Traffic-Engineering Based Routing and Channel Allocation in Wired and Wireless Networks

by

Junaid Asim Khan

B.E., NED University of Engineering & Technology, Karachi, 1997
M.Sc., King Fahd University of Petroleum & Minerals, 2001

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

THE FACULTY OF GRADUATE STUDIES

(Electrical and Computer Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

September 2005

Abstract

Goal of traffic engineering (TE) in packet networks is to improve the network performance by providing support for congestion management, higher bandwidth utilization (or throughput), and QoS. There are two ways to provide congestion management, either by avoiding congestion before routing packet flows or by eliminating congestion after routing packet flows. Congestion can be eliminated in a network by capacity re-planning, however in wired networks it is not possible to perform capacity planning periodically. Therefore, wired networks rely on congestion avoidance that can be accomplished by using explicit path support in MPLS. This thesis proposes a fuzzy logic based TE routing algorithm to calculate these explicit paths. Simulation results have shown that proposed algorithm outperforms the well-known widest shortest path (WSP) algorithm and minimum interference routing algorithm (MIRA). The thesis also provides a TE solution in broadband fixed wireless networks with directed (or physical) mesh topologies. The solution approach exploits the fact that in wireless networks it is possible to perform capacity re-planning by re-planning the frequency channel allocation to links in a network. Unlike wired networks, wireless networks do not require any infrastructure upgrade to support channel reallocation in a short scale of time. The proposed solution is based on a distributed dynamic channel allocation algorithm that is capable of finding a solution at the time of network initialization and also dynamically fine tunes the channel allocation to eliminate congestion to provide traffic engineering. The proposed distributed dynamic channel allocation is highly scalable and hence is suitable for large networks. Simulation results have shown that channel allocation based on distributed dynamic channel allocation provides much better results than a fixed channel allocation based scheme.

Contents

List of Tables

List of Figures

Acknowledgements

All praise be to the God, for his limitless blessing and guidance. I acknowledge the support and facilities provided by The University of British Columbia, Vancouver, Canada.

All my family members, especially my father and wife, were constant source of motivation and support. Their love and care carried me through some difficult moments in my life. Their prayers, guidance and inspiration lead to this accomplishment.

I would like to express my profound gratitude and appreciation to my supervisor Dr. Hussein M. Alnuweiri, for his guidance, patience, and sincere advice throughout this thesis. I acknowledge him for his valuable time, constructive criticism, and stimulating discussions. Thanks are also due to my thesis committee members, Dr. Victor Leung and Dr. Vincent Wong for their comments and critical review on my thesis proposal.

I also acknowledge the financial support from NSERC, in the form of PGS-B award and from department of electrical and computer engineering in the form of teaching assistantship and NSERC top up grants.

I am also thankful to my fellow graduate students and all my friends on the campus especially Ayman, Amr, Tamer, Tariq, and Yasser for all their help.

Chapter 1

Introduction

1.1 Introduction

The high cost of network resources such as bandwidth, equipment, frequency spectrum combined with the competitive nature of providing commercial internet services have been pressing large network Service Providers (SPs) to look for ways to generate more revenue without substantially increasing their investment in upgrading infrastructure. One key to achieving such a goal is for the SPs' to provide "high value" IP-based services to customers, while simultaneously lowering their network costs by achieving maximal operational efficiency.

The problem with reaching such a goal dates back to the original design of the Internet. To maintain ubiquity and scalability, the internet relies on a best-effort mechanism for packet forwarding implemented by the Internet Protocol, or IP. As the Internet evolved, the concept of a converged IP network has emerged, and much research and industrial effort have gone into implementing communication services over IP. Beside data and web traffic, converged IP-based networks are required to carry realtime packet voice, video, time-sensitive financial transactions, virtual private networking, etc. Guaranteeing the quality of service is essential for migrating such delay/bandwidth sensitive applications to IP networks, and consequently for the viability and success of the Internet for commercial and business applications. As a result, current network development is focusing on Quality of Service (QoS) in packet networks, which aims to enable different types of information to be transported with different levels of service guarantees.

The problem of guaranteed QoS has proved to be complex despite the great deal of effort that has been dedicated to this subject. Without proper long-term network capacity planning and dynamic mechanisms for allocating and sharing network resources, providing QoS can be an expensive and unsuccessful exercise for network service providers. In recent years, it has become obvious that there is a need for a new paradigm that takes into consideration network wide topology and bandwidth resources to enable SPs to optimize their resource utilization, by evenly distributing and balancing the traffic load on the various links, while maintaining the promised levels of services to customer flows [1, 2, 3]. This is known as Traffic Engineering (TE).

1.2 Traffic Engineering

Internet Traffic Engineering is defined as that aspect of Internet network engineering dealing with the issues of performance evaluation and performance optimization of operational IP networks. Traffic Engineering encompasses the application of technology and scientific principles to the measurement, characterization, modeling, and control of Internet traffic. [4]

Traffic Engineering optimizes network performance at traffic and resource levels. At the traffic level, the job of TE is to provide QoS guarantees such as delay, delay jitter, packet loss etc. Resource level performance optimization includes efficient utilization of network resources to support maximum traffic throughput with required QoS. Resources of particular interest are link bandwidth, buffer space, and computational resources.

Transferring information from a source node to a destination node is the main function of an operational network. In packet networks this information is embedded inside IP packets. These packets then travel from source to destination. Thus, one of the most significant optimization areas is the routing of these packets. An improper routing decision may force one part of the network to share all the load and become congested while under-utilizing other network sections. Congestion increases transit delays, delay variation, packet loss, and reduces the predictability of network services. Clearly, congestion is a highly undesirable phenomenon. Congestion avoidance or/and elimination may optimize traffic and resource level performance objectives to provide better traffic engineering.

Congestion can be avoided by using efficient routing. One such solution is to use explicit paths for each traffic flow with constraint based routing. A traffic flow can be explicitly defined for each network user for every destination or several flows with the same ingress and egress network nodes can be combined into one aggregated flow to avoid complexity. Explicit path support for every flow makes it possible to use different routes for different flows even though they may have the same ingress/egress pair. Splitting

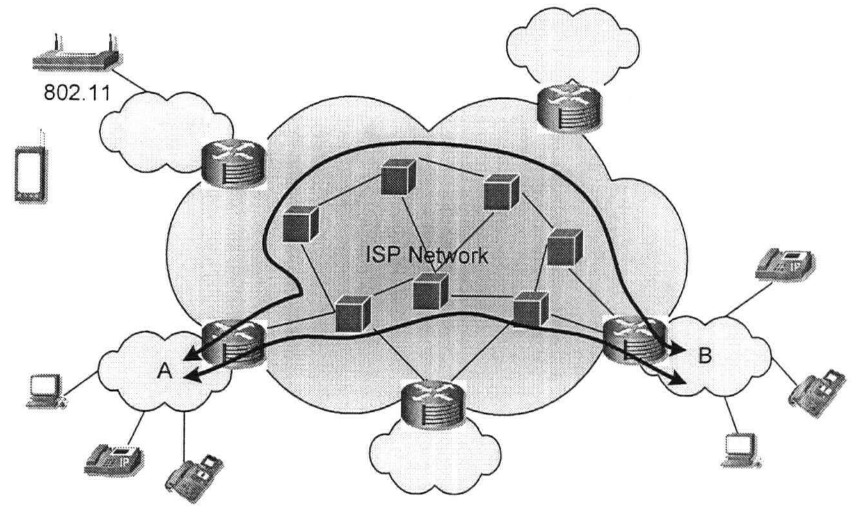


Figure 1.1: Explicit paths for same ingress/egress pair.

traffic using explicit paths avoids the congestion in one part of the network and distributes the traffic all over the network. To provide better optimization, explicit paths must be capable of selecting longer than shortest path to achieve better traffic load balancing. Figure 1.1 represents an example where two different explicit paths are used to route traffic for same ingress/egress pair to avoid congested links in shortest path. It is to be noted here that it is needed to create short term routes for individual users using explicit path support for applications such as Voice over IP (VoIP). In such a scenario these routes must be calculated using an online routing algorithm that uses readily available network load conditions.

On the other hand, congestion can be eliminated in two ways, either by increasing the capacity of the network or by restricting the traffic allowed to enter a congested section of the network. Restricting traffic reduces the network revenue and therefore is not desirable if it is possible to route the traffic in any other way. Furthermore, it is obvious in wired networks that increasing capacity at any time is not a feasible option because it needs infrastructure upgrading and network disruption. In wireless networks, capacity re-planning may not need infrastructure upgrading and can be achieved by reallocating parts of the frequency spectrum. Thus, congestion elimination by capacity re-planning is an option in wireless networks however it is not feasible for wired networks.

This thesis addresses both TE options i.e. to provide TE support by deploying congestion avoidance using efficient routing algorithm in wired networks; and congestion elimination using frequency channel reallocation in wireless networks.

1.3 Providing Explicit Path Support for Traffic Engineering

It is shown in Figure 1.1 that explicit path support can provide a way to choose multiple paths for same ingress/egress pair. This may make it possible to divert the traffic to provide better resource utilization and hence better TE. However, explicit path support to provide TE in a network has following requirements.

- A way to install and maintain explicit paths.
- A protocol to distribute current network information e.g. reserved or unused bandwidth on each link.

- An algorithm to calculate the explicit path based on network information to provide TE.

This section will cover the background on the first two requirements, the third requirement will be covered in detail in later chapters as one of the thesis objectives.

1.3.1 Multi Protocol Label Switching (MPLS)

Multi Protocol Label Switching (MPLS) is a framework that can be used to install and maintain explicit paths. In a connectionless network, a packet is analyzed at each intermediate router until it reaches its destination. Each router looks at the packet header and then makes routing decisions based on this packet header. Each router first finds the forwarding equivalent class (FEC) [1] and then this FEC is mapped to the next hop. Even in the best effort case, each router has to find a longest prefix match route to the next destination to determine its FEC, which takes more time than an exact prefix match. In an MPLS framework, assignment of a particular packet to a particular FEC is done only once and then all the intermediate decisions are made based on an exact match. An MPLS domain [5] is divided into two types of routers, (1) edge router, known as Edge Label Switch Router (ELSR) and (2) the core router known as LSR. At the time a packet arrives at an ingress ELSR, an extensive header search can be done to assign the packet to an FEC. All the packets belonging to a particular FEC follow the same path and are treated in the same manner. At an ingress ELSR, all packets belonging to a

[1] An FEC is a set of rules, based on which, forwarding decisions are made at the router.

particular FEC are assigned the same label, this label is used for routing decisions at the intermediate LSRs. At each intermediate LSR, an exact match is done based on the label of the incoming packet and then the packet is forwarded to the next hop after swapping the label with the next hop label at the intermediate LSR. At the egress ELSR, the exact match is done based on the incoming label and then that label is popped off before forwarding the packet to a non-MPLS domain. MPLS is a connection oriented network architecture, where for each FEC an explicit path is established before forwarding the packets. For each path, labels are distributed to the intermediate routers using a label distribution protocol. Since these paths use label switching when a packet has traveled via these paths, they are known as Label Switch Paths (LSP). Some of the advantages of MPLS given in [5] are as follows,

- Since MPLS uses explicit paths, some information that is not included in the packet header can be used in the forwarding decision, such as the input port of the edge LSR. This information can be vital to classify arriving packet into different flows.
- Packets entering the network from different ingress routers but with the same destination may have different LSPs which is not possible in connectionless networks.
- The rules that select the FEC can be more complicated because the FEC is determined only once at ELSR. This may be useful in per-flow services.
- Since MPLS uses explicit paths, it is easy to efficiently apply traffic engineering at the network using traffic engineering capable routing algorithms. The explicit

path support allows different paths (LSPs) for different flows even for the same ingress/egress pair, resulting in better traffic management.

The label switching mechanism in MPLS is used to maintain and use explicit paths in MPLS domain. However, to install/remove these explicit paths, MPLS uses Constraint Label Distribution Protocol (CR-LDP) [6] or Resource Reservation Protocol with TE extensions (RSVP-TE) [7]. CR-LDP and RSVP-TE distribute the MPLS labels on an explicit path and reserve the requested resources from ingress ELSR to egress ELSR. Furthermore, when needed, it is possible to remove these labels and release the reserved network resources such as bandwidth from the explicit path.

1.3.2 Open Shortest Path First with Traffic Engineering Extensions (OSPF-TE)

Designed to run internal to an autonomous system (AS), OSPF is classified as an interior gateway protocol (IGP) [8, 9]. The main function of OSPF is to distribute link state information across the network using reliable flooding so that each network element is able to make routing decisions based on a consistent view of the network. OSPF in its basic form only distributes connectivity information and static link costs, which can further be used by each router in the system by implementing Dijkstra's algorithm [10, 11] to calculate shortest path routes to each destination. However, to provide efficient traffic engineering using traffic engineered explicit paths, it is needed to distribute dynamic resource information such as bandwidth information on each link in the AS. An extension to OSPF, namely OSPF-TE serves this purpose [8, 12]. Along with connectivity information, OSPF-TE also distributes network wide information about *maximum bandwidth on each link*, *maximum reservable bandwidth on each link*, and *Unreserved bandwidth on each link*. OSPF-TE also distribute some other information in its Link State Advertisements (LSA) but the above three are of more concern in a traffic engineering framework. The maximum bandwidth on a link in OSPF-TE represents link capacity, whereas maximum reservable bandwidth is used to allow oversubscription on certain links, in such a case maximum reservable bandwidth can be greater than link capacities. Unreserved bandwidth on each link is the amount of bandwidth that is not yet reserved.

1.3.3 Explicit Path Calculation

The third requirement to provide explicit path support is to use an efficient routing algorithm that is able to avoid current and potential future congestion in the network. The algorithm must be able to make use of the information available from OSPF-TE to provide traffic engineered explicit paths. Furthermore, because of the online nature of certain traffic types, such as voice over IP where connection requirements arrive online, such algorithm must not be computationally expensive. Also to provide better network scalability and to avoid server bottlenecks, the algorithm must not rely on a centralized server and each ingress ELSR must be able to calculate its own path with the assumption that it has all the current network wide static and dynamic information available through OSPF-TE.

Developing such algorithms is the major objective of this thesis. Details of the explicit path routing problem, and our proposed solution will be presented in later chapters.

1.4 Thesis Contributions

The main purpose of this thesis is to provide traffic engineering in wired and wireless networks. However, because of the general nature of this problem, the work in this thesis is restricted to specific areas. More specifically this thesis contributes the following,

- Constraint based online routing algorithm to calculate explicit paths to provide TE. Our algorithm using fuzzy membership functions will be explained in Chapter 3.
- Providing traffic engineering in fixed broad band wireless mesh networks using distributed dynamic channel allocation. The particular network model used in this work and proposed solution will be explained in Chapters 4 and 5.

1.5 Thesis Organization

The rest of the thesis is organized as follows. Chapter 2 covers the literature review and problem formulation for TE based explicit path routing. A fuzzy logic based solution for this problem is proposed in Chapter 3. Problem formulation and literature review for traffic engineering in Broadband Fixed Wireless (BFW) systems is presented in Chapter 4. The problem is solved using proposed dynamic distributed channel allocation in Chapter 5. The thesis concludes with recommendations for future research work in Chapter 6.

1.6 Summary

This chapter has covered introduction to traffic engineering, and covered two solution scenarios for TE i.e. congestion avoidance in wired networks and congestion elimination in wireless networks. This chapter also covered a summary of MPLS and OSPF with TE extensions to support explicit paths which are needed to provide TE in a network. The last sections of the chapter have covered thesis objectives and thesis organization.

Chapter 2

TE-Based Routing in Wired Networks

2.1 Introduction

In today's networks one of the main goals of a service provider is to satisfy their customer demands while using network resources efficiently. Currently, the prevalent use of topology-driven IP routing protocols with shortest-path computations is causing serious imbalance of packet traffic distribution when least-cost paths converge on the same set of links leading to unacceptable delays or packet loss even when feasible paths over less utilized links are available. However, recently proposed enhancements to common routing protocols are promising to overcome such shortcomings by providing the means to distribute link-state information that is more pertinent to traffic engineering (TE) in routed networks [8, 12]. Also, emerging TE routing techniques are promising to enable network service providers to optimize their resource utilization by evenly distributing or balancing the traffic load on their network links while maintaining the promised levels of service to their customer traffic [13]. Load balancing is a process that attempts to improve network utilization by choosing lightly loaded links for routing new path requests so that the maximum and average link utilization is minimized. When successful this process can eliminate, or at least delay, link congestion until network utilization approaches saturation. However, load balancing may result in selecting paths that are not necessarily the shortest between peers, potentially leading to increased delays and using more network nodes and resources.

Previous studies [13] have indicated that load-balancing algorithms perform better than shortest path algorithms under low network load conditions. At higher loads, shortest path algorithms normally outperform load-balancing algorithms. It is worth noting that most of the current TE routing methods provide either load balancing in the network or reduced path-request blocking, but not both [1, 14, 15].

We will start this chapter by providing literature review of constraint based routing algorithms, then formulate the problem that will be solved in the next chapter.

2.2 Previous Related Work

Computing the best paths on a graph under two or more constraints (or objectives) is a well known NP-hard problem. For this reason, several heuristic algorithms with polynomial time complexity have been proposed in the past. Below we discuss a few of the most common approaches to solve this problem while highlighting their pros and cons.

2.2.1 Constraint Shortest Path (CSPF)

A widely used routing algorithm for guaranteed-bandwidth connection requests is the Constrained-based Shortest Path First (CSPF) algorithm. CSPF works by first removing all links that have less available bandwidth than that required by the path request, then computing the shortest path in the residual graph. CSPF is guaranteed to find a guaranteed bandwidth path, if one exists, but does not provide any load balancing because it keeps selecting the same shortest path until it saturates. This will lead to blocking future route requests through links in this path because of saturated links on the path. If the same path was not chosen until some links saturated then these future route requests could be routed.

2.2.2 Widest Shortest Path (WSP) Algorithm

The WSP [14] method is an enhancement of CSPF that works as follows. WSP maintains a list of multiple shortest paths then selects the widest path among them for routing the pending request. The widest path is defined as the path that provides the highest residual bandwidth on its bottleneck link(s). Even with this improvement, WSP will still not be able to select highly underutilized paths which are slightly longer than the shortest path, thus resulting in an unbalanced network. Moreover, WSP will keep selecting the same shortest paths until some links on these paths are saturated. At this point many future path requests will get rejected, which could have been avoided if earlier requests had chosen slightly longer paths over underutilized links. The main advantage of WSP is its low computational complexity, which is the same as the complexity of Dijkstra's Algorithm. This makes WSP suitable for online routing.

2.2.3 Minimum Interference Routing Algorithm (MIRA)

MIRA's path selection is based on information about the location of the ingress/egress router pairs involved in the routing process [1]. For each connection request, MIRA finds the route that provides the least interference with paths for other ingress/egress router pairs. MIRA does this by finding critical links for each potential path for a router pair. If a path is routed through a critical (bottleneck) link then it reduces the maximum possible bandwidth flow (max-flow) between corresponding ingress/egress pairs. MIRA assigns higher weights to the links that are critical for more ingress/egress pairs. It was shown in [1], that MIRA provides a good solution only when the number of potential ingress/egress router pairs is reasonably small, but its performance degrades quickly when the number of such router pairs is large relative to the network size (number of routers). This is because in this situation, most of the links are critical and it is difficult for the routing algorithm to differentiate between them in the path selection process. The second major problem with the MIRA approach is its high time complexity, which may render it impractical for online routing. As reported in [1], the time complexity of MIRA is $O(p \cdot n^2 \cdot \sqrt{e})$, where p is the number of ingress/egress router pairs, n is the number of routers, and e is the number of links (edges) in the network. Since it is quite possible, and perhaps desirable, to have most of the network routers configured as "edge" routers, the number of ingress/egress router pairs in this case will be $O(n^2)$. This means that $p = e = n^2$ (for a full mesh topology), which results in a worst case time complexity of $O(n^5)$. By contrast WSP requires $O(n \cdot log(n) + e)$ time. The third significant drawback of MIRA is its need for a centralized route server to keep track of the ingress/egress information because it is not distributed by a standard routing protocol such as OSPF or OSPF-TE [8]. This results in the single point of failure and communication overhead between the route server and edge routers. Another problem of MIRA in its original form is that it does not differentiate between ingress/egress pairs having different max-flow values.

2.2.4 Enhanced MIRA

With almost all the disadvantages of MIRA, enhanced MIRA [16] modifies the basic MIRA by differentiating between ingress/egress pairs having different max-flow values. The new approach generalizes the concept of critical links to Δ-critical links defined as those links which become critical when additional Δ units of bandwidth are to be routed in a flow residual graph[1]. However, finding Δ-critical links has proved to be an NP-hard problem in the same paper. Therefore, a heuristic algorithm is used that cannot guarantee providing the optimum set of Δ-critical links. The problem of differentiating between ingress/egress pairs having different max-flow values, is solved by using lexicographic ordering. In this approach, first the ingress/egress pairs are sorted in ascending order of

[1]Flow residual graph for an ingress/egress pair is a graph, which is obtained after routing the bandwidth (over one or more paths) equals to the max-flow value between the ingress/egress pair.

their max-flow value. Then before calculating the cost for each link, the weight of each ingress/egress pair is calculated as follows,

$$\alpha_i = \quad 1 \qquad\qquad if\ i = p \tag{2.1}$$

$$mD(1 + mD)^{p-i-1} \qquad if\ i = (p-1), ...1 \tag{2.2}$$

where m is the number of links, p is the number of potential ingress/egress pairs, D is the bandwidth demand (in bits per second), and i is the position of ingress/egress pair after sorting. This approach gives more weight to the ingress/egress pairs having less max-flow in such a way that the pair with the smallest max-flow will have a weight larger than the sum of all other weights. However, this may not be a good idea, because it is possible that the critical-link for one ingress/egress pair has a higher cost than any of the links that are critical for more than one pair ingress/egress pairs. This may result in congestion of more ingress/egress pairs.

With almost all the disadvantages of MIRA, enhanced MIRA is also not the best choice for online explicit path routing for the same reasons which were mentioned in the last section.

2.2.5 Profile Based Routing

In order to reduce the time complexity of MIRA while simultaneously providing the admission control, a profile based routing algorithm was proposed in [2, 3]. This approach assumes that expected traffic demand for each ingress/egress pair is known in advance, either from the Service Level Agreement (SLA) or from previous traffic data measurements. The algorithm distributes these traffic profiles into classes and the bandwidth allocation problem is solved as a multi commodity problem[2]. The output of this phase is the allocation of a portion of bandwidth in each link to each class. This computation is done off line, because it uses extensive computations. In the online routing phase of the algorithm, a shortest-path like algorithm is applied with a constraint that flow from each class can use only its allocated bandwidth in each link. If no such path is available, then the path request is rejected.

This algorithm provides a good alternative to MIRA but still has several drawbacks. For example, it is possible that bandwidth allocation to few demanding connection, may result in rejecting many other flows, which may be routed otherwise. Also it is possible that, at a certain time, a particular ingress/egress pair has a large demand (deviating from its profile) whereas all the others have a very small demand. As such, profile based routing does not support this situation. Furthermore, a link failure requires that the multi commodity problem be solved again.

2.3 Problem Formulation

We model the service provider's network as a directed Graph $G(N, L)$, where N is the set of routers (or nodes) and L is the set of links (or edges) in the network graph. For each link j in the set L, we define the parameter c_j to represent link capacity and r_j to

[2]A maximum-flow problem involving multiple commodities, in which each commodity has an associated demand and source-sink pairs.

represent the link's residual bandwidth. Each connection request is represented by the triplet (s, d, b), representing the source, destination and requested bandwidth.

It is worth mentioning that other QoS requirements, such as delay, can be incorporated in this model through a straight forward mapping to the bandwidth requirement. For example, the delay requirement can be mapped into a bandwidth requirement by deploying scheduling techniques such as Weighted Fair Queuing (WFQ) [17] in the network nodes (routers). WFQ can provide a delay guarantee to each flow based entirely on a bandwidth reservation parameter with a token-bucket shaping function applied to the traffic source. Therefore, in this thesis we will consider satisfying only bandwidth requests knowing that delay guarantees can be provided by deploying the appropriate packet schedulers and traffic shapers in the network. It is important to note, however, that routing packets over less congested paths enables packet schedulers to provide better QoS guarantees, e.g. less delay, hence the need for effective load balancing. We also assume that a TE-capable link-state routing protocol, such as OSPF-TE, is operational in the network [8]. Such a protocol floods timely link-state information, such as residual link bandwidth (or maximum reservable bandwidth in OSPF-TE terminology), to all network routers for use by TE-based routing algorithms.

We will be concerned mainly with an online TE routing mechanism that admits ingress to egress path requests based on the current network state. We will assume that connection requests arrive one by one and there is no a priori knowledge about future requests. These assumptions are considered reasonable and have been used in prior studies [1, 2, 3, 16], although they may be more applicable to on-demand connection requests such as voice-over-IP tunnels.

The objective of the routing problem is to maximize the acceptance ratio of incoming path requests (s, d, b) such that the network will remain load-balanced after routing each accepted request while providing better quality routes. The need for maximizing the number of accepted requests is economically driven. The underlying premise is that a load-balanced network maintains an even load on all network resources, which reduces queuing delays [13], and results in fewer customers being affected in case of a link failure or a sudden surge in demand.

2.3.1 An NP-hard problem

If all the future demands are known a priori (off-line routing), then the TE routing problem can be mapped to the bandwidth packing problem [18], whose objective is to route the maximum number of already known demands while satisfying bandwidth constraints for all demands. The bandwidth packing problem is a well known NP-complete problem. If these demands are not known a priori (as it is the case in online routing) then the problem becomes much more difficult to solve. With the fact that by solving online TE routing problem, the bandwidth packing problem can be solved but not vice versa, it can be concluded that the online TE routing is NP-hard.

2.4 Summary

This chapter, in the first section, has covered an introduction and motivation for online TE routing problem. This is followed by a literature review in the next section, where CSPF, WSP, MIRA, enhanced MIRA, and profile based routing algorithms are covered with their pros and cons. The third and last sections of this chapter has covered the problem formulation and NP-hard nature of the problem.

Chapter 3

TE-Based Fuzzy Routing Algorithm

3.1 Introduction

This chapter proposes a fuzzy-logic based algorithm for routing under traffic engineering constraints. The proposed Fuzzy Routing Algorithm (FRA) modifies the well-known Dijkstra's shortest path algorithm [10], by using fuzzy-logic membership functions in the path-cost update (node relaxation) process. FRA is a low complexity on-line routing algorithm that computes the "best" paths under multiple TE optimization objectives. In this chapter, we consider route optimization subject to multiple concurrent objectives related to balanced end-to-end path utilization, link utilization, and number of hops, and subject to bandwidth constraint. The details are presented in Section 3.2.

The results reported at the end of the chapter show that FRA achieves load balancing at higher loads without increasing the path-request blocking probability. The results also show that the routes discovered by FRA have much better quality-of-service attributes (due to lower congestion) than competitor algorithms. These results are presented in

[0]A version of this chapter has been published. Khan, J.A. and Alnuweiri, H.M. (2004) A fuzzy constraint-based routing algorithm for traffic engineering. Globecom. 3:1366-1372.

Section 3.3.

It is worth emphasizing that our work specifically targets flow or path-oriented networking based on Multi Protocol Label Switching (MPLS) [19] or its generalized version (G-MPLS). MPLS provides traffic-engineering solution by supporting explicit path establishment in packet networks. These explicit paths, also known as label switched paths or LSPs, are calculated at the ingress node (edge router) using a TE database. A label distribution protocol, like RSVP-TE or CR-LDP is then used to setup the paths [20]. All the packets belonging to the same connection are then routed through the same path. This is achieved by using a label-switching mechanism in MPLS. With explicit path support in MPLS, it is possible to choose a different route for each connection request. This helps in balancing the utilization of network resources. However, selecting explicit paths for TE has been proven to be NP hard [1, 21], but several heuristic solutions have been proposed in literature for routing guaranteed-bandwidth LSPs [1, 14, 21].

