State Dependent Multicast Routing
for Single Rate Loss Networks

A thesis submitted to
The Hong Kong University of Science and Technology
in partial fulfillment of the requirements for
the Degree of Doctor of Philosophy in
Electrical and Electronic Engineering

By

Chi-Chung CHEUNG

M.Phil.(ELEC), City University of Hong Kong, 1994

August, 1998
State Dependent Multicast Routing
for Single Rate Loss Networks

by

Chi-Chung Cheung

Approved by:

[Signature]
Dr. Danny H. K. Tsang
Thesis Supervisor

[Signature]
Prof. Madeline Wu
Thesis Examination Committee Member (Chairman)

[Signature]
Dr. C. T. Lea
Thesis Examination Committee Member (EEE)
Dr. T. M. Ko
Thesis Examination Committee Member (EEE)

Dr. J. Gu
Thesis Examination Committee Member (CS)

Dr. Philip C. H. Chan
Head of Department

Department of Electrical and Electronic Engineering
The Hong Kong University of Science and Technology

August, 1998
To the memory of my aunt, TSZ Karn-Hing
ACKNOWLEDGMENT

I would like to thank my supervisor, Dr. Danny H. K. Tsang, for his guidance, enthusiasm, inspiration, encouragement, and support during my studies and research. His helpful insights into problems helped me use several methods for tackling problems. I appreciate that he was able to guide my research, even though he was responsible for guiding many students with diverse research interests. I am thankful for his friendly approach to my questions and enthusiasm for a discussion even in his busy schedule. I would also like to thank the Department of Electrical and Electronic Engineering at Hong Kong University of Science and Technology and the Hongkong Telecom Institute of Information Technology for their financial supports.

I gratefully acknowledge the valuable evaluation of this thesis by my examining committee, Dr. J. Gu, Dr. K. T. Ko, Dr. T. M. Ko, Dr. C. T. Leard and Dr. Danny H. K. Tsang.

I would like to thank Dr. Sanjay Gupta, who has also given many valuable advice on my research direction. I would like to take this opportunity to express my gratitude to Dr. H. W. Chu, my fellow postgraduate student at Hong Kong University of Science and Technology, for giving his valuable suggestions and discussions on my research.
ABSTRACT

Given the popularity of multicast services and applications, we study the problem of state dependent multicast call routing for single rate loss networks. We formulate the problem and discuss the difficulties to implement the optimum solution. Because of these difficulties, a heuristic approach is proposed: we apply Minimum Spanning Tree (MST) searching with a suitable link cost function to search a connected tree for a multicast call connection request such that its normalized revenue loss can be reduced. If there is no connected tree available, the call connection request will be rejected.

We develop analytical models of Least Load Multicast Routing (LLMR), which is a well-known multicast routing algorithm, for fully connected networks. The analytical models that we develop for calculating blocking probabilities are based on the Reduced Load Approximation (RLA) with the link independence assumption. For symmetrical networks, our analytical models include both state aggregation, i.e. ALLMR, and alternative routing for point-to-point calls. The agreements between the simulation and analytical results in both scenarios are very good and we find that the agreements are not affected significantly by the link independence assumption.

Four new link cost functions are proposed to improve the network performance in different ways: Aggregated Least Load Multicast Routing (ALLMR) gives a simpler implementation, lower signaling traffic for establishing connection requests and lower sensitivity to the design parameters, compared with LLMR. Least Load Multicast Routing with Maximum Occupied Circuits (LLMR-MOC) and Least Load Multicast Routing with Minimum Measured Blocking Time (LLMR-MMBT) are modified LLMR algorithms which have moderate improvement with minimum additional cost, compared with LLMR. We also dis-
cuss their implementation issues. The Maximum Mean Number of New Calls Accepted Before Blocking Multicast Routing (MCBMR) algorithm can more accurately capture the current and future loading of a network. Simulation results show that this algorithm, compared with LLMR, not only has a smaller network revenue loss, but also results in smaller call blocking probabilities for all classes of traffic. We also discuss the implementation issues of this proposed algorithm and develop two approximation methods, state approximation and curve fitting, which can reduce the measurement complexity significantly with only a slight performance degradation.
PUBLICATIONS

Some preliminary results and parts of this thesis have been published and have appeared in conference proceedings while some submitted papers are still under review.


