let p_{ij} is the probability that the random walk moves from node i to j and $\mathbf{P}$ the transition probability matrix. If t is the target node, then the matrix $\mathbf{Q}$ obtained from $\mathbf{P}$ by removing row t and column t contains the information needed for the hitting time computation. The hitting time of t starting from i is the $i - th$ element of the column vector $\mathbf{H}$, where

$$\mathbf{H} = (\mathbf{I} - \mathbf{Q})^{-1}\mathbf{1}$$

In this expression, $\mathbf{1}$ is a column vector all of whose entries are 1 and $(\mathbf{I}-\mathbf{Q})^{-1}$ is the fundamental matrix [2][74]. $\mathbf{I}$ is identity matrix. The computation of the hitting time is usually done numerically.

For a grid the probability transition matrix is formed as follows. Let the points of the grid are assigned coordinates w.r.t. the cartesian axis, with origin at the bottom-leftmost point of the grid. Let $\mathbf{s} = (i, j)$ be a grid point and $||\mathbf{s}_1, \mathbf{s}_2|| = |i_1 - i_2| + |j_1 - j_2|$ the distance between $\mathbf{s}_1$ and $\mathbf{s}_2$. Then, we have:

$$p_{\mathbf{s}_1\mathbf{s}_2} = \begin{cases} 1/d(\mathbf{s}_1) & ||\mathbf{s}_1, \mathbf{s}_2|| = 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

where $d(\mathbf{s})$ is the number of neighbors of point $\mathbf{s}$. $d(\mathbf{s})$ has a value 2 if the node is a corner node, 3 if the node is an edge node but not a corner node and 4 if the node lies inside the edges of the grid.

Transient analysis

It is clear that the hitting time h_{ij} is a random variable, then the mean hitting time is given by

$$\begin{aligned} E[h_{ij}] &= \sum_{k=1}^{\infty} k Pr\{h_{ij} = k\} \\ &= \sum_{k=1}^{\infty} k[Pr\{h_{ij} \leq k\} - Pr\{h_{ij} \leq k-1\}] \end{aligned}$$

We can find the $Pr\{h_{ij} \leq k\}$ using Markov chain in which the state j is the absorbing state. This will give the probability of transition from i to j until step k which is the probability of reaching state j in step $k, k-1, k-2, ..., 1, 0$. Let $p_{ij}^{(k)}$ be the probability of transition from i to j until k steps and $p_{ij}^{(k-1)}$ be the probability of transition from i to j until $k-1$ steps, then the probability of transition from i to j for the first time at step k is given by $p_{ij}^{(k)} - p_{ij}^{(k-1)}$. Thus the mean hitting time is

$$h_{ij} = \sum_{k=1}^{\infty} k\{p_{ij}^{(k)} - p_{ij}^{(k-1)}\} \tag{3.4}$$

Chapter 4

An Efficient Neighbor Selection Protocol for Random Walk in Wireless Ad hoc Networks

4.1 Motivation

Random walk is a special kind of approach for solving many recurrent problems arising in distributed systems. Due to their simplicity random walk based algorithms are specially suitable for dynamic systems. To implement a random walk we need to forward a packet from a node to one of its neighbor node selected at random. Moreover, wireless networks are particularly favorable to implement random walks due to the broadcast nature of the transmissions. Concrete applications of random walk algorithms include membership services [12], group based communication [39], search/query [8], and routing [89].

Random walk is inherently a sequential process, i.e., nodes are visited serially one after the other. A small forward delay is thus an important requirement, especially when the number of nodes in the network is high. Due to the sequential propagation of the walker, the improvement in per hop delay is subject to a multiplicative factor. If we are able to reduce the delay by an amount, say δt, the overall reduction perceived by the end user is $k\delta t$, where k is the average number of steps required before the random walk terminates. For example, the number of steps before the algorithm, described in [39], terminates varies as $k = O(n^3)$, where n is the number of nodes in the network. A reduction in per hop delay is then highly visible.

To implement a random walk based algorithm the usual approach is to design the protocol at the application level. Although this approach is clearly adequate in wired networks, e.g., p2p applications, wireless networks often have different constraints, like to minimize energy consumption.

The critical part of a random walk is the implementation of a step of the walk, i.e., selecting a node at random and then sending the walker to it. This apparently simple problem is not trivial to solve efficiently in dynamic wireless networks. The usual approach to deal with changing neighbors requires the forwarding node to first discover its current neighbors and then forward the walker to one of them selected at random. Application layer neighbor discovery can be implemented straightforwardly by periodic beacons, which is a clear source of inefficiency. Each node is required to announce itself at a rate sufficiently high to be properly detected as soon as it enters the transmission range of the forwarding node. An alternative solution requires the forwarding node to discover the neighbors only when the walker is to be forwarded. The forwarding node in this case engages in a probing phase during which all neighbors are discovered. The problem with this solution is that a step of the walk may require a long delay because in order to reduce collisions the response to the neighbor discovery packet has to be randomized over a sufficiently long time interval.

Contribution: In this chapter we propose a new neighbor selection protocol for random walk for wireless ad hoc networks. This protocol assumes IEEE 802.11 MAC layer but uses only its broadcast primitive and it basic medium access mechanism. We compare the proposed protocol with a unicast based neighbor selection random walk protocol (section 4.3) which uses the basic access mechanism and RTS/CTS collision avoidance mechanism of IEEE 802.11. The proposed protocol is energy efficient, incurs less overheads and delays and is robust under mobility which makes it very suitable for mobile ad hoc networks (MANETs).

The rest of the chapter is organized as follows. Section 4.2 discusses some important background that is needed to understand the presented work. In Section 4.4 we propose and discuss an unbiased random walk protocol. Section 4.5 gives the simulation details followed by the simulations results in Section 4.6. Section 4.7 concludes our work and points out the directions for future work.

4.2 Background

4.2.1 Medium access mechanisms in IEEE 802.11

The protocols presented in this chapter assume MAC IEEE 802.11 standard. Let us first briefly explain IEEE 802.11 basic access mechanism of Distributed Coordination Function(DCF). Suppose a node j wants to send DATA to node i, node j senses the channel to determine whether it is idle. If it is idle for a specified time interval the node j is allowed to transmit, otherwise the transmission is deferred until the ongoing transmission terminates. The deferred time known as backoff time is chosen at random. The backoff time decreases as long as the channel is idle. The node transmits when the backoff time reaches zero. The backoff procedure guarantees fair access to the shared medium to different nodes. The details of the backoff algorithm can be found in [19], [30], [85] and [33].

The DCF is extended with RTS/CTS mechanism [3] to solve the hidden terminal problem. Suppose a node j is interested in sending data to a node i, then the node j broadcasts a special Request-To-Send (RTS) packet. When a neighbor node other than node i receives this packet, it defers it's transmission for a specified period. When node i receives this packet it broadcasts a reply message called Clear-To-Send (CTS) to give permission to node j to send DATA. Any other node that receives CTS defers transmission until it receive ACK packet. In the third phase node j broadcast DATA to node i. In the fourth phase node i replies to node j with ACK packet. When node j receives this packet, it will stop its retry timer and if other nodes receive this packet they will resume their scheduled transmissions [13]. The RTS/CTS mechanism and the hidden terminal problem are further explained in [47], [42] and [25].

4.3 Standard protocol for one step of RW

A Random Walk (RW) on a given graph is a random sequential visits that a token performs on the nodes of the graph. There are several nice properties that make the RW appealing for many concrete applications. An interesting property of RW is that it eventually visits all nodes of the graph. Thus, if a node being searched is in the network, it will be eventually found. Please see chapter 3 for details.

When RW is used as a search algorithm the token is implemented as a query packet which contains the description of the node that is being searched. The query packet is forwarded from a node to one of its neighbor selected at random. This process is continued until either the target node is found or a termination condition like TTL expire, is met.

Algorithm 4 shows the pseudocode of this protocol assuming that the underlaying network topology is time variant. When a query packet arrives at a particular node, the algorithm terminates if it is a destination node (lines 2-3), otherwise the node starts a neighbor discovery (*nd*) phase (line 11 to 19). During the *nd* phase the node discovers its current neighbors. This is required since the neighbors change over time. The *nd* phase is initiated at line 6 by sending a *nd* packet. When a neighbor node receives a *nd* packet, it simply replies by unicasting the *nd* packet to the node that initiated the

Algorithm 4: RW-CS on a node i

```
1  Upon reception of query packet
2  if $query.destNode = i$ then
3      terminate algorithm
4  if $query.destNode \neq i$ then
5      clear neighborInfoTable
6      broadcast nd packet
7      set selectionDelayTimer(t2)
8      Upon selectionDelayTimer expire
9          select randomly from neighborInfoTable
10         send(query) to the selected neighbor // unicast

11 Upon reception of nd packet
12 if nd.bcastNode = $i$ then
13     store $nd.bcastNode$ in neighborInfoTable
14 if $nd.bcastNode \neq i$ then
15     $nd.destNode \leftarrow nd.bcastNode$
16     $nd.bcastNode \leftarrow i$
17     set ndTimer(t1) //
18     Upon ndTimer expire
19     send(nd packet) to node $i$ // unicast
```

nd phase (line 15 to 16). The unicast packet has the id of the neighbor node (line 16). The unicast is done with a delay t1 to avoid collisions at the node i. After *nd* phase, a neighbor is selected at random and query is forwarded to it (lines 9-10). A delay of time t2 is introduced between initiating the *nd* phase and selection of a neighbor. This delay ensures that maximum neighbor information is collected before selection is done. As the neighbor selection is done explicitly by the forwarding node i, we refer this protocol as Random Walk with Centralized Selection (RW-CS).

4.4 Proposed protocol for one step of RW

In this section we propose an energy efficient and robust distributed neighbor selection protocol for RW. The protocol is designed for dynamic systems e.g., systems composed of mobile nodes. Throughout this chapter we shall refer to this protocol as Random Walk with Distribution Selection mechanism (RW-DS).

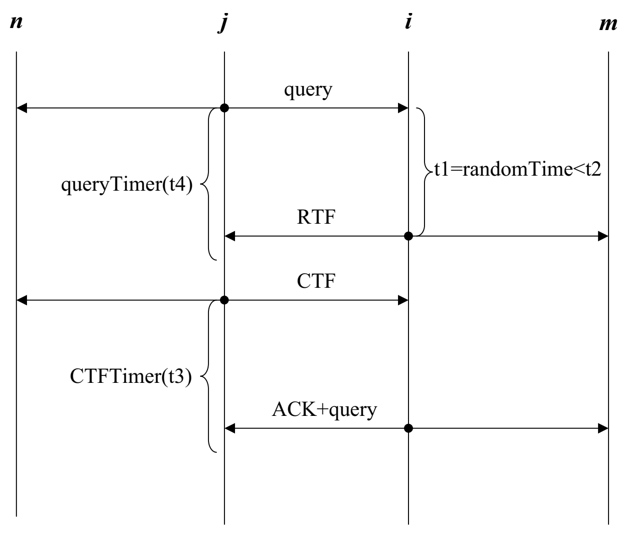


Figure 4.1: RW-DS protocol sequence diagram

4.4.1 Basic assumptions

The system consists of mobile nodes which can form wireless ad hoc network based on IEEE 802.11. The target node is present within the network. The network remains connected and there is a bidirectional connectivity between neighbor nodes at any time. The nodes can communicate only via a $bcast(m)$ primitive. This primitive sends the packet m using the local broadcast facility of the underlying MAC protocol, which uses the carrier sense and binary backoff mechanism to access the medium.

4.4.2 Protocol description

Our protocol is designed to work directly on top of data link layer. This allows to skip the hidden sources of inefficiency that characterizes the classical implementation over the network layer. The proposed protocol uses a four-phase messages exchange pattern, DATA/RTF/CTF/ACK as shown in the Figure 4.1, which embeds an efficient and robust distributed selection logic. In the first phase, suppose a node j broadcasts DATA that is a query packet. In the second phase, each neighbor that receives the query replies by broadcasting a special packet called Request-To-Forward (RTF), with a random delay t1. In the third phase, after receiving the first RTF packet from a neighbor node, say node i, the node j broadcasts another special packet called Clear-To-Forward (CTF) that does two things. First it suppresses the transmission of any further CTF packet from the neighbors if they have not yet transmitted and second it acts as a permission for node i to forward the query packet. In the fourth phase, the node i broadcasts the query, which also acts as an acknowledgement that CTF was received. Now if node i is a destination node then neighbors do not generate any further RTFs and the algorithm terminates otherwise the same four-phases are repeated.

We first, briefly explain the packets used by our protocol. The query packet is a search packet that hops from node to node in search of a destination. The query packet header has three main fields. The $query.bcastNode$ field contains the address of node that has broadcasted the query packet, $query.destFound$ is 1 if node broadcasting the query is the destination node and 0 otherwise and $query.destNode$ field contains the address of the final destination node. The query packet also acts as an acknowledgment that the CTF packet was properly received. RTF packet informs the query sending node, that a neighbor node has received the query and is waiting for its approval to forward the query. The RTF packet header has three main fields. The $RTF.bcastNode$ field contains the address of node that has broadcasted the RTF packet, $RTF.destFound$ is 1 if node broadcasting the RTF is the destination node and 0 otherwise and $RTF.queryBcastNode$ field contains the address of the node that has sent the query. CTF packet gives permission to forward the query packet to the node whose RTF packet was received first. The important CTF packet fields are similar RTF packet fields.

Algorithm 5: RW-DS on a node i, when query packet is received

```
1  **Upon reception of** *query* **packet**
2  **cancel** CTFTimer **if pending**
3  **if** $query.destNode \neq i$ ***and*** $query.destFound = 1$ **then**
4      **terminate algorithm**
5  **if** $query.destNode \neq i$ ***and*** $query.destFound = 0$ **then**
6      $RTF.destFound \leftarrow 0$
7  **if** $query.destNode = i$ **then**
8      $RTF.destFound \leftarrow 1$
9  $t1 \leftarrow jitter(t2)$
10 $RTF.queryBcastNode \leftarrow query.bcastNode$
11 $RTF.bcastNode \leftarrow i$
12 **set** RTFTimer(t1)
13 **upon** RTFTimer **expire**
14     $CTFReceptionEnabled \leftarrow 1$
15     **bcast**(RTF)
```

Now let us discuss the details of the protocol. The proposed neighbor selection protocol RW-DS is a distributed algorithm that has three sections given by algorithm 5, algorithm 6 and algorithm 7, all running on the same node, say, node i. The algorithm 5 is triggered on the reception of a query packet that can come from the upper layer of the same node i or from some neighbor node j. After receiving the query packet the node i broadcasts an RTF packet after a random delay t1 (lines 9 and 12-15 of algorithm 2). The delay is controlled by *RTFTimer* which is initiated on the reception of the query packet and expires after time t1. The time t1 is equal to jitter(t2), where jitter is a function that selects a random time in the range [0, t2]. When the RTF packet is broadcasted, the variable *CTFReceptionEnabled* is enabled (line 14) meaning that the node is now expecting and is able to receive a CTF packet. A CTF packet can get lost due to collisions. So there is a need to send the CTF packet again if an acknowledgement of CTF packet is not received. The reception of query packet also acts as an acknowledgment that CTF packet was received by that neighbor node whose RTF was the first to be received and so now there is no need to send CTF packet again. The reception of query packet thus cancels any pending *CTFTimer* (line 2).

Algorithm 6 is activated when RTF packet is received by a node. The RTF packet is only received by a node, say node j if two conditions are met (line 3 of algorithm 3). First, RTF packet is coming from the node which has generated RTF in response to receiving query from node j and second, the flag *RTFReceptionEnabled* is enabled. The first condition make sure that only that RTF packet is received which is part of the on going four-phase message exchange process of RW and not due to some uncanceled *RTFTimer* of previous steps of the RW. As soon as the first RTF packet is received the flag *RTFReceptionEnabled* is disabled to stop any further reception of RTF packet until the flag is enabled again. The operations of checking of conditions and setting of flag are synchronized as one atomic operation to avoid changing of flag by another RTF packet while these two statements are being executed. Now the first thing an RTF packet does after entering the algorithm 6, is to cancel any pending *queryTimer* (line 5). After that a CTF packet is broadcasted immediately with the destination address of that node id which just sent the RTF packet. It may happen that the CTF packet may not be able to reach the intended destination node due to collisions. In this case it has to be rebroadcasted so that the RW continue. A *CTFTimer* expires after every t3 seconds rebroadcasting the CTF packet until either it is canceled or it reaches its maximum τ times (lines 10-14).

let now us describe algorithm 7 of the protocol. This algorithm is activated by the reception of CTF packet. The CTF packet acts as a green signal to the node whose first RTF was received, to broadcast the query packet. The CTF packet is processed only if three conditions are met given on lines 3-4. These conditions make sure that the CTF packet is generated in response to RTF packet which in turn was generated in response to receiving a query by the

Algorithm 6: RW-DS on a node i, when RTF packet is received

```
1  Upon reception of $RTF$ packet
2  synchronized {
3  if $RTF.queryBcastNode = i$ and $RTFReceptionEnabled = 1$ then
4      $RTFReceptionEnabled \leftarrow 0$ }
5  cancel queryTimer if pending
6  $CTF.RTFBcastNode \leftarrow RTF.bcastNode$
7  $CTF.destFound \leftarrow RTF.destFound$
8  $tCTFbcasted \leftarrow 0$
9  set CTFTimer(0.0)
10 upon CTFTimer expire
11     broadcast(CTF)
12     $tCTFBcasted \leftarrow tCTFBcasted + 1$
13     if $tCTFbcasted \leq \tau$ then
14         set CTFTimer(t3)
```

node that is now receiving CTF and also that the flag *CTFReceptionEnabled* is enabled. This guarantees that the CTF being processed is a part of the on going four phase message exchange of RW and not a stray CTF coming from some node due to uncanceled *CTFTimer* or due to MAC delays because of carrier sensing. Once the CTF packet is received, the flag *CTFReceptionEnabled* is disabled (line 4). The checking of condition and setting of flag are implemented as synchronized one atomic operation so that during these operations another CTF cannot access that flag. All the neighbor nodes which receive RTF but were not the one whose RTF was accepted, cancel there RTF timers if they are still pending (lines 20-21). This cancelation of a scheduled RTF packet avoid unnecessary broadcasts of RTF packets. It may happen that the query is broadcasted but it does not reach any of its neighbors due to collisions. RW will stop in such cases. To keep the RW going, the query packet thus has to be rebroadcasted (lines 16-19). The *queryTimer* with expiry time of t4 controls the time delay between rebroadcasts of the query packet. The rebroadcasts are done maximum τ times before finally giving up (line 18). Note that when a node broadcast a query packet, it is expecting RTF packet from any of its neighbors. For this purpose the flag *RTFReceptionEnabled* is enabled before broadcasting the query packet (line 11).