3.2 Fuzzy Constraint-Based Routing

The proposed FR algorithm employs a simple, but very effective, multi-constraint load balancing technique to reduce call blocking as well as network delays for QoS routing. In TE-based routing there is a fundamental trade off between traffic load-balancing and minimizing path length or number of hops. While shortest path routing can lead to congestion and route blocking, load balancing tends to use more network links than necessary to spread out and reduce the traffic load on network links. In its current formulation, FRA incorporates multiple route optimization objectives including number of hops, maximum link utilization, and the utilization of links other than the bottleneck link. Unlike WSP, which performs load balancing for a path request based on the path's bottleneck link only, FRA load-balancing can be based on all links on the path. These additional path optimization objectives provide FRA with a significant performance advantage over other routing algorithms as will be demonstrated by results reported in the next sections.

3.2.1 Performance Objectives

In reservation-based online routing, there is only one constraint, which is the amount of requested bandwidth. Achieving better load balancing, however, requires satisfying several resource oriented objectives. We define the following three resource-oriented performance objectives for routing a new request with load balancing:

Objective 1: Maximize path bandwidth, i.e. route the request while maximizing the residual bandwidth on the bottleneck link(s), defined as the link(s) with the least residual bandwidth on the path.

Objective 2: Maximize bandwidth on links other than bottleneck link. When there is more than one path with the same bottleneck bandwidth, then the path with higher residual bandwidth on links other than the bottleneck link is the better path.

Objective 3: Minimize the number of hops. This objective is needed because a path which is slightly better in terms of the first two objectives but with a larger number

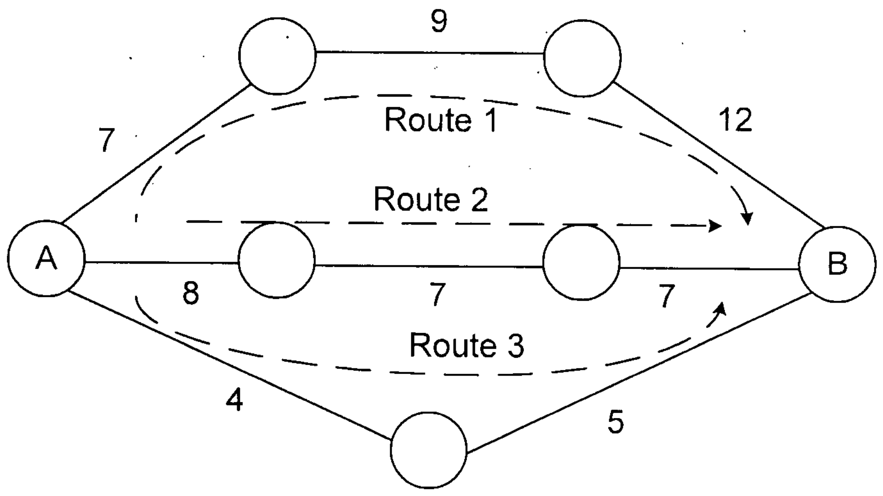


Figure 3.1: Example network with multiple paths. The link labels indicate residual bandwidth.

of hops is more likely to create interference with other path requests on one of the links.

The application of the above objectives is illustrated in Figure 3.1 which shows a simple network with three paths between an ingress node A and an egress node B. The graph also shows the residual bandwidth on each link. Given a path request from A to B with a bandwidth demand of 4 units or less, the following is true according to the above objectives:

- The bottleneck link bandwidth on route 1 and route 2 is 7 units, and on route 3 is 4 units.
- Routes 1 and 2 are equal (and better than route 3) according to objective 1, since both routes result in maximizing the residual bandwidth after routing the request.

- Route 1 is better than route 2 in terms of objective 2, because it provides more residual bandwidth (for future requests) on links other than the bottleneck link.
- Route 3 minimizes the number of hops, thus providing the best route according to objective 3, but results in the least amount of residual link bandwidth after routing the request.

Solving a multi-objective routing problem with more than one cost metric is NP-hard problem [22]. The problem is further complicated by conflicting optimization objectives. Fuzzy logic is a particularly useful vehicle for solving hard optimization problems with potentially conflicting objectives. Fuzzy optimization allows mapping the values of different criteria into linguistic values that characterize the level of satisfaction with the numerical value of the objectives. The numerical values are chosen typically to operate in the interval $[0, 1]$ according to the membership function of each objective [23].

The FRA method starts by satisfying the main constraint in the path request, namely, the requested bandwidth b between source router s and destination router d. This can be done by removing all links j having residual bandwidth $r_j < b$ from the network graph $G(N, L)$. The next step of the algorithm computes the path feasibility or membership according to a well-defined fuzzy criterion. FRA uses this information to route each path request while maintaining a load balanced network. The following fuzzy-logic rule defines a node reachability criterion.

Rule R1: **IF** *A path to node y through node x has low bandwidth utilization on bottleneck link* **AND** *path to y through x has low bandwidth utilization on links other* *than bottleneck link* **AND** *path to y through x has less number of hops* **THEN** *the node y is highly reachable through x.*

FRA uses this rule as follows: when routing a path-request, the path through which the destination router d is most reachable will be selected as the "best" route.

To understand the above rule, we need to present some standard fuzzy logic background. A fuzzy logic rule is an **If-Then** rule. The **If** part (antecedent) is a fuzzy predicate defined in terms of linguistic values and fuzzy operators **Intersection** and **Union**. The **Then** part is called the consequent. In our case, the linguistic value used in the consequent part identifies the fuzzy subset of reachable nodes through node x. There are many implementations of fuzzy union and fuzzy intersection operators. Fuzzy union operators are known as **s-norm** operators while fuzzy intersection operators are known as **t-norm**. Generally, the **s-norm** and **t-norm** operators are implemented using the *max* and *min* functions, i.e.

$$\mu_{A\cap B} = min(\mu_A, \mu_B) \tag{3.1}$$

$$\mu_{A\cup B} = max(\mu_A, \mu_B) \tag{3.2}$$

where μ_A and μ_B are the membership functions of fuzzy sets A and B respectively. This is known as the $min - max$ logic initially introduced by Zadeh [24]. According to the $min - max$ logic the rule above evaluates to the following:

$$test_y = min(p_{xy}, l_{xy}, h_{xy}) \tag{3.3}$$

$$M_y = max(M_y, test_y) \tag{3.4}$$

The above equations can be adapted to TE routing by using M_y to represent the membership of node y in the fuzzy set of reachable nodes, $test_y$ to represent a test value computed for the path through node x. If node y already has a better reachability then it will retain its previous path and membership, otherwise it accepts the path through x and changes its membership value by the amount $test_y$. The function p_{xy} represents the membership of a path through node x to node y in the fuzzy set of low-utilization paths. Also, l_{xy} and h_{xy} represent, respectively, the membership values of a path from s to y through x in the fuzzy sets of low-utilization links and smaller hop-count paths.

Some formulations of multi-criteria decision functions may not desire pure "ANDing" of **t-norms** nor the pure "ORing" of **s-norms**, mainly because of the complete lack of compensation of the **t-norm** for any partial fulfillment of all criteria and the complete submission of the **s-norm** to the fulfillment of any criteria. For this purpose, the Ordered Weighted Averaging (OWA) operator [25] was developed. This operator falls in the category of compensatory fuzzy operators and allows for the adjustment of the degree of ANDing and ORing. According to [25], the "OR-like" and "AND-like" OWA for two fuzzy sets A and B are implemented as follows.

$$\mu_{A \cap B} = \beta \times min(\mu_A, \mu_B) + (1-\beta) \times \frac{1}{2}(\mu_A + \mu_B) \tag{3.5}$$

$$\mu_{A \cup B} = \beta \times max(\mu_A, \mu_B) + (1-\beta) \times \frac{1}{2}(\mu_A + \mu_B) \tag{3.6}$$

Where β is a constant parameter in the range $[0, 1]$ that represents the degree to which OWA operator resembles the pure OR, or pure AND. In our work we will set $\beta = 0.8$.

With the AND fuzzy operator implemented as OWA operator, rule **R1** can be expressed as follows

$$test_y = \beta \times min(p_{xy}, l_{xy}, h_{xy}) + (1-\beta) \times \frac{1}{3}(p_{xy} + l_{xy} + h_{xy}) \quad (3.7)$$

$$M_y = max(M_y, test_y) \quad (3.8)$$

It should be noted that the equation for M_y is the same as before. The reason is that this equation does not represent a fuzzy rule. It is used to select the path that provides better reachability than other candidate paths.

3.2.2 Membership Functions

FRA uses membership functions to compute node reachability. We define three membership functions for FRA corresponding to the three performance objectives listed in previous section. For objective 1, we define the function p_{xy} as the membership value of a path in the fuzzy set of feasible paths with respect to the bottleneck link. A path has the highest value, $p_{xy} = 1$, when its bottleneck link bandwidth is equal to the maximum residual link bandwidth (BW_{max}) in the network graph. A path with bottleneck bandwidth equal to the minimum residual link bandwidth (BW_{min}) in the graph, is still a candidate path and is assigned a value $1 > p_{xy} > 0$. In this paper, we define p_{xy} by a simple linear function that maps the bandwidth range $[BW_{min}, BW_{max}]$ to the range $[0.25, 1]$, as shown in Figure 3.2. By definition, a path with value $p_{xy} < 0.25$ is not a feasible path because it does not have sufficient bandwidth. The minimum value of $p_{xy}(= 0.25)$ was chosen based on some performance heuristics, but other values can be

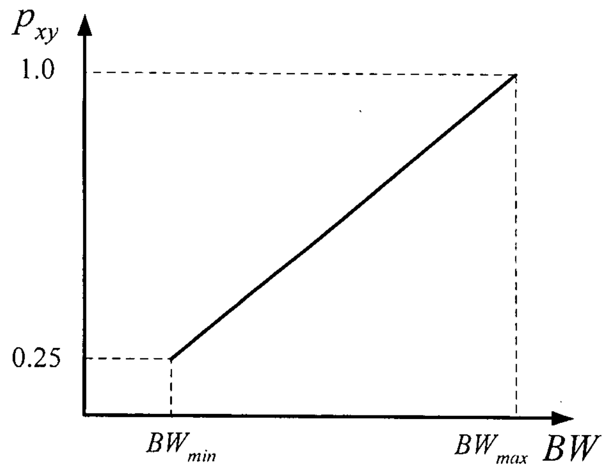


Figure 3.2: p_{xy} in terms of path bandwidth. BW_{min} and BW_{max} are the minimum and maximum residual bandwidths in the graph. BW is the path bandwidth.

used within the above range.

For objective 2, we define l_{xy} as the membership value of a path in the fuzzy set of feasible paths based on all path links, including the bottleneck link. The derivation of a suitable function for l_{xy} is not a trivial task since it requires considering bandwidth on almost all the links on the path. We define the function l_{xy} by the following equation

$$l_{xy} = max(1 - \sum_{j \in L_{xy}} S_j, 0) \tag{3.9}$$

where S_j is a positive real value for link j and L_{xy} is the set of links in the candidate path to destination y through node x.

Essentially, l_{xy} is a piece-wise linear function whose slope depends on both the path length (number of hops) and the residual bandwidth range on all path links including the bottleneck link. To understand the role of this function, we start by explaining two boundary conditions for l_{xy}. In the first condition, all the links in the candidate path have equal residual bandwidth $r_j = BW_{max}$, in which case $l_{xy} = 1$. In the second boundary condition, all links in the path with H_{min} hops (minimum number of hops in the bandwidth residual graph to reach the destination) have equal residual bandwidth $r_j = BW_{min}$, for which $l_{xy} = \delta$, where δ is a very small value larger than 0. Also , $l_{xy} = 0$ for a path with $H_{min} + 1$ hops or more (longer than H_{min}) and with all links having $r_j = BW_{min}$. This condition imposes the rule of allowing only paths with minimum number of hops (H_{min}) to be selected when all the links have BW_{min} available for routing the request.

First Boundary condition (for lower bound S_l): This is the case when all the links in the path have equal residual bandwidth $r_j = BW_{max}$, which means that all links have the same value of S_j, and $l_{xy} = 1$. Putting these values of S_j and l_{xy} in Equation 3.9, we get,

$$\begin{aligned} 1.0 &= max(1 - \sum_{\in L_{xy}} S_l, 0) \\ \Rightarrow S_l &= 0 \end{aligned} \qquad (3.10)$$

Second Boundary condition (for upper bound S_u): This is the case when all the links in the path have equal residual bandwidth $r_j = BW_{min}$. In this case, we set $l_{xy} = 0$ for a path of $length = H_{min} + 1$ hops. Again, using Equation 3.9, we get

$$\begin{aligned}
& 0 = max(1 - \sum_{j \in L_{xy}} S_u, 0) \\
\Rightarrow\ & 0 \geq 1 - \sum_{j \in L_{xy}} S_u \\
\Rightarrow\ & 0 \geq 1 - (H_{min} + 1) \times S_u, \quad and \\
\Rightarrow\ & S_u \geq \frac{1}{H_{min} + 1}
\end{aligned} \tag{3.11}$$

Also a path with path *length* $= H_{min}$ under this scenario is still a candidate path and must have $l_{xy} > 0$. Using these values in Equation 3.9 we get.

$$\begin{aligned}
& l_{xy} = max(1 - \sum_{j \in L_{xy}} S_u, 0) > 0 \\
\Rightarrow\ & 1 - \sum_{j \in L_{xy}} S_u > 0 \\
\Rightarrow\ & 1 - H_{min} \times S_u > 0,
\end{aligned}$$

Since all links have same value S_u and the path has H_{min} hops

$$\Rightarrow\ S_u < \frac{1}{H_{min}} \tag{3.12}$$

Combining Equations 3.11 and 3.12 we get

$$\frac{1}{H_{min} + 1} \leq S_u < \frac{1}{H_{min}}$$

The value S_u that can satisfy this condition is

$$S_u = \frac{1}{H_{min} + 1} \tag{3.13}$$

Based on S_u and S_l and using a linear equation, we calculate the value S_j for link j,

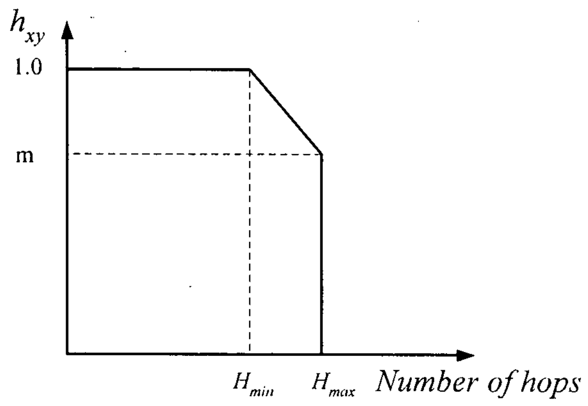


Figure 3.3: Membership function for number of hops.

having residual bandwidth r_j as follows,

$$S_j = S_u - \frac{r_j - BW_{min}}{BW_{max} - BW_{min}} \times (S_u - S_l) \qquad \text{if } BW_{max} \neq BW_{min}$$
$$S_j = S_u \qquad \text{otherwise}$$

Putting $S_l = 0$, we can also write the equation as follows.

$$S_j = S_u - \frac{r_j - BW_{min}}{BW_{max} - BW_{min}} \times S_u \qquad \text{if } BW_{max} \neq BW_{min} \tag{3.14}$$
$$S_j = S_u \qquad \text{otherwise} \tag{3.15}$$

Finally, to meet objective 3, the function h_{xy} is defined as the decision variable for the case when a new path $P1$ is slightly better than other candidate paths, i.e. it offers small advantage in terms of l_{xy} and p_{xy}, but uses a larger number of hops. In this case, h_{xy} can be used to force the algorithm to choose the path with the smaller number of hops based on some performance threshold. A simple fuzzy membership function for h_{xy} is shown in Figure 3.3. By choosing the value of m we can increase or decrease the importance of this objective in the optimization process. In our work we have chosen $m = 0.75$.

Once we are able to calculate the membership of a node, through a particular path, in the fuzzy set of reachable nodes we can modify the Dijkstra's algorithm to use this membership value. The FRA algorithm is presented in Figure 3.4.

Like WSP, FRA inherits the breadth-first search feature of Dijkstra's algorithm but modifies the path update process by using fuzzy logic membership values in place of the simple additive link costs. However, fuzzy logic membership proves to be a simple but powerful tool for routing under multiple performance objectives (load balancing, link utilization, and number of hops). As will be demonstrated in the next section, FRA outperforms WSP [14] and even the more complex MIRA [1] algorithms both in terms of maintaining low link utilization and low rejection rate for path requests.

3.2.3 Algorithm Analysis

Computational Complexity

For a network graph with n nodes and e edges (links), the outer loop of the algorithm runs once for each node, thus requiring $O(n)$ iterations, with each iteration performing an operation to extract the node with the maximum membership value. In the inner loop, each of the e edge is traversed once and only once. Balanced heap data structure enables such operation to be performed in $O(log\ n)$ time, resulting in $O(n \cdot log\ n + e)$ complexity for the entire algorithm.

```
ALGORITHM $FRA(G, R, C, ingress, egress, b)$
NOTATION
$G = G(N, L)$ = Input Graph.          $R$ = Set of link residual bandwidth $r_i$.
$C$ = Set of link capacities $c_i$.   $ingress$ = Ingress node.
$egress$ = Egress node.               $b$ = Bandwidth demand.
$Path_y$ = Set of nodes in the path from $ingress$ to node $y$.
Begin
  1. Remove all links which does not satisfy bandwidth constraint "$b$" from $G$.
  2. Run Dijkstra's Algorithm to calculate $H_{min}$ for each node.
  3. $P = \{\}$ , $Path_y = \{\}$ $\forall y$, $M^r_{ingress} = 1$, and $M^r_i = 0$ $\forall i \neq ingress$.
  Loop
    4. Find $x \notin P$ such that $M^r_x$ is maximum $\forall x \notin P$;
    5. $P = P \cup \{x\}$. If $P$ contains $egress$ then exit loop;
    6. Loop $\forall y \notin P$ having a link $xy$
      Update
        $test_y = \beta \times min(p_{xy}, l_{xy}, h_{xy}) + (1 - \beta) \times \frac{1}{3}(p_{xy} + l_{xy} + h_{xy})$;
      If $test_y > M_y$ then
        $Path_y = Path_x \cup \{x\}$;
        $M_y = max(M_y, test_y)$;
      End If
    End Loop
  End Loop
  Return $Path_{egress}$
END $FRA$
```

Figure 3.4: Structure of Fuzzy Routing Algorithm (FRA).

Correctness

To prove correctness, it is sufficient to show that the use of fuzzy-logic membership functions in FRA does not introduce any cycle, i.e. it does not result in a path that visits a node more than once. The proof is based on simple induction. In each iteration, the path to a node x is chosen when it is included in the set P and is reached through a node in P. This node x can be revisited (which creates a cycle) only if x is reached through some other node in $N - P$ with a larger membership value. However, in the current iteration, all the nodes in $N - P$ have membership values less than the membership of node x and also less than the membership of any other node in P. Further more, Equation 3.8 guarantees that membership value of a node z is always less than the membership value of a node y, if it is reached through y. Since all the nodes in $N - P$ can only be reached through a node in P, therefore they cannot have a higher membership value than any node in P. Hence no node in P will ever be revisited to create a cycle.

3.2.4 Comparison with WSP and MIRA

Table 3.1 presents a comparison among FRA, MIRA and WSP algorithms (reviewed in Section 2.2). The table shows that FRA has a much lower time complexity than MIRA, and lower path request rejection rate than WSP. Unlike MIRA, FRA does not need a route server. Moreover, FRA does not need ingress/egress information and only needs the residual link bandwidth information which can be easily obtained from TE routing protocols such as OSPF-TE. FRA performs load balancing based on all links, MIRA does not provide good load balancing, whereas WSP provides load balancing based on bottleneck link only.

Table 3.2 provides a practical comparison of the algorithms in terms of the time required to compute a new route for the two network graphs used in this paper (Figures 3.5 and 3.16). In Figure 3.5, $n = 15$, $p = 90$, and $e = 54$, in Figure 3.16, $n = 20$, $p = 380$, and $e = 92$. It should be emphasized that this comparison is based on computational complexity of FRA, MIR, and WSP. The comparison of Table 3.2 uses the parameter T defined as the time it takes to compute a route in WSP for the given network graph.

We will show in the next section that path request rejection rate of FRA is much lower than WSP rejection rate and is lower or the same as MIRA's rejection rate. Also that the quality of FRA routes (route QoS) is better than the other techniques because of it maintains end-to-end paths with less congested links. The claims stated in Table 3.1 are all substantiated by the performance results reported in the next section.

3.3 FRA Performance

We have preformed extensive simulations to test the performance of the proposed algorithm. Our simulator is capable of modeling different routing algorithms for different input graphs and different flow request distributions. We have generated several random sets of flow requests (random ingress/egress path requests) each having a random bandwidth demand uniformly distributed between 1 and 4 units to compare FRA against WSP and MIRA. This type of bandwidth demand was also chosen in [1]. We have chosen

Table 3.1: FRA comparison with WSP and MIRA.

Comparison Criteria	**FRA**	**WSP**	**MIRA**
Computational complexity	$O(n.log\ n + e)$	$O(n.log\ n + e)$	$O(p.n^2.\sqrt{e})$ worst case is $O(n^5)$
Route server	not needed	not needed	required
Ingress/egress router information	not needed	not needed	required
TE routing protocol (e.g. OSPF-TE)	required	required	required
Load Balancing based on bottleneck link	High	High	Low
Load balancing based on all path links	High	Low	Low
Path request rejection rate	Low	High	Low
Route QoS	High	Medium	Low

Table 3.2: FRA comparison with WSP and MIRA.

Comparison Criteria	**FRA**	**WSP**	**MIRA**
Average time to compute new route in Network of Figure 3.5	$2T_1$	T_1	$1300T_1$
Average time to compute new route in Network of Figure 3.16	$2T_2$	T_2	$8000T_2$

the same demand distribution to provide comparison of FRA with MIRA based on the traffic trend used in [1].

Furthermore, most of the previously reported algorithms [1, 21] were tested for path requests routed between a small set of ingress/egress router pairs. Such restricted scenario is not representative of deployed ISP networks where a considerable portion of the routers are edge routers. In our simulations, we have modeled networks with a large number of edge routers then generated path requests between all possible pairs of edge routers.

3.3.1 Network Topologies

We have used two network topologies to test the performance of the proposed FRA. For the comparison purpose, the first network graph used in the simulation is the same as the one used for MIRA in [1]. This graph is shown in Figure 3.5 , where we have chosen 10 edge routers (nodes 0, 1, 4, 8, 7, 10, 11, 12, 13, and 14 in the graph) to generate $10 \times (10 - 1) = 90$ ingress/egress router pairs. To check the performance consistency, a larger network topology graph of Figure 3.16 is also used in the simulation. It is to be noted here that both graphs are good representers of actual network topologies with following characteristics.

- Large number of customer-facing (edge) routers as compared to the number of core routers. It is needed because only customer-facing routers are the revenue generating routers.

- Links connected to core routers usually have higher bandwidth capacities than links

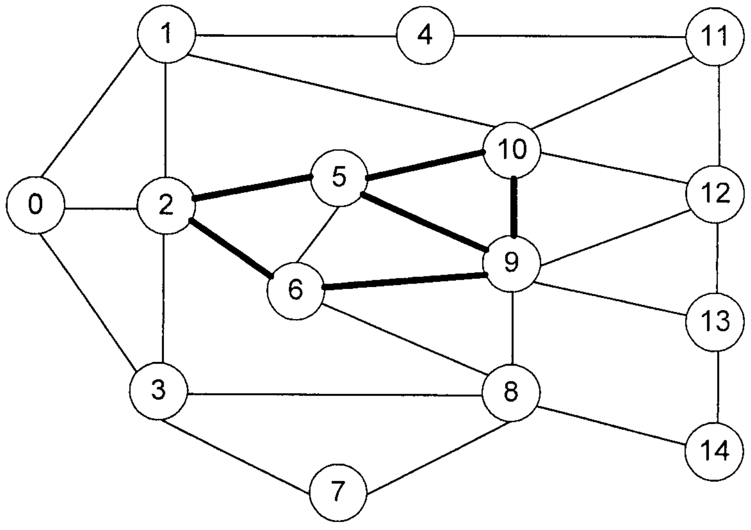


Figure 3.5: Network graph G1, thin lines have capacity of 1200 bandwidth units and thick lines have capacity of 4800 units.

connected to edge routers, because core routers are generally the bottleneck if they do not have high capacity links.

- In a typical OSPF-TE environment number of routers usually not larger than 20.
- Actual network topologies normally support large number of possible paths for each ingress/egress pair to provide fault tolerance. This is also needed to check or to enhance the performance of a TE based routing algorithm.

3.3.2 Test Scenarios

We have compared FRA against WSP and MIRA under both static and dynamic traffic conditions. In the static scenario, once a request is routed it will not be removed throughout the simulation. In the dynamic case, for network graph G1, the network is initially loaded with 7000 static routes, but the next 2000 requests arrive with exponentially distributed interarrival times with rate λ. Each new route persists for a period of time that is exponentially distributed with an average hold time of $\frac{1}{\mu}$ seconds. After the expiration of its hold time, the route is removed. We have selected the value $\frac{\lambda}{\mu} = 1000$ in our simulations.

It should be mentioned that ultimately, the performance of routing algorithm can be compared based on the number of admitted path requests and the computational complexity. In TE-routing, however, the comparison is not straightforward since it depends on many factors including network topology and path request patterns. Therefore, in addition to path request acceptance/blocking rate, results based on load balancing performance and route quality provide for more insightful comparison among the routing methods.

3.3.3 Results for Static Routes Scenario for Graph G1

For static routes, we compare the three algorithms based on path request blocking, load balancing and route quality.

Load balancing Performance

In this simulation, we have compared FRA load balancing performance with WSP and MIRA. We have generated 6500 random requests between all 90 ingress/egress router pairs. For load balancing comparison, it is necessary for the three algorithms to route the same requests and either do not reject any request or reject exactly the same subset of requests in the same order. However, since the second scenario is not possible to achieve with different algorithms, we have tested the algorithms for 6500 requests when there were no path request rejections by any algorithm.

We use the maximum link utilization as the main load-balancing performance metrics to compare the performance of FRA, MIRA, and WSP, as shown in Figure 3.6. Figure 3.6 shows maximum link utilization attained by each of the algorithms after routing path requests. The results indicate clearly that that FRA outperforms MIRA and WSP. The reason is that MIRA does not provide any load balancing, whereas load balancing in WSP is restricted to shortest paths only, which saturate certain network links while leaving others underutilized. FRA provides better load balancing by choosing slightly longer paths if they provide better load balancing.

Figure 3.7 plots the average link utilization after routing each flow request for the three algorithms. Again, FRA outperforms the other two algorithms. However, its performance in terms of average link utilization is close to WSP. For additional insight, we have also plotted the standard deviation (STD) of link utilization in Figure 3.8. The smaller STD of FRA is an indicator of its better load balancing performance.

To show that the performance of FRA is independent of the pattern of requests, we have compared the algorithms for 20 different sets of flows. In Figure 3.9, we have plotted the average link utilization at the end of each trial. It can be easily seen that FRA performs the best in all the trials.

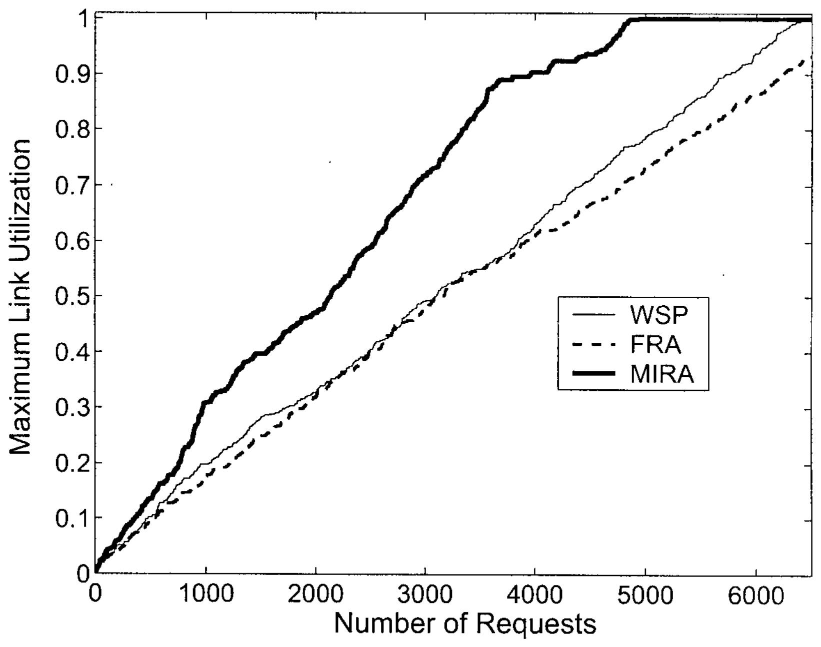


Figure 3.6: Maximum Link Utilization versus Number of Requests for Static Scenario for network graph G1.