# Table of Contents

## 1 Introduction
1.1 Definition of a Single Rate Loss Network .......................... 2
1.2 Routing for Point-to-point Communication .......................... 2
1.3 Previous Work .................................................................. 4
1.4 Reduced Load Approximation (RLA) ................................. 6
1.5 Organization of the Thesis .............................................. 7

## 2 Problem Formulation ......................................................... 10
2.1 System Model .................................................................. 10
2.2 Performance Measures .................................................... 11
2.3 Global Optimization Problem and its Limitation ................. 12
2.4 Minimum Spanning Tree (MST) searching .......................... 14

## 3 Performance Analysis of LLMR Algorithm ......................... 18
3.1 Least Load Multicast Routing (LLMR) algorithm ................. 18
3.2 System Model and Symbolic Notations ............................. 20
3.3 Performance Modeling ..................................................... 21
3.4 Numerical Results ......................................................... 26
3.5 LLMR for Symmetrical Networks .................................... 29
3.5.1 LLMR with No Alternative Nodes ............................... 30
3.5.2 LLMR with An Alternative Node Allowed for Class 2 only 35
3.5.3 Evaluation of Performance Metrics ............................. 38
3.5.4 Numerical Results ..................................................... 39
4 Modified Least Load Multicast Routing (MLLMR) 42

4.1 Aggregated Least Load Multicast Routing (ALLMR) 42

4.1.1 The System Model and The Algorithm 44

4.1.2 Numerical Results 45

4.1.3 Analytical Models of ALLMR for Symmetrical Networks 48

4.2 Least Load Multicast Routing with the Consideration of Secondary Parameters 53

4.2.1 The Network Model and The Proposed Routing Algorithm 54

4.2.2 Illustrative Example 57

4.2.3 The Implementation Issues 58

4.2.4 Numerical Results 60

5 Maximum Mean Number of New Calls Accepted Before Blocking Multicast Routing (MCBMR) 64

5.1 The Mean Number of New Calls Accepted Before Blocking (MCB) 64

5.1.1 The Network Model and The Proposed Routing Algorithm 66

5.1.2 Illustrative Example 68

5.2 The Implementation Issues and Approximation Methods 70

5.2.1 Indirect Measurement 70

5.2.2 Approximation Method 1: State Approximation 73

5.2.3 Approximation Method 2: Curve Fitting 74

5.3 Numerical Results 75

5.3.1 Performance of MCBMR Algorithm 76

5.3.2 Performance of the Two Approximation Methods 77

6 Conclusions and Future Work 86

6.1 Conclusions 86

6.2 Future Works 89
List of Figures

1.1 The generic loss network ........................................ 2

2.1 Example to illustrate the heuristic approach without alternative nodes .................................................. 15

2.2 Example to illustrate the heuristic approach with an alternative node .................................................. 15

3.1 Example to illustrate LLMR ....................................... 19

3.2 A 3-node network .................................................. 23

3.3 Call blocking probability versus network loading ........ 28

3.4 % error in mean call blocking probability of class 4 versus the ratio of the loading of unicast connections to that of multicast connections .................................................. 28

3.5 Call blocking probability versus network loading (N = 30, C = 20, D = 4, and no alternative node is allowed) ............ 31

3.6 Call blocking probability versus network loading (N = 30, C = 20, D = 4, trunk reservation = 4 free circuits and an alternative node is allowed for class 2) ........................................ 31

4.1 The state dependent arrival rate versus link state (occupied circuits) .................................................. 33

4.2 The state occupancy distribution versus link state (occupied circuits) .................................................. 33
4.3 MLLMR algorithms ........................................ 56
4.4 Example to illustrate our proposed algorithm without alternative nodes ........................................ 58
4.5 Mean call blocking probabilities of MLLMR and LLMR (no alternative node is allowed) .............. 62
4.6 Mean call blocking probabilities of MLLMR and LLMR (at most one alternative node is allowed) .................. 62
4.7 Relative improvement of revenue loss of MLLMR over LLMR (no alternative node is allowed) ........................................ 63
4.8 Relative improvement of revenue loss of MLLMR over LLMR (at most one alternative node is allowed) ........................................ 63
5.1 The MCBMR algorithm ........................................ 68
5.2 Example to illustrate our proposed algorithm without alternative nodes ........................................ 69
5.3 Example to illustrate our proposed algorithm with an alternative node ........................................ 70
5.4 General M/M/N/N queueing model ........................................ 71
5.5 Mean call blocking probabilities of MCBMR and LLMR (no alternative node is allowed) .............. 78
5.6 Mean call blocking probabilities of MCBMR and LLMR (at most one alternative node is allowed) ........................................ 79
5.7 Relative improvement of revenue loss of MCBMR over LLMR (no alternative node is allowed) .............. 79
5.8 Relative improvement of revenue loss of MCBMR over LLMR (at most one alternative node is allowed) ........................................ 80
5.9 Normalized revenue loss vs. time (at most one alternative node is allowed) ........................................ 80
List of Tables

3.1 $s(n, k, l)$ as a function of $n$ and $k$. 

3.2 $q(n, k)$ as a function of $n$ and $k$. 

4.1 The effect of the number of aggregated states, $K$, on mean call blocking probabilities (10% overload, no alternative nodes are allowed). 