4.4.3 Non-splitting property of the protocol RW-DS

We define splitting of RW as division of RW into two or more RW during its progress. This means that a single query packet was set out to search for a destination but during its progress, some node instead of forwarding just

Algorithm 7: RW-DS on a node i, when CTF packet is received

```
Upon reception of $CTF$ packet
synchronized {
if $CTF.RTFBcastNode = i$ and $CTF.bcastNode = query.bcastNode$ and $CTFReceptionEnabled = 1$ then
    $CTFReceptionEnabled \leftarrow 0$ }
if $CTF.destFound = 0$ then
    $query.destFound \leftarrow 0$
if $CTF.destFound = 1$ then
    $query.destFound \leftarrow 1$
    print Destination_Found
$query.bcastNode \leftarrow i$
$RTFReceptionEnabled \leftarrow 1$
$tQueryBcasted \leftarrow 0$
set queryTimer(0.0)
upon queryTimer expire
    bcast(query)
    if $query.destFound = 0$ then
        $tQueryBcast \leftarrow tQueryBcast + 1$
        if $tQueryBcast \leq \tau$ then
            set queryTime(t4)
if $CTF.RTFBcastNode \neq i$ then
    cancel RTFTimer if pending
```

one query packet that it received, broadcasted, in response, the same query packet more than once. Now each query packet starts a new RW. The negative effect of this is that even if one of the RW found the destination, other RW may still continue, consuming resources like computational power, energy and bandwidth of devices in the network. It is thus essential for any RW algorithm to have non-splitting property.

Property: The random walk protocol RW-DS given by the distributed algorithms 5, 6 and 7 never splits.

Proof: RW due to protocol RW-DS can split only in the following two cases:

1. Due to RTF: Splitting occurs if more than one RTF packet from different neighbor nodes is processed, as each processed RTF packet will trigger a CTF packet to a different node thus starting a different RW.

2. Due to CTF: CTF packet gives permission to broadcast a query packet. A node receiving a CTF packet triggers the broadcast of a query packet, which starts RW. If a node receives a CTF packet, which is not part of the current four-phase message exchange protocol of RW-DS, then it will trigger the broadcast of another query packet thus causing splitting RW.

Now we shall show that the conditions 1 and 2 will not occur in the protocol RW-DS.

The condition 1 due to the reception of RTF packet cannot occur because the algorithm 6 processes only the first received RTF packet and all RTF packets that are received later are discarded. This is done with the help of a flag *RTFReceptionEnabled*, which is initially enabled. When an RTF packet is received, it will only be processed if it meets the conditions given in line 3 of algorithm 3 and then disables the flag *RTFReceptionEnabled* stopping any further RTF reception until the node receives again a RW query packet. The checking of if condition and setting the flag are synchronized atomic processes so that during these operations another received RTF packet cannot access the flag.

The condition 2 due to the reception of CTF packet cannot occur because of the three checks implemented in the algorithm 7 to ensure that the received CTF packet is the only CTF packet of the current four-phase message exchange of RW-DS protocol and not due to some uncanceled CTFTimer. Note that a CTFTimer is not canceled if a node does not receive a query packet due to collisions, as can be seen from line 2 of algorithm 5. The checks on the reception of CTF packet that ensure that the received CTF packet is the only CTF packet of the current four-phase message exchange are given below.

(i) the received CTF packet has been generated in response to the RTF packet which in turn has been generated by the same node i (condition 1 in line 3 of algorithm 4).

(ii) the received CTF packet has been generated by the node which just sent the query packet to the node i (condition 2 in line 3 of algorithm 4).

(iii) a query packet was received and processed but it has not yet been broadcasted (condition 3 in line4 of algorithm 4). This condition makes sure that once a node has broadcasted a query, it will not process another CTF packet until the node again receives a query packet that is a part of the on going RW.

Hence the protocol RW-DS does not split. □

4.4.4 Characteristics of the protocol

Lookahead implementation

Lookahead means that if one of the neighbors of a node is a destination node, then the RW will proceed to the destination node instead of randomly selecting a neighbor. Lookahead can be very useful in case of service discovery using RW. In RW-DS we can easily implement lookahead for a service discovery where as this is difficult to implement in RW-CS. In RW-DS when a node receives a query packet it checks if this node has the service or not. If this node has the service then query receiving node will reply immediately so that its RTF is able to reach first among all neighbor node RTFs and can be the winner to be selected for the next hop by the query-forwarding node. In case of RW-CS neighbor ids are discovered without any information of the services a node can provide. Note that the service information can be quite long and it is not feasible to include long service information in the neighbor discovery packet reply. In such a case when neighbor discovery packet reply has only the neighbors id, there is no way to identify which id to forward in the next hop.

Bandwidth efficient

The proposed protocol is bandwidth efficient as it generates less number of packets as compared to RW-CS. The efficiency comes from the fact that in RW-DS the selection and forwarding mechanisms are implicit, which means that the same broadcast packets are contributing for selection and forwarding. The RW-CS, due to distinct neighbor discovery and forwarding phases, results in more packets. Also since RW-CS uses unicast, so the IP address has to be resolved before sending the unicast frame. To resolve the IP address ARP packets are used. Moreover unicast uses the RTS/CTS packets. Thus the aggregate effect is the large number of packets generation.

Energy efficient

Energy consumption of a node depends not only the number of packets transmitted but also on the number of packets received. In the proposed protocol RW-DS, the total number of sent and received frames are much less as compared to RW-CS. This is because in RW-DS less number of packets are generated as explained previously and hence less number of packets are received resulting in less energy consumption as compared to RW-CS.

Robustness

In the proposed protocol, the selection takes place when the first RTF packet is successfully received by a node. If the first RTF packet is lost due to a collision then RTF from another neighbor can be received. So actually the robustness of the proposed protocol comes from the fact that reception of RTF from any neighbor node is sufficient where as in RW-CS unicast is the cause of the fragility of the protocol. During the unicast mechanism the packets RTS/CTS/DATA/ACK are sent to specific nodes. If these nodes do not receive any of these packets then the RW will die out. Also in unicast the IP address has to be resolved to MAC address before transmission. This is accomplished by using ARP protocol in which an ARP request frame is broadcasted with a specific node as an intended recipient. In IEEE 802.11 the broadcast mechanism is unreliable as there is no acknowledgement scheme and no retransmission of broadcast in case of collisions.

Decreased source to destination delay

In RW-CS the forwarding node has to wait for the discovery of all or at least most of the expected neighbors before it decides the neighbor to send the query. This delay increases with the increased neighbor density. Whereas in RW-DS the first neighbor to reply is selected and the node can forward the query as soon as it receives the first RTF reply.

How this protocol is different from IEEE 802.11 RTS/CTS mechanism

This protocol seems like the IEEE 802.11 RTS/CTS mechanism but actually it is quite different in its purpose and also in the way it works. The RTS/CTS mechanism is basically an access method and is an extension of Distributed Coordination Function (DCF) of 802.11 MAC protocol. This mechanism is used to get rid of hidden node problem in ad hoc networks [13]. The proposed protocol on the other hand is concerned with the random selection of one of the neighbor nodes and forwarding the query to that node. Also in the proposed protocol the DATA is sent in the first phase while in IEEE 802.11 RTS/CTS mechanism DATA is sent in the third phase. Sending the data in the first phase has the advantage of implementing look ahead, which can greatly reduce the number of hops to search a destination. Due to space limitation of this chapter we shall not go into the details of look ahead.

4.5 Simulation setup

To evaluate the proposed protocol we did detailed simulations using *ns*-2.30 [1]. We used 914MHz Lucent WaveLAN DSSS radio interface model available in *ns*-2 [1]. Each node has a transmission range of 250m and carrier sense range of 550m. We used two ray ground reflection model as the wireless propagation model.

Each reading of the simulation was taken after minimum of 250 runs. To have independent different runs of each the simulation experiment we set the seed of the random number generator of *ns*-2 to heuristic.

We compared the proposed protocol with RW-CS given in section 4.3, using three different scenarios. The first scenario chosen was the grid scenario to validate the protocol. We chose the extreme diagonal nodes as the source and destination nodes. We then compared the mean hitting time results obtained from this scenario with the corresponding mean hitting times using Markov Chain [2] simulations for a grid, as explained in subsection 3.4.3. In the second scenario the static nodes were placed randomly. Such type of scenarios have their importance in wireless sensor networks. The third scenario is for mobile ad hoc networks in which mobile nodes are initially placed uniformly at random and then these nodes move with random way point model. As we are interested in looking at the effects of mobility exclusively, so we choose zero pause time for the nodes in the third scenario. In the later two scenarios we chose terrain a flat area of 2000m x 1000m in which the source was fixed at (500m, 500m) and destination fixed at (1500m, 500m). Fixing source and destination assure that they are always at least some hops away.

4.5.1 Tuning the protocol timers

The purpose of tuning the timers of the proposed protocol is to achieve sustainable RW that finds the destination with a minimum delay.

RTFTimer t2 has to be adjusted with the density of the neighbors. For more dense networks t2 should be chosen large to allow more neighbor nodes to equally participate in winning for the query forwarding. A small value of

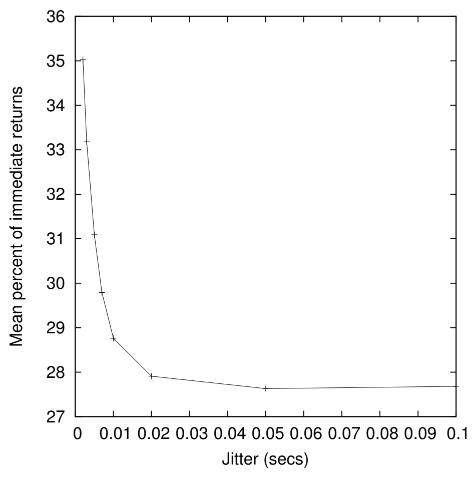


Figure 4.2: Immediate returns on a 10x10 grid

t2 will be sufficient for less dense networks. But t2 cannot be very small. We have seen that if RTFTimer t2 is made very small the hitting time increases and deviates from the graphs obtained from the hitting time analysis using Markov chain (section 3.4.3), showing that the RW does not remain unbiased. This increase in the mean number of hops to find the destination is due to the increased immediate returns of the query packet, that is, the cases in which a packet goes from node i to j and then comes back to i in the following hop, as shown in the Figure 4.2 for a grid topology. We found that the reason for such a behavior is due to the backoff algorithm of IEEE 802.11. This behavior is explained as follows. Suppose that a query hops from node i to node j. Now node j has to send the query to a node selected randomly from one of its neighbors, which also include node i. As t2 is small so neighbors of node j reply with RTF within the short interval [0,t2], resulting in lot of collisions at node j. So the probability of receiving RTF from neighbors other than node i becomes very small. The node i after sending CTF had a backoff and now receives the query from a node j. The node j on the other hand had collisions and supposedly did not receive any RTF. After some time when node i broadcasts RTF due to the expiry of backoff time, the RTF, probably being the last to reply, is received by node j and thus becomes the winner to forward the query. Thus the query has gone from node j back to node i. We set the t2 timer with a value little greater than time where the mean number of immediate returns becomes comparatively less in the graph shown in Figure 4.2. A greater value of t2 on one hand allows us to use the same protocol in greater neighbor densities but on the other hand it increases per hop delay.

For our simulation experiments we set $t2 = 0.01$ secs.

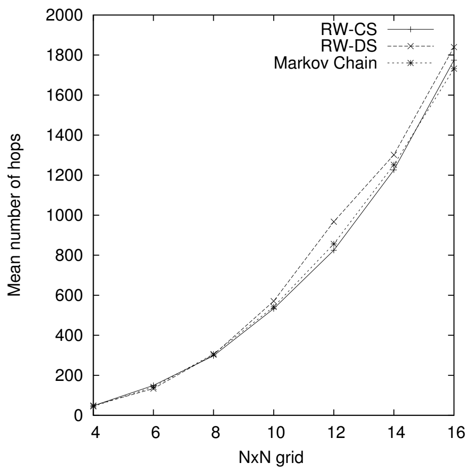
CTFTimer If CTFTimer t3 is short then there would be redundant CTF rebroadcasts and if t3 is large then in case of CTF collision a node has to weight unnecessarily long to receive a rebroadcast of CTF thus increasing overall delay. An ideal CTFTimer setting would be to have t3 a litter greater than the time required between sending the CTF and receiving the query packet, which also acts as an acknowledgment that CTF has been received. But this time depends on the network density. With more neighbor density more time should be given before CTF is rebroadcasted, and with less neighbor density the interval between sending CTF and receiving the query can be shorter. In our experiments we set this time $t3 = 0.1$ secs.

queryTimer If the value of queryTimer t4 is short then there will be unnecessary rebroadcasts of query, which would consume resources like bandwidth and power of transmitting and receiving nodes. Selecting larger value of t4 would introduce unnecessary delays in case of collisions that render no RTF. To set t4 we suppose that at the most first RTF is received within time t2. Although MAC delay and rtt can also contribute in delay but since we are only interested in receiving the first RTF so we just select this timer equal to t2. We can also select t4 a little greater or less than t2 but this will not have much effect on the delay per hop as only the first received RTF is processed. In our experiments we set time $t4 = 0.01$ secs.

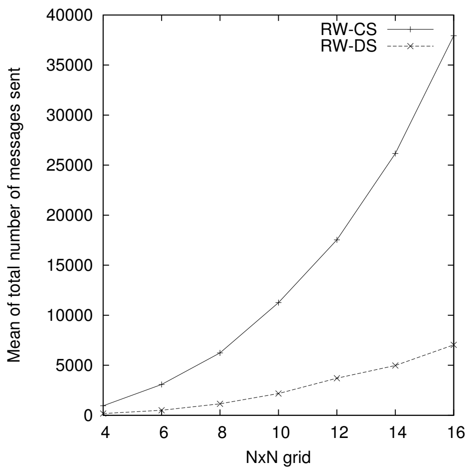
4.6 Simulation results

We first simulated the protocols in grid scenarios. The graphs in Figure 4.3(a) show that random walks RW-CS and RW-DS have almost the same mean number of hops with respect to increasing grid nodes. To validate the proposed protocol RW-DS and the standard protocol RW-CS as an unbiased RW protocol, we did Markov Chain simulations [2] as discussed in chapter 3 section 3.4.3. These simulations confirm that our both algorithms implement a unbiased RW and are not biased towards having increased or decreased mean number of hops for hitting the target. Similarly the simulation graphs in Figure 4.4(a) and Figure 4.5(a) verify that the proposed protocol and the standard protocol behaves in a similar way in randomly placed nodes and mobile nodes scenarios respectively.

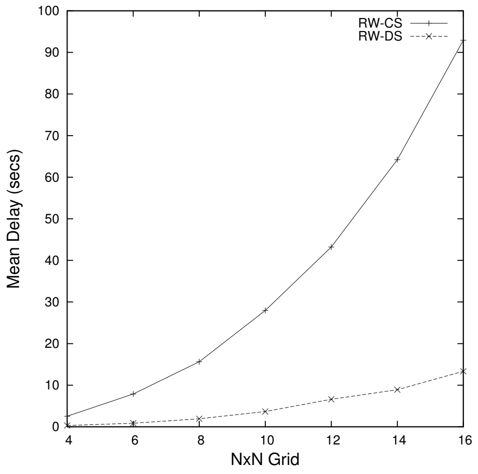
The remaining simulation results confirm our discussion regarding the characteristics of the protocol discussed in the section 4.4.4. The plots in Figures 4.3(d), 4.4(d), and 4.5(d) show that total energy consumed by the system is much less in the RW-DS as compared to the RW-CS. Also we see that the decreased delay in searching time from source to destination in the proposed protocol is much less as compared to RW-CS. This can be seen for all the three



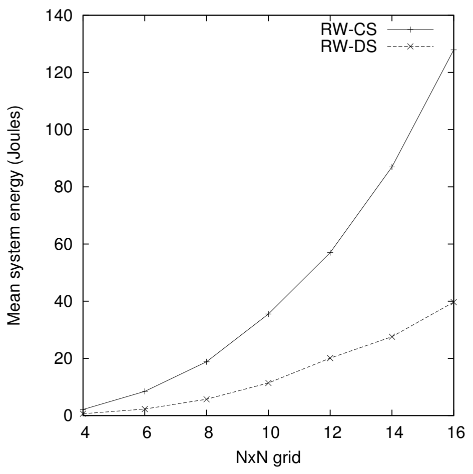
(a) Mean no. of hops



(b) Mean no. of messages sent

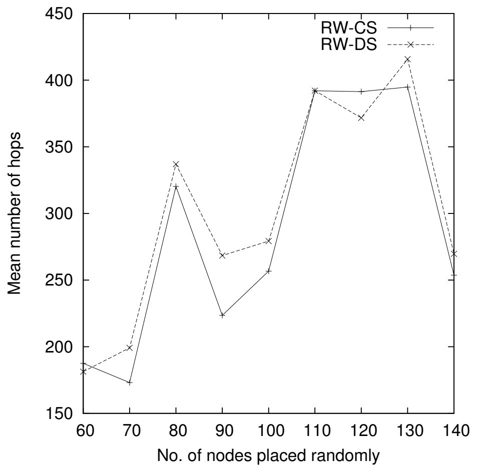


(c) Mean delay

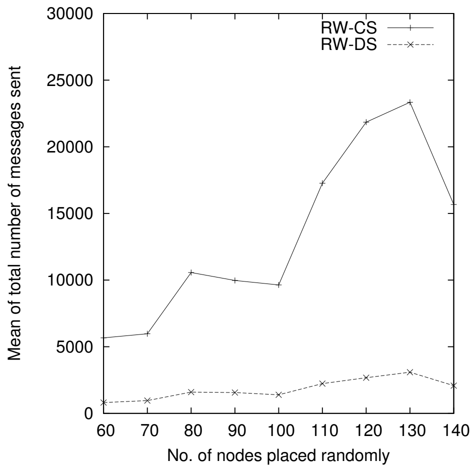


(d) Mean energy consumption

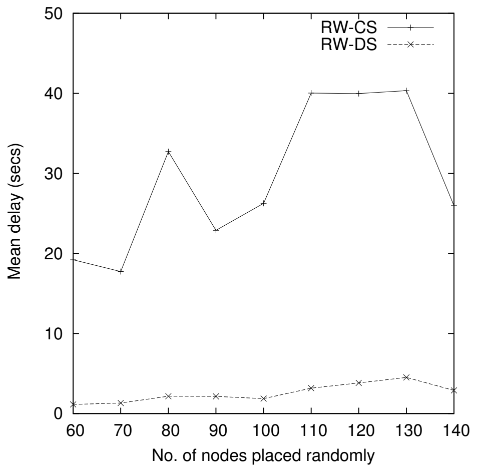
Figure 4.3: A random walk search on NxN grids