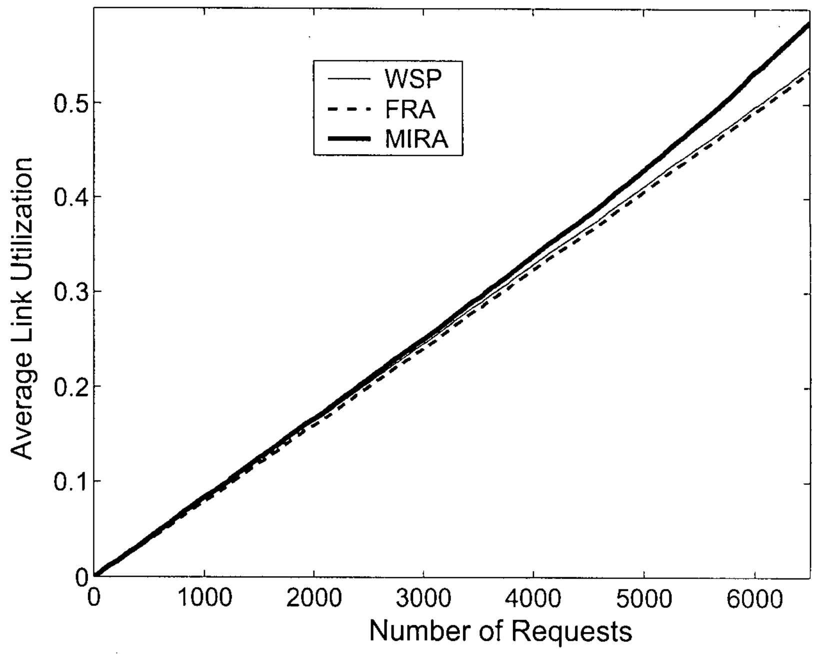


Figure 3.7: Average Link Utilization versus Number of Requests for Static Scenario for network graph G1.

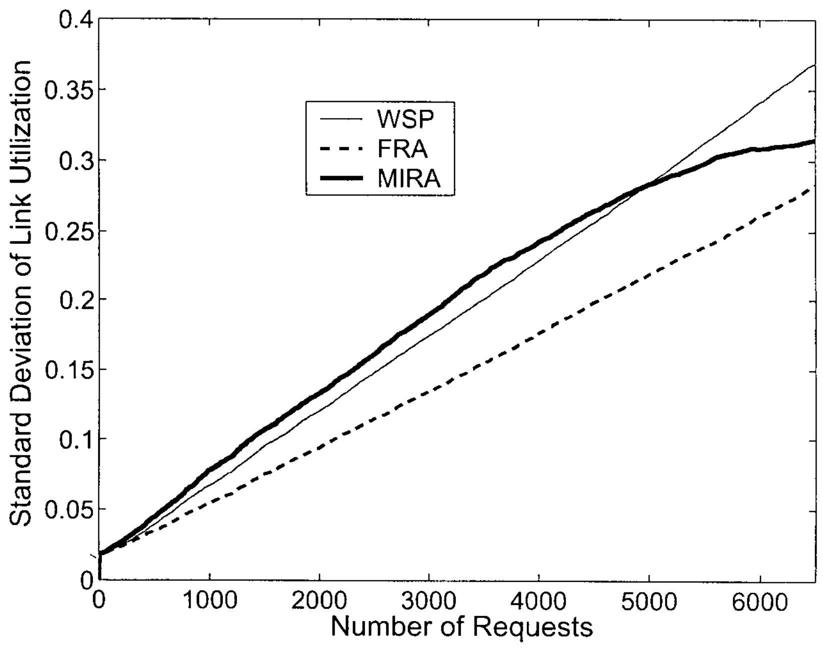


Figure 3.8: Standard Deviation of Link Utilization versus Number of Requests for Static Scenario for network graph G1.

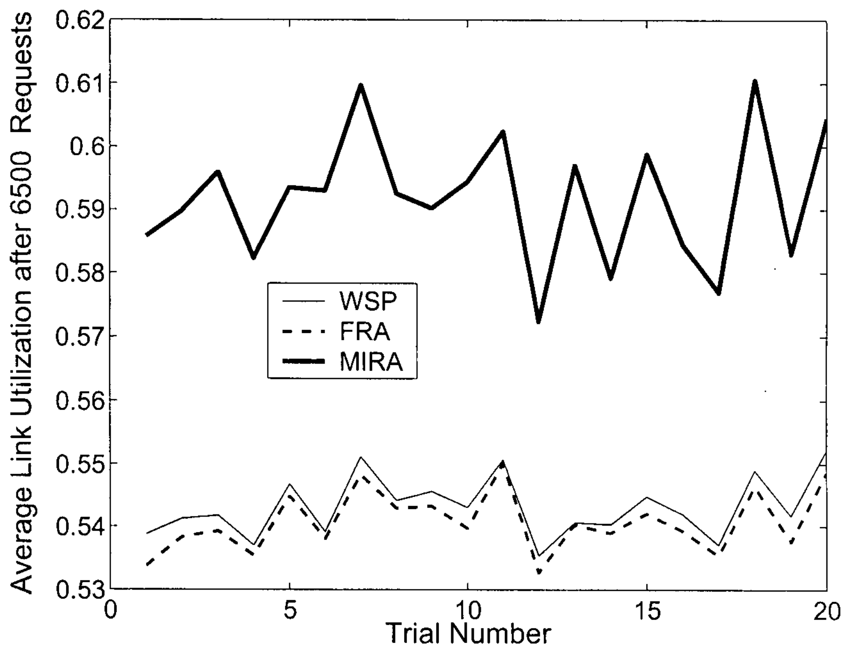


Figure 3.9: Average Link Utilization versus Trial Number for Static Scenario for network graph G1.

Path Request Blocking Performance

Figure 3.10 shows the number of rejected requests after each request arrival for 9000 requests, clearly demonstrating that FRA rejects the least number of requests. Figure 3.11 shows that this behavior is consistent over 20 trials and FRA always provides the least path request blocking.

Route Quality

The QoS of the traffic routed over a given path can be degraded considerably if the path is routed through one or more congested links. As packet queues build up on congested links, the effectiveness of scheduling techniques can be adversely impacted resulting either in higher end-to-end delays or high packet drop for all flows through these links. Therefore, an additional useful metric for the performance of a routing algorithm is a measure that we call route-quality and defined as in the number of accepted paths that are routed through congested links. Let $Q(\mu, R)$ be the route-quality metric that calculates the number of times congested links with link utilization grater than μ used to route a requested path, and R is the number of path requests. Then,

$$Q(\mu, R) = \sum_{i \in L} \sum_{r=1}^{R} g_i(r) \tag{3.16}$$

where

$$g_i(r) = \begin{cases} 1 & If\ request\ r\ uses\ link\ i\ and\ utilization(i) \geq \mu \\ 0 & Otherwise \end{cases} \tag{3.17}$$

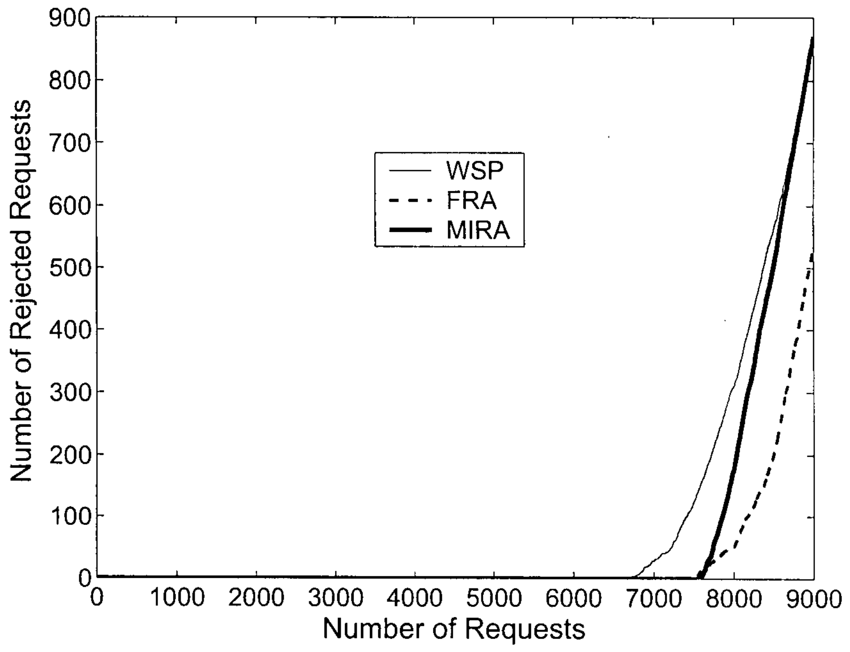


Figure 3.10: Number of Rejected Route Requests versus Number of Requests for Static Scenario for network graph G1.

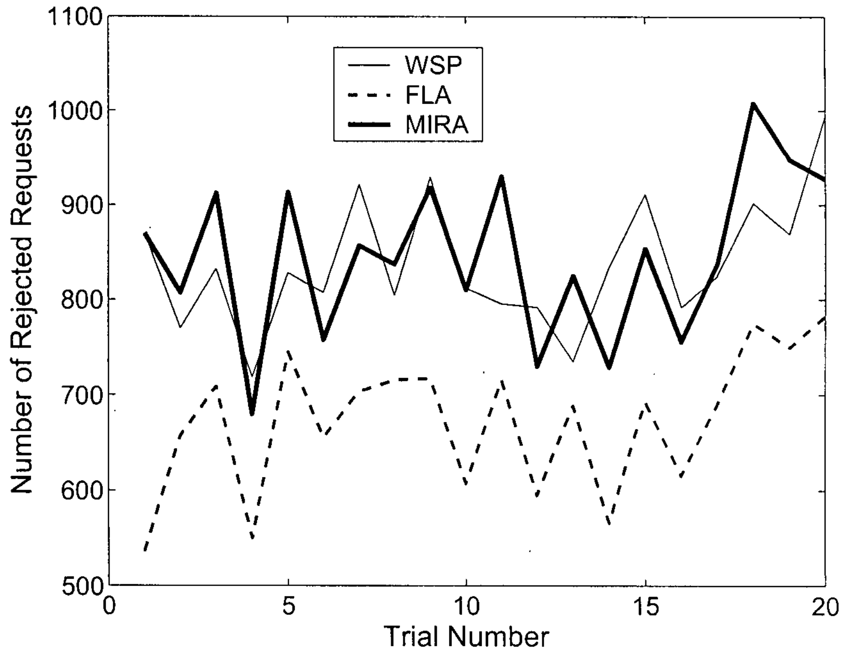


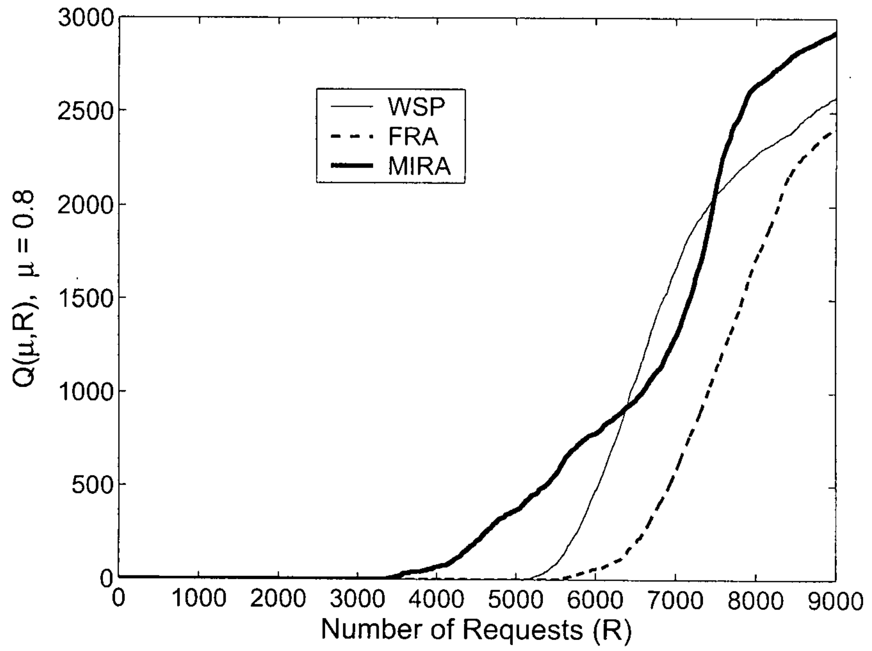
Figure 3.11: Number of Rejected Route Requests versus Trial Number for Static Scenario for network graph G1.

Note that route-quality also differentiates between two paths based on how many congested links are traversed. Accordingly, a path routed over multiple congested links has worse quality than a path routed over fewer congested links.

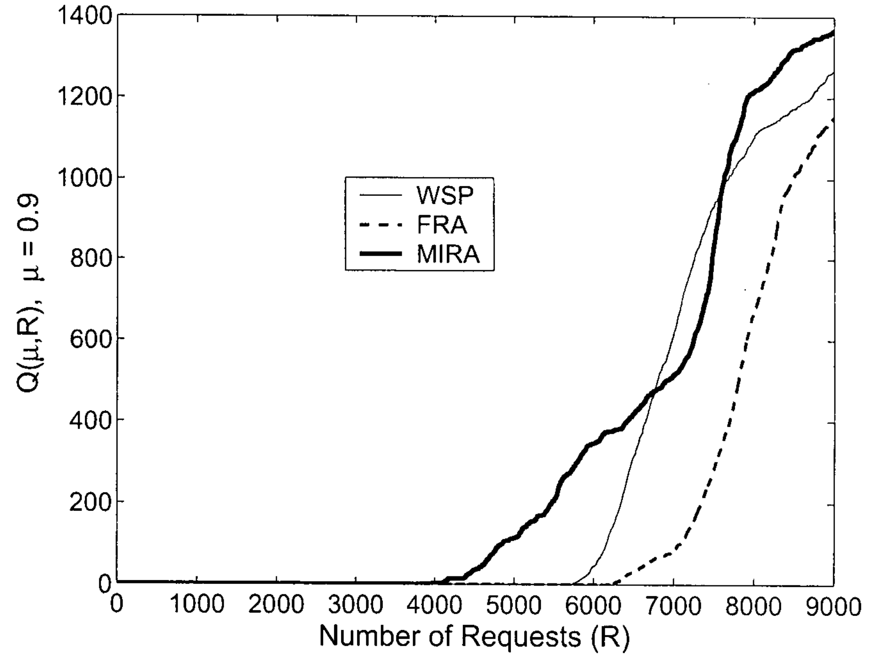
We have compared our algorithm for two different link utilization (or congestion) levels: $\mu = 0.8$, and $\mu = 0.9$. The simulation results of Figure 3.12 show that MIRA starts using congested links after routing just 3500 requests (Figure 3.12(a)), whereas FRA uses the first congested link after routing 5500 requests. This clearly indicates FRA's superior ability to maintain a load-balanced network even after routing thousands of requests. The same effect can be seen for highly congested links (with $\mu = 0.9$). It is worth noting that, according to Figure 3.10, FRA has successfully routed more paths while using less congested links than either MIRA or WSP.

3.3.4 Dynamic Routing Scenario for Graph G1

Figures 3.13, 3.14, and 3.15 compares FRA against other algorithms in a more realistic dynamic routing scenario where we first route then tear down paths as described earlier. Figure 3.13 compares the algorithms in terms of rejected requests versus the original number of requests. The results for the more realistic dynamic scenario show a marked advantage of FRA over the other two algorithms. To show that the behavior is not for a particular set of requests, we have tested the algorithms on 20 different sets of requests in Figure 3.14 and observed the same performance behavior. Figure 3.15 shows that FRA is still using less number of highly utilized links to provide better quality routes.



(a)



(b)

Figure 3.12: Quality of LSP's, for Static Scenario for network graph G1.

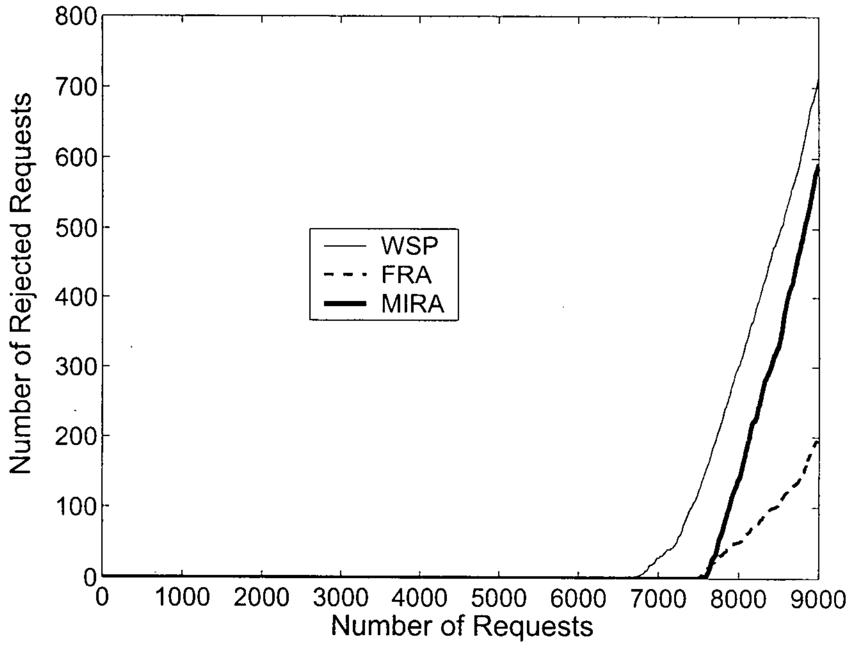


Figure 3.13: Number of Rejected Route Requests versus Number of Requests for Dynamic Scenario for network graph G1.

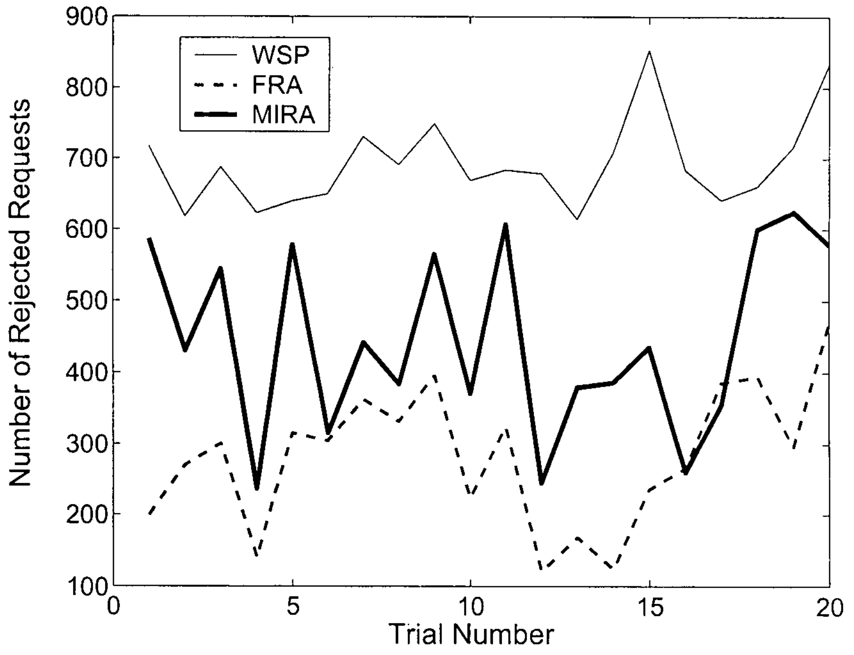


Figure 3.14: Number of Rejected Route Requests versus Trial Number for Dynamic Scenario for network graph G1.

It should be clarified that the figure indicates that WSP is using a smaller number of congested links after 8500 requests, but this is because WSP has rejected a considerably larger number of path requests as compared to FRA (see Figure 3.13).

3.3.5 Second Test Network Graph G2

To verify the consistency of our results, we have compared the three algorithms on a larger network topology, shown in Figure 3.16. All the routers in this topology are considered to be edge routers and form ingress/egress pairs with all other routers. With 20 routers, the total number of ingress/egress pairs is 20(20-1) = 380. The requests are again generated randomly between 1 and 4 bandwidth units (Mbps). We have tested the algorithms under 15,000 requests for load balancing and 20,000 requests for all other tests. Selected results for static scenario are shown in Figures 3.17-3.21. These results demonstrate almost the same performance behavior as with the previous network graph (G1) of Figure 3.5. The only exception for the current network is that the number of rejected requests is nearly identical for FRA and MIRA. However FRA outperformed MIRA for all other tests.

3.4 Summary

This chapter introduced the concept of fuzzy membership functions as a very effective tool for solving the problem of TE-constrained Internet routing. It has been shown that the main algorithm for implementing routing with fuzzy membership functions, i.e. FRA, outperforms other previously proposed techniques in terms of its low computational

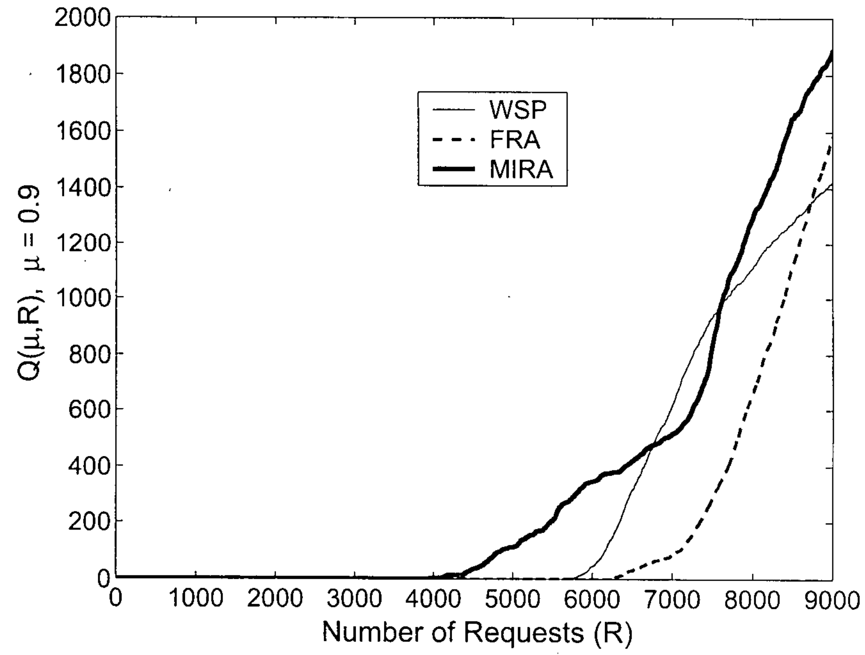


Figure 3.15: Quality of LSP's, for Dynamic Scenario for network graph G1.

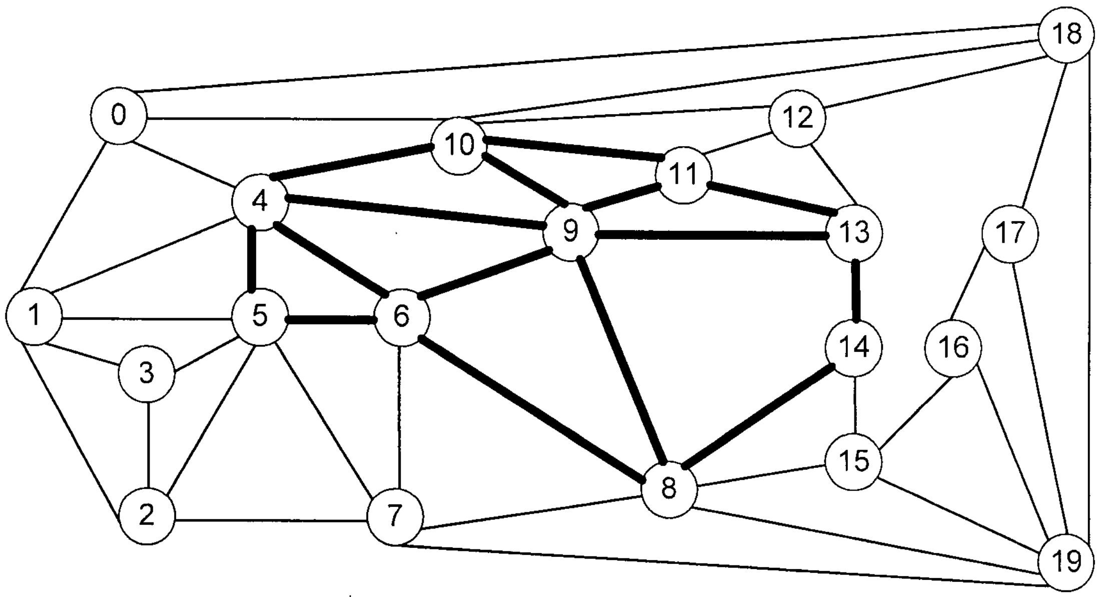


Figure 3.16: Network graph G2, thin lines have capacity of 1.2 Gbps bandwidth units and thick lines have capacity of 4.8 Gbps.

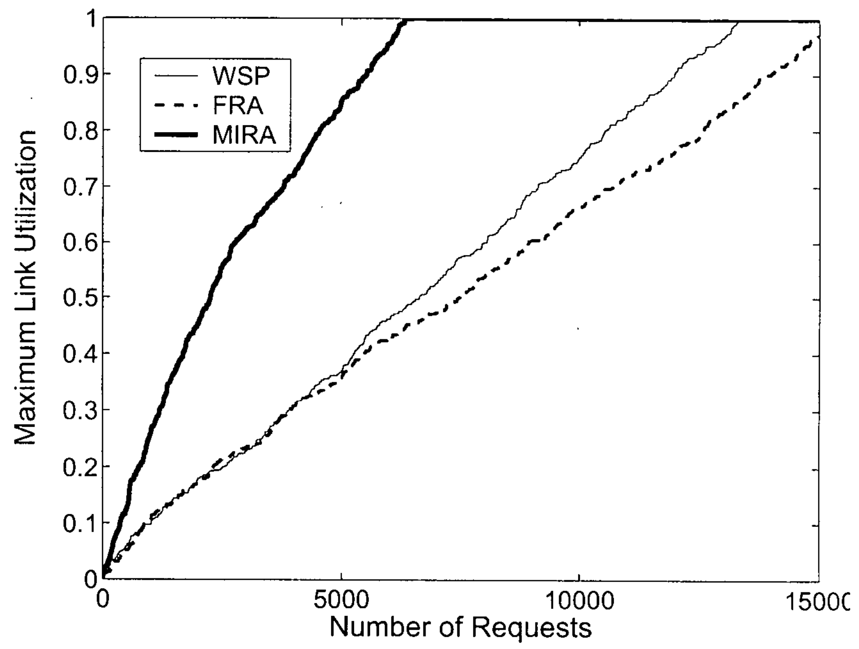


Figure 3.17: Maximum Link Utilization versus Number of Requests for Static Scenario for network graph G2.