4.2 The effect of the network loading on mean call blocking probabilities comparing ALLMR ($K = 8$) to LLMR ($K = C(e) + 1$) (no alternative nodes are allowed). 

4.3 The effect of the number of aggregated states, $K$, on mean call blocking probabilities (10% overload, trunk reservation = 5, at most one alternative node is allowed). 

4.4 The effect of the network loading on mean call blocking probabilities comparing ALLMR ($K = 8$) to LLMR ($K = C(e) + 1$) (trunk reservation = 5, at most one alternative node is allowed). 

4.5 The effect of number of aggregated states ($N = 30$, $C = 20$, $D = 4$, no alternative node is allowed and $L = 0.9$) (Note that the number inside the braces represents the actual number of free circuits). 

33

34

46

47

47

48

52
4.6 The effect of number of aggregated states \((N = 30, \ C = 20, \ D = 4)\), an alternative node is allowed for class 2, trunk reservation = 4 free circuits and \(L = 0.9\) (Note that the number inside the braces represents the actual number of free circuits) 52

5.1 The link information of two links 65

5.2 The effect of the number of measured data points on the performance of the two approximation methods (10% overload and no alternative nodes are allowed) 82

5.3 The effect of the number of measured data points on the performance of the two approximation methods (10% overload, at most one alternative node is allowed and trunk reservation is 5 circuits) 83

5.4 The effect of overloading on the performance of the two approximation methods (the number of measured data points is 6 and no alternative nodes are allowed) 84

5.5 The effect of overloading on the performance of the two approximation methods (the number of measured data points is 6, at most one alternative node is allowed and trunk reservation is 5 circuits) 85
Chapter 1

Introduction

The performance of a network depends on many factors such as network configuration, offered load, and network management methods. An important element of network management called network routing consists of the decision rules used to connect the calls when they arrive at the network. A variety of methods are now possible. In the past, most of the research work has been focused on routing for point-to-point communication. However, current trends in networking applications indicate that there will be an increasing demand in future networks for multicasting, which refers to the ability of a set of more than two nodes or end-users in a communication network to communicate simultaneously with each other. Given the popularity of multicast end-user services and applications, our research interests are focused on the multicast routing for single rate loss networks.

In this chapter, we define the network of our research interest: a single rate loss network, in Section 1.1. Section 1.2 describes some important routing schemes applied in point-to-point communication. Previous research work in multicast communication are discussed in Section 1.3. Section 1.4 presents the concept of Reduced Load Approximation (RLA) which will be applied in our analytical model of Least Load Multicast Routing (LLMR). The organization of the rest of the thesis is provided in Section 1.5.
1.1 Definition of a Single Rate Loss Network

A loss network is a collection of resources to which calls, each with an associated holding time and class, arrive at random instances (see Figure 1.1) [31]. An arriving call either is admitted into the system or is blocked and lost; if the call is admitted, it remains in the system for the duration of its holding time. The admission decision is based on the call’s class and the system’s state. Note that class here refers to the size of destination sets or the number of nodes involved in a call connection request. A single rate loss network is a communication network in which each connection request requires that a unit capacity is reserved on each link that it traverses. Examples of single rate loss networks include telephone networks and homogeneous VP-based ATM networks [15].

![Figure 1.1: The generic loss network](image)

1.2 Routing for Point-to-point Communication

For hierarchical architectures used in some circuit switched networks, routes are selected based on the following alternative routing principle: first a direct route is selected; if it is unavailable, the first alternative route is tried; if this is not available, the second alternative route is tried next; and so on. Finally,
a final route is tried. If the final route is busy, the call is blocked. The main disadvantage of hierarchical routing is the inefficiency in selecting an available route. In large public networks, the time to search for an available route may range from a few seconds to more than ten seconds.

Because of the introduction of electronic switches such as Stored Program Control (SPC), routing schemes have dramatically been changed from hierarchical to nonhierarchical routing. The first implementation was introduced in 1984 in the AT&T’s long distance telephone network in the United States, and it was called Dynamic NonHierarchical Routing (DNHR) [2]. DNHR is a time-dependent routing method, which selects routes using different sequences of paths for different times of day. The 24-hour day is divided into 15 Load Set Periods (LSPs). For each LSP, routes consisting of two link paths for all node pairs are calculated and stored in switches. Typically, 14 alternate routes for each node pair are calculated and stored. A primary reason for introducing nonhierarchical routing was its cost effectiveness. Studies indicate that DNHR based methods can provide approximately a 15% cost reduction over hierarchical based routing methods [4].

Dynamic Call Routing (DCR), which is a state-dependent routing method, was developed by Northern Telecom for the Canadian national and local telephone networks [6]. The idea of DCR is as follows: route selection is based on paths with the largest number of free trunks. The network processor periodically computes the alternative routes for each source and destination pair. Calls blocked on the direct route choose an alternative route at random. If a call is blocked from all alternative routes, it is rejected.