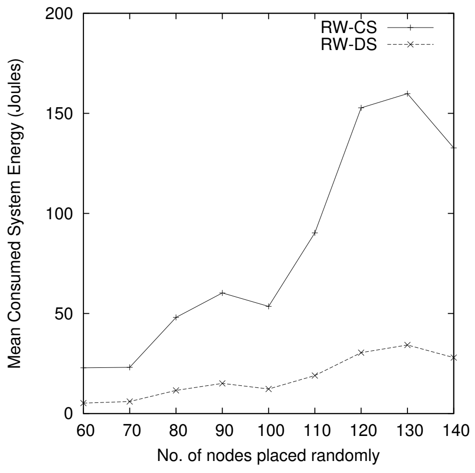
(a) Mean no. of hops



(b) Mean no. of messages sent

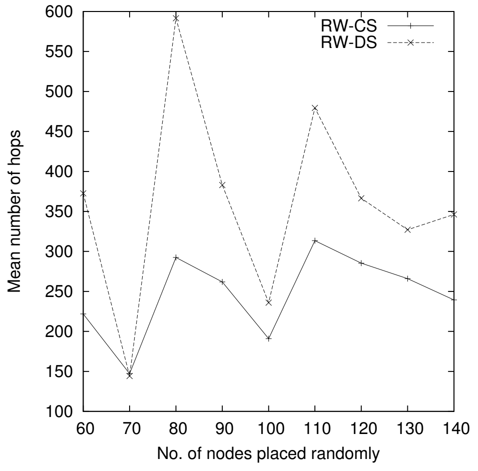


(c) Mean delay

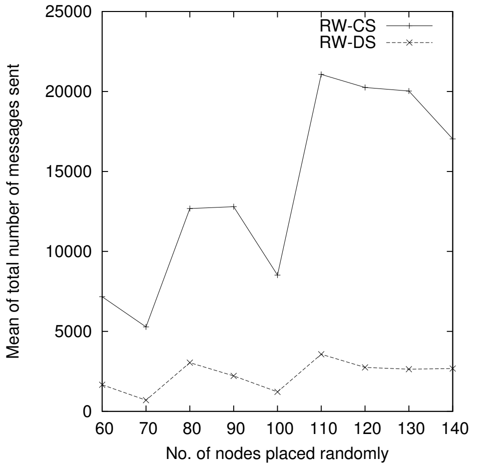


(d) Mean energy consumption

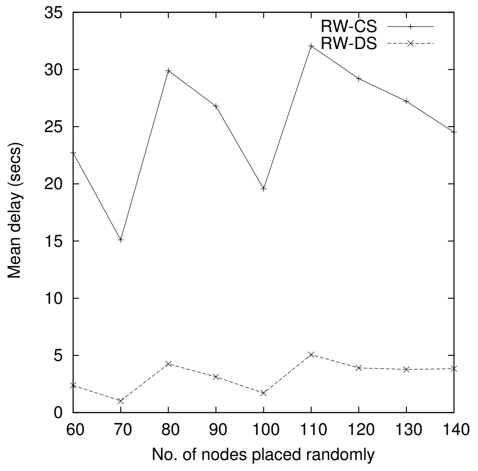
Figure 4.4: A random walk search on static randomly placed nodes



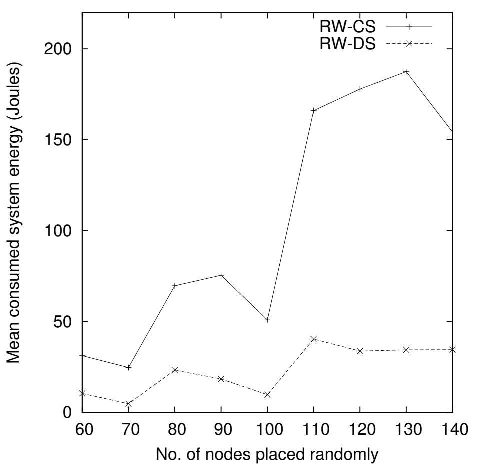
(a) Mean no. of hops



(b) Mean no. of messages sent



(c) Mean delay



(d) Mean energy consumption

Figure 4.5: A random walk search on mobile nodes

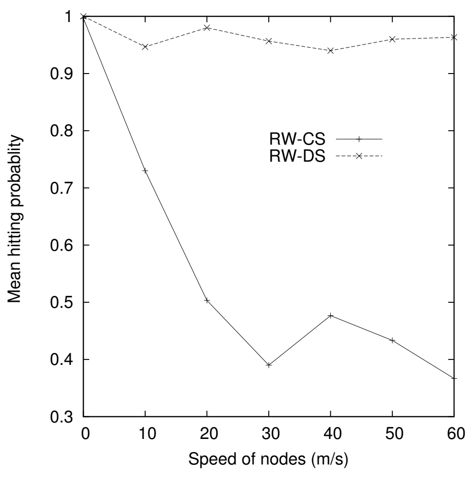


Figure 4.6: Hitting probability w.r.t speed for 60 nodes

different scenarios in Figures 4.3(c), 4.4(c), and 4.5(c).

We note that there are fluctuations in the simulation plots for randomly placed static node scenarios shown in Figure 4.4 and in the mobility scenarios as shown in Figure 4.5. The fluctuations that appear in these plots are the consequence of the variability in the density of nodes that we can observe in some regions of the area passing from one deployment to another in case of random topologies. We can explain this fact further as follows. We have fixed source and destination to make sure that they remain at least some hops away from each other. When a scenario is created with two fixed nodes and remaining nodes are randomly placed, there can be scenarios in which nodes can be placed in such way that more paths lead to the destination from the source, thus leading to less number of hops of RW. In some scenarios there are not many paths leading from source to destination, which results in large hitting times.

But in spite of the fluctuations in the graphs with respect to increasing number of nodes, these graphs are sufficient to study the real focus of the study, which is the comparison between the two protocols, RW-DS and RW-CS. Thus the two protocols are compared for the same fixed scenarios and the the most meaningful information which is the difference in the performances of the two protocols rather than the absolute performance of a specific protocol, can be easily judged. It can be seen from these graphs that RW-DS always out performs RW-CS. This fact is visible in smooth graphs obtained from the grid topologies of Figure 4.3 as well as from the graphs showing fluctuations obtained for random topologies given in 4.4 and 4.5.

Figure 4.6 shows the hitting probability with respect to mobility of nodes. As expected, the proposed protocol RW-DS is more robust under mobility scenarios as compared to the RW-CS. The hitting probability remains much high in RW-DS as compared to RW-CS as seen in Figure 4.6.

4.7 Conclusions

In this chapter we proposed a protocol for unbiased random walk in which the selection for next hop node is achieved by a distributed algorithm instead of a centralized algorithm. The proposed protocol is broadcast based and has a four-phase message exchange procedure. We compared it with a unicast based standard random walk protocol based on a centralized algorithm. Simulation studies showed that the proposed protocol (i) is bandwidth efficient (ii) energy efficient and (iii) incurs less delays in searching from source to destination.

Chapter 5

Biased Random Walk

5.1 Motivation

Random walk (RW) on a given network is a sequential process in which a walker moves from one node of the network to another, selected at random among its neighbors. RW algorithms do not require the knowledge of underlying network, hence they are very well suited for wireless networks which can undergo structural changes due to mobility, switching off of the resource constraint devices, failure of devices and many other factors [9].

Algorithms that depend on topology are not favorable due to the cost associated with the maintaining the topology information. Wireless networks are also particularly favorable to implement random walks due to the broadcast nature of the transmissions. The use of Random walk have been proposed in membership services [12], group based communication [39], search/query [8], [9], and routing [89].

A Simple Random Walk (SRW) arises when the choice of the next node to visit is done uniformly at random. SRWs have the property visiting all nodes of the network, provided that the topology is connected. Thus, the target is eventually reached. However, in a SRW the expected number of steps required to reach a given target node, namely the hitting time, can be very high. This makes SRW potentially not always convenient w.r.t. to other unstructured search solution, most notably flooding.

The hitting time can be reduced by enriching the selection policy with some form of biasing, that pushes the walker towards the target. If there is no external information or hints about where the target can be located, then the only option is to force the walker to always explore new parts the network. Or, equivalently, to avoid visit the same nodes less number of times. This is the central idea of our chapter.

The biasing strategy proposed in this chapter is completely distributed and rely only on the information locally known to a node. We show that with this biasing strategy the search efficiency and other metrics of random walk greatly improves as compared to existing strategies. To evaluate our proposal we have chosen grid topology. The grid topology, on one hand allows us to use a simple analytical model for having a first flavor of the effect of the position of the target node on the hitting time and on the other hand, by increasing the grid connectivity we can approximate a random geometric graph. Moreover, a grid topology is a good model for wireless mesh networks.

This chapter is organized as follows. In section 5.2 we give some the essential definitions of terms used in the chapter, section 5.3 present the proposed strategy, section 5.4 give experiment details and discuss the results in which we compare the proposed strategy with other strategies, section 5.5 gives some of the important work done in biasing strategies and the last section 5.6 concludes and points out some directions for future work.

5.2 Background

5.2.1 Some important definitions

Let us define various terms used in this study

- *Visit:* A node is visited when it receives the walker. We define two types of visits. If a node after receiving the walker either forwards it to another node or it is the target node then the visit is called *active* visit, otherwise if the does not forward the walker then we call the visit as *passive* visit. Passive visits occur due to the broadcast nature of transmissions.
- *Covered node:* A node is said to be covered if it has been visited at least once.
- *Partial cover time:* It is the expected number of steps required by RW to visit a constant fraction of nodes of the network [8].

5.2.2 Hitting time in Simple Random Walk

Before presenting the proposed biasing strategy, it is worth analyzing the behavior of a SRW on a grid topology. For the SRW on a grid we can use the steady state analysis to find the mean hitting time of the random walk as discussed in section 3.4.3. Using this method we found the mean hitting time with respect to different positions of destination when source is placed at (0,0). This is shown by the graphs in Figure 5.1. It can be seen that the hitting time is maximum when source and destination are at extreme diagonal node. This is easy to understand since the nodes at the extreme diagonals are

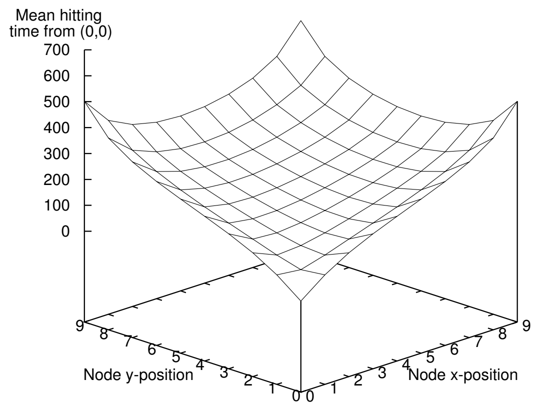


Figure 5.1: Mean hitting time on 10x10 grid with source placed at (0,0) and destination placed at different positions.

farthest apart and also at corners they have the least number of neighbors and thus connectivity. We shall use these results in section 5.4.2.

5.3 Proposed biasing strategy

In the proposed biasing strategy each node records the number of times it has been actively and passively visited. When a node has to select a neighbor for forwarding the walker, the nodes probes its neighbors and then selects the neighbor which have a least value of active and passive visits. In case of tie, a node is selected randomly from these least visited neighbors. Probing can be done very efficiently as described in [70].

We compare the proposed biasing strategy with the strategies given in Table 5.1 in which the next hop is selected at random without look-ahead (strategy A) or with look-ahead (strategy B), or selected at random among the least visited nodes without look-ahead (strategy C) or with look-head (strategy D). Strategy C corresponds to random walk with choice proposed in [9], where the number of choices equals the number of neighbors. The proposed is the strategy E in which the neighbor is selected at random among the least actively and passively visited neighbors. Note that this strategy implicitly includes look-ahead. Figure 5.2 shows a few steps of random walker moving according to strategy E, using the notion of active visit (AV) and passive visit (PV).

Table 5.1: Different RW strategies used for comparison

Strategy	Selection strategy for next-hop
A	random
B	random with lookahead
C	random among the least actively visited nodes
D	random among the least actively visited nodes with lookahead
E	random among the least actively and passively visited nodes

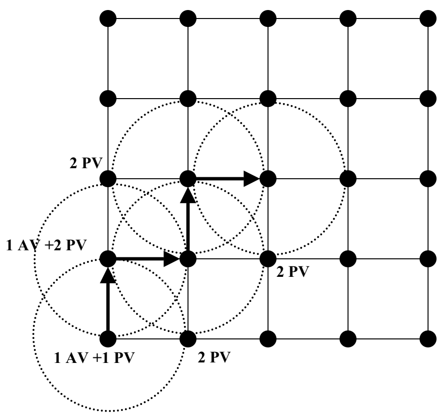


Figure 5.2: Random walk using active and passive visits

5.4 Simulation results

We did simulations on Linux SUSE 9 platform using custom built C simulator. We did not used a packet level simulator like ns-2 [1] on purpose. We experimented [70] with ns-2 and found that ns-2 is not scalable. It took large amount of time to simulate the behavior of random walk with just a few hundred nodes. In this work since we are not interested in packet level simulations and want to observe the behavior of the proposed biasing strategy with large number of nodes, we built a simulator with C. We assume that within a step of the random walk there are no communication failures when probing for neighbors and forwarding the walker to a selected neighbor using, for example, distributed selection or centralized selection mechanisms [70]. We also assume that a node do not fail during the short interval when it is being visited by

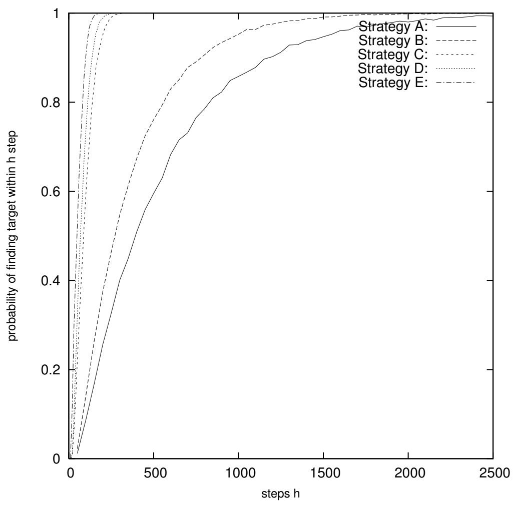


Figure 5.3: Hitting probability distribution

the walker.

5.4.1 Comparison against flooding

Flooding is a well known approach for searching in unstructured networks. It is important to compare our solution with flooding. Comparing random walk search cost with that of flooding is not straight forward. Random walk search performance is measured in terms of hitting time and hitting probability whereas for flooding there are no useful notion of these two parameters. Instead, in flooding we can measure the search cost in terms of number of transmissions required to search a target. This parameter is indicative of the energy consumption in searching a target. The transmissions can be due to query packets or due to control packets such as MAC level packets.

Note that it is not only the number of transmissions but also the length of transmission that is the measure of energy consumption. In this respect if we observe the query packet we see that it is the longest of all other associated control packets as it carries detailed information, which can be an XML file describing the service or the target node. Whereas the control packets are of very short duration. We can thus safely assume that actual energy consumption is due to the transmissions of the query packets and therefore can consider the number of transmissions as the number of query packets transmissions. The hitting time, as defined before for a random walk, is therefore the number of transmissions of the query packet.

The number of transmissions in case of flooding the whole network is the total number of nodes N, if we assume that each node transmits the query exactly once. The Figures 5.4(a) and 5.4(b) compare flooding with different RW strategies. We see that in random walk with the proposed biasing strategy E, the mean number of transmissions and hence the search cost is less than that of flooding. In case of many query requests, this result encourages the use of RW using the proposed strategy for searching a network because on one hand the mean number of transmissions will be less than that of flooding and and on the other hand there is always a probability of hitting a randomly places target within few steps and thus stopping the RW, whereas in case of flooding the whole network has always to be searched without being stopped even when the target is found earlier.

5.4.2 Comparison against other strategies

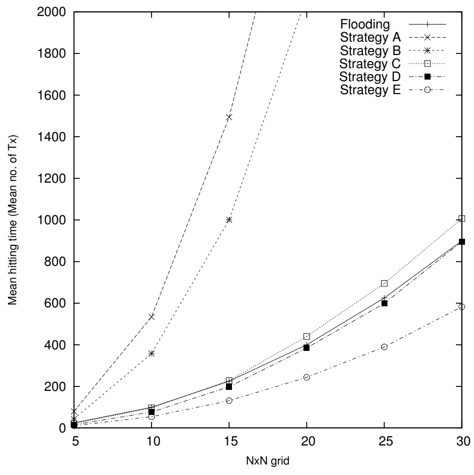
In this section we shall compare random walks having selection strategies given in table 5.1 with respect to different metrics.