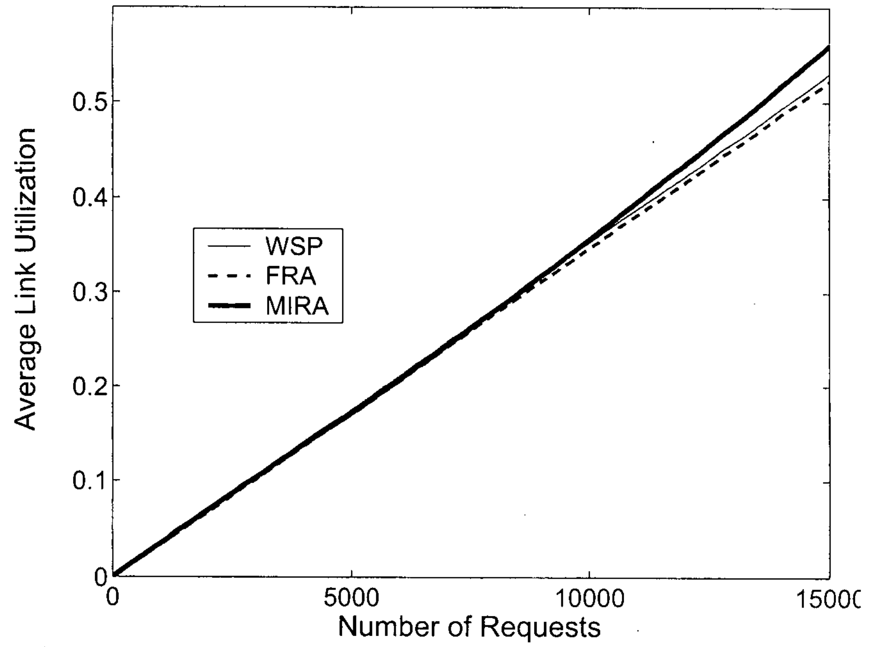


Figure 3.18: Average Link Utilization versus Number of Requests for Static Scenario for network graph G2.

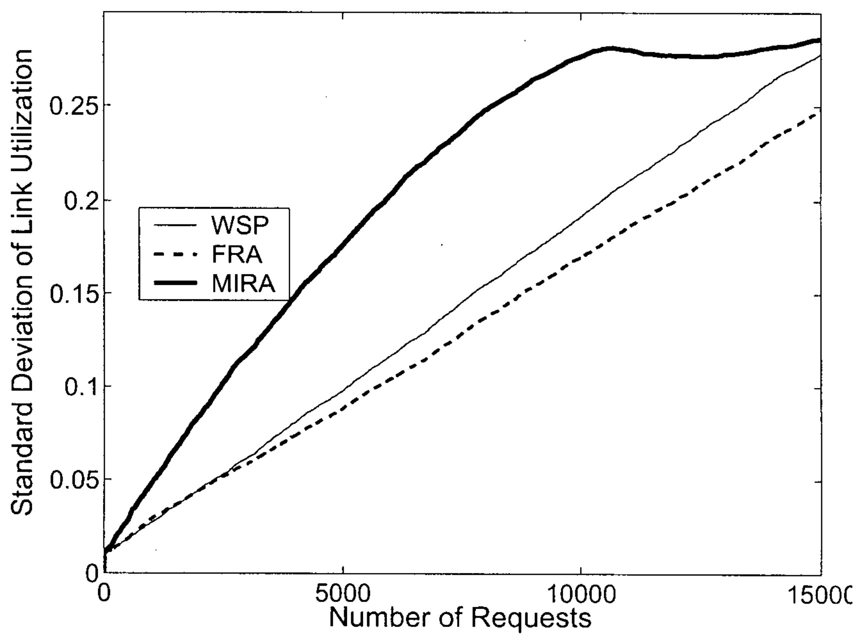


Figure 3.19: Standard Deviation of Link Utilization versus Number of Requests for Static Scenario for network graph G2.

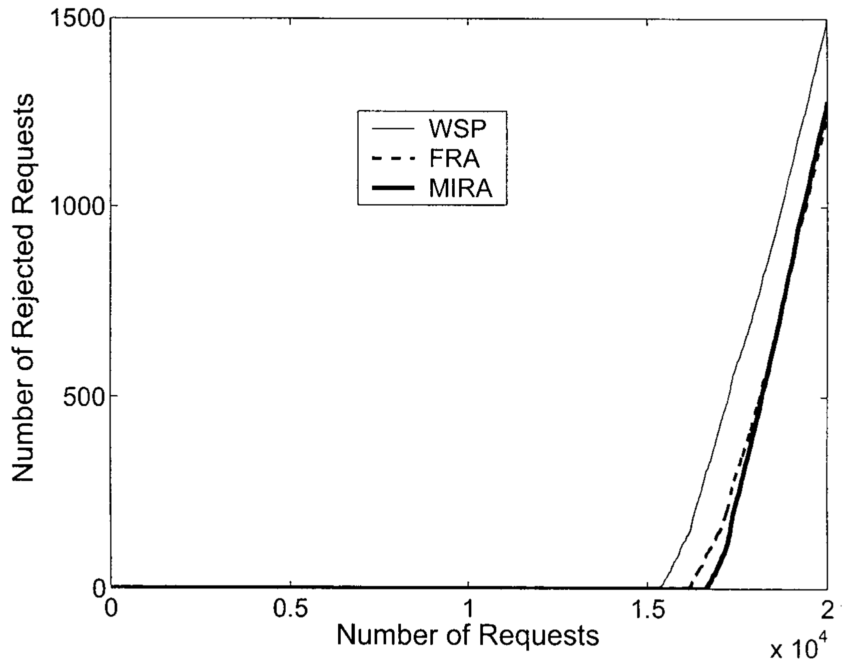
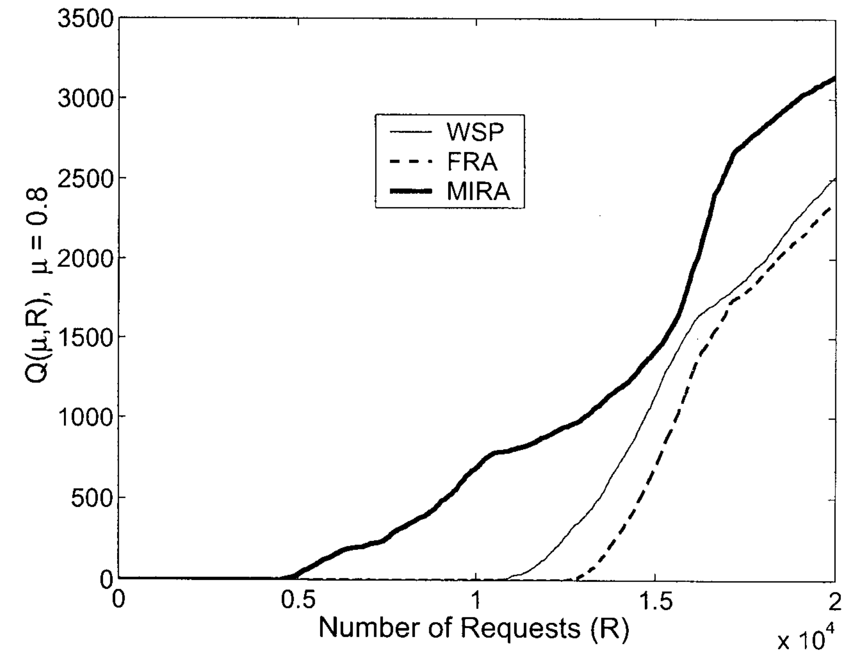
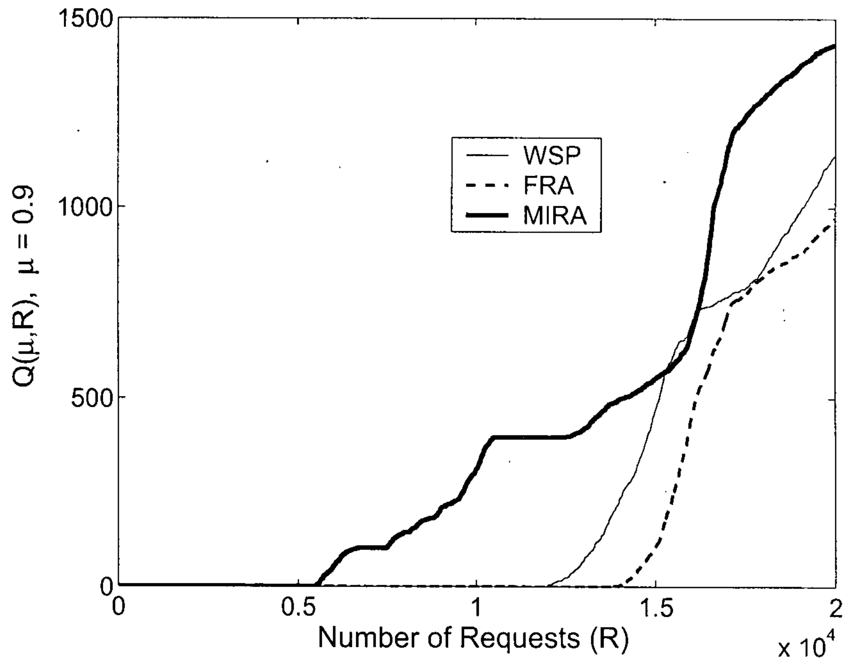


Figure 3.20: Number of Rejected Route Requests versus Number of Requests for Static Scenario for network graph G2.



(a)



(b)

Figure 3.21: Quality of LSP's, for Static Scenario for network graph G2.

complexity as well as its excellent routing performance. The significance of the fuzzy membership approach was demonstrated by showing that FRA always performs better than WSP and MIRA in terms of path request rejection ratio and traffic load balancing. It should be emphasized that although FRA employs fuzzy-logic concepts, the algorithm outcome is deterministic. All the information that FRA needs can be obtained from TE-enhanced routing protocols such as OSPF-TE.

Chapter 4

TE-Based Channel Allocation in Fixed Wireless Mesh Networks

4.1 Introduction

The high capacity offered by a Broadband Fixed Wireless (BFW) Systems has induced a general interest in the communication community. Furthermore, success in wireless Local Area Network (WLAN) using 802.11 has increased the demand for wireless solutions versus wired based solutions [26].

For a service provider to provide acceptable QoS to its customers, it must be in the full control of its resources. In Broadband Fixed Wireless (BFW) systems this means that the use of licensed radio spectrum must be carefully managed to provide reliable QoS. Unlicensed bands are not immediately useful to service providers because of the risks associated with the presence of other potential users in the same spectrum, which makes it difficult to provide reliable QoS. In the 2 to 11 GHz frequency range, two licensed frequency regions are available, namely Wireless Communication Services (WCS), and Multichannel Multipoint Distribution System (MMDS) [27]. WCS provides only 30 MHz of bandwidth at 2.3 GHz frequency making it unsuitable for broadband services. Whereas MMDS provides 200 MHz of spectrum in the range 2.5 GHz to 2.7 GHz. The problem with MMDS is that it is not widely available because licenses have been issued for this spectrum a long time ago in most places. Further licensed spectra are available above the 18 GHz range such as Local Multipoint Distribution Service (LMDS). One portion of LMDS, 27.5 GHz to 28.35 GHz, provides almost 850 MHz of spectrum to use. To provide sufficient services to its customers, a broadband service provider needs at least 100 MHz or more of the spectrum [28]. Such abundance of bandwidth makes it suitable to use higher frequencies at millimeter-wave bands for BFW systems.

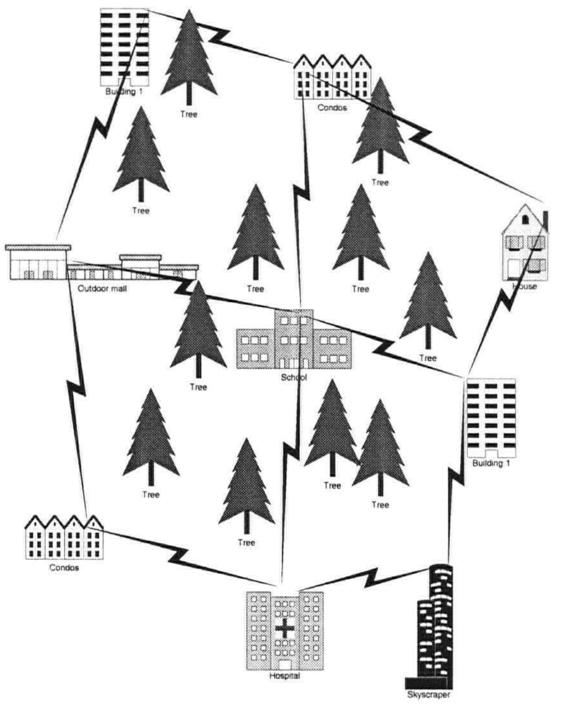
Inherent difficulties in millimeter-wave radio operations, such as higher atmospheric attenuation, especially during rainy times, limit the range of millimeter-wave links to a few kilometers (1-2 km). Furthermore, Line of Sight (LOS) operation is necessary at these frequencies, and multipath effect is not significant. These requirements make it harder to use these frequencies in traditional Point-to-Multi-Point (PMP) network architecture because it needs expensive base station every few kilometers. The authors in [29] have proposed a mesh architecture to overcome these difficulties. The work in this thesis is based on this particular wireless mesh network (WMN) architecture.

4.2 WMN Architecture

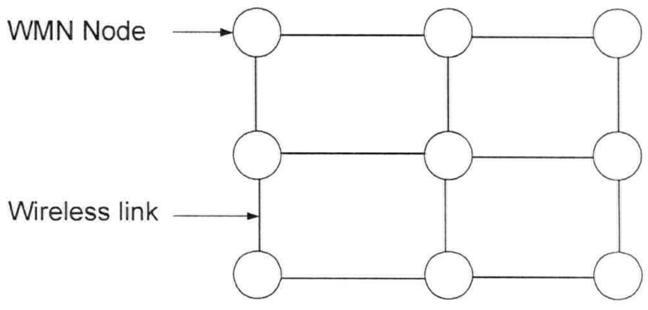
There are two main categories of WMNs, *logical mesh*, and *physical* or *directed mesh* [30, 31]. The logical mesh typically uses omni-directional antennas to minimize complexity.

On the other hand, a physical mesh requires the use of strongly directional antennas to reduce interference. An example physical WMN and its network architecture is presented in Figure 4.1. Each node in this network is a wireless router capable of routing traffic from a source node to a destination node. Some of these nodes are connected to access networks, some are points of interface to the core network (Internet) or some other networks and some are just relay nodes. Each node is connected to other nodes through point-to-point wireless links using directional antennas. The whole network is a multipoint-to-multipoint network, where a traffic flow may use multiple hops to reach destination node. As shown in Figure 4.1(a), with WMN it is possible to walk around trees and obstacles while still receiving a signal which may not be possible for a PMP network.

We assume a WMN model where each wireless link between two WMN nodes consists of several radio channels. In this respect the bandwidth capacity of a link is equal to the bandwidth sum of all constituent channels. Since the number of channels assigned to different links can vary, the WMN links will not necessarily have uniform bandwidth capacity. This distinguished feature provides more flexible routing in WMNs because bandwidth can be added or subtracted from individual links simply by moving channels. It is possible by using adaptive modulation techniques that a wireless link will be up most of the time at the expense of capacity. It is also possible to control the capacity of each link by allocating or removing wireless channels from the link. Two adjacent nodes can communicate to allocate or remove a channel from a link. The only constraint is that no



(a)



(b)

Figure 4.1: (a) Example of wireless mesh network, and (b) corresponding network graph.

two links connected to the same node can share a channel to avoid interference. In the presence of directional antennas the interference is reduced considerably, however a node transmitting signal on one link can interfere with the signal received on one of its other links. Also each node maintains a list of the channels allocated to the links connected to it, and consequently knows the capacity of the links. The node also knows the reserved bandwidth (or bandwidth utilization) on such links all the time. It is appropriate to assume that MPLS (Multi Protocol Label Switching) tunnels are operational in the network by reserving the required bandwidth on each route. This assumption provides a better approximation of bandwidth utilization (or reserved bandwidth) on each link. Another assumption is the use of time division duplexing (TDD) on each link. TDD makes it possible to use same channel on a link for traffic on both directions. TDD avoids the unused channel problem present in Frequency Division Duplexing (FDD) and hence provides better link utilization. In FDD it is possible that, for a short period of time, there is no traffic in one direction and thus the frequency channel allocated to that direction was not in use. Furthermore, it is not appropriate to reallocate channels for such a short period of time. TDD shares the channels on both directions, thus if there is no traffic in one direction for a short time then all the channels can use most of their time in one direction with TDD bursts, providing better link utilization. Finally, we assume the presence of a control radio channel in the system. Although the same control radio channel is used on each link, using a contention avoidance mechanism such as CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) can eliminate interference problem between two adjacent links. The use of CSMA/CA may introduce some overhead, but for the purpose of control signals, it may not be very significant. However, once a channel is assigned to a link, then the control radio channel is no longer needed for that link. A control channel is needed only at the initial network configuration or when a node joins a network. Also, each node is capable of detecting the presence of a link. If a link is down, the node removes all channels from this link.

4.2.1 Benefits of WMNs

It is the goal of any network service provider to deploy a reliable network in shortest time, with lower cost while providing with all the customers' needs. WMN is a right solution in this scenario for access networks and/or for backbone networks where bandwidth demand does not require a fiber optic based network.

WMN nodes are capable of receiving/transmitting and routing data traffic. Nodes in WMN are interconnected in a mesh topology to provide multiple hop paths to traffic flows [32, 33]. The mesh topology in WMN makes it possible to walk around obstacles such as trees, tall buildings etc., to provide a better coverage, which may not be possible in Point-to-Multi-Point (PMP) networks. In PMP networks long distances between a base station and a subscriber unit may cause poor carrier to noise ratio (CNR) and carrier to interference ratio (CIR), whereas WMN uses shorter wireless links to eliminate this problem [34]. With larger CNR and CIR in WMNs, it is possible to use higher bit rates utilizing the same analog frequency band as in PMP networks [29]. By using highly directed antennas, a WMN may also provide better frequency reuse than PMP networks. Even with omni-directional antennas, the frequency reuse distance is smaller because of smaller link distances (lower transmitted power) [32].

In directional or physical WMN, each node requires more than one antenna to connect itself to multiple nodes. By using directional antennas, the link range can be extended or conversely the transmission power can be reduced. The directional pattern at the receiver and transmitter ends provides better frequency reuse than the logical mesh. It is to be noted that it is advantageous to use physical mesh topology in backhaul networks as well. The use of narrow Point-to-Point (PP) radio beams to establish a link means that energy is not transmitted where it is not needed and the spectrum can be reused many more times [35]. Hence more data traffic can be accommodated in the backhaul network because of better frequency reuse. Better CNR and CIR can be achieved with directional antennas because of higher antenna gain, this results in better link availability even in degraded situations e.g. from rain and bad weather. With higher CNR and CIR it is possible to obtain higher bit rate utilizing the same frequency band. Higher antenna gains, using directional antennas, make it possible to have long-range links, which are common in backhaul network. Since service providers operate backhaul network nodes, a node failure is not as common as for the nodes at customer premises. Therefore, it is not needed to reconfigure the network topology quite often, however a link failure may be more common than in a wired network. Although nodes with directional antennas are more expensive than omni-directional devices, still they are cheaper than base stations in Point-to-Multi-Point (PMP) networks.

4.3 WMN Node Architecture

A hardware architecture of a node that can work in a directed mesh architecture was described in [36, 37]. Figure 4.2 represents the architecture of wireless nodes at both ends of a wireless link. The node consists of a set of Outdoor Units (ODU), Indoor Unit (IDU) and a Network Control Unit (NCU). The ODUs consists of receivers, transmitters and narrow beam directional antennas working at millimeter band frequencies. The narrow beam directional pattern can be obtained by either using array antennas or steerable antennas. A mechanically steerable antenna system is provided by Radiant Networks[1] in their product MESHWORKS, working at 26 GHz and 28 GHz ranges. The set of four mechanically steerable antennas in this product are capable of providing narrow beam-width of 4.5 degrees azimuth and 9 degree elevation. Their claim is that such antennas are not costly and are appropriate for narrow beam scenarios.

The indoor unit consists of a Switch Unit (SW Unit) and set of Channel Units (CHU) and a Channel Manager (CHM). Each CHU is responsible for modulation and demodulation of a carrier for a radio channel. The CHU is an expensive resource and normally limited number of channel units can be provided in a single node. However using digital modulation techniques in base band, this cost can be considerably minimized and single broadband RF transceiver can be used per link. CHM manages the attributes of each

[1]Radiant Networks: www.radiantnetworks.com (last accessed on May 17, 2005.)

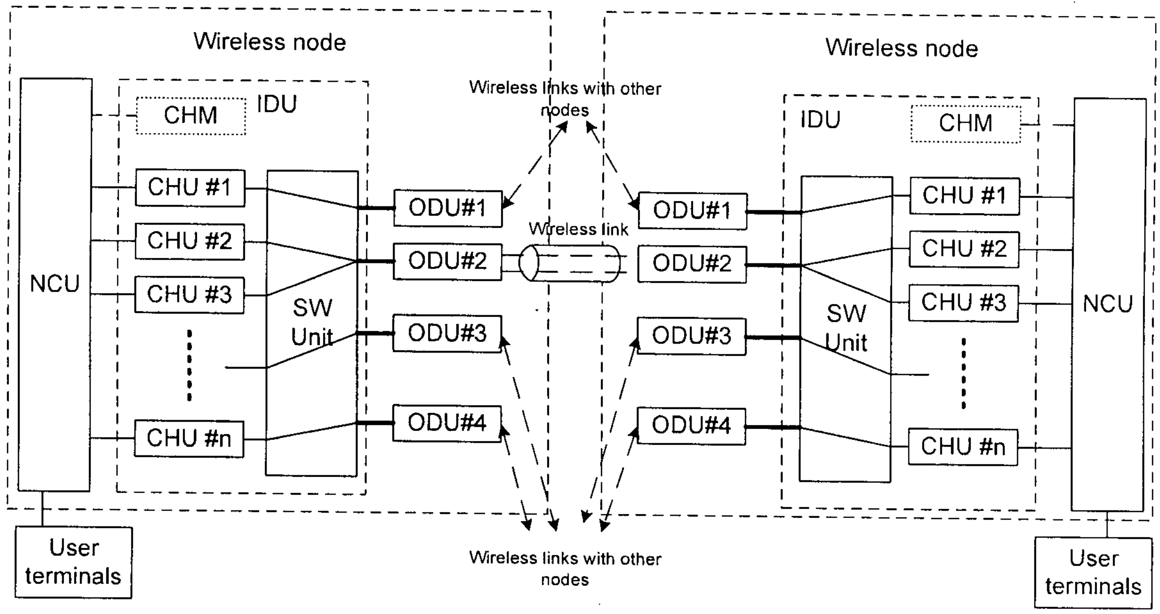


Figure 4.2: Architecture of wireless node and link.

channel such as carrier frequency, transmission power and modulation level.

The NCU provides several functions, such as (a) network control and management functions; (b) adaptive packet routing function; and (c) adaptive link control function. Network control and management functions include acquiring network information and adaptive control of link capacities by adding or removing channels on each link. Packet routing is based on MPLS connection oriented routing. Lastly, adaptive link control function is needed, because packets from a single flow can be forwarded on the same link but using different channels, this may result out of sequence packets to arrive on the other side of the link. Link control function reorder them to increase TCP efficiency for end-to-end connections.

In spite of all the advantages mentioned, WMNs are still in their early stage of development and several real problems remain to be solved before an efficient deployment of WMNs is achieved. Our work proposes to solve one major issue, i.e. to provide TE solution in physical WMN. TE is a proven technique which has been used with considerable success in wired networks. TE's main objective is to increase network utilization and maximize the number of admitted connections through load balancing. There are two ways to provide TE in a network; (a) by using a well-designed TE based routing algorithm to avoid hot spots or (b) by reconfiguring the network to eliminate hot spots by balancing traffic load on all links. It is obvious that hot spot elimination is a better solution and can be achieved by increasing the link capacities of congested links. However, in wired networks it is not possible to change link capacities and thus they rely on solution (a) in general. However it is possible in WMNs to change link capacities by assigning or removing channels from each link. Our solution has adopted the approach in (b).

4.4 Problem Formulation

The objective of TE with dynamic channel allocation is to assign channels in the available spectrum, using a distributed dynamic algorithm, to each link in the network, such that

(a) No two links connected to the same node share the same channel. This is to avoid interference.

(b) Each link has at least one channel.

(c) Most of the available channels in the spectrum are used/reused to achieve better resource utilization. In other words, summation of link capacities is maximized.

(d) The distribution of channels is balanced. For an unloaded network, it means that channels are distributed as evenly as possible. For a network with some bandwidth reservation, it means that channels are allocated such that the utilization of links remains balanced.

(e) Maximum link utilization is minimized.

(f) Minimum numbers of future unknown route requests are blocked because of reconfigured channel allocation.

Constraint (a) is to avoid interference between adjacent links (links connected to same node). Although the use of directional antennas reduces this possibility, the presence of side lobes in antenna pattern does not eliminate interference completely. As mentioned earlier, transmitted signal on a link can interfere with receiving signal on another link which is sharing the same node and same channel. It is, therefore, needed to assign different channels to adjacent links. However, non-adjacent links have the freedom to choose the same channels, which may not be always possible if omni directional antennas are used. Constraint (b) is needed to provide a link with minimum capacity. Objective (c) is used to avoid waste of resources and to provide more capacity to links. It is to be noted here that poor channel allocation may result in many unused channels in a node and hence reducing the sum of link capacities. Objective (d) provides a balanced capacity planning. Objectives (e) and (f) are byproducts of objective (d) because a dynamically reconfigurable load balanced network is also optimized with these two objectives. One can argue that, channel allocation to links must be coupled with already estimated traffic demand. However, in real life it is not possible to perfectly predict future traffic demand [38]. In such a case, a distributed and dynamic channel allocation helps to provide better traffic engineering, where channels may be re-allocated based on current network load conditions.

4.4.1 Mathematical Model

A WMN is represented by a graph $G(N, L)$, where N is the set of nodes and L is the set of links. Let n, and l, be the wireless node and wireless link IDs. Other parameters are defined as follows

E_n : Set of links connected to node n.

C_s : Set of radio channels in the available spectrum.

U_n : Set of unused radio channels in node n.

R_l : Set of radio channels allocated to link l.

c_l : Single channel capacity on link l.

r_l : Reserved bandwidth on link l.

μ_l : Bandwidth utilization of link l.

SD_n : Standard deviation of bandwidth utilization of E_n.

SDR_n : Standard deviation of the number of radio channels allocated to links in E_n.

$F_i(s,d,b)$: Future unknown i^{th} route request (e.g. MPLS tunnel) to be routed from s to d with at least b units of guaranteed bandwidth.

X_i : $X_i = 0$ if $F_i(s,d,b)$ has successfully routed, else it is 1.

The objective of the TE with dynamic channel allocation is to route $F_i(s,d,b)$ and assign channels in C_s to every set R_l ($\forall l \in L$) dynamically to optimize certain objectives while fulfilling certain constraints. Formally,

$$\textbf{Minimize} \quad \begin{cases} \sum_{\forall n \in N} SD_n & ; \text{ If network has some reserved bandwidth.} \\ \sum_{\forall n \in N} SDR_n & ; \text{ Otherwise.} \end{cases} \tag{4.1}$$

$$\textbf{Minimize} \quad \sum_{\forall n \in N} |U_n| \tag{4.2}$$

$$\textbf{Minimize} \quad MAX(\mu_l) \tag{4.3}$$

$$\textbf{Minimize} \quad \sum X_i \tag{4.4}$$

Constraints

$$R_i \cap R_j = \{\}\,; \qquad \forall i, j \in same\ E_n; \quad \forall n \in N \tag{4.5}$$

$$|R_l| > 0; \qquad \forall l \in L \tag{4.6}$$

$$MAX(\mu_l) \leq 1.0 \tag{4.7}$$

The objective functions in Equation 4.1 represent two scenarios. In the first scenario, the network has reserved some bandwidth based on connection requests seen before. In this case, the objective is to assign channels to links, such that links connected to the same node will have nearly the same amount of bandwidth utilization. The second scenario represents the moment when the network is initialized with no reserved bandwidth. Recall that a link is a grouping of channels. In such case, the objective is to assign nearly the same number of channels to all links connected to the same node. This objective ensures that if a link connected to a node has a high bandwidth utilization then it should get a channel from one of its neighboring links to reduce utilization. The objective function in Equation 4.2 increases the efficiency of channel usage by minimizing the number of unused channels (maximizing the channel reuse) in every node in the network. Note that if all the channels for n are in use, then $|U_n| = 0$. The objective in Equation 4.3 is a byproduct of objective in Equation 4.1 as every node tries to reduce the maximum utilization of links connected to it by reducing the standard deviation of the links' utilizations. The last objective in Equation 4.4 is to route maximum number of route requests through the network. We assume that connection route requests arrive one by one and there is no a priori knowledge about future requests. These assumptions are considered realistic and

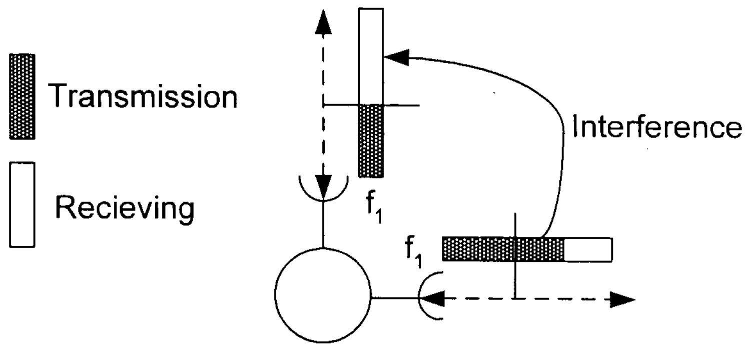


Figure 4.3: Interference between two adjacent links.

have been used in prior art [1, 38].