Hitting time

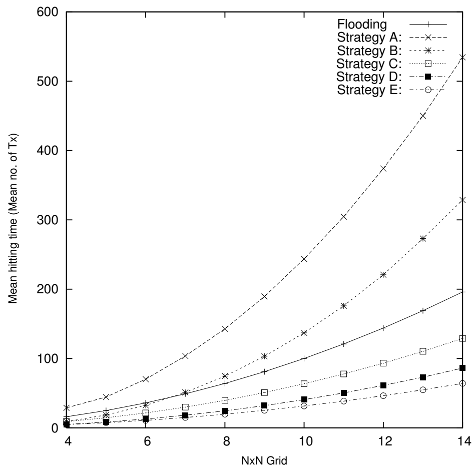
In this section we will consider different metrics related to the hitting time. The worst case scenario is the one in which the source and destination are placed at the extreme diagonals as suggested in section 5.2.2 and in random case scenario source and destination are placed at random in the network.

Hitting probability distribution, worst case The graphs in Figure 5.3 show the probability of finding a target within a specific number of steps for different random walks. A better random walk would have more probability of finding a target in comparatively less number of steps. In this regard we observe that for the proposed policy E, the probability of hitting a target is highest within the given number of steps as compared to other random walks. This is because the random walk tries to avoid the nodes which it has already search either actively or passively and thus probability of finding a target always increases with each step.

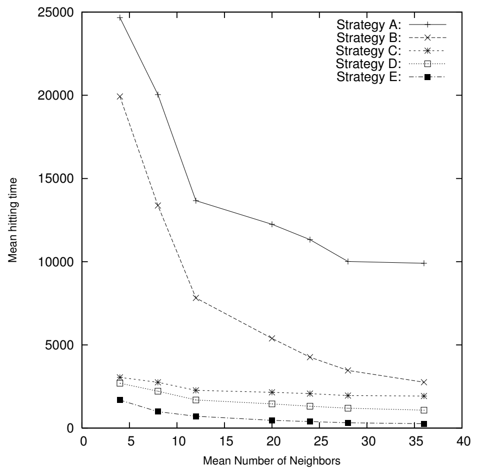
Mean hitting time, worst case Figure 5.4(a) shows the variation of hitting time for increasing network size for different biasing strategies for the worst case search scenario, that is when the source and target are placed at the extreme diagonal nodes of the grid. We see that the pure random walk performs worst with the maximum hitting time where as the proposed strategy E performs the best with the least hitting time.



(a) Hitting time variation with grid size in worst case



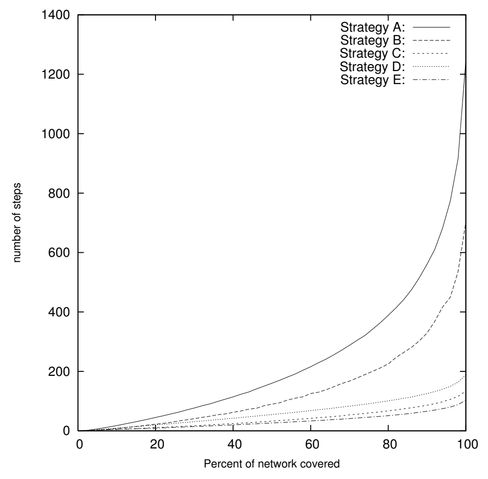
(b) Hitting time variation with grid size in random case



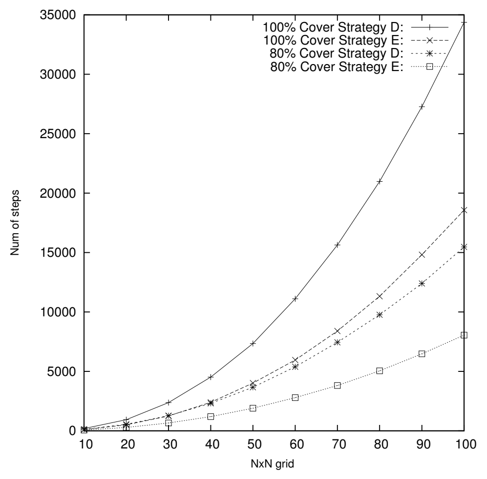
(c) Hitting time variation with neighbor density in worst case

Figure 5.4: Hitting time variation of different biasing strategies

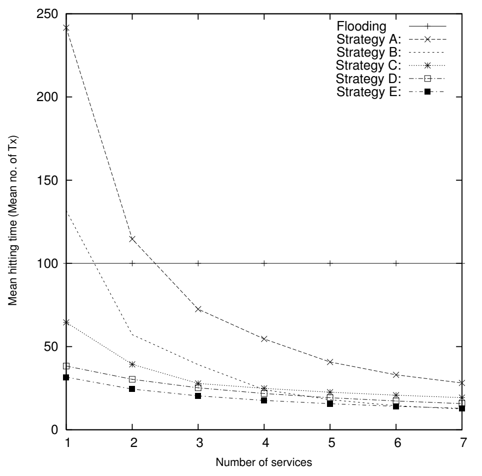
Mean hitting time, random case Figure 5.4(b) shows the variation of mean hitting time with respect to increasing number of nodes when any source and destination can be chosen from the grid. Here we also observe that the policy E has always the least mean hitting time with increasing number of nodes as compared to other random walks. This is because the strategy used for biasing exploits the broadcast nature of the wireless medium and with one broadcast it can passively visit it neighbors and can later effect the random



(a) Partial cover time with increasing fraction of cover



(b) Cover time with increasing grid size for strategies D and E for 100% and 80% network cover



(c) Effect of replicated services on hitting time

Figure 5.5: Comparison of biasing strategies w.r.t. different metrics

walk by avoiding further visits.

Mean hitting time variation with density of neighbors, worst case Figure 5.4(c) shows the variation of mean hitting time with the density of neighbors. We see that for all strategies the mean hitting time decreases. The variation of mean hitting time is maximum in strategy A, which is SRW whereas the variation and also it value is always least in the proposed policy E. This is a significant result as it tells us that the RW biased with the proposed strategy is least effected by varying neighbor density.

Cover Performance

It is often important to see how a network is covered by a RW biasing strategy. Cover time and partial cover time are two important metrics in knowing the efficiency of cover performance. The Figure 5.5(a) shows how the partial cover time is related to the fraction of cover of the network for different biasing strategies for a 10x10 grid. We see that the proposed biasing strategy E performs best through out different percent of the network cover as compared to other strategies. We also note that in all biasing strategies the number of steps and hence energy and effort to cover almost 80% of the network is less as compared to covering the remaining network. In the compared RWs the best is performed with biasing strategy E in which the increase in number of steps is almost linear till about 80% of network cover.

Now let us see how the strategies would scale with the increasing grid size. Instead of comparing all strategies in the table 5.1 we just select the best two i.e., strategy D and E and compare them for 100% and 80% cover of network with increasing network size. The Figure 5.5(b) shows this comparison. It is interesting to see that in strategy E the number of steps required to cover 80% of network increases almost linearly with the increasing size of the grid. We thus observe that RW with strategy E is more scalable than other strategies with respect to network coverage.

Effect of service replication

The final aspect we consider is service replication. Often we have more than one similar services in a network and we are interested in searching any one of the available services. In Figure 5.5(c) we compare the performance of random walk biased with different policies when the number of similar available service increases. In Figure 5.5(c) we see that proposed policy E performs best as compared to RWs biased with other strategies. One interesting point here is that although with smaller number of similar services the difference between hitting times of different random walks is prominent but with the increasing number of similar services this difference of hitting time decreases. We also observe that with policy E the mean hitting time remains more or less constant with increasing number of similar services whereas in case of other random walks the hitting time depends on the number of available similar services. Thus with the strategy E there is not big advantage in replicating services. The advantage of replicating services is quite visible in other biasing strategies.

5.5 Related work

There has been an extensive analytical study of simple random walk in literature. Lovasz [62] gives a detailed analytical study of SRWs. To reduce the hitting time and improve other metrics of SRW, different biasing strategies have been used in literature. We give here some of the recent related work in biased random walk in which the biasing does not depend on the information available globally to the network nodes, like gps.

Braginsky and Estrin in [15] propose a rumor routing mechanism used for searching. The nodes with services launch mobile agents which execute random walks in the network resulting in event-paths. The queries are also mobile agents that follow random walks. Whenever a query agent intersects with an event-path, it uses that information to bias the query to the location of the service provider. This strategy may not be efficient if services are replicated in the network as each service provider would initiate a random walk [80].

Avin and Brito [8] have proposed a biased RW in which the biasing depends on a biasing parameter $bias$ where $0 \leq bias \leq 1$. When $bias = 0$ it leads to SRW and when $bias = 1$ the RW is not allowed to visit an already visited node and if all neighbors are visited then any neighbor is selected at random to avoid a deadlock situation. When $0 < bais < 1$ the selection of visited and unvisited neighbors is probabilistic. The RW performs best when $bias = 1$. But in this case i.e., when $bias = 1$ as the network is covered more and more nodes are marked visited thus reducing unvisited neighbors. Since in this case when a node finds all neighbors visited it does a random selection, so we expect to have a diminishing effect of bias with increasing network cover.

The RW proposed in [8] was later improved by Avin and Krishnamachari [9]. They proposed RW with choice. The random walk moves to a neighbor which is selected from a small number of neighbors d sampled at random and is least visited. They compare SRW with the proposed RW and show that for $d \geq 2$ there is a significant improvement over the SRW w.r.t different metrics like cover time and visit distribution. To the best of our knowledge this is the best biasing strategy proposed so far which does not rely on any external or global information for biasing. The biasing strategy proposed in this chapter actually extends and improves this strategy.

Adamic et al. [4], Kim et al. [52] and Thadakamalla et al. [88] have studied searching using RW in scale free and power law graphs in which highest degree neighbor is chosen with no retracing. Such a RW may lead to infinite loops [37].

A detailed comparative study is made by Dhillon and Mieghen [36]. They compare RW in which next step is chosen proportional to the degree, RW with memory, RW with lookahead, RW that steps to the highest degree neighbor and RWs having different combinations of these RW strategies in random geometric graphs. In RW with memory, a memory list keeps record of node ids visited. The next hop is selected randomly from neighbors that are not in the memory list. Such a strategy leads to deadlock. In RW with lookahead if destination node is among the neighbors, that node is chosen for the next hop. Such a RW can have loops, but no deadlock. In RW that steps to the highest degree neighbor node and select randomly if some nodes have the same highest degree, the RW can lead to infinite loop. In case of RW proportional to the degree, next step is chosen probabilistic in which the probability of choosing the neighbor node is proportional to the degree of the neighbor nodes. There are no deadlock in this type of RW but the drawback is that it becomes equivalent to a simple RW in regular graphs. The authors have shown that RW that uses highest degree with lookahead and memory is the most efficient strategy among the compared in random graphs.

5.6 Conclusions

In this chapter we first introduced the notion of active and passive visits and then proposed an efficient random walk biasing strategy that exploit the active and passive visits of a node. We compared our proposed strategy with different random walk strategies which do not rely on any global information and just use local information available at the node. The simulation results on a grid topology showed that the RW with the proposed strategy is better than flooding with respect to search cost. It is most search cost effective, least effected by varying neighbor density, not only least effected but also perform best when services are replicated and scales best.

Chapter 6

β Stopping Rule: An Adaptive Random Walk

6.1 Motivation

In this chapter we focus on an important question of deciding when to stop RW when it is used to search for a service or a node having a specific property. A stopping rule for a RW is a condition or a set of conditions which when satisfied cause a RW to stop. We address the problem stopping rule in the context of a wireless ad hoc network whose size can not be known exactly and the target node may not be in the network. In such dynamic systems nodes may join and leave due to churn or node mobility and thus it is not possible to know the size of the network and also about the presence of the target in it. The real dilemma behind this problem is to decide about the lifetime of the walk. If RW stops the search too early, then the algorithm can fail even if the target is available. On the other hand RW can continue to search while the target may not be present in the network, wasting network resources.

Stopping rules have been discussed in [63]. The simplest and the most commonly used stopping rule is Time-To-Live (TTL). In this stopping rule the query does certain number of hops equal to TTL before finally stopping. The drawback of using this stopping rule is that the network size must be known to set TTL to visit a specific fraction of the network. But in case in of dynamic networks it is not always possible to know the network size. In this chapter we propose a stopping rule which has a nice property that the RW stops after visiting a minimum fraction of the network on the average and this fraction of coverage is very less sensitive to the network size. In other words, the RW adapts its lifetime according to the size of the network.

Contribution of this chapter: In this chapter we analyze and then compare RW with TTL and the proposed stopping rules, which we shall call β stopping rule, for a torus grid. The torus grid is wrapped square grid. This graph model is often used to model wireless ad hoc networks. However before analyzing the stopping rules TTL and β for a torus grid, we give their probabilistic and mean value analysis for a complete connected graph and then using this analysis we compare both stopping rules. Although complete graph case is not a model for wireless ad hoc network, but the study of the stopping rules for this simple graph helps in identifying the adaptive property of RW using the β stopping rule and gives a guideline for further investigating the property in a grid model.

This chapter is organized as follows. In the next section we shall define and explain the proposed stopping rule. Then in the next section we discuss TTL and the proposed stopping rule with respect to complete graph model. We shall first give the probabilistic model of covered nodes with TTL as a stopping rule and then the mean value analysis of the covered node and then compare TTL and proposed stopping rule with respect to the mean fraction of coverage and the standard deviation of coverage. We then discuss the stopping rule for a torus grid model. We then compare both the TTL and the proposed stopping rule in this model and then finally discuss the an important property of the RW with the proposed stopping rule.

6.2 β stopping rule

Let β be a real number then the β stopping rule is defined as the condition for the termination of RW when, during RW, the ratio $\frac{\text{number of steps}}{\text{covered nodes}}$ becomes greater than β.

A RW using β stopping rule requires four variables, three of which are part of the random walking packet, with the field name BETA, STEPS and COVERED and one, called IS-COVERED, which is stored at nodes. These variables are used for the following purpose.

- The field BETA is a real number sets when a random walk is initiated and regulates the coverage requirement.
- The field STEPS is a counter of the steps done by the walker so far.
- The field COVERED is a counter of the total number of covered nodes.
- IS-COVERED is a bit that indicates if a node is covered or not.

The field BETA is set according to the required coverage and maximum number of nodes in the network. The value of this field, β, is to be determined experimentally. All the other variables are initialized to 0. The STEP field is incremented at each new step, while the COVERED field is incremented only when a node with the IS-COVERED bit unset is visited. Before making a new step, the ratio STEPS/COVERED is calculated. If this ratio becomes greater than BETA then the walk stops.

6.3 RW on a complete graph

In this section we shall we give a probabilistic and mean value coverage analysis of RW with TTL stopping rule and with the β stopping rule and then compare both stopping rules.

6.3.1 TTL stopping rule

Probabilistic analysis

Let the RW be terminated at a specific step $k = TTL$. It is then clear that the covered nodes at step k, C_k are random. In the following study we develop a probabilistic model for C_k. The Figure 6.1 shows all the possible number of covered nodes at each step with their respective transition probabilities. We assume that at step $k = 0$ the source node is visited, thus at step $k = 1$ two nodes are visited. At $k = 2$ there are two possibilities. Firstly, the previously visited node is visited again and thus the number of covered nodes remain same. This happens with probability $\frac{1}{N-1}$, where N is the total number of nodes. Secondly, a new node is visited thus making the number of covered nodes 3. This happens with probability $(1 - \frac{1}{N-1}) = \frac{N-2}{N-1}$. Thus for different values of k, we can write

$$\begin{array}{lrcl}
\text{k=1} & Pr\{C_1 = 2\} & = & 1 \\
\text{k=2} & Pr\{C_2 = 2\} & = & \dfrac{1}{N-1} \\
 & Pr\{C_2 = 3\} & = & \dfrac{N-2}{N-1} \\
\text{k=3} & Pr\{C_3 = 2\} & = & Pr\{C_2 = 2\}\dfrac{1}{N-1} \\
 & Pr\{C_3 = 3\} & = & Pr\{C_2 = 2\}\dfrac{N-2}{N-1} + Pr\{C_2 = 3\}\dfrac{2}{N-1} \\
 & Pr\{C_3 = 4\} & = & Pr\{C_2 = 3\}\dfrac{N-3}{N-1} \\
\vdots & \vdots & &
\end{array}$$

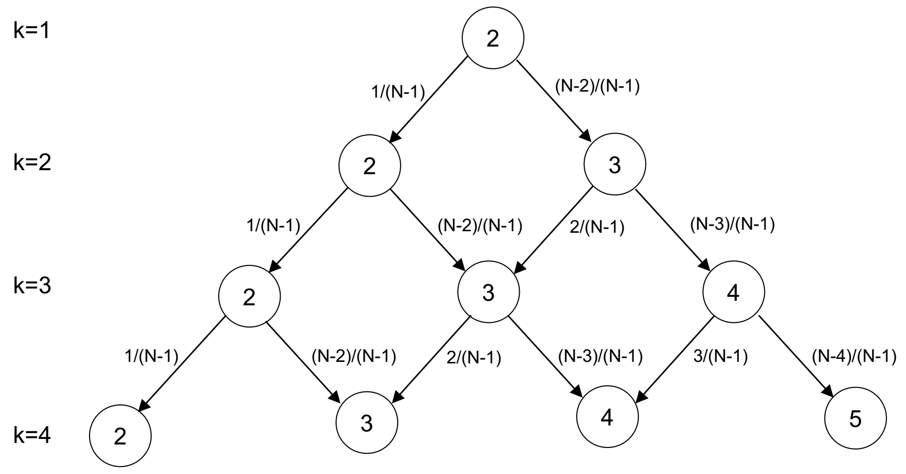


Figure 6.1: Probability analysis of covered node showing all possible states of covered nodes at each step till $k = 4$

Hence if X is value that the random variable C_k can have then the generally the probabilistic distribution of C_k is given as

$$Pr\{C_k = X\} = Pr\{C_{k-1} = X - 1\}.\frac{N - X + 1}{N - 1} + Pr\{C_{k-1} = X\}.\frac{X - 1}{N - 1}$$

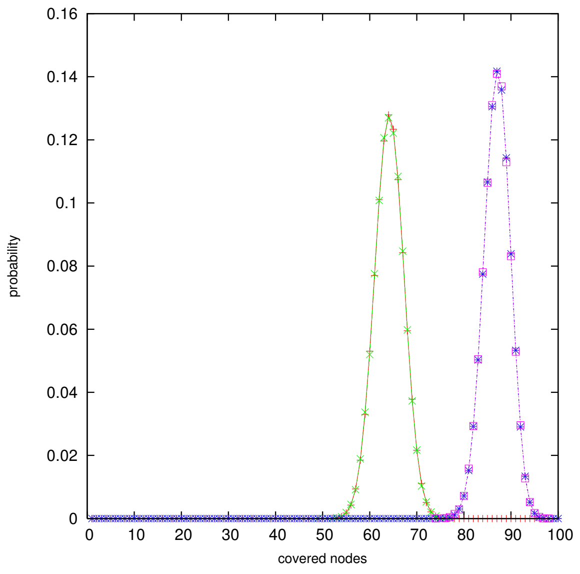
To validate the above probabilistic model of covered nodes we did simulations of the RW on a complete graph. For a values of TTL we had a minimum of 10^5 iterations for calculating the probability of different covered node. The Figure 6.2 validates the above model and shows the probability distributions calculated from the model and by simulation for 100 and 900 nodes for different values of TTL. We shall use this model later when we compare TTL with the proposed stopping rule.

Mean value analysis

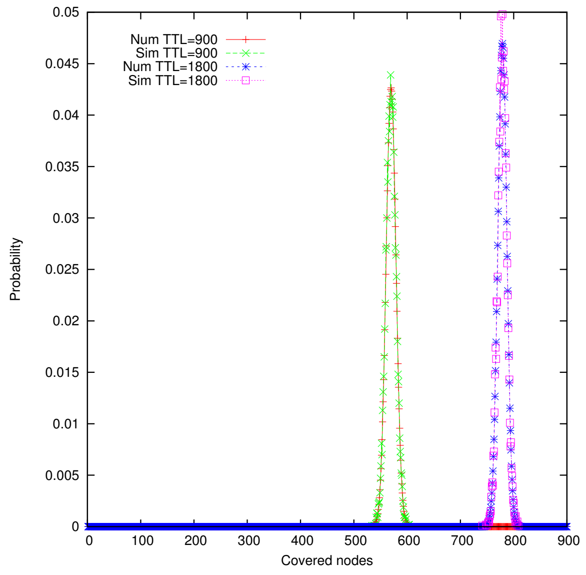
Mean network coverage with TTL can be found as follows. Let $E[C_k]$ and $E[C_{k+1}]$ represent the expected value of covered nodes at step k and $k + 1$ respectively, then

$$E[C_{k+1}] = E[C_k] + E[C_{k\to k+1}] \tag{6.1}$$

where $E[C_{k\to k+1}]$ represent the mean number of nodes covered when the RW moves from step k to step $k + 1$. This could be either 1 or 0. Thus from the



(a) Prob. Analysis for 100 nodes



(b) Prob. Analysis for 900 nodes

Figure 6.2: Validation of the probabilistic model when TTL is the stopping rule

definition of the expectation of a random variable,

$$\begin{aligned} E[C_{k\to k+1}] &= 1.Pr\{C_{k\to k+1}=1\} + 0.Pr\{C_{k\to k+1}=0\} \\ &= Pr\{\text{visiting a new node from step } k \text{ to } k+1\} \qquad (6.2) \\ &= \frac{\text{Mean uncovered nodes at step } k}{\text{Number of nodes that can be selected}} \\ &= \frac{N - E[C_k]}{N-1} \qquad (6.3) \end{aligned}$$

Hence from equations 6.1 and 6.3, we have

$$E[C_{k+1}] = E[C_k] + \frac{N - E[C_k]}{N-1}$$

This is an iterative equation which leads to the solution,

$$E[C_k] = N - (N-1)\left\{\frac{N-2}{N-1}\right\}^k$$

Let $E[\rho] = \frac{E[C_k]}{N}$ be the mean partial cover, then we can write the above equation as

$$E[\rho] = 1 - \left\{\frac{N-1}{N}\right\}\left\{\frac{N-2}{N-1}\right\}^k \qquad (6.4)$$

To validate we also did simulations in which each value of mean covered nodes at step k was calculated after a minimum of 1000 iterations. Figure 6.3 shows that mean partial coverage given by above equation and the simulations perfectly agree. The results obtained in this section shall be used later.

6.3.2 β stopping rule

Probabilistic analysis of C_β

Let C_β be the covered nodes by the RW when β stopping rule is satisfied. Figure 6.4 shows the state transitions for different values of the ratios $\frac{\text{number of steps}}{\text{covered nodes}}$. This ratio is shown at the bottom whereas number of covered nodes are shown at the top of each circle. Suppose we want to find the probability of 2 covered nodes when $\beta = 0.51$. The fraction $\frac{k}{X} \geq 0.51$ for the first time when $k \geq X\beta$ for the first time. This happens at step $k = \lceil X\beta \rceil = \lceil 2*0.51 \rceil = \lceil 1.02 \rceil = 2$. Hence we have $Pr\{C_{\beta=0.51} = 2\} = Pr\{C_{k=2} = 2\} = \frac{1}{N-1}$. Similarly the probability of 3 covered nodes for $\beta = 0.51$ is same as the probability of 3 covered nodes at step $k = \lceil X\beta \rceil = \lceil 3 * 0.51 \rceil = \lceil 1.53 \rceil = 2$. That is

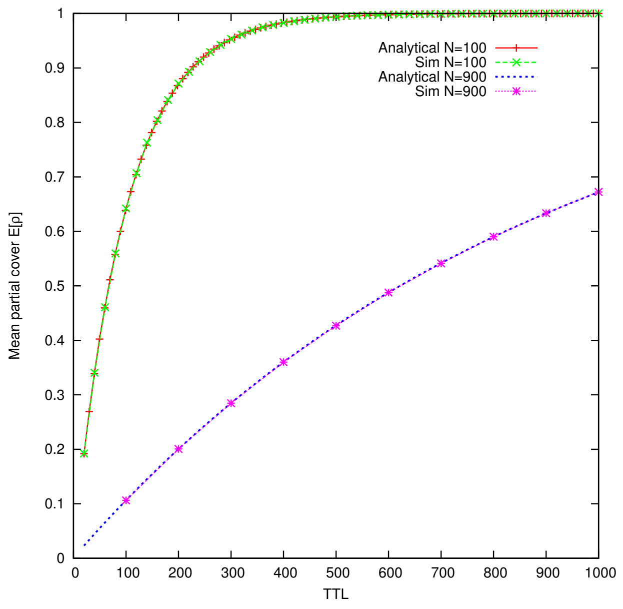


Figure 6.3: Validation of mean value analysis of partial cover with TTL as a stopping rule

$Pr\{C_{\beta=0.51} = 3\} = Pr\{C_{k=2} = 3\} = \frac{N-2}{N-1}$. The probability of covered node ≥ 4 with stopping value $\beta = 0.51$ is 0 because the RW will always terminate at step $k = 2$.

From the Figure 6.4 it is clear that the probability distribution of covered nodes will not change when $0.5 \leq \beta \leq 0.67$. It changes when $\beta > 0.67$. This results in steps which are quite visible for small values of β, as can be seen in Figure 6.6.

Let us take another example of finding the probability distribution of covered nodes with β stopping rule. Suppose we want to find the probability distribution of covered nodes at $\beta = 0.68$ then we have

$$\begin{aligned}
Pr\{C_{\beta=0.68} = 2\} &\Rightarrow k = \lceil 2 * 0.68 \rceil = 2 \Rightarrow Pr\{C_{k=2} = 2\} = \frac{1}{N-1} \\
Pr\{C_{\beta=0.68} = 3\} &\Rightarrow k = \lceil 3 * 0.68 \rceil = 3 \Rightarrow Pr\{C_{k=3} = 3\} = \frac{N-2}{N-1}\frac{2}{N-1} \\
Pr\{C_{\beta=0.68} = 4\} &\Rightarrow k = \lceil 4 * 0.68 \rceil = 3 \Rightarrow Pr\{C_{k=3} = 4\} = \frac{N-2}{N-1}\frac{N-3}{N-1} \\
Pr\{C_{\beta=0.68} \geq 5\} &\Rightarrow k = \lceil 5 * 0.68 \rceil = 4 \Rightarrow Pr\{C_{k=4} \geq 5\} = 0
\end{aligned}$$

The probability $Pr\{C_{k=4} \geq 5\} = 0$ because the RW will always terminate at step $k = 3$.

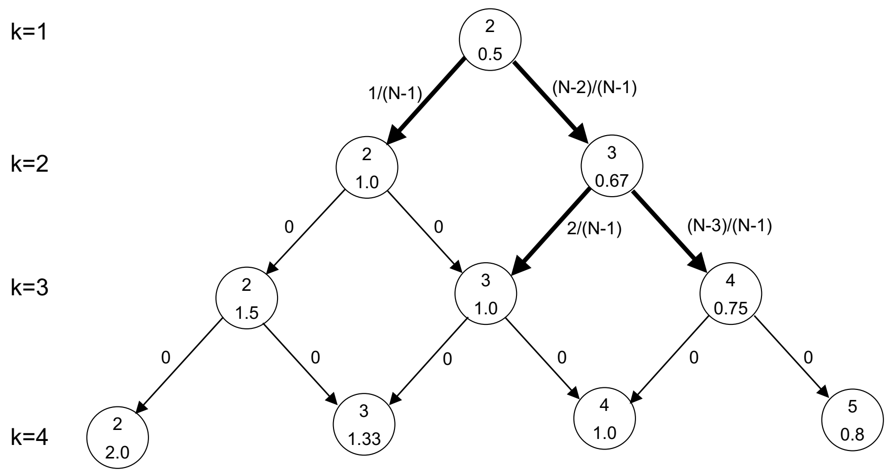


Figure 6.4: Probability transitions to different states of λ when $\beta = 0.68$

Hence generally the probability distribution of covered nodes with β stopping rule can be written as

$$Pr\{C_\beta = X\} = Pr\{C_{k^*} = X\} \tag{6.5}$$

where k^* is an integer such that

$$X\beta \le k^* < X\beta + 1$$

The probability distribution of C_k, when $k = k^*$ is

$$\begin{aligned} Pr\{C_k = X\} &= Pr\{C_{k-1} = X-1\}Pr\{(k-1, X-1) \to (k, X)\} \\ &\quad + Pr\{C_{k-1} = X\}Pr\{(k-1, X) \to (k, X)\} \end{aligned} \tag{6.6}$$

where the probabilities $Pr\{(k-1, X-1) \to (k, X)\}$ and $Pr\{(k-1, X) \to (k, X)\}$ represent the transition probabilities from state $(k-1, X-1)$ to (k, X) and $(k-1, X)$ to (k, X) respectively. These transition probabilities are defined below.

$$Pr\{(k-1, X-1) \to (k, X)\} = \begin{cases} \frac{N-X+1}{N-1} & \frac{k-1}{X-1} < \beta \\ 0 & \text{otherwise} \end{cases}$$

$$Pr\{(k-1, X) \to (k, X)\} = \begin{cases} \frac{X+1}{N-1} & \frac{k-1}{X} < \beta \\ \\ 0 & \text{otherwise} \end{cases}$$

We have validated the above model with the simulations. Each probability was calculated after 10^5 iterations. Figures 6.5(b) and 6.5(a) show probability distribution of normalized coverage or partial coverage. We see that the simulations and numerical values from the probabilistic model perfectly agree.

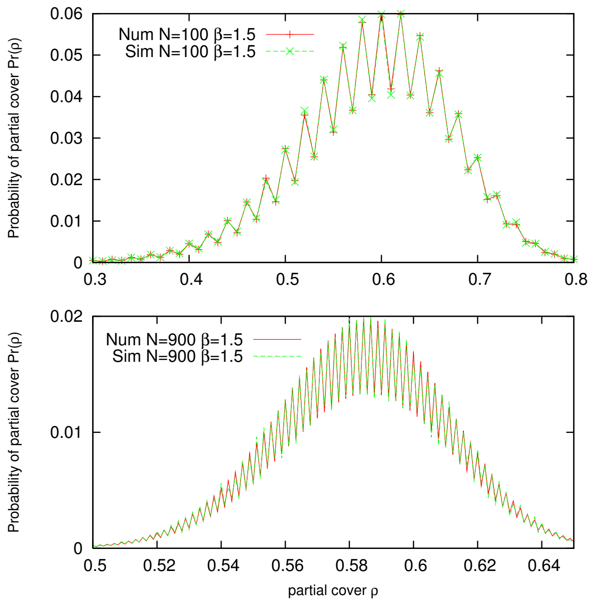
We observe fluctuations in Figure 6.5(a). Generally for a specific value of β the probability of covered nodes $Pr\{C_\beta = X\}$ increases with increasing X and then it decreases as in Figure 6.5(b) but there can also be fluctuations in this increasing probabilities. This can be explained from the Figure 6.4 in which it is can be seen that it may happen that at a particular step k the total transition paths leading to a particular state may result in lower probabilities even when the probability $Pr\{C_\beta = X\}$ is increasing.

Mean Value Analysis of C_β

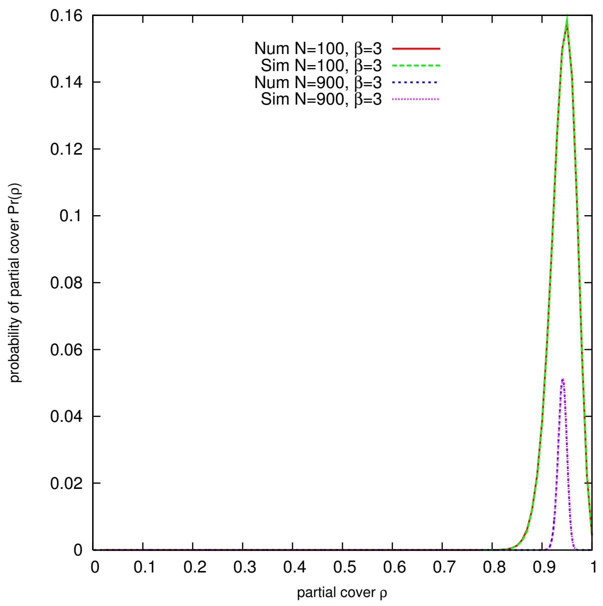
The probabilistic model is given by equations 6.5 and 6.6 which gives the probability distribution of covered node with the stopping rule β. We can find the mean covered nodes from this model by using the definition of mean of random variable given as

$$E[C_\beta] = \sum_{\text{all } x} x Pr(C_\beta = x)$$

To have comparison for different values of N we normalize $E[C_\beta]$ with N, which is also the mean partial cover $E[\rho]$. The graphs in the Figure 6.6 shows the variation of $E[\rho]$ with respect to the stopping rule β. We see that all the graphs start from $\beta = 0.5$. This is because this is minimum value of β that exist since at the first step, two nodes are visited. We also observe an interesting fact that with β little greater than one, the variation of partial cover does not depend on the number of nodes. We shall further investigate the dependence of mean partial cover with N in the next section.



(a) Validation with $\beta = 1.5$



(b) Validation with $\beta = 3.0$

Figure 6.5: Validation of probabilistic analysis of β

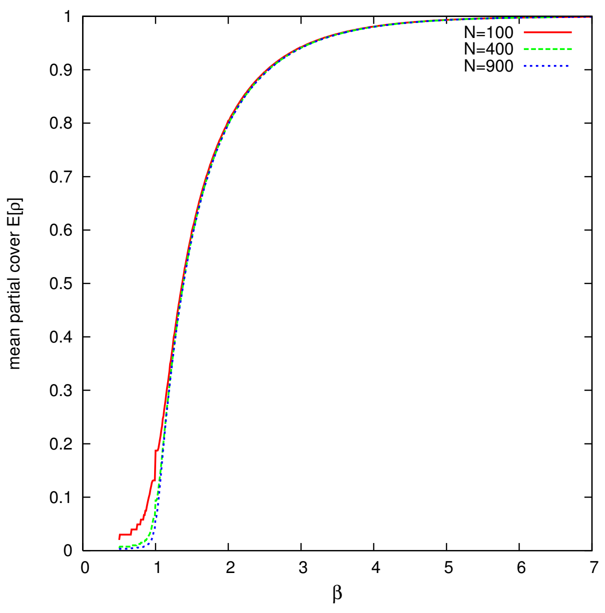


Figure 6.6: Mean covered node variation with β stopping rule

6.3.3 Comparison between TTL and β stopping rules

Now we shall compare TTL and β stopping rules using the mathematical model that we have developed. In this regard first we shall compare with respect to mean partial cover and then standard deviation of partial cover.

Variation of mean partial cover with respect to N

The plots in the Figures 6.7 and 6.8 are plotted from the mathematical models discussed in the previous sections. We infer from these plots that mean partial cover with a specific value of TTL depends on N where as the mean partial cover with a specific value of β that slightly greater than one does not depend on N. It is clear from the graphs of Figure 6.7 that with a constant TTL, partial cover decreases with increasing number of nodes and it is not possible to estimate partial cover without knowing N. On the other hand the graphs of Figure 6.8 show that with a specific value of β partial cover remains practically constant with increasing N. In this case we can estimate the mean partial cover even if we not know N because partial cover is only dependent on β as shown by graph in Figure 6.6. The graphs of Figure inside the Figure 6.8 shows that for small values of N the mean partial cover decreases and quickly settles for almost constant values.