There are three constraints in channel allocation. Constraint 4.5 states that no two links connected to the same node can share a channel. This constraint avoids channel interference on adjacent links. Constraint 4.6 makes sure that each link has at least one radio channel assigned to it to avoid zero capacity links. Constraint 4.7 is a maximum capacity constraint that makes sure that no link will have reserved bandwidth more than its capacity at anytime.

The use of directional antennas considerably reduces the interference between non-adjacent links (links not connected to the same node). However, interference cannot be completely eliminated between adjacent links because of finite side lobes present in directional antenna patterns. This phenomenon can be explained using Figure 4.3 [39]. Consider that the two links in Figure 4.3 are using the same channel. In this case, the transmitted signal from one link interferes with the received signal on the other adjacent link. For low cost antennas, side lobes are typically not below -20 *db* from the main lobe. The received power on a link from an interfering signal from an adjacent link can be as high as -20 db, whereas the received power of the desired signal for millimeter band is much lower than this level. In the best-case scenario, we can consider only free space path loss represented by $P_{loss} = 20 \cdot log_{10}(\frac{4\pi \cdot f \cdot d}{c})$, where f is the radio frequency, d is the separation between transmitting and receiving nodes, c is the speed of light and P_{loss} is free space loss in dB. For the 28 GHz band with 100-meter separation between transmitting and receiving nodes, the path loss is almost -100 dB. Therefore, the desired received signal will have much lower power than the interfering signal, meaning that it is not desirable to use the same channel on adjacent links. One way to overcome this problem is to use same TDD boundaries on adjacent links sharing the same channel. This means that links will transmit and receive signals simultaneously. However, this technique does not suit data networks since the traffic loads on links are asymmetric and it is feasible to adjust TDD boundaries independently for each link according to its traffic load.

4.5 Compatibility with IEEE 802.16-2004

IEEE 802.16-2004 standard [40] does not provide any frequency allocation mechanism for licensed band. For license exempt band there is mandatory Dynamic Frequency Selection (DFS) mechanism. However, DFS only provides mechanism to avoid channels used by some primary users. In the absence of a standardized solution for frequency allocation, it is possible to use dynamic and distributed channel allocation to dynamically change link capacities to provide better resource utilization in the network. It is to be noted that for better utilizing the proposed approach, it is needed to have large number of channels in the system. This problem is solved in 802.16 standard for 2-11 GHz licensed spectrum where it is possible to choose up to 1.25 MHz/channel to divide the allocated spectrum into large number of channels. For millimeter wave bands it is mentioned that a norm is to use 20 or 25 MHz/channels however in our opinion it is better to choose smaller bandwidth per channel to have large number of channels. Also at higher frequencies it is relatively easier to allocate larger spectrum per service provider because of abundant bandwidth as compared to centimeter band.

4.6 Previous work

Despite the need for an efficient distributed dynamic solution for this problem, surprisingly not many have been reported in the literature. This section will review the few reported methodologies to solve the problem. In [39], the authors presented a dynamic solution to minimize the number of links which interfere with each other while using the same frequency. Their decision criteria, however, was not given in a formal algorithm to solve the problem. Also the objective of their paper is to minimize interference rather than avoiding interference completely. Almost the same group of authors presented a solution for simultaneous routing and channel allocation in [41]. Their objective was to provide traffic engineering in WMN by reconfiguring all the routes and channel allocation every few minutes. However, with the presence of large number of routes, it is not feasible to recalculate routes and channel allocation and to restore new paths every few minutes. Furthermore, interference is not considered in this work. The entire solution is based on a centralized server, thus limiting the scalability of WMN. The centralized server introduces an extra bottleneck, because failure of the server may stop the whole process.

The above two reported solutions were for the exact network model that is considered in this thesis. However, because of some drawbacks in their problem formulation they cannot be compared with the work in this thesis. The first solution, allows the links which interfere with each other to share a channel that is not appropriate if one wants to remove interference completely. The second solution did not assume interference at all hence many adjacent links can share the same channel.

It is worth to mention here that the problem under consideration is different from the channel allocation in PMP networks such as cellular networks or BFW with PMP architecture. In PMP networks the problem is to assign channels to nodes and not to links. Several solution for this problem exists such as those mentioned in [42, 43, 44]. However the problem under consideration is different from these, as we have to assign channels to links and not to nodes. One may argue that a dual graph may serve this purpose, but if we use the dual graph to use algorithms for channel allocation in PMP networks then we have to consider that wireless links have computational capabilities, whereas in reality it is wireless nodes which can compute. Based on these it is needed to develop a distributed dynamic channel allocation for the problem discussed in Section 4.4.

4.7 Summary

This chapter has covered a detailed overview of wireless mesh architecture at millimeter band. This includes mesh architecture details, architecture of the nodes in the network under consideration. The chapter has also formulated the traffic engineering problem based on distributed dynamic channel allocation. Compatibility of the problem with IEEE 802.16 standard was discussed and at the end the chapter has covered some previous works to solve the problem.

Chapter 5

Dynamic Channel Allocation Algorithm

In this chapter, we present a complete solution for routing with channel allocation in fixed wireless mesh networks. The proposed solution combines two different techniques. We first propose a distributed dynamic channel allocation (DDCA) algorithm that optimizes the usage of expensive licensed frequency channels based on current network load. To realize DDCA, we proposed a new decentralized protocol that allows neighboring nodes to borrow or release channels using simple control messages. We provide full analysis of the DDCA algorithm to prove that it indeed optimizes its objectives and that it converges to a stable solution without creating loops. In the second part of the chapter, we present a novel technique for wireless mesh networks that combines the DDCA algorithm with route bandwidth reservations to provide a complete TE solution. The proposed solution is truly distributed since no node needs network wide information to optimize its frequency

[0]A version of this chapter has been published. Khan, J.A. and Alnuweiri, H.M. (2005) Traffic engineering with distributed dynamic channel allocation in BFWA mesh networks at millimeter wave band. IEEE-LANMAN.

channel usage. Although the proposed solution targets a particular type of wireless networks, viz. BFW mesh networks at the millimeter band frequency range, the same approach of dynamic channel allocation to provide TE can be applied to other types of networks, such as WiFi based (e.g. 802.11a) mesh networks with multiple active channels.

5.1 Distributed Dynamic Channel Allocation (DDCA)

The proposed solution to the DDCA problem employs message passing to distribute channel information among wireless nodes. Each node is capable of invoking the channel allocation process based on certain conditions. Whenever, a node joins a network, or detects a change in bandwidth reservation on a link connected to it, the node checks if there is a need to invoke local channel allocation/reconfiguration process. A node n invokes channel allocation/reconfiguration process when at least one of the following conditions is true (See Chapter 4 for definitions).

Condition 1: There are unused channels available in node n i.e.

$$|U_n| > 0 \tag{5.1}$$

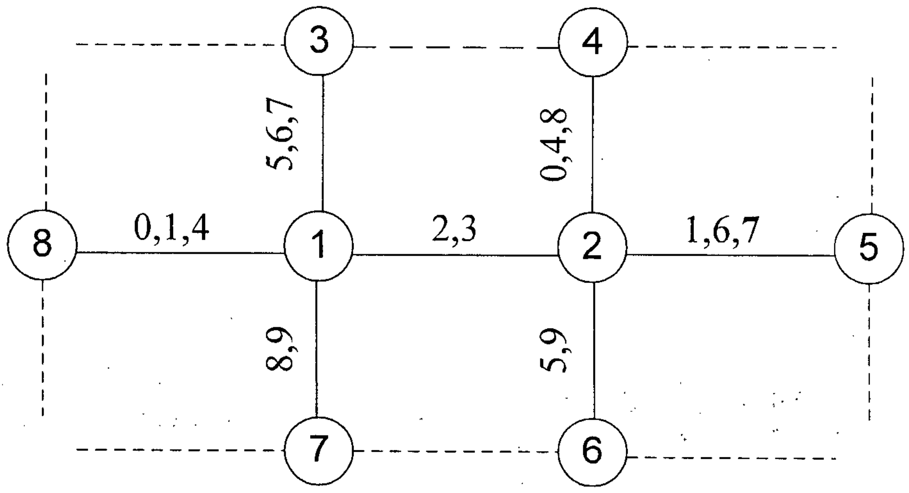
Condition 2: If

$$\frac{r_A}{c_A \times (|R_A| - 1)} + \delta < \frac{r_B}{c_B \times |R_B|} \tag{5.2}$$

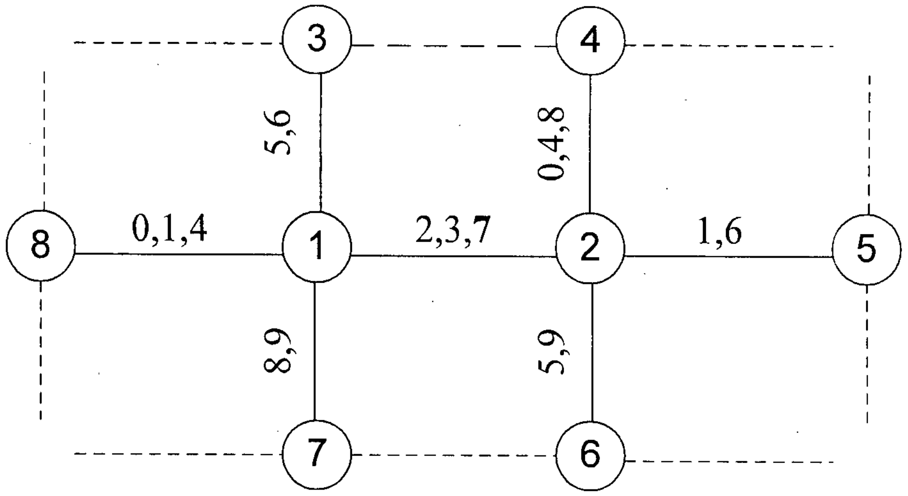
for any two links A and B connected to node n, where $\delta > 0$ is a constant, and A has at least two channels available.

The first condition is self-explanatory and is used to achieve objective in Equation 4.2. The second condition states that removing a channel from link A and allocating it to link B will provide a better load balancing thus optimizing other objectives. A positive threshold of δ (e.g. $\delta = 0.1$) is added to avoid instability in WMN system. The above channel allocation scenario can be explained with the example in Figure 5.1 (numbers beside each link corresponds to the channels allocated to the link). Each channel in this example is modulated to have 10 Mbps. In Figure 5.1(a) this gives 20 Mbps to link 1-2 and 30 Mbps to links 1-3 and 2-4 each. Suppose the bandwidth reservations on link 1-2 take 18 Mbps resulting in a bandwidth utilization of 0.9, and on links 1-3 and 2-4 bandwidth reservations take 15 Mbps resulting in bandwidth utilization of 0.5 on each of the two links. In this case Condition 2 is true and nodes 1 and 2 can start the channel (re)allocation process resulting in the channel configuration shown in Figure 5.1(b) whereby channel 7 is moved to link 1-2 and removed from links 1-3 and 2-4. The resulting bandwidth utilization for link 1-2 is now 0.6, whereas for other two links it is 0.75. It is worth observing that the network is now better load-balanced and the congestion in link 1-2 has been eliminated.

If a node n needs to (re)allocate the channels to the links connected to it, node n sends a Channel allocation Request (CR) message to the node on a selected link indicating that n needs an extra channel on the link. The link with highest bandwidth utilization



(a)



(b)

Figure 5.1: Channel reallocation example. Bandwidth capacity per channel is 10 Mbps; link 1-2 has 18 Mbps reservation whereas links 1-3 and 2-4 have 15 Mbps reservations.

is selected by node n to send CR message. A tie (specially if there is no bandwidth reservation) is broken by selecting the link with smallest number of allocated channels $|R_l|$. This optimizes objective in Equation 4.1 for unloaded network. Selecting a link at random breaks further ties.

5.1.1 Message Passing

A flow of CR and its responses is shown in Figure 5.2. In this figure, node x has decided that it needs an extra channel on the link connecting x and y. Node x sends a CR message to node y with list of available channels on link xy. The list of available channels in node n for link B are the channels in set U_n and all the channels on links connected to n which satisfy condition in Equation 5.2.

Figure 5.2 shows four different message response scenarios. In Figure 5.2(a), node y sends a positive acknowledgment that it has an available channel for xy that matches one of the requested channels in CR. Node x then sends a positive confirmation to y to assure allocation of the channel on the link. In this case nodes x and y also remove the channel from other adjacent links. However, it is possible that node x has received an earlier CR from a node other than y and has already sent a positive acknowledgment for the same channel. In this case, node x sends a negative confirmation to y. Figure 5.2(b) shows a case with negative acknowledgment of CR. It is also possible that node y does not have an available channel that matches a channel in CR, but it has some other available channels for xy. In this case, node y sends a negative acknowledgment but with

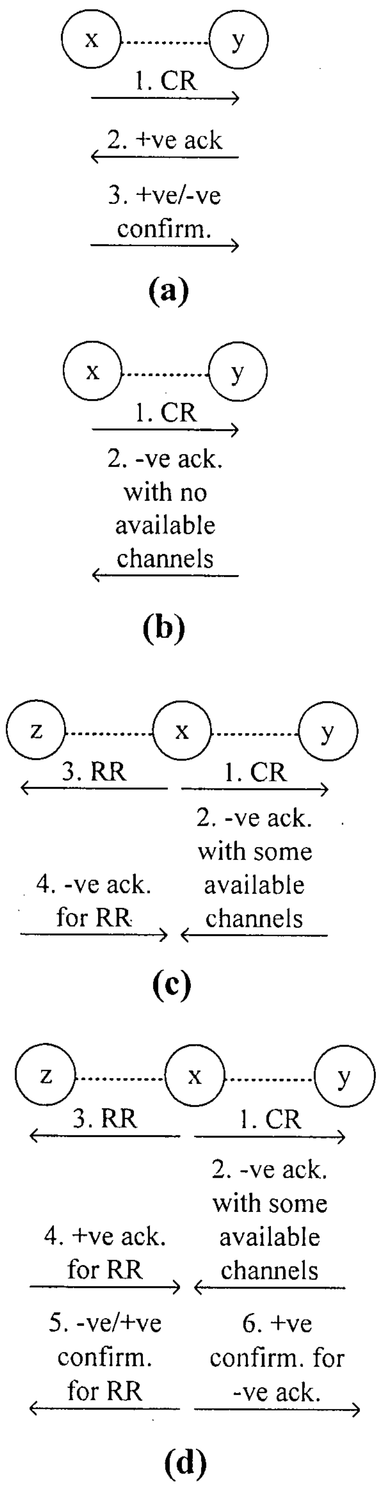

x
y
1. CR
2. +ve ack
3. +ve/-ve confirm.
(a)
x
y
1. CR
2. -ve ack. with no available channels
(b)
z
x
y
3. RR
1. CR
2. -ve ack. with some available channels
4. -ve ack. for RR
(c)
z
x
y
3. RR
1. CR
2. -ve ack. with some available channels
4. +ve ack. for RR
5. -ve/+ve confirm. for RR
6. +ve confirm. for -ve ack.
(d)

Figure 5.2: Flow of channel request (CR) and its responses. (a), (b), (c) and (d) represent different scenarios.

a list of available channels as shown in Figure 5.2(c). When node x receives this type of negative acknowledgment, it sends a replacement request (RR) to its neighboring nodes. These are the nodes which are connected by links having at least one channel common with those listed in the negative acknowledgment. The RR message contains the list of channels in the negative acknowledgment, and list of channels in CR. The purpose of RR is to replace a channel in a link xz that matches the available channel in y with an available channel in x. If z does not accept RR then it sends a negative acknowledgment for RR. However, if z accepts the RR based on channels in U_z, or on condition 2 for link xy as shown in Figure 5.2(d), then it sends a positive acknowledgment. After receiving positive acknowledgment for RR, node x checks whether it has already received another positive acknowledgment from some other node and replied to it. If not, x checks for its available channels again before sending a positive confirmation to z. After sending a positive confirmation to z, node x also sends a positive confirmation to y to allocate the channel moved from link xz to link xy. After receiving positive confirmations from x, both y and z remove the corresponding channels from other adjacent links, if needed. If x finds that it has already accepted another positive acknowledgment then it sends a negative confirmation to z. In such a case, x does not send any message to y.

It is possible that node x is unable to allocate a channel to link xy in the above algorithm. For further optimization, node x again checks for need to invoke local channel (re)allocation process. However, this time node x does not include links to which it has just failed to allocate a channel. A node x tracks failed attempts by using a flag for each link connected to it. This flag for a link is set after sending a CR message on this link. Once node x achieved a successful channel allocation then it removes the flag from all links connected to it. The flags are also removed from all the links connected to node x, if node x has unsuccessful attempts on all links connected to it. However, node x does not sends a CR message in such a case. The flags are removed in this situation because channel reconfiguration is not possible at this time but it may be possible in future. After a successful channel allocation, a node x again checks for channel allocation. This is needed because it is possible that there are some unused channels in U_x which can be assigned to some links connected to x.

To complete the DDCA algorithm specification, we need to consider the scenario when a node n receives a new CR or RR message from a node m while n itself is waiting for a response to a CR or RR message that it has sent earlier. In such a case, n will compare the two links for which it has sent and received CR or RR messages, then chooses the one with higher utilization. Further ties are broken using the number of channels assigned to those links, followed by random selection. If node n decides not to serve an incoming CR or RR message then it asks node m to check for a need for channel (re)allocation again and sends the CR or RR message again, if needed.

The above-mentioned DDCA algorithm uses 13 different message types. The algorithm will be further specified by showing how these messages are processed when received by a node, using formal sub-algorithms. Before discussing the message processing we introduce a formal algorithm that a node uses to decide about the need to invoke local channel (re)allocation by sending a CR message (CRM). We call this Channel Request Initiation Check.

5.1.2 Channel Request Initiation Check

The algorithm is shown in Figure 5.3. The algorithm starts by checking for a flag for each link in E_n. The flag indicates that an unsuccessful CR request has already been sent for this link. If this flag is asserted for all links, then it means that it is not feasible to initiate another channel request at this time, because it will be unsuccessful again. In such a scenario, the algorithm resets the flag for all links in E_n and exits without sending CR message. The flag is removed because it may be possible in the future to initiate a successful CR on these links. This flag is also used in the condition given in Equation 5.2 to exclude links with unsuccessful CR.

The most needed link in the algorithm description is a link connected to n that has the highest bandwidth utilization and did not have unsuccessful CR. A tie (specially if there is no bandwidth reservation) is broken by selecting the link with smallest $|R_l|$. This tie breaking technique optimizes objective in Equation 4.1 for unloaded network. Selecting a link at random breaks further ties.

5.1.3 Message Types

The following thirteen message types are used in this work.

```
ALGORITHM CR_Check(n)
Begin
    1. If $Flag(l) == 1; \forall l \in E_n$ then
        $Flag(l) = 0; \forall l \in E_n$
        goto 4;
    End If;
    2. If $|U_n| > 0$ then
        Send Channel Request Message on most needed link $l$;
        $Flag(l) = 1$;
        goto 4;
    End If;
    If for any $A$ and $B \in E_n$
        $Flag(A) == Flag(B) == 0$ and $|R_A| > 1$ and $\frac{r_A}{c_A \times (|R_A|-1)} + \delta < \frac{r_B}{c_B \times |R_B|}$ then
          Send Channel Request Message on most needed link $l$;
          $Flag(l) = 1$;
          goto 4;
    End If;

4. END CR_Check.
```

Figure 5.3: Channel Request Initiation Check.

M1. Channel Request Message (CRM): CRM from a source node s to a destination node d consists of all unused channels in s (i.e. U_s), and channels on all links $A \in E_s$ that satisfy condition 2 for CR initiation check. Where link B in condition 2 is the link connecting s and d here.

M2. Positive Ack for CRM (Ack_{cr}^{+}): Ack_{cr}^{+} consists of positive acknowledgment for the CRM and a channel c that matches a channel in CRM. It indicates that the requested node (node d) could assign this channel c to the link.

M3. Negative Ack for CRM (Ack_{cr}^{-}): Ack_{cr}^{-} consists of negative acknowledgment for the CRM. It indicates there is no channel available for the link that matches a channel in CRM. However, if there are other channels available, then Ack_{cr}^{-} also includes a list of the available channels. We call these channels replacement request channels (RRC). Ack_{cr}^{-} also contains the list of channels in CRM.

M4. Positive Confirmation for CRM (PCCR): When the source node receives back an Ack_{cr}^{+}, it checks that the acknowledged channel is still available. If it is still available then PCCR is sent. PCCR contains the channel in Ack_{cr}^{+}.

M5. Negative Confirmation for CRM (NCCR): NCCR is sent when the source node realizes that the channel in Ack_{cr}^{+} is not longer available for the link.

M6. Replacement Request Message (RRM): RRM is sent by source node in response to Ack_{cr}^{-} with at least one RRC. RRM contains RRC and the channels in the original CRM. The RRM is sent on all the links connected to each neighbor node which has at least one channel that matches a channel in the RRC list. The purpose of RRM is to replace a channel in RRC by a channel in CRM. The replaced channel is then assigned to the original requesting link. RRM also contains the number of channels in the original requested link, and the link's bandwidth utilization.

M7. Positive Ack for RRM (Ack^{+}_{rr})**:** Ack^{+}_{rr} contains positive acknowledgment of RRM. It also lists the replaced channel on the link and the new allocated channel.

M8. Negative Ack for RRM (Ack^{-}_{rr})**:** Ack^{-}_{rr} is sent when a node cannot replace a channel on the link where it has received an RRM.

M9. Positive Confirmation for RRM (PCRR): PCRR is sent after receiving Ack^{+}_{rr}, provided that replacing channel is still available.

M10. Negative Confirmation for RRM (NCRR): NCRR is a response of Ack^{+}_{rr}, when NCRR sending node has already responded to another Ack^{+}_{rr} on another link.

M11. Positive Confirmation for Ack^{-}_{cr} **(PCNA):** A PCNA is sent by the source node after sending PCRR to the node that sent Ack^{-}_{cr}. The purpose of PCNA is to allocate a channel in RRC to the requesting link. This is the channel that is replaced on another link by using RRM.

M12. Remove Channel Message (RCM): A node sends RCM to another node if it wants to remove a channel form the link connecting these two nodes. RCM contains channel to be removed from the link.

M13. Channel Request Check Message (CRCM): A source node sends CRCM to ask another destination node (d) to invoke a new $CR\_Check(d)$ again.

5.1.4 Message Processing

Figure 5.2 provides an overview of the proposed distributed algorithm. However, it is needed to understand message processing at each node whenever it receives a message. This section covers message processing for each message type in detail.

CRM Processing:

Whenever a destination node (*dst*) receives a CRM, it runs the algorithm given in Figure 5.4. In this algorithm, the node *dst* first checks that either the node itself has sent a CRM or RRM and waiting for their response. In such a scenario, the *dst* node compares the link k for which it has sent RRM or CRM with the link l where it has received current CRM. If k has higher utilization than l, indicating that k's need for a channel is more than l's, then node *dst* sends a CRCM to the sending node so that it will reschedule its channel request for later time. The *dst* node also sends CRCM if k and l have the same utilization and number of channels assigned to k is less; further ties are broken at random. If node *dst* does not send a CRCM then if a channel in the list of requested channels matches a channel in its list of unused channels, then the node *dst* sends a positive acknowledgment for this channel. The node also removes the channel from its list of unused channels. If the node does not find a matching channel in its set of unused

```
ALGORITHM $Handle\_CRM(src, dst, ch\_list)$
//$src$: source node.   $dst$: destination node.   $l$: link connecting $src$ and $dst$.
// $ch\_list$: list of channels that $src$ can assign to link.
Begin
 1. If $dst$ is waiting for a CRM or RRM response then
   // $k$: link for which $dst$ has sent CRM or RRM.
   2. If $\mu_k > \mu_l$ then  Send CRCM ($src$,$dst$);  goto 10;  End If 2;
   3. Else If$\mu_k == \mu_l$ then
     4. If $|R_k| < |R_l|$ then  Send CRCM ($src$,$dst$);  goto 10;  End If 4;
     5. Else If $random > 0.5$ then  Send CRCM ($src$,$dst$);  goto 10;  End If 5;
   End If 3;
 End If 1;
 //$R\_ch\_list$: Replacement channel list to be sent with $Ack^-_{CR}$;
 // $SC$: Selected channel for $Ack^+_{CR}$;
 $R\_ch\_list = \{\} \cup U_{dst}$
 6. If  $ch\_list \cap U_{dst} \neq \{\}$ then
   $SC$ = Any one channel from $ch\_list \cap U_{dst}$;
   Send $Ack^+_{CR}(src,dst,SC)$;  $U_{dst} = U_{dst} - SC$;  goto 10;
 End If 6;
 7. For  $\forall j \in E_{dst}$
   8. If  $\frac{r_j}{c_j \times (|R_j|-1)} + \delta < \frac{r_l}{c_l \times |R_l|}$ and $|R_j| > 1$ then
     9. If $R_j \cap ch\_list \neq \{\}$ then  $SC$ = Anyone channel from $R_j \cap ch\_list$;
          Send $Ack^+_{CR}(src,dst,SC)$;  goto 10;  End If 9;
     $R\_ch\_list = R\_ch\_list \cup R_j$
   End If 8;
 End For 7;
 Send $Ack^-_{CR}(dst,src,ch\_list,R\_ch\_list)$)
10. END $Handle\_CRM$.
```

Figure 5.4: Algorithm to handle CRM.

channels then it check all its links which can give a channel to requested link and have a channel that matches one of the channel in the list of requested channels. If it finds a channel on such links, satisfying the given condition, then a positive acknowledgment is sent for that channel. If the node does not find a channel for positive acknowledgment then it sends a negative acknowledgment. The negative acknowledgment contains all the requested channels and channels that the node can assign to link l but are not in requested channel list.

Ack_{cr}^{+} Processing

After receiving positive acknowledgment for a CRM, a node runs the algorithm given in Figure 5.5. First the flags for the all the links connected to this node are set to 0. Some of these flags were set to 1 in *CR_Check*. The algorithm also makes sure that channel for which it has received positive acknowledgment is still available. If the channel is still available then the node will send a positive confirmation and assign the channel to requesting link. However, it is possible that channel is not available anymore because it has been assigned to some other link or the condition given in the equation in the algorithm is not valid anymore. In such a case, a negative confirmation is sent. Also if the node is sending positive confirmation after removing a channel from another link, then it will send RCM message to remove channels from the link that is giving the channel. A *CR_Check* is also performed in case there are some more unused channels in the node.

```
ALGORITHM $Handle\_Ack^{+}_{CR}(src, dst, SC)$
//$src$: source node.    $dst$: destination node.    $l$: link connecting $src$ and $dst$.
// $SC$: Channel in positive $ACK$.
Begin
  1. For $\forall j \in E_{dst}$
     $Flag(l) = 0$ ;
  End For 1;
  2. If $SC \cap U_{dst} \neq \{\}$ then
     $U_{dst} = U_{dst} - SC$;    $R_l = R_l \cup SC$;    $CR\_Check(dst)$;
     Send PCCR($dst$,$src$,$SC$);    goto 6;
  End If 2;
  3. For  $\forall j \in E_{dst}$
     4. If $\frac{r_j}{c_j \times (|R_j|-1)} + \delta < \frac{r_l}{c_l \times |R_l|}$ and $|R_j| > 1$ then
        5. If  $R_j \cap SC \neq \{\}$ then
             // $d$ : node connected to $dst$ through link $j$;
             $R_j = R_j - SC$;    Send RCM($dst$,$d$,$SC$);    $R_l = R_l \cup SC$;
             $CR\_Check(dst)$;    Send  PCCR($dst$, $src$,$SC$);    goto 6;
        End If 5;
     End If 4;
  End If 3;
  Send NCCR($dst$,$src$,SC);
6. END $Handle\_Ack^{+}_{CR}$.
```

Figure 5.5: Algorithm to handle Ack^{+}_{CR}.

```
ALGORITHM $Handle\_Ack^{-}_{CR}(src, dst, ch\_list, R\_ch\_list)$
//$src$: source node.    $dst$: destination node.    $l$: link connecting $src$ and $dst$.
// $ch\_list$: Channel list sent in CRM.
// $R\_ch\_list$: Available channel in $src$, but does not match any channel in $ch\_list$
Begin
  1. If $R\_ch\_list == \{\}$ then
     $CR\_Check(dst)$;
  Else
     2. For $\forall j \neq l \in E_{dst}$
        3. If $R_j \cap R\_ch\_list \neq \{\}$ then
             // $d$ : node connected to $dst$ through $j$;
             Send RRM($dst$,$d$,$ch\_list$,$R\_ch\_list$,$\mu_l$,$|R_l|$);
        End If 3;
     End If 2;
  End If 1;
4. END $Handle\_Ack^{-}_{CR}$.
```

Figure 5.6: Algorithm to handle Ack^{-}_{CR}.