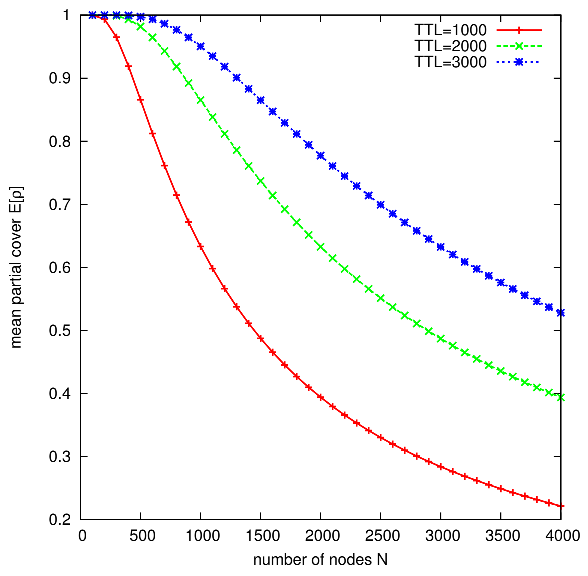


Figure 6.7: Variation of mean partial cover with N for TTL stopping rule

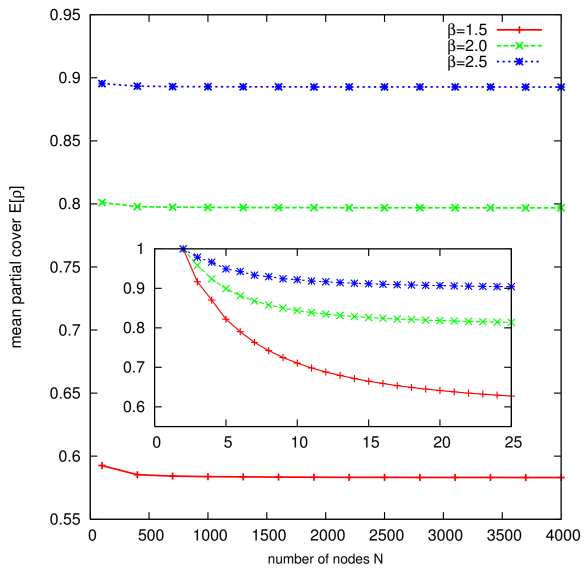


Figure 6.8: Variation of mean partial cover with N for β stopping rule

Variation of standard deviation of partial cover

We know that partial cover ρ is a random variable. The standard deviation of partial cover is thus a measure of how the distribution of partial cover is spread out around the mean partial cover. The best would be to have least spreading of partial cover around its mean, that is, least standard deviation. The standard deviation for the probabilistic model given by 6.5 and 6.6 can be calculated using the following equation [77].

$$SD(\rho) = \sqrt{\sum_{\text{all } x} \rho^2 Pr(\rho = x) - \left[\sum_{\text{all } x} \rho Pr(\rho = x)\right]^2}$$

The Figures 6.9 and 6.10 show the $SD(\rho)$ with respect to TTL and β respectively. It is interesting to note that both have nearly the same kind of pattern, that is, the standard deviation initially increases rapidly and then slowly decreasing with increasing the stopping rule. We also note that standard deviation is low in both cases with increasing number of nodes. It is generally less in case of TTL as compared to β stopping rule. The Figure 6.11 shows $SD(\rho)$ in both cases with respect to the mean partial cover. It is clear that the $SD(\rho_\beta)$ remains always higher than $SD(\rho_{TTL})$ from different values of N. The nice thing is that the difference between $SD(\rho_\beta)$ and $SD(\rho_{TTL})$ is less for higher values of mean partial cover.

From the plots of standard deviation SD for a complete graphs, we also see that although with β, the partial cover has more SD than with TTL but with higher values of mean partial cover the difference between both SDs decrease. We can thus have lower SD for higher values of β. But on the other hand from the Figure 6.6 we observe that increasing β beyond a certain value, which is around 2.5, the rate of increase of mean partial cover decreases which means that more effort is required to cover additional nodes. As often we might not be interested in covering the whole network, a good compromise between a small SD and high transmissions per node is to cover 0.8 fraction of the network.

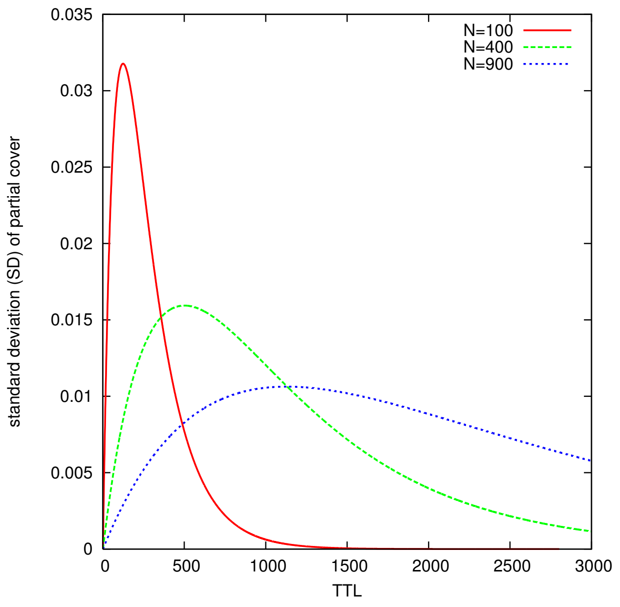


Figure 6.9: Variation of standard deviation of partial cover with TTL

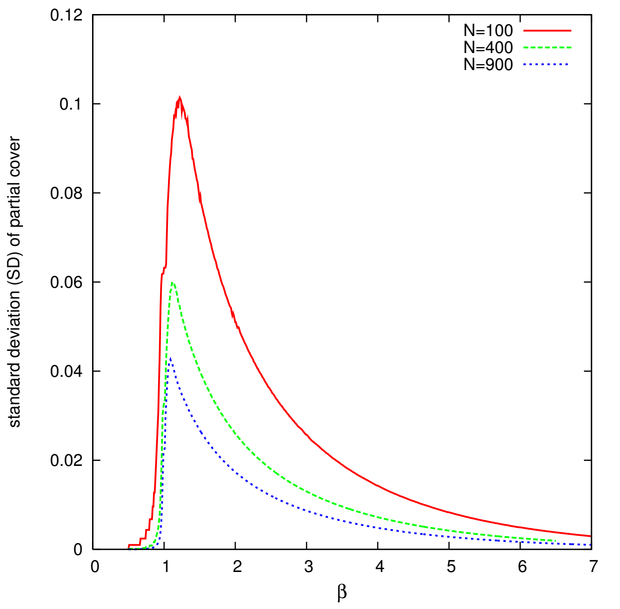


Figure 6.10: Variation of standard deviation of partial cover with β

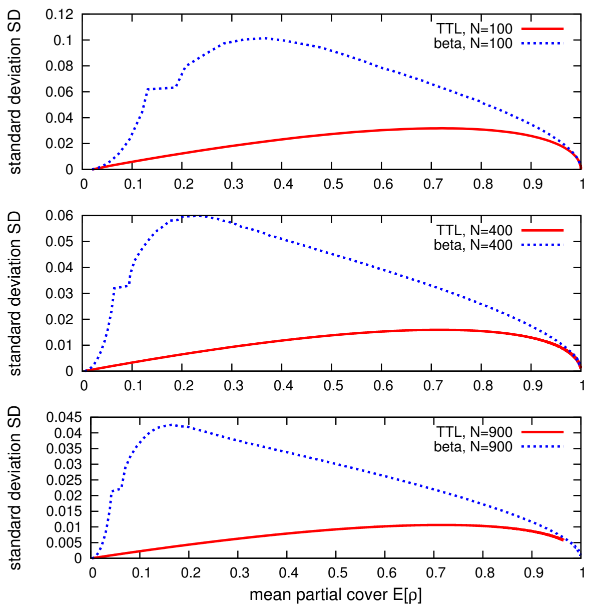


Figure 6.11: standard deviation of partial cover with corresponding mean partial cover

6.4 RW on a finite torus grid

Encouraged by the results obtained for RW using β stopping rule on a complete graph, we shall now study both TTL and β stopping rules for RW on a torus grid, which is graph model often used for wireless ad hoc networks. We shall only do the mean value analysis for both stopping rules.

6.4.1 TTL stopping rule

We cannot find the number of covered nodes as a function of steps using Markov chain in a torus grid because the number of covered nodes at a step not only depend on the present state of the RW but also on the previous state i.e., path of RW. There has been studies regarding mean coverage analysis on a grid but these studies either assume infinite grid [17] [20] or study the asymptotic behavior of coverage as the number of steps tends to infinity. Our analysis of mean partial cover of RW on a finite torus grid with finite number of steps is based on the notion of return time of RW and absorbing state Markov chain.

Definition *Return Time (RT)* is the number of steps for a walk starting at a node to return to that node after leaving it.□

Let the probability of visiting a new node at step k be γ_k, then from equation 6.1 and 6.2 the expected covered nodes at step $k+1$ are

$$E[C_{k+1}] \quad = \quad E[C_k] + \gamma_{k+1}$$

We suppose that at step 0 the initial node is visited, that is, $E[C_0] = \gamma_0 = 1$. Note also that at step 1 the RW always visits a new node, that is, $\gamma_1 = 1$. The above is an iterative equation which results in the sequence for $k \geq 0$ as given

$$E[C_k] \quad = \quad \gamma_0 + \gamma_1 + \gamma_2 + \gamma_3 + ... + \gamma_k \tag{6.7}$$

Let N_k represent a node visited at step k. At each step the node N_k select one of its d neighbors with the uniform probability $1/d$. Let $\mathcal{N}_k$ be the set of nodes visited in steps $k \geq 0$, that is, $\mathcal{N}_k = (N_0, N_1, ..., N_k)$

$$\begin{aligned}
\gamma_k &= Pr\left\{\text{visiting a new node at step k}\right\} \\
&= Pr\left\{N_k \notin \mathcal{N}_k\right\} \\
&= Pr\left\{N_k \notin (N_0, N_1, ..., N_{k-1})\right\} \\
&= Pr\left\{N_k \not\equiv N_0, N_k \not\equiv N_1, ..., N_k \not\equiv N_{k-1}\right\}
\end{aligned}$$

The event $N_k \not\equiv N_1$ means that the node at step k is not the same as the node at step 1. The probability of this event is same as the probability of the event $N_{k-1} \not\equiv N_0$. This is because of the regularity of the torus grid topology, that is, at each step a node can select one node from its d neighbors with same probability $1/d$ and thus the probability of two nodes not being identical after same number of steps (that is, $k-1$ steps) is same in case of both events. Thus we have

$$\begin{aligned}
Pr\left\{N_k \not\equiv N_1\right\} &= Pr\left\{N_{k-1} \not\equiv N_0\right\} \\
Pr\left\{N_k \not\equiv N_2\right\} &= Pr\left\{N_{k-2} \not\equiv N_0\right\} \\
\vdots \qquad & \qquad \vdots \\
Pr\left\{N_k \not\equiv N_{k-1}\right\} &= Pr\left\{N_1 \not\equiv N_0\right\}
\end{aligned}$$

Thus

$$\begin{aligned}
\gamma_k &= Pr\left\{N_k \not\equiv N_0, N_{k-1} \not\equiv N_0, N_{k-2} \not\equiv N_0, ..., N_1 \not\equiv N_0\right\} \\
&= Pr\left\{N_0 \not\equiv N_i, \text{where } i = 1, 2, ..., k\right\} \\
&= Pr\left\{\begin{matrix}\text{The node visited in step 0 is not} \\ \text{visited in previous } k \text{ steps}\end{matrix}\right\} \\
&= Pr\left\{\begin{matrix}\text{RW does not return to origin in} \\ \text{any of previous } k \text{ steps}\end{matrix}\right\} \\
&= Pr\{RT \geq k\}
\end{aligned} \tag{6.8}$$

We then have the interesting result that the probability of visiting a new node at step k is the same as the probability of the return time being grater than k. Please note that this result is valid only for a regular topology, like finite torus grid. In case of a finite square grid we cannot use this result because the edge nodes do not have the same neighbor distribution as the nodes which are inside the edges.

Another useful relation follows from the regularity of the tours grid. It can be seen that for such a topology the probability of not returning to origin at any of the previous k steps is the same as probability of not returning to origin at any of the previous $k+1$ steps for $k = 0, 2, 4, ...$, that is, for k even we have $\gamma_k = \gamma_{k+1}$.

We find can the $Pr\{RT \geq k\}$ using absorbing state Markov chain as follows. It is clear that

$$Pr\left\{\begin{array}{l}\text{RW does not return}\\ \text{to orgin in any of the}\\ \text{previous } k \text{ steps}\end{array}\right\} = 1 - Pr\left\{\begin{array}{l}\text{RW returns to origin}\\ \text{in any of the previous}\\ k \text{ steps}\end{array}\right\} \tag{6.9}$$

The probability that RW returns to origin in any of the previous k steps can be found using an absorbing state Markov chain as we are interested in the behavior of the process only up to the time it reaches an absorbing state.

To find the probability of returning to origin in any of the previous k steps, we have to model the return of RW to the starting state. In the probability transition matrix of Markov chain it is not possible to have same state as starting and absorbing state. For this reason we choose one of the four neighbors of the source node as the starting state i. The source node j is made as the absorbing state. Thus we can have

$$Pr\left\{\begin{array}{l}\text{RW returns to origin in any}\\ \text{of the previous } k \text{ steps}\end{array}\right\} =$$
$$\left(\begin{array}{l}\text{number of}\\ \text{neighbors}\end{array}\right) Pr\left\{\begin{array}{l}\text{transition from } j \text{ to}\\ \text{one of the neigh-}\\ \text{bors } i \text{ in step 1}\end{array}\right\} Pr\left\{\begin{array}{l}\text{transition from } i \text{ to}\\ j \text{ in } k-1 \text{ steps}\end{array}\right\}$$

Let the probability of transition from state i to state j at step $k-1$ is represented by $p_{ij}^{(k-1)}$. We know that each node has 4 neighbors then from above equation we have

$$Pr\left\{\begin{array}{l}\text{RW returns to origin in any}\\ \text{of the previous } k \text{ steps}\end{array}\right\} = p_{ij}^{(k-1)}$$

We thus have from equation 6.9

$$\gamma_k = 1 - p_{ij}^{(k-1)}$$

From equation 6.7, we have

$$E[C_k] \quad = \quad 1 + (1 - p_{ij}^{(0)}) + (1 - p_{ij}^{(1)}) + (1 - p_{ij}^{(2)}) + ... + (1 - p_{ij}^{(k-1)})$$

where $k = 1, 2, 3, ...$ We thus have the following expression

$$E[C_k] \quad = \quad 1 + k - \sum_{n=0}^{k-1} p_{ij}^{(n)} \tag{6.10}$$

The probability $p_{ij}^{(n)}$ is the ij^{th} element of the n^{th} power of the probability transition matrix of the Markov chain that describe RW on a torus grid in which each node has the same neighbor degree. In torus grid is a square grid in which the nodes on the two vertical edges are made neighbors and similarly the nodes on the two horizontal edges are also made neighbors. Let $\mathbf{s}_1$ and $\mathbf{s}_2$ be any two points on the torus and $||\mathbf{s}_1, \mathbf{s}_2||$ the distance between $\mathbf{s}_1$ and $\mathbf{s}_2$. Then the probabilities of the transition matrix are given by

$$p_{\mathbf{s}_1\mathbf{s}_2} = \begin{cases} 1/d(\mathbf{s}_1) & ||\mathbf{s}_1, \mathbf{s}_2|| = 1 \\ 0 & \text{otherwise} \end{cases}$$

where $d(\mathbf{s})$ is the number of neighbors of point $\mathbf{s}$ which is always 4 in case of torus grid.

We simulated the Markov chain given in equation 6.10 and also has simulations of RW on a torus grid to verify the model described by equation. The plots in the Figure 6.12 show that the numerical values calculated using the model perfectly agrees with the simulation results.

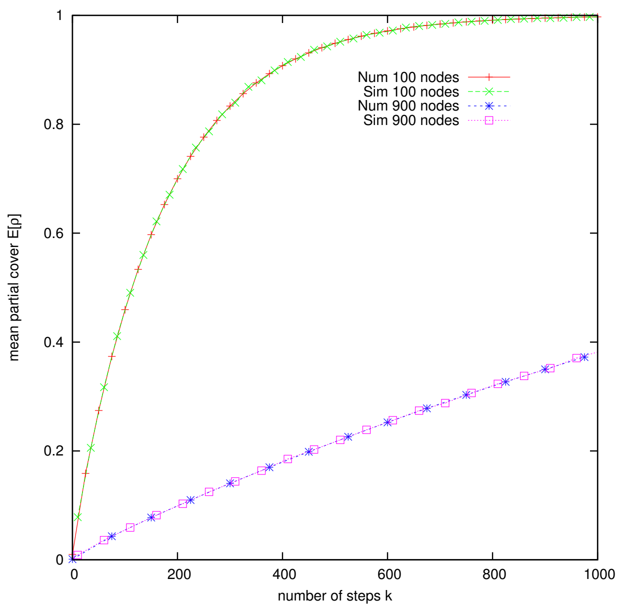


Figure 6.12: Validation of equation 6.10 for RW on a torus grid

6.4.2 β stopping rule

Once the expected number covered nodes at each step is known, we can approximate the lifetime of a RW using β stopping rule by assuming that the walk lasts for k steps, where k is the smallest value such that $k \geq \beta C_k$, hence replacing the actual number of covered nodes with its expected value. Note that k/C_k is an increasing function of k. By doing so, we are able to study the coverage Vs β relationship.

We simulated RW with β stopping rule to find relation between the coverage and β. The plots in Figure 6.13 show that the model provides a very good approximation of the relationship between coverage and β. Since for analyzing the RW with β stopping rule we have replaced the actual number of nodes covered when the walk stops with the expected number of covered nodes corresponding to the stopping time, the difference among the two relationships is minimum when the probability distribution of the covered nodes is narrow around its mean, which indeed occurs when the number of covered nodes is very low or very high (β small and β high). The difference is wider in the other cases.