Ack_{cr}^{-} Processing

The algorithm to handle Ack_{cr}^{-} is shown in Figure 5.6. If the negative acknowledgment does not include any replacement channel list (empty *R_ch_list*) then the receiving (*dst*) node runs channel request initiation check again. The Flag in Figure 5.3 prevents the node from sending another CRM to the link from which it has just received a negative acknowledgment. It is possible that the replacement channel list is not empty, meaning that there are channels available in the message source node (*src*) but these channels do not match the channels in CRM. In this case, the node *dst* tries to replace a channel that matches a channel in *R_ch_list* on some other link. For this purpose an RRM message is sent with two additional parameter μ_l (utilization of link l) and $|R_l|$. RRM is sent to all the links connected to the node that have at least one assigned channel matching a channel in *R_ch_list*.

PCCR/NCCR Processing

PCCR-receiving node assigns SC in PCCR to the link on which PCCR was received. If SC is assigned to some other link, then first SC is removed from that link and an RCM message is sent on this link for SC.

If SC in NCCR is not assigned to any link connected to *dst*, then NCCR-receiving node (*dst*) includes it in U_{dst}.

RRM Processing

The algorithm to handle an RRM is shown in Figure 5.7. The purpose of RRM is to switch a channel in *R_ch_list* to a channel in *ch_list* on the link where RRM is received. This channel switching makes available a channel in the sending node that can be assigned to the link from which original CRM was sent.

When a node *dst* receives RRM (and it has already sent either a CRM or RRM), then the node checks which link needs a channel the most, the link for which it has sent CRM or RRM or the link for which it has received an RRM. If it finds that the link for which the node has sent a CRM or RRM needs most then it sends a negative acknowledgment for RRM. Otherwise it tries to replace the channel. If the node did not send any CRM or RRM itself then it also tries for channel replacement. If *dst* (RRM-receiving node) finds a channel in its list of unused channels that matches a channel in *ch_list* representing that this channel can be switched with a channel on the link where RRM was received. In such a case, *dst* sends a positive acknowledgment. A positive acknowledgment is also sent if a link connected to *dst* will satisfy the conditions given in step 8 of the algorithm. The first condition represents that link j has enough capacity and could remove a channel so that link l (link for which original CRM was sent) can get a channel. This condition also make sure that j has at least two channels, so that after giving a channel it will have at least one channel. The second condition check that if removing a channel does not give much advantage in terms of utilization then the decision is made based on the number of channels assigned to links j and l. If algorithm is unable to find a replacement

```
ALGORITHM $Handle\_RRM(src, dst, ch\_list, R\_ch\_list, \mu_l, |R_l|)$
//$src$: source node.   $dst$: destination node.   $ch\_list$: Channel list sent in CRM.
// $R\_ch\_list$: Replacement channel list as in $ACK^{-}_{CR}$.   $m$: Link connecting $src$ and $dst$.
// $\mu_l$: Utilization of link $l$ that needs a channel in original CRM.
// $|R_l|$: Number of channels assigned to link $l$.   $SC$: Selected channel for $ACK^{+}_{RR}$
// $R\_ch\_list\_new$: List of replacement channels returned in $ACK^{+}_{RR}$.
Begin
  1. If $dst$ is waiting for a CRM or RRM response then
  // $k$: Link for which $dst$ has sent CRM or RRM.
    2. If $\mu_k > \mu_l$ then   Send $ACK^{-}_{RR}$(dst,src);   goto 10;
      3. Else If $\mu_k == \mu_l$ then
        4. If $|R_k| < |R_l| + 1$ then   Send $ACK^{-}_{RR}$(dst,src);   goto 10; End If 4;
        5. If $|R_k| == |R_l| + 1$ AND $random > 0.5$ then
            Send $ACK^{-}_{RR}$(dst,src);   goto 10; End If 5;
      End if 3;
    End If 2;
  End If 1;
  $R\_ch\_list\_new = R\_ch\_lis \cap R_m$;
  6. If $ch\_list \cap U_{dst} \neq \{\}$ then   $SC$ = Any channel in $ch\_list \cap U_{dst}$;
      $U_{dst} = U_{dst} - SC$;   Send $ACK^{+}_{RR}(dst, drc, SCR\_ch\_list\_new)$;   goto 10;
  End If 6;
  7. For $\forall j \neq m \in E_{dst}$
    8. If $(\frac{r_j}{c_j \times (|R_j|-1)} + \delta < \mu_l$ and $|R_j| > 1)$   OR
       $(\frac{r_j}{c_j \times (|R_j|-1)} + \delta == \mu_l$ and $|R_j| > |R_l| + 1)$ then
      9. If $ch\_list \cap R_j \neq \{\}$ then   $SC$ = Any channel in $ch\_list \cap R_j$;
          Send $ACK^{+}_{RR}(dst, src, SCR\_ch\_list\_new)$;   goto 10;
      End If 9;
    End If 8;
  End For 7;
  Send $ACK^{-}_{RR}(dst, src)$;
10. END $Handle\_RRM$.
```

Figure 5.7: Algorithm to handle RRM.

channel then it sends a negative acknowledgment for RRM.

Ack^{+}_{RR} Processing

The algorithm to process Ack^{+}_{RR} is shown in Figure 5.8. When a node (*dst*) receives this message then first it checks that either it has received another positive acknowledgment from some other link (because *dst* has sent multiple RRM) or not. If *dst* has already received another Ack^{+}_{RR}, then it will discard the acknowledgment and send a negative confirmation to the node that sent this acknowledgment. The *dst* node can keep track of all these acknowledgment by including a sequence number in the messages. If the node has not received any positive acknowledgment yet, then it will remove the flag from all links connected to it. This is to make sure that these links can be included in future *CR_Check*. The *dst* also removes one channel in *R_ch_list_new* from the link where it received positive acknowledgment. This is part of channel replacement. At the end of the algorithm *dst* also sends a PCNA message, for this channel, to the node that sent negative acknowledgment in response of original CRM. In the remaining part of the algorithm *dst* gets the channels that will be replaced on the link where it has received the current message. If the channel is in list of unused channels then it is just removed from this list. If it is on another link, then it is removed from the link and RCM is sent to the node connected to *dst* through this link. At the end, this channel is added to the link where current positive acknowledgment is received and a positive confirmation is send to *src* node. After sending successful positive confirmations, *dst* node schedules another

CR_Check for itself. It is helpful in assigning remaining unused channels to links.

Ack^{-}_{RR} Processing

When a node receives negative acknowledgments for all RRMs it has sent for current cycle then, it just schedules another *CR_Check* for itself. The reason for another *CR_Check* is to possibly send another CRM to links where node has not sent a CRM yet in current cycle.

PCRR/NCRR Processing

After receiving a message PCRR(src, dst, SC, SC_{new}), the dst replace SC_{new} from the link on which it has received PCRR by SC and adds SC_{new} to its list of unused channels. If SC is assigned to another link then dst removes SC from that link and sends RCM to ask the node on other side of the link to remove SC as well from the link.

If SC in NCRR is not assigned to any link in E_{dst} then NCRR receiving node dst includes SC in U_{dst}.

PCNA Processing

If the channel for which PCNA-receiving node has received PCNA is in its list of unused channels then it removes the channel from the list. If the channel is assigned to some other link then the node removes the channel from this link and sends RCM to the node on the other side of this link. At the end the channel is assigned to the link on which PCNA is received.

```
ALGORITHM $Handle\_Ack^{+}_{RR}(src, dst, SC, R\_ch\_list\_new)$
//$src$: source node.    $dst$: destination node.    $l$: link for which original CRM was sent.
// $k$: Link connecting $src$ and $dst$.    $SC$: Channel to be added to $k$.
// $R\_ch\_list\_new$: list of channels that can be removed from $k$.
//$x$: Node connected to $dst$ through link $l$.
Begin
  1. If $src$ already received another $Ack^{+}_{RR}$ on some other link then
       Send NCRR($dst, src, SC$);    goto 7;
  End If 1;
  2. For $\forall j \in E_{dst}$    $Flag(j) = 0$;    End For 2;
  3. If $R\_ch\_list\_new \cap R_k \neq \{\}$ then
    $SC_{new}$ = Any one channel in $R\_ch\_list\_new \cap R_k$;
    $R_k = R_k - SC_{new}$
  End If 3;
  4. If $SC \cap U_{dst} \neq \{\}$ then    $U_{dst} = U_{dst} - SC$
  Else
    5. For $\forall j \in E_{dst}$
      6. if $R_j \cap SC \neq \{\}$ then
        // $d$: node connected to $src$ through link $j$.
        $R_j = R_j - SC$;
        Send RCM($dst, d, SC$);
        Break For 5;
      End If 6;
    End For 5;
  End If 4;
  $R_k = R_k + SC$;
  Send PCRR ($dst, src, SC, SC_{new}$);
  Send PCNA ($dst, x, SC_{new}$);
  $CR\_Check(dst)$;
7. END $Handle\_Ack^{+}_{RR}$.
```

Figure 5.8: Algorithm to handle Ack^{+}_{RR}.

RCM Processing

After receiving RCM, the receiving-node removes the channel from the link where it has received RCM and adds it to its list of unused channels. The node also runs *CR_Check* to assign the newly released channel to some other link.

CRCM Processing

When a node receives a CRCM then it runs *CR_Check* algorithm for itself.

5.1.5 DDCA State Diagram

Based on the sub algorithms used in DDCA, each node follows the state diagram shown in Figure 5.9.

5.2 Algorithm Analysis

This section provides the analysis of the DDCA algorithm, covering the following issues.

1. Complexity of individual sub-algorithms.
2. Message complexity i.e. the number of messages passed to allocate single channel to a link.
3. Computational latency to allocate a single channel.
4. Algorithm scalability.

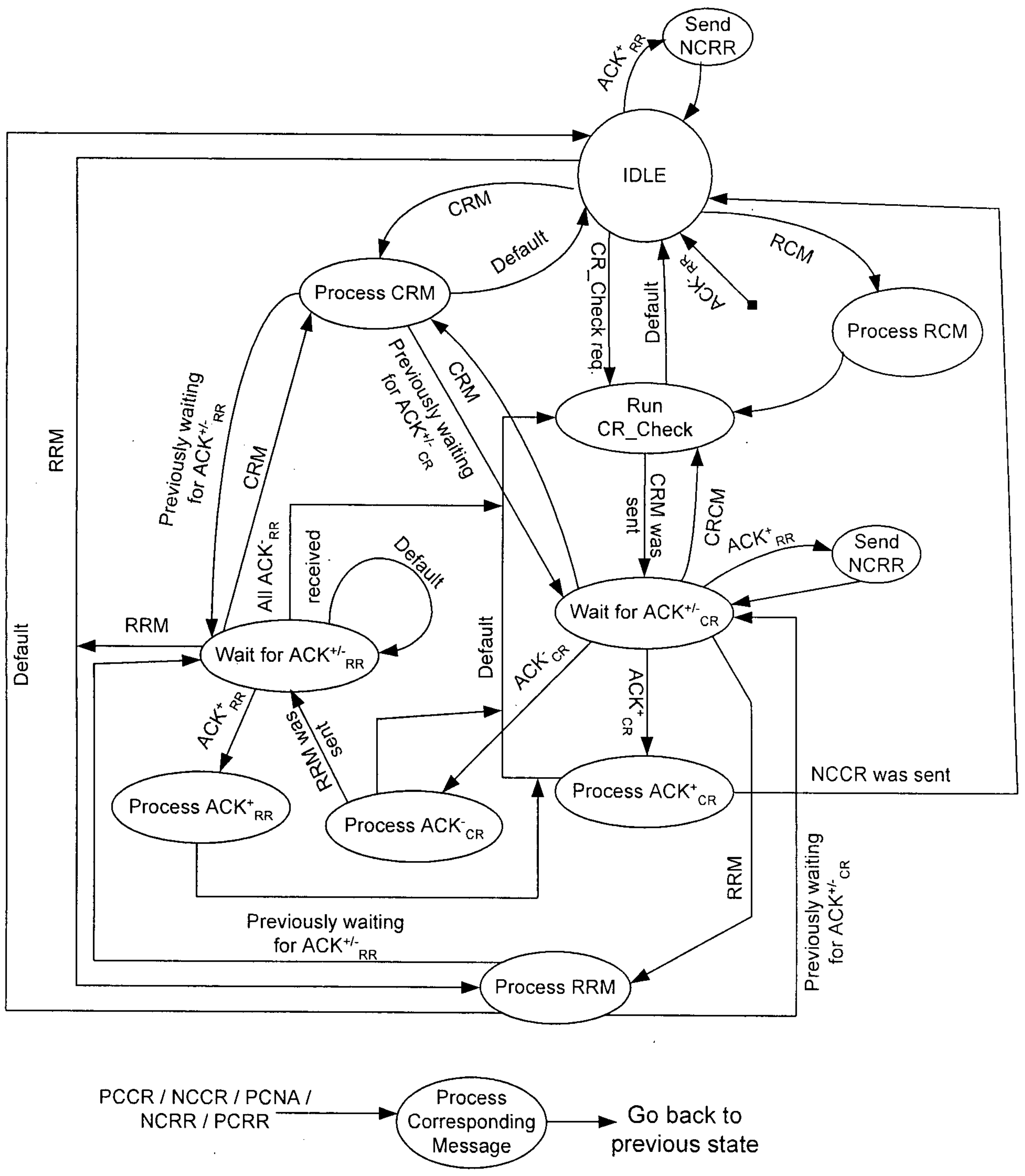


Figure 5.9: State diagram for a node running DDCA.

Table 5.1: Computational complexity of individual sub-algorithms used in the proposed distributed dynamic channel allocation.

Sub Algorithm	**Complexity**
$CR\_Check(n)$	$O(\|E_n\|)$
CRM Processing or *RRM Processing*	$O(\|E_n\| \cdot \|C_s\|)$
Ack^{+}_{CR} *Processing*	$O(\|E_n\|)$
Ack^{-}_{CR} *Processing*	$O(\|E_n\| \cdot \|C_s\|)$
Ack^{+}_{RR} *Processing*	$O(\|C_s\|)$
Ack^{-}_{RR} *Processing* or *RCM Processing* or *CRCM Processing*	$O(\|E_n\|)$
PCCR/NCCR Processing or *PCRR/NCRR Processing* or *PCNA Processing*	Constant time.

5. Algorithm Stability.

6. Convergence.

7. Optimization.

5.2.1 Complexity of Sub Algorithms

Table 5.1 shows the complexity of each individual sub-algorithm used in DDCA algorithm.

The complexity of $CR\_Check(n)$ algorithm is determined from finding the most requested link adjacent to a node n. This link can be found using linear search in $O(|E_n|)$ time, where E_n is the set of links adjacent to n. The other operations in the algorithm can be done within $O(|E_n|)$ as well. In WMN number of links connected to a node are constraint by number of antennas and hence $|E_n|$ is a small number typically in the range

[2, 4]. For CRM or RRM processing (Figures 5.4 and 5.7), complexity came from the *for-loop* at Step 7 of these algorithms. In the worst case, this *for-loop* runs for $O(|E_n|)$ time for each link connected to a node receiving CRM or RRM. Inside each loop, if the condition in Step 8 is true, then it takes $O(|C_s|)$ time to check the condition in Step 9. The worst case complexity is calculated when, $|Ch\_list| = |U_{dst}| = \frac{|C_s|}{2}$ and $Ch\_list \bigcap U_{dst} = \{\}$. In this case every entry in Ch_list has to be checked for its presence in U_{dst}, which requires $O(|E_n| \cdot |C_s|)$ time. Similarly, Ack^{-}_{CR} processing requires $O(|E_n| \cdot |C_s|)$ time to check condition in Step 3 (Figure 5.6), $|E_n|$ times. Ack^{+}_{CR} processing requires checking a single channel on every link; which can be done in $O(|E_n|)$. Ack^{+}_{RR} processing needs $O(|C_s|)$ time to check condition in Step 3 (Figure 5.8). All the algorithms in second last row of Table 5.1 require running $CR\_Check(n)$, hence their time complexity is same as $CR\_Check(n)$. It is also worth noting that the $O(C_s)$ time complexity in most of these algorithms can be reduced to constant time if presence of a channel in a message is represented by binary flags. In such a case the intersection operation can be done on these binary flags in constant time.

5.2.2 Message Complexity

The message complexity for this algorithm is defined as the number of messages sent to allocate a single channel to a link. This can be calculated with the help of Figure 5.2. From Figure 5.2(a) it is clear that the best case message complexity is 3 messages i.e. one CRM, one Ack^{+}_{CR} and one $PCCR$. In order to calculate worst case complexity we have to look at worst case scenario. The worst case scenario is explained with the help of Figure 5.2(c) and (d). In this scenario node x has $|U_x| > 0$; to allocate channels in U_x, x will send CRM on every link in E_x, one by one. The worst case scenario happens when all CRM, except the last one, will be unsuccessful and results in RRM, and Ack^-_{RR}. The last CRM will result in RRM, Ack^+_{RR}, $PCRR$, and $PCNA$. This scenario will generate the maximum number of messages to allocate a single channel that is successful after last CRM.

Now we calculate the number of messages used in this worst case scenario as follows. After receiving Ack^-_{CR} for each unsuccessful CRM, with available replacement channels, node x will send at most $|E_x| - 1$ RRM and will receive $|E_x| - 1$ Ack^-_{RR}. Therefore total number of messages for $|E_x| - 1$ unsuccessful channel requests will be

$$(|E_x| - 1) \times (CRM + Ack^-_{CR} + (|E_x| - 1) \cdot (RRM + Ack^-_{RR})) \tag{5.3}$$

Considering CRM, Ack^-_{CR}, RRM, and Ack^-_{RR} each as a single message. We can rewrite the number of messages for $|E_x| - 1$ unsuccessful channel requests as

$$\begin{aligned} & (|E_x| - 1) \times (2 + 2 \cdot (|E_x| - 1)) \\ = \quad & 2 \cdot ((|E_x| - 1) + (|E_x| - 1)^2) \\ = \quad & 2 \cdot E_x(E_x - 1) \end{aligned} \tag{5.4}$$

Again for worst case scenario the last successful CRM will have following messages

$$CRM + Ack^-_{CR} + (|E_x| - 1) \cdot (RRM + Ack^+_{RR} + NCRR/PCRR) + PCNA \tag{5.5}$$

It is to be noted here that only one PCRR and PCNA will be generated. In terms of number of messages we can also write the above as

$$\begin{aligned} & 3+3(|E_x|-1) \\ = \; & 3|E_x| \end{aligned} \tag{5.6}$$

Combining number of messages in successful and unsuccessful $CRMs$ the total number of messages used in worst case scenario is given as follows

$$2|E_x|^2+|E_x| \tag{5.7}$$

5.2.3 Latency to allocate a single channel

In this thesis the latency to allocate a single channel is defined as the communication time to allocate a single channel to a link. The unit of time T is considered as the time to relay one message from one node to another node. Best case latency is related to best case scenario as shown in Figure 5.2(a). In this scenario, only three messages are passed sequentially hence latency is $3T$ i.e. a constant in asymptotic sense.

The worst case latency depends upon worst case scenario as discussed in Section 5.2.2. In this scenario for each unsuccessful CRM, all RRMs are sent in parallel, hence it is sufficient to include time for only one RRM and its responses. Based on this, worst case latency is $4T$ for a single worst case unsuccessful attempt as shown in Figure 5.2(c). For a worst case successful attempt as shown in Figure 5.2(d), the latency is $5T$; please note that PCRR and PCNA are sent in parallel. Therefore, for a worst case scenario i.e. for

$|E_x| - 1$ unsuccessful CRM and one successful CRM the latency is given as follows

$$Latency_{worst} = 4T \times (|E_x| - 1) + 5T \tag{5.8}$$

$$= (4|E_x| + 1)T \tag{5.9}$$

5.2.4 Algorithm scalability

The proposed algorithm is highly scalable because all the decisions are made locally and to allocate a single channel to a link connected to a node n, messages are passed at most to the extended neighborhood of n. The extended neighborhood of n is the set of nodes that can be reached from n over at most two links. These are nodes that will receive the RCM messages.

5.2.5 Algorithm Stability

To determine if the DDCA algorithm is stable, we first have to consider what can cause the system to be unstable. By instability we mean that following a channel (re)allocation operation. The DDCA algorithm never converges to a stable channel configuration for all links. The two possible causes of instability are

1. If an unsuccessful CRM will continue to be generated by same node.
2. The channel allocation process is triggered very rapidly, meaning that even for very slight changes in link utilization, network-wide channel (re)allocation is performed.

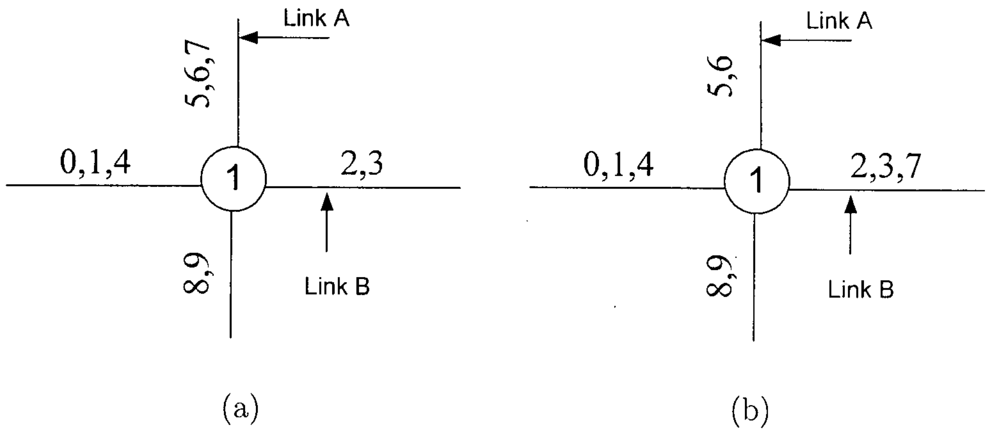


Figure 5.10: Example to understand oscillation. Bandwidth capacity per channel is 100 Mbps. The numbers next to each link correspond to channels allocated to the link.

It is possible that a node has some unused channels that it is unable to allocate to any of its links, in which case unsuccessful *CRM*s are sent indefinitely. To avoid this problem, the proposed algorithm uses a separate flag for each link connected to a node *n*. This flag is set by the *CR_Check* algorithm before sending a *CRM* on a link. If the *CRM* is unsuccessful then only those links whose flags are not set will be tested in future *CR_Check*s. When *CRM*s fail on all the links then *CR_Check* stops sending anymore *CRM*s to avoid instability. Thus, unsuccessful *CRM*s are not sent indefinitely. After detecting that all flags are set, *CR_Check* removes these flags so that, if the process in invoked again based on changes in link capacities then more *CRM*s can be sent. Because after changes in the link capacities, network conditions are not the same and it may be possible to allocate the unused channels again.

The second instability scenario is controlled by the factor δ defined in Condition 2 in Section 5.1. The possible oscillation can be understood from Figure 5.10. Figure 5.10(a) represents the initial condition. Suppose link A has reserved bandwidth of 100 Mbps and link B has reserved bandwidth of 101 Mbps. If δ is not used, then Condition 2 of Section 5.1 will be satisfied and channel allocation process will be invoked to get the allocation in Figure 5.10(b). Now, if link A attempts to reserve only 2 Mbps more resulting in 102 Mbps of reserved bandwidth, then channel allocation will change to that in Figure 5.10(a) again, causing oscillation by very nominal changes in bandwidth reservations. On the other hand by introducing $\delta = 0.1$, the channel allocation process will not be invoked until link B reserves bandwidth of 120 Mbps. Therefore using δ prevents the DDCA algorithm from oscillating during channel allocation. It is to be noted here that value of δ can be configured by a network administrator in the beginning.

5.2.6 Convergence

Lemma 5.1: Under the same network load conditions, DDCA converges to a stable solution.

Proof: To prove that DDCA algorithm converges, it is sufficient to prove that DDCA does not create cycles i.e., DDCA does not exchange a channel between two adjacent links indefinitely under same network load conditions.

For a given node n, with at least two incident links A and B, DDCA removes a channel from a link A and assigns it to a link B, iff

$$\frac{r_A}{c_A \times (|R_A| - 1)} + \delta < \frac{r_B}{c_B \times |R_B|} \tag{5.10}$$

After removing a channel from A and assigning it to B,

$$|R_{A-new}| = |R_A| - 1$$

$$|R_{B-new}| = |R_B| + 1$$

Now, if after this assignment there is no change in the bandwidth reservations in the network, then to prove convergence we need only to prove that B will not give a channel back to A, or formally:

$$\frac{r_B}{c_B \times (|R_{B-new}| - 1)} + \delta \geq \frac{r_A}{c_A \times |R_{A-new}|}$$

By substituting the values of $|R_{A-new}|$ and $|R_{B-new}|$ using the above equations, we get,

$$\frac{r_B}{c_B \times |R_B|} + \delta \geq \frac{r_A}{c_A \times (|R_A - 1)|} \tag{5.11}$$

Considering inequality 5.10 to be true for a positive δ, it is obvious that inequality 5.11 is also true, and, consequently, link B will not give a channel back to link A.

The other possible scenario is when link A gives a channel to another link C after giving a channel to link B. We will use a contradiction proof to show that no loops can occur under this assignment. Assume (by contradiction) that the above channel assignment results in an indefinite channel exchange between A and B. This can occur if and only if the following two inequalities are true simultaneously

$$\frac{r_B}{c_B \times |R_B|} > \frac{r_C}{c_C \times |R_C|} \tag{5.12}$$

and

$$\frac{r_B}{c_B \times (|R_{B-new}| - 1)} + \delta < \frac{r_A}{c_A \times (|R_{A-new}| - 1)} \tag{5.13}$$

We prove that inequalities 5.12 and 5.13 cannot be true simultaneously. The link A gives a channel to the link C after giving a channel to link B iff

$$\frac{r_A}{c_A \times (|R_{A-new}| - 1)} + \delta < \frac{r_C}{c_C \times |R_C|} \tag{5.14}$$

Comparing inequalities 5.12 and 5.14 we get

$$\frac{r_A}{c_A \times (|R_{A-new}| - 1)} + \delta < \frac{r_B}{c_B \times |R_B|}$$

Replacing the value of $|R_B|$ with $|R_{B-new}| - 1$, we get

$$\frac{r_A}{c_A \times (|R_{A-new}| - 1)} + \delta < \frac{r_B}{c_B \times (|R_{B-new}| - 1)} \tag{5.15}$$

Clearly, inequalities 5.13 and 5.15 cannot be true simultaneously and since inequality 5.15 is true only if inequality 5.12 is true then inequalities 5.12 and 5.13 cannot be true simultaneously. Therefore, links A and B do not exchange a channel indefinitely and DDCA converges to a stable channel assignment

5.2.7 Optimization

To avoid bandwidth bottlenecks in a network, it is desirable to reduce maximum link utilization. For the network with same bottleneck link utilization, reducing the average link utilization and standard deviation of link utilizations achieves further load balancing. A network with low average link utilization and low standard deviation of link utilization means that most of the links are not congested. This section provides proofs that DDCA indeed optimizes its objectives.