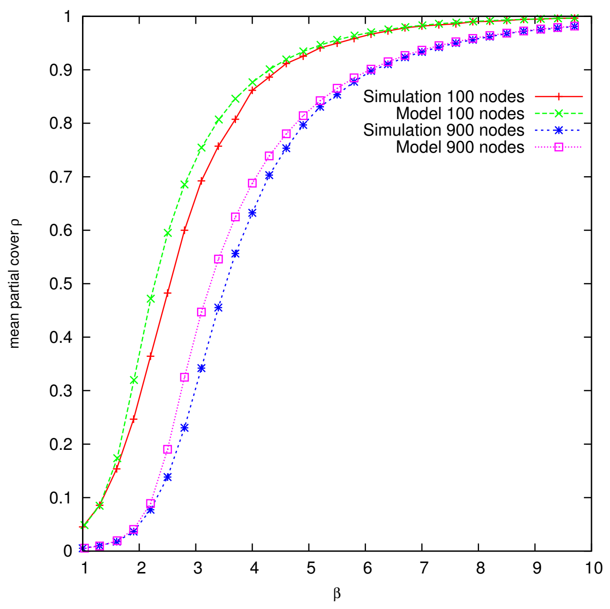


Figure 6.13: Validation of coverage with β stopping rule

6.4.3 Comparison between TTL and β stopping rules

The expected covered nodes for TTL stopping rule for a finite torus grid can be found by putting $k = TTL$ in the model given by equation, that is,

$$E[C_{TTL}] = 1 + TTL - \sum_{n=0}^{TTL-1} p_{ij}^{(n)}$$

where i and j are any two neighbor nodes in the torus grid. Note that in this Markov chain model, number of nodes N does not appear explicitly, but in fact the probability of RW starting from node i and reaching its neighbor j i.e., $p_{ij}^{(n)}$ depends on the number of nodes N. Using this model we calculated numerically the variation of estimated partial cover $E[\rho]$ with increasing number of nodes for a specific TTL, shown by the plots given in Figure 6.14.

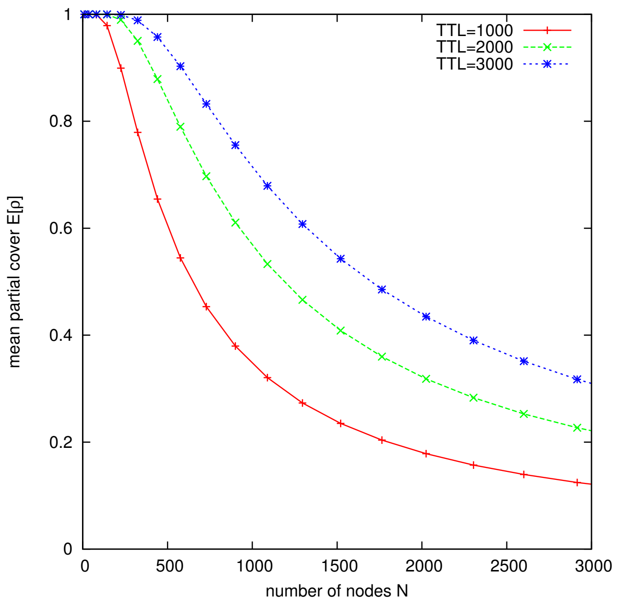


Figure 6.14: Variation of $E[\rho]$ with N for different TTL on a torus grid

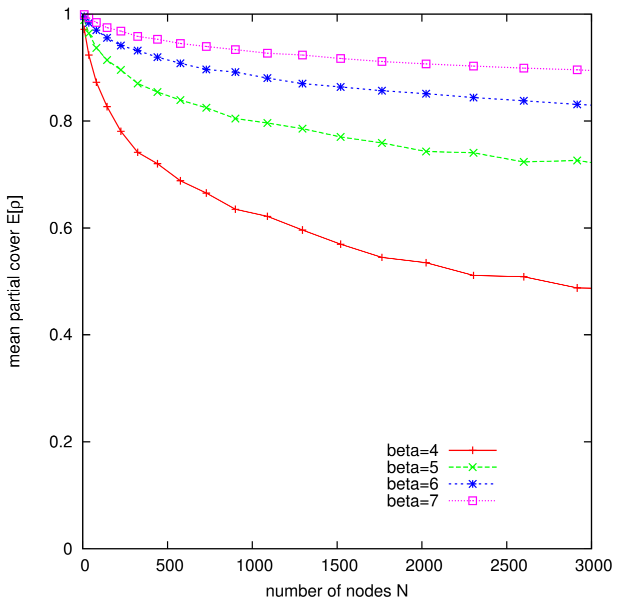


Figure 6.15: Variation of $E[\rho]$ with N for different β on a torus grid

We see that with increasing number of nodes the partial cover constantly decreases for any value of TTL. This is quite understandable as for the same number of steps the fraction of network covered would always decrease with increasing N and expected to approach zero when N approaches infinity for any value of TTL.

The behavior of mean partial cover with increasing number of nodes for different values of the β for a torus grid was found using RW simulations on the torus grid. We calculated each value after a minimum of 1000 iterations. The plots are shown in the Figure 6.15. We observe that for a particular value of β as the number of nodes increase, the expected partial cover initially decrease rapidly and then this decrease slows.

It is interesting to note that these observations are also true for the case of complete graph, in which the asymptotic values very quickly tend to stabilize even for small values of N, as can be seen in the in plots of the Figure 6.8.

We also observe from Figures 6.15 and 6.8 that the asymptotic values of expected partial cover depends on the connectivity degree of the nodes. For a more connected network small values of β tend to cover more network with increasing number of nodes as compared to a network with less connectivity degree.

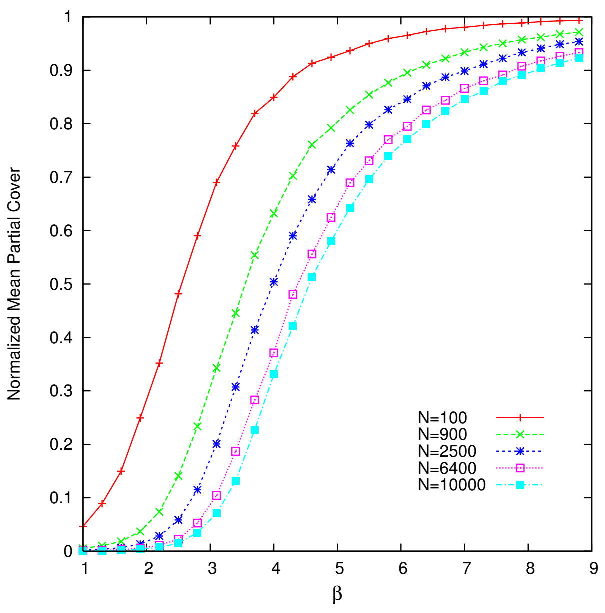


Figure 6.16: Coverage β relationship.

Figure 6.16 shows the effect of the network size on the coverage-β relationship. These plots are obtained from the model. For any value of β, as the size of the network decreases the covered nodes increases. For example, for $\beta = 4$ and $N = 900$, the mean fraction of covered nodes is about 0.65 while for $N = 100$ the fraction is 0.85. Due to such inverse relationship, once β is fixed on the basis of the required coverage and of the nominal maximum size, the required coverage is guaranteed for any lower size.

Of course, the target coverage can also be satisfied trivially by assign a static TTL value to the lifetime, where TTL is calculated for the maximum size. The real advantage of the proposed β-stopping rule is that the RW lifetime decreases according to the network size, thus meeting the target coverage at a lower cost compared to using TTL stopping rule.

Figure 6.17 shows such a self-adapting capability by reporting the average number of steps as a function of the network size for a target coverage of 0.8. The lifetime of a RW with perfect adaptable TTL is also provided. Such an ideal walk stops when the expected number of covered nodes is exactly 0.8. We can see how the lifetime of our proposed protocol decreases with the size and follows the same behavior of such an ideal case.

For example, from the plot we can see that for a grid of $30 \times 30 = 900$ nodes the lifetime of the walk should be about 3500. Then, in the case of RW with TTL, the value of TTL is to be set to such a value. The same coverage of 0.8 is obtained with a RW using β stopping rule with $\beta = 4.77$ (see also Figure 6.16).

Let now suppose the actual size of the network be of $25 \times 25 = 625$ nodes instead of 900 nodes. This reduction in size can due, for example, to a partial deployment of the network, or to the failures of some nodes (in this case we are approximating the real network with a shrink version of the grid).

On this network, a RW with β stopping rule with the same value of $\beta = 4.77$ stops after 2460 steps, on the average. The minimum number of steps required to cover 80% of the network is now 2267; thus, the RW with β stopping rule performs only about 8% more steps than the strictly required one, while for a RW with TTL stopping rule, the steps are about 50% more than what it is required.

6.5 Conclusions

In this chapter we proposed and studied a novel stopping rule based on the number of steps and covered node. Both of these quantities can be calculated online. The proposed stopping rule, which we have called β stopping rule, terminates the RW when the ratio between the number of steps and number of covered nodes becomes greater than a value β. We studied analytically and also through simulations β stopping rule and mostly used stopping rule TTL. We first studied RW using both stopping rules on complete graph. Motivated

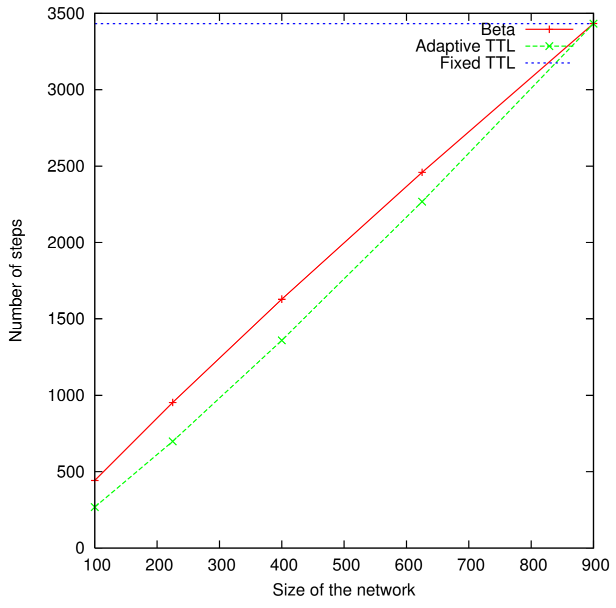


Figure 6.17: Lifetime of random walks vs network size (target coverage is 80%).

with the results on a complete graph, we then studied the stopping rules with RW on torus grid. We conclude that β stopping rule seems to show a nice property that the RW coverage with this stopping rule on the average depends very less on the network size as opposed to RW coverage from TTL stopping rule, which highly depends on network size. In case of β stopping rule we are sure of approximately covering at least some portion of the network coverage before the RW stops. The most important interpretation of the results is that the RW with β stopping rule adapts its lifetime according to the network size with the mean partial coverage mainly depending on β.

Chapter 7

Conclusions and future directions

7.1 Summary and Conclusions

In this thesis we discussed and proposed RW for searching in wireless ad hoc network. We first analyzed some of existing SDPs for MANETs. We gave a general frame for the SDPs in MANETs, categorized the SDPs and identify some open problems in the field. We found that most of the SDPs used flooding for searching a service. Flooding can be simple or overlay based or probabilistic, that is, gossiping. Each type has it own problems which make them unsuitable for large networks with high dynamism. We proposed to use RW in wireless ad hoc networks for searching. RW has been used in wired P2P networks for searching for a long time and even in wireless ad hoc networks for tasks other than searching, but still we see not many applications using RW as search tool in wireless domain. We identified three main problems related to RW in wireless domain. First is the efficient and robust implementation of one step of RW, second reducing the overall cost of search, that is, the mean number of steps to reach from search initiating node to destination and third is defining when to stop RW. We worked on each problem separately and come up with solutions presented in this thesis.

We presented an efficient distributed protocol for implementing one step of RW. This protocol, called as RW-DS, is a four phase message exchange protocol. To make one step of query packet this protocol uses two extra packets. There are three timers whose proper tuning is needed to have a sustainable RW. The latency in one step of query depends on the tuning of these timers. We discussed how to tune the timers for low latency. We then compared the proposed protocol with a standard centralized selection protocol called as RW-CS using detailed simulations. We thoroughly explained the reasons of RW-DS being more robust than RW-CS in mobility scenarios, RW-DS being more bandwidth efficient, energy efficient and having low latency as compared with RW-CS. We also proved that the RW-DS has a non-splitting property which means that during the course of RW it will never happen that the same RW gets divided into two or more RWs.

The problem of reducing the overall number of steps from query source to destination was dealt by proposing a biasing strategy for the RW. Biasing can be done mainly based on information in two different ways. One is using global information source such as location aware mechanism and the other is just using the information available local to nodes. As we do not assume the presence of a global observer and that the network devices to have hardware to access some global information source, we focussed on the second method of biasing the RW. We used the notions of active and passive visits and used these notions for biasing the RW. We compared the cost of this biasing strategy with that of existing biasing strategies that only rely on local information and with flooding, and found that the proposed biasing strategy performs best among them. The proposed biasing strategy performs even better than gossiping.

In the last we investigated the problem of termination of RW. Usually TTL is used but this is a very inefficient way to terminate RW as it requires to know the size of the network in advance for searching a specific portion of the network. We studied analytically the TTL stopping rule and found shortcomings in this stopping rule. In wireless ad hoc networks the number of nodes may, for example, decrease, due to churn. If TTL has been set to cover a fraction of network, then with decreased network size RW will be covering more fraction of network than originally set, thus consuming network resources. We proposed another stopping rule and called it as β stopping rule, which depends on two metrics that can be calculated online with just the local information available to devices. We have studied both TTL and β stopping rules analytically and through simulations on a complete graph and torus grid. We found that with β stopping rule the RW adapts its lifetime with network size to approximately covering the fraction of network as originally set. Also we found that minimum expected partial coverage before the RW stops is very less depended on the network size as compared with RW with TTL stopping rule.

7.2 Future directions

The protocol presented in Chapter 4 is robust in mobility but we have seen that the under high mobility sometimes RW suddenly stops. We found that this is due to the selection of neighbors that are close to the edge of transmission range of the forwarding node. This phenomenon can be further studied and methods devised to avoid selecting such neighbors to make the protocol even more robust. The setting of the timers of the proposed protocol mainly depend on the neighbor density of the network. In a realistic network a node's neighbor density can vary with respect to time and position. One can investigate methods for adaptive CTFTimer and queryTimer which can adjust the timers dynamically depending on a node's neighbor density to achieve low latency.

We have proposed a stopping rule for the RW with which it is possible to stop the RW at least after covering an expected fraction of network with just nominally knowing the size of the network and also the lifetime of the RW adapts according to the network size. We have analyzed and then verified thought simulation the proposed stopping rule for RW on a complete graph and on a torus grid. A more practical approach would be to study these properties of the proposed stopping rule on random geometric graphs.

In this thesis we have proposed an efficient one step implementation of RW, an efficient biasing strategy and a stopping rule for the RW in which the portion of cover nodes are insensible to network size. These all aspects are studied in detail one by one. These results can be integrated as a single protocol for implementing on real devices.

We know that the pure RW is a Markovian process as it does not depend on the history of the process and just depend on the present state of the process. The biased RW, on the other hand is a non-Markovian process in which the next state of RW not only depend on the present state but also the history of the RW. Such processes are difficult to model. It would be interesting to look into the possibility of an approximate model of biased RW that can be studied with the help of Markov theory.

An emerging approach for streaming in wireless networks is P2P streaming over wireless ad hoc network. Instead of contacting the central server the chunks of data are retrieved from any device in P2P manner. These chunks have to be searched and retrieved within time constraints. It is not possible to flooding the network for every chunk. The use of RW for searching in such a scenario can be very efficient as many RW searches can be initiated without being overloading the network with query requests.

Bibliography

[1] Network simulator ns-2. `http://www.isi.edu/nsnam/ns/`.

[2] *Introduction to Probability.* American Mathematical Society, Providence, RI, USA, 2nd edition, 1997.

[3] I. S. 802.11-1999. *Part11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications.* IEEE, 1999 edition.

[4] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman. Search in power-law networks. *Physical Review E*, 64(4):046135, September 2001.

[5] M. Alanyali, V. Saligrama, and O. Savas. A random-walk model for distributed computation in energy-limited networks. In *Proceedings of the Inaugural Workshop for the Center for Information Theory and its Applications*, University of California San Diego, February 2006.

[6] D. Aldous and J. A. Fill. Reversible markov chains and random walks on graphs. Draft Monograph, online at `http://www.stat.berkeley.edu/aldous/RWG/book.html`, 2001.

[7] K. M. Alzoubi, P.-J. Wan, and O. Frieder. Message-optimal connected dominating sets in mobile ad hoc networks. In *MobiHoc '02: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 157–164, Lausanne, Switzerland, 2002. ACM, New York, USA.

[8] C. Avin and C. Brito. Efficient and robust query processing in dynamic environments using random walk techniques. In *IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks*, pages 277–286, Berkeley, California, USA, April 2004. ACM, New York, USA.

[9] C. Avin and B. Krishnamachari. The power of choice in random walks: an empirical study. In *MSWiM '06: Proceedings of the 9th ACM international symposium on Modeling analysis and simulation of wireless*

and mobile systems, pages 219–228, Terromolinos, Spain, 2006. ACM, New York, USA.

[10] C. Avin and B. Krishnamachari. The power of choice in random walks: An empirical study. *Computer Networks*, 52(1):44–60, 2008.

[11] R. Baldoni, A. N. Mian, S. Scipioni, and S. Tucci-Piergiovanni. Churn resilience of peer-to-peer group membership: A performance analysis. *Lecture Notes in Computer Science, Distributed Computing, IWDC 2005*, 3741/2005:226–237, December 2005. Springer Berlin / Heidelberg.

[12] Z. Bar-Yossef, R. Friedman, and G. Kliot. RaWMS - Random walk based lightweight membership service for wireless ad hoc networks. In *MobiHoc '06: Proceedings of 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 238–249, Florence, Italy, May 2006. ACM.

[13] S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic. *Mobile Ad Hoc Networking*. IEEE Press, John Wiley & Sons, Inc. NJ, August 2004. pages 69-116.

[14] R. Beraldi. The polarized gossip protocol for path discovery in manets. *Ad Hoc Networks*, 6(1):79–91, 2008.

[15] D. Braginsky and D. Estrin. Rumor routing algorthim for sensor networks. In *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 22–31, Atlanta, Georgia, USA, 2002. ACM, New York, USA.

[16] S. Broadbent and J. Hammersley. Percolation processes. i. crystals and mazes. In *Proceedings of the Cambridge Philosophical Society*, volume 53, pages 629–641, July 1957.

[17] M. J. A. M. Brummelhuis and H. J. Hilhorst. Covering of a finite lattice by a random walk. *Physica A Statistical Mechanics and its Applications*, 176:387–408, September 1991.

[18] R. Burioni and D. Cassi. Random walks on graphs: ideas, techniques and results. *Journal of Physics A*, 38(8):R45, March 2005.

[19] F. Cali, M. Conti, and E. Gregori. IEEE 802.11 protocol: design and performance evaluation of an adaptive backoff mechanism. *IEEE Journal on Selected Areas in Communications*, 18(9):1774–1786, September 2000.

[20] S. Caser and H. J. Hilhorst. Topology of the support of the two-dimensional lattice random walk. *Physical Review Letters*, 77:992–995, August 1996.

[21] D. Chakraborty, A. Joshi, T. Finin, and Y. Yesha. Gsd:a novel group-based service discovery protocol for manets. In *MWCN '02: Proceedings of the 4rth IEEE Conference on Mobile and Wireless Communications Networks*, September 2002.

[22] D. Chakraborty, A. Joshi, Y. Yesha, and T. Finin. Toward distributed service discovery in pervasive computing environments. *IEEE Transactions on Mobile Computing*, 5(2):97–112, February 2006.

[23] N. B. Chang and M. Liu. Revisiting the ttl-based controlled flooding search: optimality and randomization. In *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 85–99, Philadelphia, PA, USA, 2004. ACM, New York, USA.

[24] N. B. Chang and M. Liu. Controlled flooding search in a large network. *IEEE/ACM Transactions on Networking*, 15(2):436–449, April 2007.

[25] C. Chaudet, D. Dhoutaut, and I. G. Lassous. Performance issues with IEEE 802.11 in ad hoc networking. *IEEE Communications Magazine*, pages 110–116, July 2005.

[26] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker. Making gnutella-like p2p systems scalable. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 407–418, Karlsruhe, Germany, 2003. ACM, New York, USA.

[27] L. Cheng and I. Marsic. Service discovery and invocation for mobile ad hoc networked appliances. In *IWNA '00: Proceeding of Second International Workshop on Networked Appliances*, December 2000.

[28] Z. Cheng and W. B. Heinzelman. Searching strategy for multi-target discovery in wireless networks. pages 1–10, Aug. 2004.

[29] C. Cho and D. Lee. Survey of service discovery architectures for mobile ad hoc networks. Technical report, Department of Computer and Information Science and Engineering, University of Florida, USA, 2005. `http://folk.uio.no/paalee/referencing_publications/`.

[30] S. Choi and J. Yu. QoS provisioning in IEEE 802.11. In R. Shorey, A. Ananda, M. C. Chan, and W. T. Ooi, editors, *Mobile, Wireless, and Sensor Networks: Technology, Applications and Future Directions*, chapter 3, pages 45–72. IEEE press and Wiley-Interscience, 2006.

[31] E. Cohen and S. Shenker. Replication strategies in unstructured peer-to-peer networks. *SIGCOMM Computer Communication Review*, 32(4):177–190, August 2002.

[32] S. Consortium. Salutation architecture specification v. 2.1, 1999.

[33] M. Conti. Body, personal, and local ad hoc wireless networks. In M. Ilyas, editor, *The Handbook of Ad Hoc Wireless Networks*, chapter 1. CRC PRESS, 2003.

[34] D. Coppersmith, U. Feige, and J. Shearer. Random walks on regular and irregular graphs. *SIAM Journal on Discrete Mathematics*, 9(2):301–308, 1996.

[35] B. Das and V. Bharghavan. Routing in ad-hoc networks using minimum connected dominating sets. In *ICC '97: Proceedings of the IEEE International Conference on Communications*, volume 1, pages 376–380, Montreal, Canada, June 1997.

[36] S. S. Dhillon and P. V. Mieghem. Performance analysis of the antnet algorithm. *Computer Networks*, 51(8):2104–2125, 2007.

[37] S. S. Dhillon and P. V. Mieghen. Comparison of random walk strategies for ad hoc networks. In *Proceedings of the Sixth Annual Mediterranean Ad Hoc Networking Workshop*, June 2007.

[38] S. Dolev, E. Schiller, and J. Welch. Random walk for self-stabilizing group communication in ad hoc networks. In *PODC '02: Proceedings of the twenty-first annual symposium on Principles of distributed computing*, pages 259–259, Monterey, California, 2002. ACM, New York, USA.

[39] S. Dolev, E. Schiller, and J. Welch. Random walk for self-stabilizing group communication in ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(7):893–905, July 2006.

[40] U. Forum. Upnp device architecture 1.0, December 2003. `www.upnp.org/resources/documents/CleanUPnPDA10120031202s.pdf`.

[41] R. Gandhi, S. Parthasarathy, and A. Mishra. Minimizing broadcast latency and redundancy in ad hoc networks. In *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking*

& computing, pages 222–232, Annapolis, Maryland, USA, 2003. ACM, New York, USA.

[42] J. J. Garcia-Luna-Aceves and Y. Wang. Collision avoidance procotols. In P. Mohapatra and S. V. Krishnamurthy, editors, *Ad Hoc Networks: Technologies and Protocols*, chapter 2, pages 23–62. Springer, Boston, USA, 2005.

[43] C. Gkantsidis, M. Mihail, and A. Saberi. Hybrid search schemes for unstructured peer-to-peer networks. In *INFOCOM '05: Proceeding of IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1526–1537, March 2005.

[44] E. Guttman, C. Perkins, J. Veizades, and M. Day. Service location protocol (SLP), v. 2, IETF RFC 2608. June 1999. `http://www.ietf.org/rfc/rfc2608.txt`.

[45] Z. Haas, J. Halpern, and L. Li. Gossip-based ad hoc routing. In *Proceeding of INFOCOM '02*, volume 21, pages 1707–1716, June 2002.

[46] G. Hasslinger and S. Kempken. Applying random walks in structured and self-organizing networks: Evaluation by transient analysis. In *Proceedings of Value tools workshop*, Nantes, France, October 2007.

[47] R. Hekmat. *Ad-hoc Networks: Fundamental Properties and Network Topologies*, chapter 7. Springer, Dordrecht, The Netherlands, 2006. ISBN-10 1-4020-5165-4.

[48] S. Helal, N. Desai, V. Verma, and C. Lee. Konark - a service discovery and delivery protocol for ad-hoc networks. In *WCNC '03: Proceeding of the 3rd IEEE Wireless Communications and Networking Conference*, pages 2107–2113, New Orleans, LA, USA, March 2003.

[49] M. R. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork. Measuring index quality using random walks on the web. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 31(11-16):1291–1303, May 1999.

[50] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In T. Imielinski and H. Korth, editors, *Mobile Computing*, chapter 5, pages 153–181. Kluwer Academic Publishers, 1996.

[51] J. Kahn, J. H. Kim, L. Lovasz, and V. H. Vu. The cover time, the blanket time, and the matthews bound. In *FOCS '00: Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, page 467. IEEE Computer Society, 2000.

[52] B. J. Kim, C. N. Yoon, S. K. Han, and H. Jeong. Path finding strategies in scale-free networks. *Physical Review E*, 65:027103–027103.4, 2002.

[53] M. Klein, B. König-ries, and P. Obreiter. Lanes - a lightweight overlay for service discovery in mobile ad hoc network. In *ASWN '03: Proceedings of the 3rd Workshop on Applications and Services in Wireless Networks*, July 2003.

[54] M. Klein, B. Konig-Ries, and P. Obreiter. Service rings - a semantic overlay for service discovery in ad hoc networks. In *DEXA '03: Proceedings of the 14th International Conference on Database and Expert Systems Applications*, pages 180–185. IEEE Computer Society, september 2003.

[55] Y. B. Ko and N. H. Vaidya. Flooding-based geocasting protocols for mobile ad hoc networks. *Mobile Networks and Applications*, 7(6):471–480, 2002.

[56] U. C. Kozat and L. Tassiulas. Network layer support for service discovery in mobile ad hoc networks. In *Proceedings of IEEE INFOCOM-2003*, San Francisco.

[57] B. Krishnamachari and J. Ahn. Optimizing data replication for expanding ring-based queries in wireless sensor networks. In *WiOpt '06: Proceedings of 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, pages 1–10, April 2006.

[58] B. Krishnamachari, S. B. Wicker, R. Béjar, and M. Pearlman. Critical density thresholds in distributed wireless networks. In *Proceedings of Communications, Information and Network Security*, pages 1–15. Kluwer Publishers, December 2002.

[59] V. Lenders, M. May, and B. Plattner. Service discovery in mobile ad hoc networks: A field theoretic approach. In *WoWMoM '05: Proceedings of the Sixth IEEE International Symposium on World of Wireless Mobile and Multimedia Networks*, volume 1, pages 120–130. IEEE Computer Society, June 2005.

[60] J. Li, C. Blake, D. S. D. Couto, H. I. Lee, and R. Morris. Capacity of ad hoc wireless networks. In *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 61–69, Rome, Italy, 2001. ACM, New York, USA.

[61] W. Lou and J. Wu. Double-covered broadcast (dcb): a simple reliable broadcast algorithm in manets. In *INFOCOM '04: Proceedings of Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 2084–2095, March 2004.

[62] L. Lovász. Random walks on graphs: a survey. In *Combinatorics Paul Erdös in Eighty*, volume 2, pages 1–46. J'anos Bolyai Mathematical Society, Budapest, Hungry, 1993.

[63] L. Lovász and P. Winkler. Efficient stopping rules for markov chains. In *STOC '95: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 76–82, Las Vegas, Nevada, United States, 1995. ACM, New York, USA.

[64] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *ICS '02: Proceedings of the 16th international conference on Supercomputing*, pages 84–95, New York, 2002. ACM, New York, USA.

[65] R. Marin-Perianu, P. Hartel, and H. Scholten. A classification of service discovery protocols. Technical report, Department of Electrical Engineering, Mathematics and Computer Science (EEMCS), University of Twente, Netherlands, June 2005.

[66] A. N. Mian, R. Baldoni, and R. Berladi. An efficient biasing strategy for randomwalk in wireless ad hoc networks. In *IWCMC '08: Proceedings of the International Wireless Communications and Mobile Computing Conference*, pages 1087–1092, Crete Island, Greece, August 2008. IEEE.

[67] A. N. Mian, R. Beraldi, and R. Baldoni. A closer investigation of service discovery protocols in mulithop mobile ad hoc networks. *IEEE Pervasive Computing*, (To appear).

[68] A. N. Mian, R. Beraldi, and R. Baldoni. Identifying open problems in random walk based service discovery in mobile ad hoc networks. In *6th International Workshop on Innovative Internet Community Systems*, Neuchatel, Switzerland, June 2006.

[69] A. N. Mian, R. Beraldi, and R. Baldoni. Survey of service discovery protocols in mobile ad hoc networks. Technical report 4/06, Dipartimento di Informatica e Sistemistica "Antonio Ruberti", Università degli Studi di Roma "La Sapienza", Rome, Italy, April 2006. `http://www.dis.uniroma1.it/~midlab/articoli/SSDP.pdf`.

[70] A. N. Mian, R. Berladi, and R. Baldoni. A robust and energy efficient protocol for random walk in ad hoc networks with ieee 802.11. In *IPDPS '08: Proceedings of the 22nd IEEE International Parallel and Distributed Processing Symposium*, Miami, USA, April 2008. IEEE.

[71] S. Microsystems. Jini technology core platform specification, v. 2.0, June 2003. `www.sun.com/software/jini/specs/core2_0.pdf`.

[72] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu. The broadcast storm problem in a mobile ad hoc network. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 151–162, Seattle, Washington, United States, 1999. ACM, New York, USA.

[73] M. Nidd. Service discovery in deapspace. *IEEE Personal Communications*, 8(4):39–45, August 2001.

[74] J. R. Norris. *Markov Chains.* Cambridge University Press, 1998 edition.

[75] W. Peng and X.-C. Lu. On the reduction of broadcast redundancy in mobile ad hoc networks. In *MobiHoc '00: Proceedings of the 1st ACM international symposium on Mobile ad hoc networking & computing*, pages 129–130, Boston, Massachusetts, 2000. IEEE Press, Piscataway, NJ, USA.

[76] C. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing. In *WMCSA '99: Proceedings of Second IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, February 1999.

[77] J. Pitman. *Probability.* Springer-Verlag, New York US, 1993.

[78] S. Pleisch, M. Balakrishnan, K. Birman, and R. van Renesse. Mistral: efficient flooding in mobile ad-hoc networks. In *MobiHoc '06: Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, pages 1–12, Florence, Italy, May 2006. ACM, New York, USA.

[79] O. Ratsimor, D. Chakraborty, A. Joshi, and T. Finin. Allia: alliance-based service discovery for ad-hoc environments. In *WMC '02: Proceedings of the 2nd international workshop on Mobile commerce*, pages 1–9, New York, NY, USA, 2002. ACM press.

[80] N. Sadagopan, B. Krishnamachari, and A. Helmy. Active query forwarding in sensor networks. *Journal of Ad Hoc Networks*, 3(1):91–113, January 2005.

[81] Y. Sasson, D. Cavin, and A. Schiper. Probabilistic broadcast for flooding in wireless mobile ad hoc networks. In *WCNC '03: Proceedings of IEEE Wireless Communications and Networking Conference*, volume 2, pages 1124–1130. IEEE, March 2003.

[82] D. Scott and A. Yasinsac. Dynamic probabilistic retransmission in ad hoc networks. In *ICWN '04: Proceedings of the International Conference*

on Wireless Networks, pages 158–164, Las Vegas, Nevada, June 2004. CSREA Press.

[83] S. D. Servetto and G. Barrenechea. Constrained random walks on random graphs: routing algorithms for large scale wireless sensor networks. In *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 12–21, Atlanta, Georgia, USA, 2002. ACM, New York, USA.

[84] S. Shakkottai. Asymptotics of query strategies over a sensor network. In *INFOCOM '04: Proceedings of the Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 1, pages –557, March 2004.

[85] P. Sinha. QoS issues in ad-hoc networks. In P. Mohapatra and S. V. Krishnamurthy, editors, *Ad Hoc Networks: Technologies and Protocols*, chapter 8, pages 229–248. Springer, Boston, USA, 2005.

[86] H. Sivaraj and H. Sivaraj. Random walk based heuristic algorithms for distributed memory model checking. *Parallel and Distributed Model Checking (PDMC) (Satellite Workshop of CAV '03)*, 89(1):51–67, September 2003. Elsevier publishers.

[87] W. W. Terpstra, J. Kangasharju, C. Leng, and A. P. Buchmann. Bubblestorm: resilient, probabilistic, and exhaustive peer-to-peer search. *SIGCOMM Computer Communications Review*, 37(4):49–60, 2007.

[88] H. P. Thadakamalla, R. Albert, and S. R. Kumara. Search in weighted complex networks. *Physical Review E*, 72:066128, 14 pages, 2005.

[89] H. Tian, H. Shen, and T. Matsuzawa. Randomwalk routing for wireless sensor networks. In *PDCAT '05: Proceedings of the Sixth International Conference on Parallel and Distributed Computing Applications and Technologies*, pages 196–200. IEEE Computer Society, 2005.

[90] R. Tian, Y. Xiong, Q. Zhang, B. Li, B. Y. Zhao, and X. Li. Hybrid overlay structure based on random walks. In *IPTPS '05: Proceedings of the 4th International Workshop on Peer-to-Peer Systems*, pages 152–162, 2005.

[91] Y.-C. Tseng, S.-Y. Ni, , and E.-Y. Shih. Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network. In *ICDCS '01: Proceedings of the The 21st International Conference on Distributed Computing Systems*, pages 481–488, Phoenix, Arizona, April 2001. IEEE Computer Society, Washington, DC, USA.

[92] J. Tyan and Q. H. Mahmoud. A network layer based architecture for service discovery in mobile ad hoc networks. In *CCECE '04: Proceeding of Canadian Conference on Electrical and Computer Engineering*, volume 3, pages 1379–1384, May 2004.

[93] B. SIG. Specification of the bluetooth system, February 2003. `www.bluetooth.com/Bluetooth/Technology/Building/`.

[94] A. Varshavsky, B. Reid, and E. de Lara. A cross-layer approach to service discovery and selection in manets. In *MASS '05: Proceedings of the Second International Conference on Mobile Adhoc and Sensor Systems Conference*. IEEE Computer Society, November 2005.

[95] Z.Cheng and W.B.Heinzelman. Flooding strategy for target discovery in wireless networks. *Wireless Networks*, 11(5):607–618, September 2005.

[96] Q. Zhang and D. P. Agrawal. Dynamic probabilistic broadcasting in manets. *J. Parallel Distrib. Comput.*, 65(2):220–233, 2005.

[97] M. Zhong and K. Shen. Popularity-biased random walks for peer-to-peer search under the square-root principle. In *IPTPS '06: Proceedings of 5th International Workshop on Peer-to-Peer Systems*, 2006.

[98] F. Zhu, M. Mutka, and L. Ni. Splendor: A secure, private, and location-aware service discovery protocol supporting mobile services. pages 235–242. ACM Press, March 2003.

[99] F. Zhu, M. Mutka, and L. Ni. Service discovery in pervasive computing environments. *IEEE Pervasive Computing*, 4(4):81–90, October-December 2005.

[100] M. Zuniga, C. Avin, and B. Krishnamachari. Using heterogeneity to enhance random walk-baed queries. Technical Report CENG-2006-8, USC Computer Engineering, Netherlands, August 2006.