Lemma 5.2: The DDCA algorithm performs a channel exchange between two adjacent links iff this exchange reduces the maximum link utilization.

Proof: Suppose that a node n has at least two incident links A and B that satisfy the condition in inequality 5.10 and assume without loss of generality that link B has the highest link utilization among all links incident on node n. In this case, the current link utilizations is

$$\mu_A = \frac{r_A}{c_A \times |R_A|}; \qquad \mu_B = \frac{r_B}{c_B \times |R_B|}; \qquad and \qquad \mu_B > \mu_A \tag{5.16}$$

If based on inequality 5.10, DDCA successfully removes a channel from link A and assigns it to link B then the new bandwidth utilizations of A and B will be

$$\mu_{A-new} = \frac{r_A}{c_A \times (|R_A| - 1)}; \qquad and \qquad \mu_{B-new} = \frac{r_B}{c_B \times (|R_B| + 1)}; \tag{5.17}$$

From Equations 5.10, 5.16, and 5.17 it is clear that

$$\mu_A < \mu_{A-new} < \mu_B; \qquad and \qquad \mu_{B-new} < \mu_B \tag{5.18}$$

Since both μ_{A-new} and μ_{B-new} are less than μ_B (which was the maximum among all links incident on node n) then the maximum link utilization is reduced after the channel exchange between links A and B

When multiple links satisfy inequality 5.10, then DDCA chooses the link with maximum link utilization (see Figure 5.3) to assign a channel to the link by sending a CRM on that link. From inequality 5.18 it is clear that after successful channel re-allocation process, all the links involved will have lower utilization than μ_B, which was the maximum link utilization (for node n) before sending a CRM. Since DDCA runs in every node in the network, therefore DDCA minimizes the maximum link utilization.

Lemma 5.3: The DDCA algorithm reduces SD_n for individual nodes except in the case when $\mu_{B-new} < \mu_A < \mu_{A-new} < \mu_B$.

Proof: As soon as it detects that inequality 5.10 is true, the DDCA algorithm sends a CRM to exchange a channel. The links involved in this process are typically link with highest utilization and link with lowest utilization among all corresponding adjacent links. Showing that standard deviation of bandwidth utilization between these two links after channel reallocation is lower, is sufficient to prove that SD_n will be minimized. The standard deviation between two links A and B incident on node n can be calculated as

$$(SD_{A-B})^2 = \left(\mu_A - \frac{\mu_A + \mu_B}{2}\right)^2 + \left(\mu_B - \frac{\mu_A + \mu_B}{2}\right)^2$$

with some simplification, we get

$$(SD_{A-B})^2 = \frac{(\mu_A - \mu_B)^2}{2}$$

Since $\mu_A < \mu_B$ therefore the positive square root will be

$$SD_{A-B} = \frac{\mu_B - \mu_A}{\sqrt{2}} \tag{5.19}$$

Equation 5.19 represents standard deviation before channel reallocation. After channel reallocation there can be at most three possible outcomes

Case 1: If

$$\mu_A < \mu_{A-new} < \mu_{B-new} < \mu_B \tag{5.20}$$

then

$$SD_{A-B-new} = \frac{\mu_{B-new} - \mu_{A-new}}{\sqrt{2}}$$

From inequality 5.20 it is clear that $SD_{A-B-new} < SD_{A-B}$, which means that standard deviation of link utilization is minimized.

Case 2: If

$$\mu_A < \mu_{B-new} < \mu_{A-new} < \mu_B \tag{5.21}$$

then

$$SD_{A-B-new} = \frac{\mu_{A-new} - \mu_{B-new}}{\sqrt{2}}$$

From inequality 5.21 it is clear that again $SD_{A-B-new} < SD_{A-B}$ and the standard deviation of link utilization is minimized.

Case 3: If

$$\mu_{B-new} < \mu_A < \mu_{A-new} < \mu_B \tag{5.22}$$

then

$$SD_{A-B-new} = \frac{\mu_{A-new} - \mu_{B-new}}{\sqrt{2}}$$

This time we cannot say anything about standard deviation, however, from inequality 5.22

we can derive the following

$$
\begin{aligned}
\mu_A &> \mu_{B-new} \\
\mu_B &> \mu_{A-new} \\
\Rightarrow \frac{\mu_A + \mu_B}{2} &> \frac{\mu_{A-new} + \mu_{B-new}}{2}
\end{aligned}
\tag{5.23}
$$

In other words average link utilization is minimized.

In all three cases, it is clear that the DDCA algorithm reduces SD_n for individual nodes except when considerably reducing average link utilization.

Corollary 5.1: The DDCA algorithm minimizes SDR_n for individual nodes for same number of unused channels ($|U_n|$) for a network without bandwidth reservation.

Proof: Whenever DDCA sends a CRM, it selects the link with least number of channels. Therefore, it reduces SDR_n because whenever there are two links whose difference in number of channels equal to 1, DDCA chooses the link with less number of channels to send CRM to reduce this difference.

Corollary 5.2: The DDCA algorithm minimizes $|U_n|$ for individual nodes.

Proof: Whenever DDCA detects a non-empty U_n it initiates channel allocation process to allocate channels in U_n to minimize $|U_n|$.

As further consequence of the above, we can argue that, the DDCA algorithm minimizes the number of rejected future route requests. A route request is rejected when there is no path available to accommodate the required bandwidth. This happens when there exists at least one link in the network with maximum link utilization close or equal to 1. Since DDCA minimizes the maximum link utilization to reduce the number of links with link utilization close or equal to 1, it also minimizes the number of rejected requests by avoiding congestion in any part of the network.

5.3 Integrating Channel Allocation and Route Reservation

The DDCA algorithm provides distributed dynamic channel allocation. However, to provide traffic engineering, it is needed to invoke the channel request initiation check (as explained earlier) at proper instances. One way is to run this check on each node periodically; but a better way is to run this check whenever there is a change in the link reserved bandwidth. In this section, the DDCA is integrated with route reservation. It is assumed in this work that Open Shortest Path First with Traffic Engineering (OSPF-TE) routing protocol is operational in the network [8]. This assures that each node in the network has recent information about the bandwidth reservation and capacity on each link in the WMN. It is also assumed that network is capable of using a resource reservation protocol such as RSVP-TE [7].

Whenever a route request arrives, the widest shortest path (WSP) algorithm [14] is used at the ingress node to calculate the path. A successful request is followed by bandwidth reservation on the calculated path presumably using RSVP-TE. Each node in the path also checks if there is a need to invoke channel request on any link connected to it or not. If there is a need then the node sends corresponding CR message. A node also checks the need to invoke channels (re)allocation when it joins the network or it detects a link failure. Because every node invokes a channel request whenever it joins the network, the above solution provides an initial channel assignment. In the above mechanism, WSP is chosen for routing because it always selects the shortest route and thus reduces the bandwidth wastage. There are several routing algorithms that provide better TE performance than WSP, essentially by selecting longer path than the shortest path to avoid hot spots when the link capacities are fixed [1, 38], but with the flexibility of re-configurable link capacities WSP is sufficient.

5.4 Simulation Results

A discrete time simulator was developed to test the performance of the proposed solution. Two different size networks are used, one smaller lattice type networks with 16 nodes and 24 links, and one larger lattice type network with 100 nodes and 180 links. For each network different test scenarios are used to test the proposed solutions.

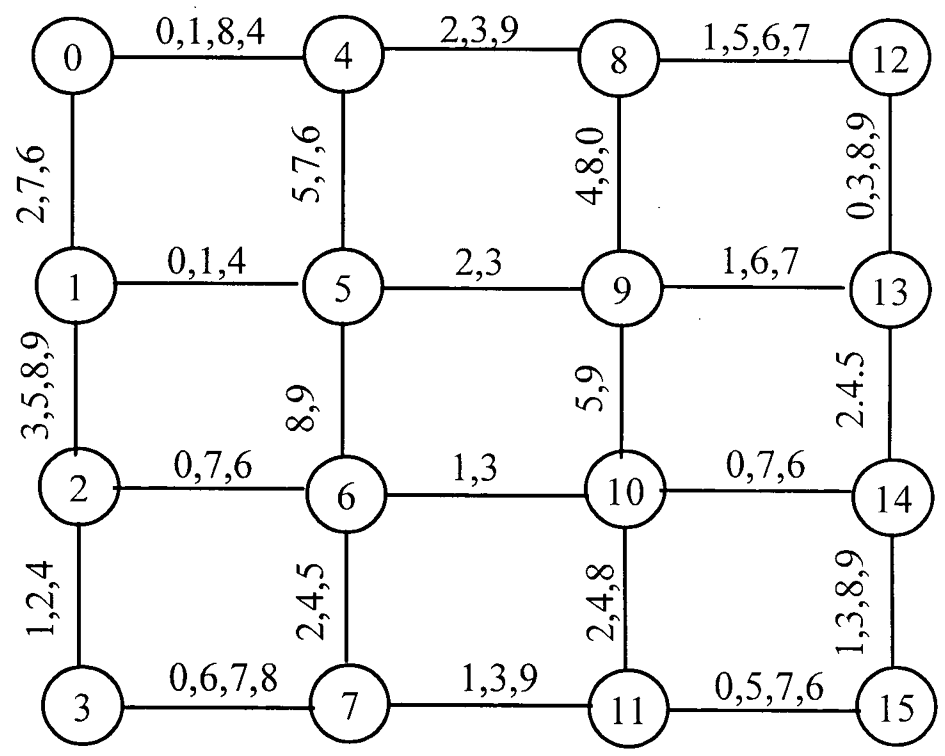


Figure 5.11: Simulation Results, showing correctness of DDCA.

5.4.1 Results for 16 nodes Network

Correctness of DDCA

It is assumed that there are 10 channels in the spectrum. All the nodes are forced to join the network simultaneously for initial channel assignment without routing any traffic demand. The resultant channel allocation is shown in Figure 5.11. It is clear from this solution that the difference between the numbers of channels on any two adjacent links is never more than one. This indicates that the algorithm has successfully balanced the number of channels on each link. Also, only the 4 corner nodes (nodes 0,3,12, and 15 in Figure 5.11) have some unused channels. The reason for these unused channels is that the corner nodes are each connected to two links only and assigning these unused channels to these links may cause removal of channels from some other links. It can also be seen in this solution that there is no interference as no two adjacent links share the same channel, and every link has at least one channel assigned to it.

Performance with Static Route Requests

This scenario uses 200 MHz of Local Multipoint Distribution Service (LMDS) band. Thirty three channels are considered with 6 MHz analog bandwidth per channel. 16QAM encoding is used to provide 24 Mbps digital bandwidth per channel. In this scenario, the performance of the proposed solution with routing is tested. Two different cases are compared. In the **Static** case, channel allocation is performed at the start and remains fixed throughout simulation. Whereas, the **Dynamic** case integrates channel allocation with routing, whereby channel allocation is reconfigured dynamically. Furthermore, it is assumed that route requests arrive one by one and once arrived and routed, they are not removed, and we call these static routes. Also bandwidth demand per request is chosen randomly in the range [0.1-0.4] Mbps. The source and destination are chosen randomly for each request. To provide fair comparison, the random route requests have been generated and saved in a text file. Then these already saved route requests are used in both cases to provide fair comparison. This scenario is tested for 9000 requests. The results are shown in Figures 5.12-5.14.

These results have shown that the proposed TE solution with dynamic channel allocation provides significantly better performance as compared to fixed channel allocation. Because of the proposed dynamic approach, the network was able to route around 900 more routes before it starts rejecting requests (Figure 5.12). The proposed solution also performed better in terms of maximum link utilization and average link utilization in the network (Figure 5.13). Although, the dynamic case average link utilization was high after routing 7000 requests (Figure 5.14), however, it is because the proposed dynamic solution was able to route more requests.

Another measure of testing the effectiveness of the proposed solution is to check how many links have bandwidth utilization above a certain threshold. Figures 5.15-5.17 shows the results for this measure. Results are shown only for route requests from 5000 to 7000, because none of the links has high utilization before 5000 requests have been made. The solution with static channel allocation has started rejecting route requests after

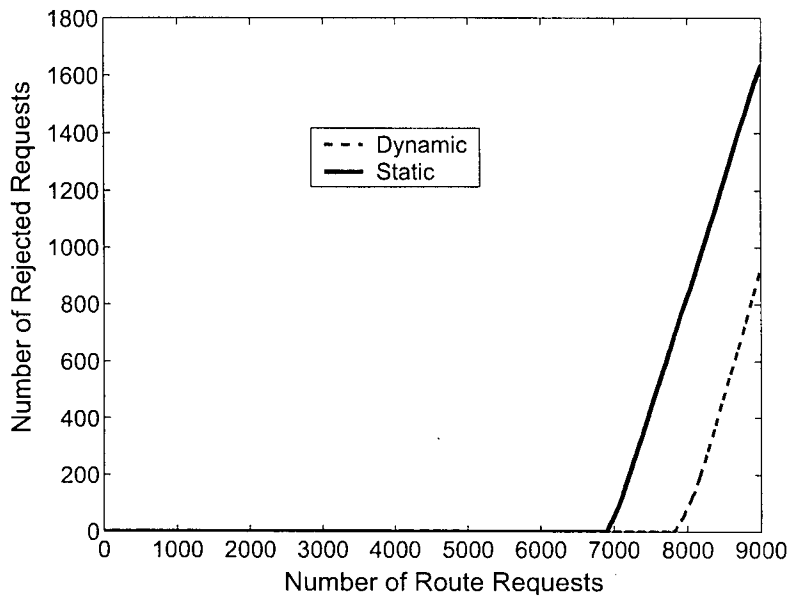


Figure 5.12: Number of rejected requests versus number of route requests, with static routes.

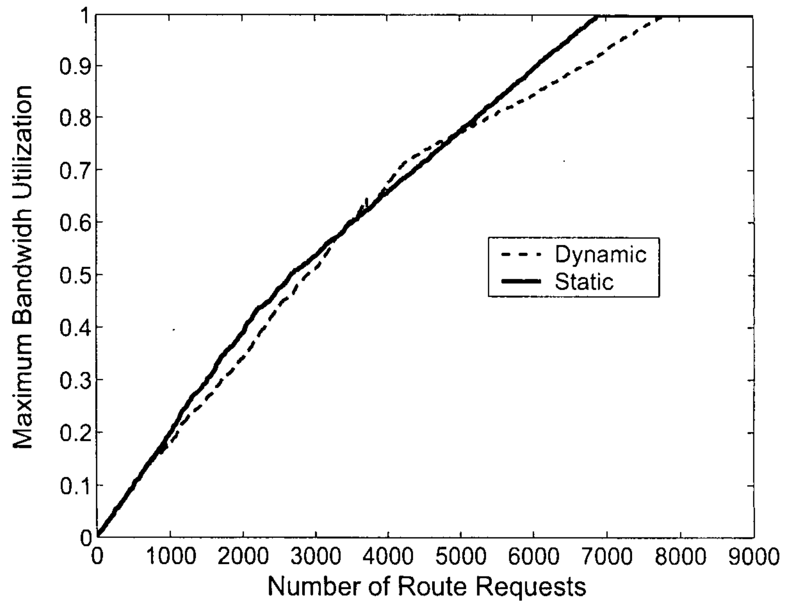


Figure 5.13: Maximum bandwidth utilization over all links versus number of route requests, with static routes.

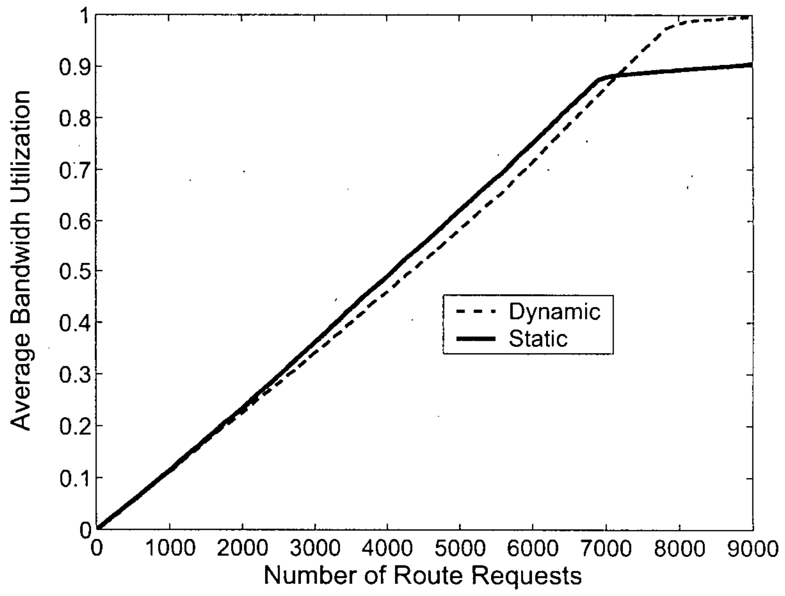


Figure 5.14: Average bandwidth utilization over all links versus number of route requests, with static routes.

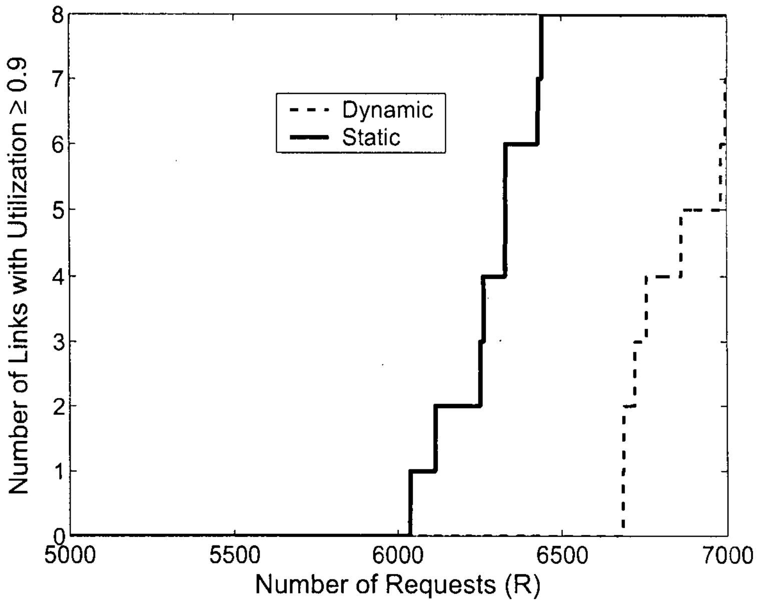


Figure 5.15: Number of links with bandwidth utilization $\geq$ 0.9 versus number of route requests, with static routes.

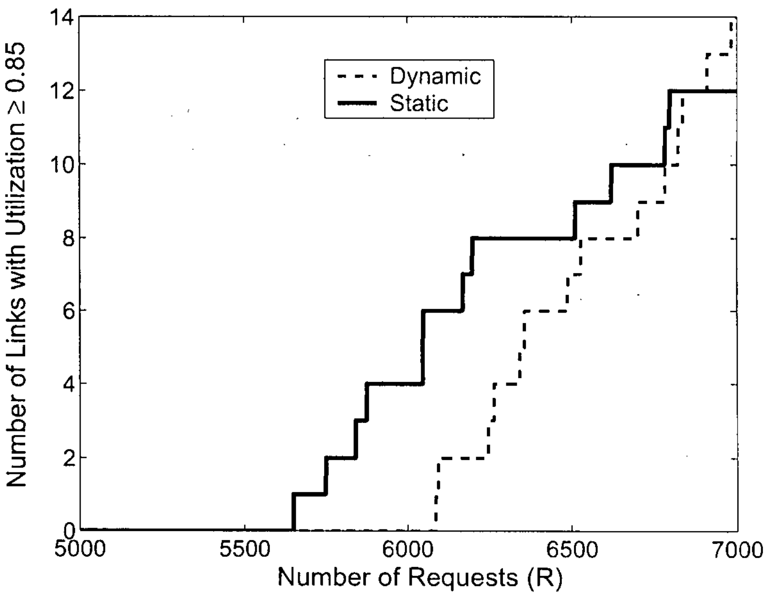


Figure 5.16: Number of links with bandwidth utilization $\geq$ 0.85 versus number of route requests, with static routes.

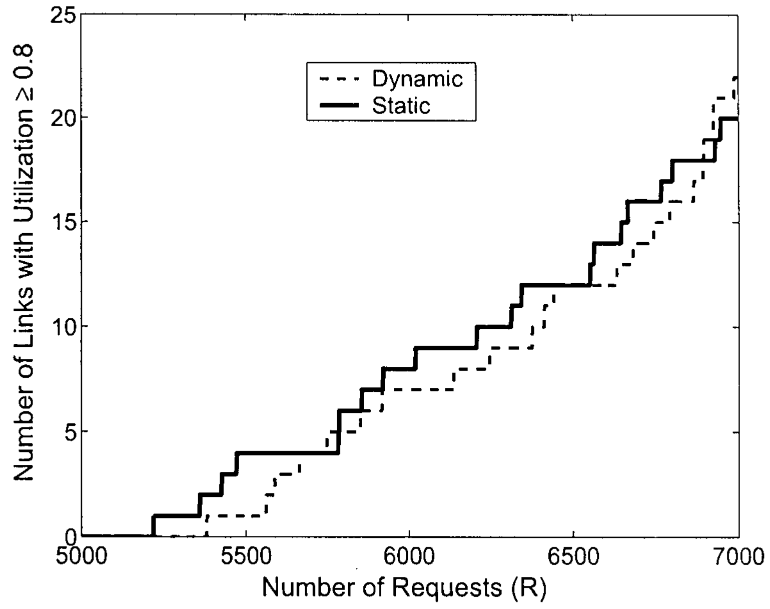


Figure 5.17: Number of links with bandwidth utilization $\geq$ 0.8 versus number of route requests, with static routes.

around 7000 requests (see Figure 5.12). Therefore to provide fair comparison between static and dynamic channel allocation with same accepted route requests the 5000-7000 requests range was chosen. Note that in Figure 5.15 that DDCA with routing was able to decrease the number of highly utilized links. Almost the same behavior is shown in Figures 5.16 and 5.17 but at few instances the proposed solution have slightly more number of links above the corresponding utilization threshold. This happens because the proposed solution tries to balanced the utilization of all links and, therefore, decrease in the utilization of highly utilized links increases the utilization of some other links because of channel reallocation. This behavior can be understood with the fact the two links with utilization of 0.85 each are better than two links with one having utilization of 0.95 and other 0.75.

Performance with Dynamic Route Requests

Every setup in this scenario is the same as in static route request scenario, except that dynamic route requests are also considered. First the network is loaded with 6950 static routes, after that dynamic route requests are issued. Dynamic route requests arrive with exponentially distributed interarrival time with rate λ. If routed then each route keeps the connection for another exponentially distributed hold time with rate μ. The value of $\frac{\lambda}{\mu} = 1000$ have been selected for this scenario. The purpose of this scenario is to test the performance of the proposed solution in more realistic environment, and when network is heavily loaded. The simulation results for this scenario are shown

Table 5.2: Comparison for number of requests routed before first rejection; and when at least one of the link become saturated.

Number of routed requests before	**Static Routes**			**Dynamic Routes**		
	Static	**Dynamic**	**Improvement**	**Static**	**Dynamic**	**Improvement**
First rejected request	6897	7793	11.5%	6897	9231	25.2%
$Max(\mu_l) = 1$ for first time	6901	7799	11.5%	6901	9502	27.3%

in Figures 5.18-5.20. Note that the "Dynamic" legend in Figures 5.18-5.20 indicates dynamic channel allocation and not dynamic route requests. Again it is clear from these results that proposed TE approach using dynamic channel allocation has outperformed the fixed channel allocation approach. Especially in terms of the number of rejected route request(Figure 5.18). Again average bandwidth utilization for the proposed solution is higher after routing 7250 requests (Figure 5.20), because the proposed solution was able to route a larger number of requests compared to fixed channel allocation.

Table 5.2 compares the number of route requests successfully routed before first route request is rejected and when at least one of the link is saturated. It is clear from this comparison that dynamic channel allocation has outperformed static channels allocation for both second and third scenarios.

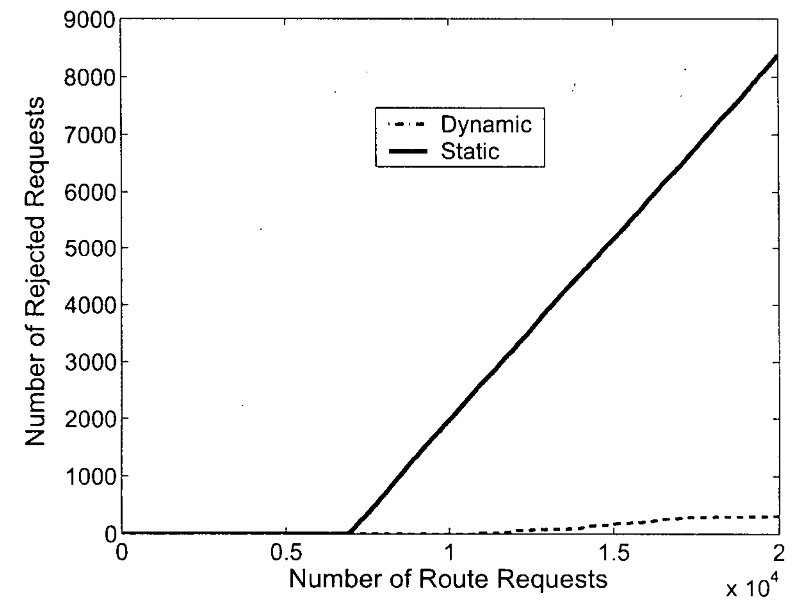


Figure 5.18: Number of rejected requests versus number of route requests, with dynamic routes.

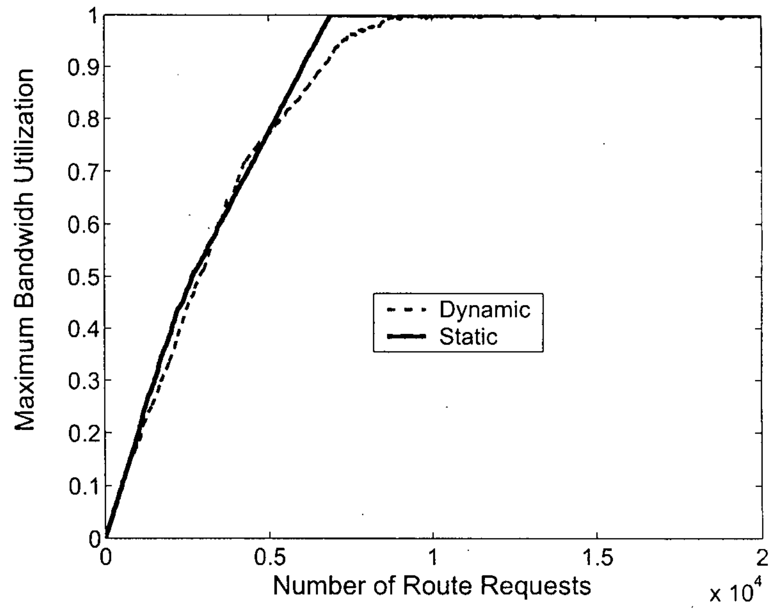


Figure 5.19: Maximum bandwidth utilization over all links versus number of route requests, with dynamic routes.

5.4.2 Results for 100 nodes Network with Gateways

To investigate the consistency, and uniformity of the DDCA algorithm with routing in larger networks, a 100 nodes lattice type network is considered. The structure of the network is the same as for 16 nodes network, except it has larger number of nodes. Another difference for the 100-node network is that only peripheral nodes (gateways) are connected to the external network. All communications must go through these gateways only. Two different scenarios are considered for this networks, as explained next.

In the first scenario 20000 static route requests are used with the same setup as the earlier static route requests scenario for 16 node network, except that route requests are generated between a node and one of the gateways. Some selected results for this scenario are shown in Figures 5.21-5.23. These results show the consistency in the performance of the proposed solution and it has outperformed the solution with static channel allocation approach.

The second scenario for 100 nodes network considered the dynamic route requests. The setup for this scenario is the same as the dynamic route requests scenario for 16 nodes network, except that the network was first loaded with 15000 static requests that are followed by another 25000 dynamic route requests. For dynamic route requests the value $\frac{\lambda}{\mu} = 2000$ has been selected, where λ is the rate for exponentially distributed route requests interarrival time and μ is the rate for exponentially distributed route hold time. The results are shown in Figures 5.24-5.26. Again the results are consistent with the results for smaller network and proposed solution has outperformed the fixed channel

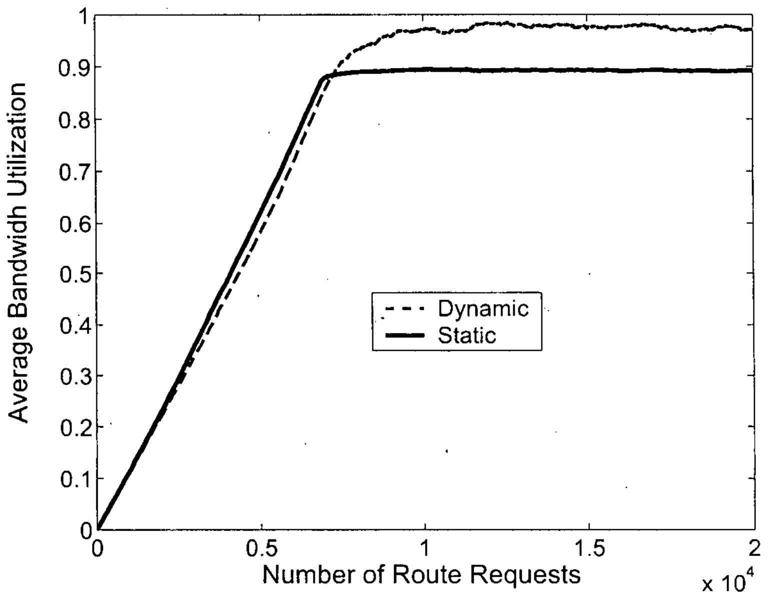


Figure 5.20: Average bandwidth utilization over all links versus number of route requests, with dynamic routes.

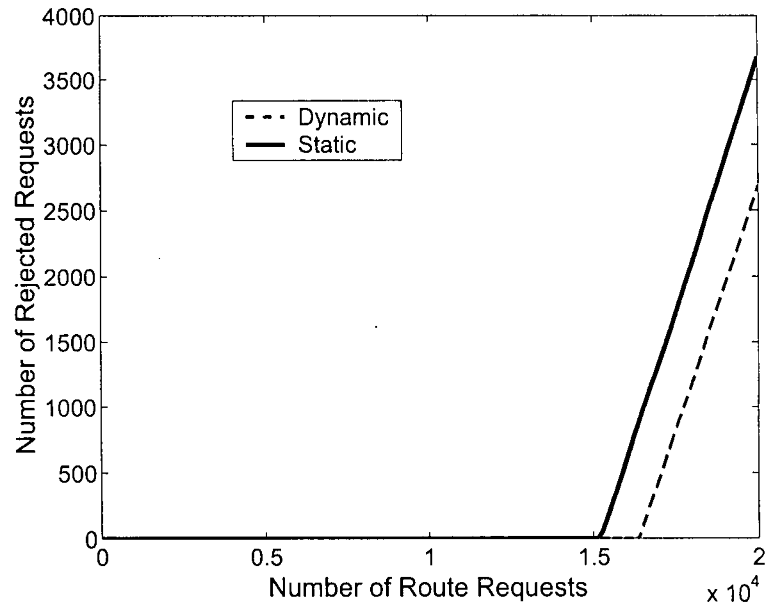


Figure 5.21: Number of rejected requests versus number of route requests, with static routes. Results for 100 node network.

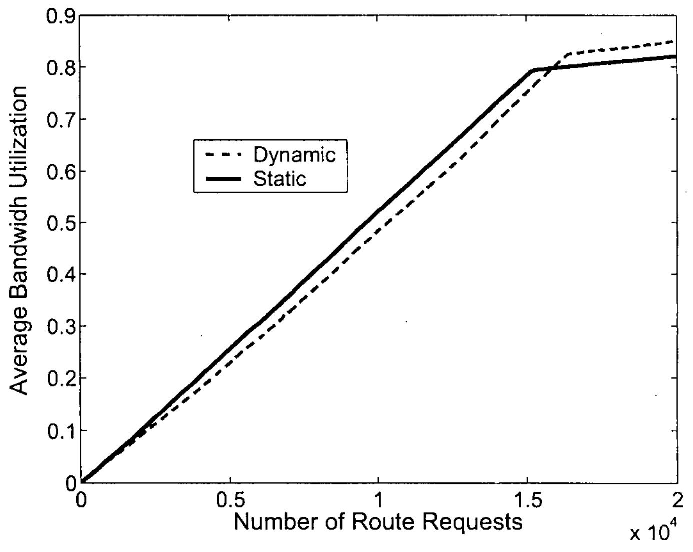


Figure 5.22: Average bandwidth utilization over all links versus number of route requests, with static routes. Results for 100 node network.

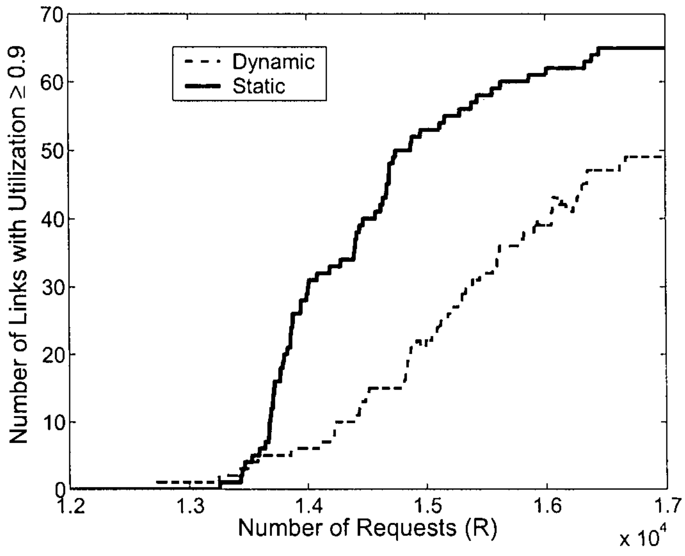


Figure 5.23: Number of links with bandwidth utilization $\geq$ 0.9 versus number of route requests, with static routes. Results for 100 node network.

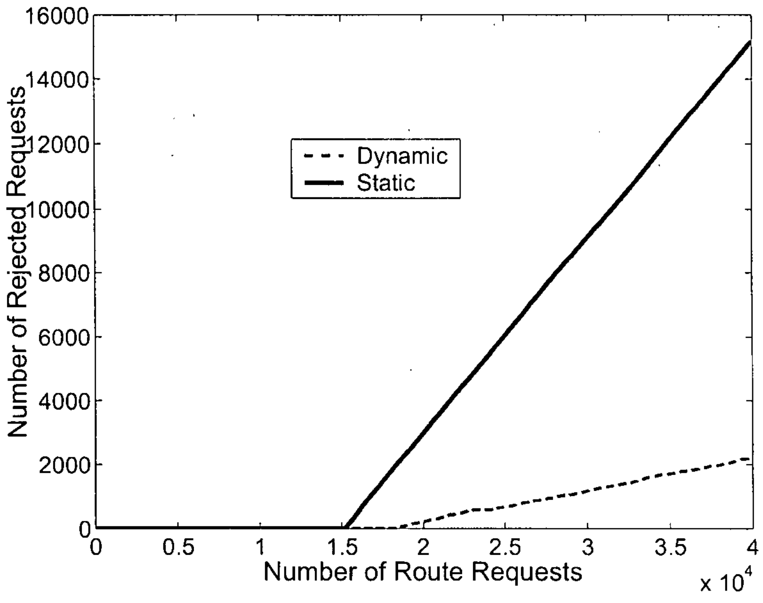


Figure 5.24: Number of rejected requests versus number of route requests, with dynamic routes. Results for 100 node network.

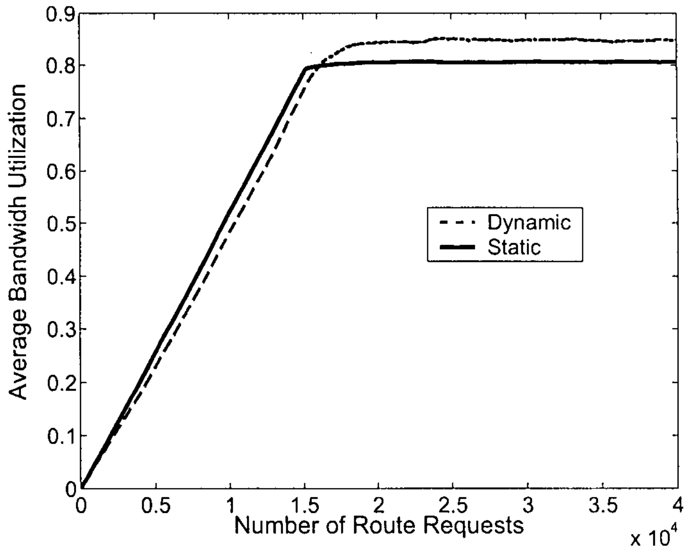


Figure 5.25: Average bandwidth utilization over all links versus number of route requests, with dynamic routes. Results for 100 node network.

allocation solution.

5.4.3 Scalability Test

One way to check the scalability of the proposed solution is to show that average number of messages sent by a node for initial channel assignment is independent of network size. For this purpose we have calculated the average number of messages sent by a node for initial channel allocation. It has been done for 33 and for 10 available channels in the spectrum. The results for this test are shown in Table 5.3. It is clear from this table that algorithm is scalable since the average number of messages per node does not depend upon the size of the network. Indeed, the number of messages per node depends upon the number of channels available in the spectrum which is expected since more messages are required to allocate more channels.

5.5 Summary

This chapter has presented the distributed dynamic channel allocation algorithm. The algorithm is then integrated with routing. The performance results has shown that algorithm performs better than a fixed channel allocation scheme and is highly scalable.

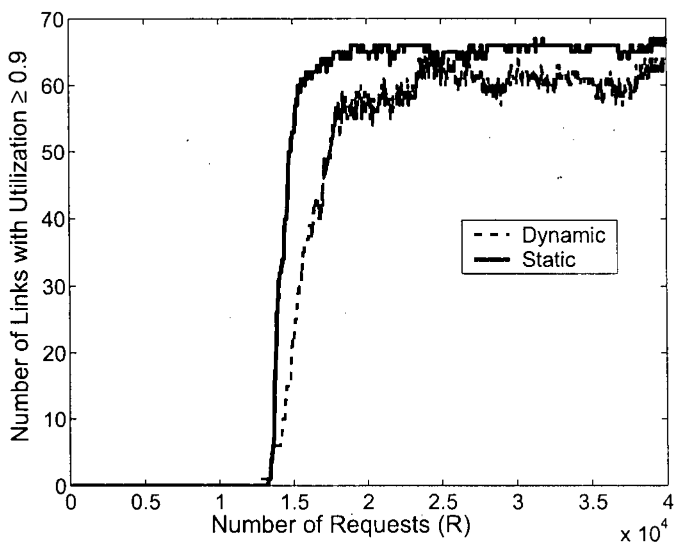


Figure 5.26: Number of links with bandwidth utilization $\geq$ 0.9 versus number of route requests, with dynamic routes. Results for 100 node network.

Table 5.3: Comparison for average number of messages sent by a node for initial channel assignment for different size networks.

Network Size	Average Number of Messages Sent by a Node	
	For 33 Channels	For 10 Channels
16 node & 24 links	78	25
100 node & 180 links	89	31

Chapter 6

Conclusions and Future Work

Supporting Traffic Engineering (TE) is an essential part for an efficient operation of a network. TE provides resource optimization, especially bandwidth resources. This optimization can be achieved either by avoiding congestion by routing traffic in an efficient manner or by eliminating congestion using link capacities reconfiguration.

This thesis has covered both aspects of bandwidth optimization of an operational network. In wired networks where it is not possible to reconfigure link capacities with network upgrading, explicit paths are used with efficient fuzzy routing algorithm to provide TE. In wireless networks, the thesis has covered TE in broadband fixed wireless networks with directed (physical) mesh topologies. In such wireless systems, it is possible to reconfigure link capacities by reconfiguring the frequency channel allocation, hence eliminating congestion.

6.1 Recommendations for TE in Wired Networks

The thesis has proposed a new fuzzy routing algorithm (FRA) to compute online TE efficient routes for unicast flows. It is assumed that MPLS is operational to provide support for explicit paths and OSPF-TE is also available to distribute network wide dynamic link information throughout the network. The proposed solution suggests optimizing three performance objective i.e. minimizing maximum link utilization, minimizing the utilization of all the links, and also minimizing number of hops for the route. The first objective is needed to avoid congestion based on single congested link. However, it is possible that several paths are available with same maximum link utilization. In such a scenario, second objective provides a decision criteria to choose the best path. The last objective is to avoid extra network resources, because choosing a longer than minimum hop path will use extra network resources.

The thesis uses fuzzy logic to optimize these three performance objectives simultaneously. A fuzzy rule, in linguistic terms, is specified to aggregate three objectives into one objective. Then fuzzy membership function for each individual objective is proposed. These individual membership functions are then aggregated in a fuzzy t-norm operator (namely AND like OWA operator) based on the fuzzy rule proposed earlier. This aggregated membership function is finally used in a Dijkstra's like algorithm to compute the TE efficient routes.

The proposed fuzzy routing algorithm (FRA) has been compared with well-known Widest Shortest Path (WSP) and Minimum Interference Routing Algorithm (MIRA) in extensive simulation to deduce following conclusion.

- FRA is less computationally expensive than MIRA and hence more suitable for online routing. Because of low computational complexity, FRA can support large number of route requests in a short interval of time.

- Unlike MIRA, FRA does not need a route server and hence it is more scalable, cannot be effected by server failure and does not need communication between server and ingress for each and every route computation. Furthermore, FRA does not need information about ingress/egress pairs in the system.

- FRA is able to provide load balancing based on bottle neck link as well as all other links. In comparison MIRA does not provide good load balancing and WSP provides load balancing based on only bottleneck link.

- FRA rejects less route requests without needing the ingress/egress information that is needed in MIRA for the same performance in terms of number of rejected route requests.

- It is also observed that FRA provides paths with better QoS than MIRA and WSP. FRA does this by selecting a path that has low utilization in most of the links. On the other hand, MIRA does not look at link utilization, and WSP optimizes a route based on a single bottleneck link.

- Furthermore, FRA does not need any extra information that is not needed by MIRA or WSP. Like WSP and MIRA, FRA needs a connection oriented service e.g. MPLS and a TE based link state protocol such as OSPF-TE.

In its current formulation FRA is capable of computing routes for unicast route requests. However, it is possible to extend the work using fuzzy logic for multicast route requests as well. Multicast is useful in many applications such as virtual private networking (VPN) etc. Almost the same approach as in FRA can be used for multicast route requests. The only difference is that instead of using modified Dijkstra's algorithm, a steiner tree based algorithm will be needed. Furthermore, the steiner tree algorithm has to be modified to accommodate fuzzy membership functions instead of any additive link cost.

It is worth to note that FRA has the ability to include multiple objectives in routing decisions. This property of FRA can be utilized to include more performance objectives, such as dropping probability, and in case of wireless links, the channel conditions. Another application of interest is the calculation of backup paths. Backup paths are needed in the case of a link failure to reroute traffic. Managing bandwidth resource in the network while providing backup paths is another problem that can be included in the future work.

6.2 Recommendations for TE in Wireless Networks

In broadband fixed wireless networks, managing analog frequency channels while optimizing network performance is more critical. Because, to provide QoS to their customers, service providers have to use licensed frequency bands which have ongoing monitory costs. This thesis provides a TE solution by exploiting the fact that frequency channels in a wireless network can be reallocated based on dynamic network conditions to optimize the use of this expensive resource. The thesis covers a specific type of wireless network, namely, physical (or directed) mesh network using millimeter wave band for broadband fixed wireless systems. The network consists of nodes with narrow beam directional (steerable or array) antennas. The directional pattern minimizes the interference. Each link in the network consists of several radio channels. By changing the number of channels on each link, it is possible to eliminate congestion in certain part of the network.

Based on the flexibility of changing link capacities in the network under consideration, the thesis purposes a dynamic distributed channel allocation algorithm. The proposed dynamic distributed channel allocation provides TE by monitoring the dynamic network conditions and changing link capacities by reallocating channels dynamically based on message passing. The proposed channel allocation algorithm is integrated with routing as well. The proposed solution is compared with solution based on fixed channel allocation using a discreet time simulator. Following conclusions are made based on algorithm properties and its simulation.

- The algorithm can provide solution from scratch or fine tunes a solution based on dynamic network conditions making it a dynamic solution.
- The algorithm is highly distributed because no node needs network wide information to make a decision. This results in highly scalable network architecture.
- The scalability is demonstrated by simulating channel allocation for two different size networks. Both networks generated almost the same average number of mes-

sages per node for similar number of channels in the spectrum.

- Simulation results have shown that algorithm is capable of finding interference free channel allocation by not allocating same channels to links connected to a same node.
- For a network without reservation, the channel allocation is balanced in the sense that links connected to a same node has almost the same number of channels.
- For a network with reserved bandwidth, it is observed that proposed distributed dynamic solution was able to route more flow requests than solution based on fixed channel allocation. This provides more revenue to a service provider by using the same spectrum.
- The dynamic solution also found to be better in terms of maximum and average utilization of a link.
- It is observed that number of links with higher utilization are considerably low in dynamic channel allocation providing better QoS to individual flows.
- Simulation results demonstrated similar behavior for tests with static route requests or dynamic route requests. The behavior is also similar for different network sizes.

The proposed solution covers the basic approach to solve the TE problem in the directed mesh networks. However, there are still improvements that can be done. One such improvement is to consider that interference is not only between the links connected to the same node but it can be from other links as well. One solution to minimize this problem is to tilt the directional antennas slightly toward the earth. This will avoid the signal to propagate farther from the desired node and hence avoid unwanted interference. However, it is still possible to have interference from the link not directly connected to a node. One future work recommendation is to improve the proposed channel allocation algorithm to avoid this kind of interference. One straight forward solution is that whenever a node n detects interference then it can remove the channel from its list U_n of available channels.

The proposed solution considers the use of OSPF-TE for calculating routes using WSP. However, since proposed dynamic channel allocation needs no information from OSPF-TE, therefore, another future work recommendation is to test the performance without considering OSPF-TE and WSP. This may degrade the performance but a trade off can be made between performance and the overhead by OSPF-TE.

Another improvement can be done by using a modified fuzzy logic routing algorithm (FRA) to consider physical channel conditions. This is needed because it is possible that some links are using lower QAM, in such a case it is better to avoid routing through these links. However, in such a case it is also worth to modify the channel allocation algorithm so that it can be integrated with FRA. It is to be noted here that in all cases, modification in distributed dynamic channel allocation algorithm will be needed only in evaluating the conditions presented in the algorithm description and not in the basic structure of the algorithm.

The proposed solution targeted a specific type of network, however it is possible to extend the approach to other network types as well. One such network is IEEE 802.11a that contains 12 non-overlapping channels. There are several algorithms available to allocate channels to base stations in PMP networks, but most of the algorithm considers interference free operation between two adjacent base stations. It is worth to note that IEEE 802.11 standards does not prevent an access point from using multiple channels simultaneously. Presence of CSMA/CA in IEEE 802.11 standard can provide extra advantage. With CSMA/CA it is possible that two IEEE 802.11a access points share the same channel. Therefore in the scenario where each access point can use multiple channels, it is possible to optimize network operation by adding or removing channels from access points dynamically. While removing a channel from one access point and adding it to another adjacent access point, it is possible that network performance does not improve a lot. However, if the channel is shared then it may give better optimization based on number of users connected to each access points. Several considerations has to be taken into account, such as, channel reallocation should not be done in a short interval of time because it may cause users to scan for channels again. Furthermore, a criteria has to be developed that decides that it is better to share channels between adjacent access points or not.

Bibliography

[1] K. Kar, M. Kodialam, and T. V. Lakshman. Minimum interference routing of bandwidth guaranteed tunnels with MPLS traffic engineering application. *IEEE Journal on Selected Areas in Communications*, 18:2566–2579, December 2000.

[2] S. Suri, M. Waldvogel, D. Bauer, and P. R. Warkhede. Profile-based routing and traffic engineering. *Computer Communications*, 26:351–365, 2003.

[3] S. Suri, M. Waldvogel, and P. R. Warkhede. Profile-based routing: A new framework for MPLS traffic engineering. *In Proceedings of 2nd International Workshop on Quality of Future Internet Services*, pages 138–157, September 2001.

[4] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. Overview and Principles of Internet Traffic Engineering. *IETF Network Working Group, RFC 3272*, May 2002.

[5] E. Rosen and A. Viswanathan. RFC 3031: Multiprotocol Label Switching Architecture. *Internet Engineering Task Force (IETF)*, January 2001.

[6] B. Jamoussi et al. Constraint-Based LSP Setup using LDP. *IETF Network Working Group, RFC 3212*, January 2002.

[7] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow. RSVP-TE: extenstions to RSVP for LSP tunnels. *IETF Network Working Group, RFC 3209*, November 2001.

[8] H. M. Alnuweiri, L-Y K. Wong, and T. Al-Khasib. Performance of new link-state advertisement mechanism in routing protocols with traffic engineering extensions. *IEEE Communication Magazine*, 42:151–162, May 2004.

[9] J. Moy. OSPF Version 2. *IETF Network Working Group, RFC 2178*, April 1998.

[10] D. Bersekas and R. Gallager. *Data Networks.* Prectice Hall, New Jersey, 2nd edition, 1992.

[11] J. F. Kurose and K. W. Ross. *Computer Networks.* Addison Wesley, New York, 2nd edition, 2003.

[12] D. Katz, K. Kompella, and D. Yeung. Traffic Engineering (TE) Extensions to OSPF Version 2. *IETF Network Working Group, RFC 2370*, September 2003.

[13] P. Fafali, C. Patrikakis, A. Michalas, and V. Loumos. Internet traffic engineering: History monitoring information featuring routing algorithms. *In Proceedings of International Conference on Automation and Information*, pages 87–90, October 2003.

[14] R. Guerin, A. Orda, and D. Williams. QoS routing mechanism and OSPF extensions. *In Proceedings of IEEE Global Telecommunication Conference (GLOBCOM)*, 3:1903–1908, November 1997.

[15] Z. Wang and J. Crowcroft. Quality of service routing for supporting multimedia applications. *IEEE Journal on Selected Areas in Communications*, 14:1288–1234, September 1996.

[16] K. Kar, M. Kodialam, and T. V. Lakshman. MPLS traffic engineering using enhanced minimum interference routing: An approach based on lexicographic max-flow. *In Proceedings of* 8^{th} *IEEE International Workshop on Quality of Service (IWQoS)*, pages 105–114, June 2000.

[17] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated service networks: The multiple node case. *IEEE/ACM Transactions on Networking*, 2(2):137–150, 1994.

[18] S. M. Sait and H. Youssef. *Iterative Computer Algorithms with Applications in Engineering, Solving Combinatorial Optimization Problems.* IEEE Computer Society, 1999.

[19] E. Rosen, A. Vishwanathan, and R. Callon. Multiprotocol label switching architecture. *IETF Network Group, RFC 3031*, January 2001.

[20] L. Andersson, P. Doolan, N. Feldman, A. Fredette, and B. Thomas. LDP specification. *IETF Network Working Group, RFC 3036*, January 2001.

[21] A. Capone, L. Fratta, and F. Martignon. Virtual Flow Deviation: Dynamic routing of bandwidth guaranteed connections. *In Proceedings of the* 2^{nd} *International Workshop on QoS in Multiservice IP Networks (QoS-IP)* , pages 592–605, 2003.

[22] E. Aboelela and C. Douligeris. Fuzzy reasoning approach for QoS routing in B-ISDN. *Journal of Intelligent and Fuzzy Systems, Application in Engineering and Technology*, 9:11–27, November 2000.

[23] S. M. Sait, H. Youssef, and J. A. Khan. Fuzzy evolutionary algorithm for VLSI placement. *In Proceedings of Genetic and Evolutionary Computation Conference (GECCO)*, pages 1056–1063, July 2001.

[24] L. A. Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on Systems, Man. and Cybernetics*, 3(1):28–44, 1973.

[25] R. R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Transactions on systems, Man. and Cybernetics* , 18:183–190, January 1988.

[26] I. Tardy and O. Grondalen. On the Role of Future High Frequency BFWA Systems in Broadband Communication Networks. *IEEE Communication Magazine*, 43:138–144, February 2005.

[27] D. Sweeney. *WiMax Operator's Manual Building 802.16 Wireless Networks.* APRESS, Berkeley, CA 94710, USA, 2004.

[28] W. Webb. Broadband Fixed Wireless Access as a Key Component of the Future Integrated Communication Environment. *IEEE Communication Magazine*, 39:115–120, September 2001.

[29] Y. Kishi, S. Konishi, S. Nanba, and S. Nomoto. A proposal of millimeter-wave multi-hop mesh wireless network architecture with adaptive network control features for broadband wireless access. *In Proceedings of IEEE Radio And Wireless Conference, RAWCON*, pages 17–20, August 2001.

[30] D. Beyer. Wireless mesh networks for residential broadband. *In Proceedings of National Wireless Engineering Conference*, November 2002.

[31] P. Piggin, B. Lewis, and P. Whitehead. Mesh networks in fixed broadband wireless access, multipoint enhancements for the 802.16 standard. *http://www.ieee802.org/16/docs/.*

[32] H. R. Anderson. *Fixed broadband wireless system design.* John Wiley & Sons, 2003.

[33] P. Whitehead. Mesh Networks: A new architecture for broadband wireless access system. *In Proceedings of IEEE Radio And Wireless Conference RAWCON*, pages 43–46, Septemeber 2000.

[34] B. Shrick and M. J. Riezenman. Wireless broadband in a box. *IEEE Spectrum*, 39:38–43, June 2002.

[35] T. Fowler. Mesh networks for broadband access. *IEEE Review*, 47:17–22, January 2001.

[36] Y. Kishi, K. Tabata, T. Kitahara, Y. Noishiki, A. Idoue, and S. Nomoto. Implementation of the Integrated Network and Link Control Functions for Multi-hop Mesh Networks in Broadband Fixed Wireless Access Systems. *In Proceedings of IEEE Radio and Wireless Conference RAWCON*, pages 43–46, September 2004.

[37] Y. Kishi, K. Tabata, T. Kitahara, Y. Noishiki, A. Idoue, and S. Nomoto. Wireless Node Architecture and Its Implementation for Multi-Hop Mesh Networks in IP-Based Broadband Fixed Wireless Access. *IEICE Transaction on Communication*, E88-B(3):1202–1210, March 2005.

[38] J. A. Khan and H. M. Alnuweiri. A fuzzy constraint-based routing algorithm for traffic engineering. *In Proceedings of IEEE Global Telecommunication Conference (GLOBCOM)* , 3:1366–1372, November 2004.

[39] S. Konishi, S. Nanba, Y. Kishi, and S. Nomoto. Dynamic and autonomous frequency assignment method for MP-MP BFWA systems. *In Proceedings of IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, 1:359–363, September 2002.

[40] *IEEE Standard for Local and metropolitan area networks Part 16: Air Interface for Fixed Broadband Wireless Access Systems.* Institute of Electrical and Electronics Engineers, Inc., New York, 2004.

[41] Y. Kishi, T. K., S. Konishi, and S. Nomoto. A proposal of an adaptive channel allocation and traffic engineering algorithm in multi-hop mesh networks for broadband fixed wireless access. *In Proceedings of IEEE Wireless Communication and Networking Conference, WCNC*, 2:1043–1048, March 2003.

[42] A. Boukerche, S. Hong, and T. Jacob. A distributed algorithm for dynamic channel allocation. *ACM Mobile Networks and Applications*, 7:115–126, April 2002.

[43] K. Leung and Byoung-Jo "J" Kim. Frequency Assignment for Multi-Cell IEEE 802.11 Wireless Networks. *In Proceedings of IEEE Vehicular Technology Conference*, pages 1422–1426, October 2003.

[44] M. Papatriantafilou, D. Rutter, and P. Tsigas. Distributed Frequency Allocation Algorithms for Cellular Networks: Trade-offs and tuning strategies. *In Proceedings of IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS 2001)*, pages 339–344, 2001.