There are other popular routing methods such as Dynamic Alternative Routing (DAR) developed by British Telecom (BT) and State- and Time-dependent routing (STR) developed by Nippon Telephone and Telegraph (NTT). We would
like to refer the reader to [34] for some of the popular routing methods.

1.3 Previous Work

Applications that require multicast capability will be important in the near future. Typical applications where multicasting is critical include the following:

1. Wire services used by new agencies, where a news agency such as the Associated Press needs to distribute news reports from their bureaus to newspaper and radio stations throughout the world;

2. Multi-party video conferencing, where every member needs to send his audio/video to all the other members simultaneously;

3. Remote lecturing, where a single speaker can address a large number of audiences; and

4. the recently developed collaborative system, where any change made to the document must be sent to all the people working together; and

Recent research efforts in the area of multicast routing have followed the following heuristic approach: the underlying network is modeled as a graph (directed or undirected) with links as edges and switches as nodes; and each edge is assigned a “cost” (for instance the cost could be a measure of expected delay). The multicast routing problem is then reduced to finding a “tree” that spans the nodes that wish to participate in a given multicast connection that has minimal cost [21] [10] [35] [37] [38]. The above problem is known in the literature as the Steiner tree problem (an NP-complete problem [24]) for which several heuristics have been proposed.

The above studies develop static routing policies, i.e., the routes to be used for communication between a set of destination nodes remain fixed. Therefore,
these routing schemes cannot exploit the inherent statistical nature of network traffic to achieve higher network utilization or reduce connection blocking probability. Moreover, given that the above studies assume that routing is done on a packet by packet basis limits their applicability to networks where routing is done on a per circuit or virtual circuit basis. Additionally, since no appropriate analytical techniques exist, simulations have been employed to evaluate the performance of routing algorithms. The lack of analytical tools for performance evaluation makes network dimensioning and planning difficult.

Given the limitations of the above studies, we consider the problem of developing routing policies for multicast connections that do not decide routes to be used beforehand, but take into account the state of the network at the instant a connection request arrives to assign a route. Such routing schemes are popularly referred to as state dependent routing policies in the literature. State dependent\(^1\) routing of calls (point-to-point) has long been regarded in the telephone industry as a means of increasing call throughput and robustness in the telephone network. We would like to refer the reader to [4] [20] [3] [25] [28] [7] for some of the popular state dependent routing algorithms. The objective of routing policies is to either maximize the network revenue or minimize the network revenue loss, which will be defined in Chapter 2.

Some previous research work in the area of dynamic multicast routing schemes in single rate loss networks are reported in [17] and [8] as Least Load Multicast Routing (LLMR). The LLMR algorithm is to find a connected tree for the multicast call by applying minimum spanning tree searching with a link cost function defined as the negative value of the number of free circuits, i.e. the current residual capacity. Note that finding the minimum spanning tree is equivalent to finding a multicast connected tree with maximum number of free circuits. By

\(^1\)State refers to a set of variables that describe completely the current status of the network.
considering the loading of links involved, LLMR chooses the links with the least loading to establish the connection.

### 1.4 Reduced Load Approximation (RLA)

Because the exact method to compute call blocking probability of a loss network is very difficult to develop, we are therefore compelled to consider alternative approaches for evaluating its performance. The alternative approach studied in this section is Reduced Load Approximation (RLA). This approximation assumes that blocking is independent from link to link, giving rise to a set of fixed-point equations whose solution supplies approximations for blocking.

Consider a loss network supporting only single-rate calls — that is, a network with bandwidth requirement of 1 unit for all classes. With \( L_j \) denoting the approximate probability that link \( j \) is full, we thus have

\[
L_j = ER \left[ \sum_{k \in \mathcal{K}_j} \rho_k t_k(j), C_j \right]
\]  

(1.1)

where \( \mathcal{K}_j \) is the set of classes that use link \( j \), \( \rho_k \) is the class-\( k \) offered load, \( t_k(j) \) is the probability that there is at least one bandwidth unit available in each link along the path of class \( k \) calls excluding link \( j \), \( C_j \) is the capacity of link \( j \), and

\[
ER[\rho, C] = \frac{\rho^C / C!}{\sum_{c=0}^{\infty} \rho^c / c!}.
\]  

(1.2)

If we further assume that blocking is independent from link to link, we obtain

\[
t_k(j) = \prod_{i \in R_k \setminus \{j\}} (1 - L_i)
\]  

(1.3)

where \( R_k \) is the route of class-\( k \) calls, \( R_k \subseteq \{1, \ldots, J\} \), and \( J \) is the total number of links in a network. The above two expressions combined give the following fixed-point equation satisfied by the approximate link blocking probabilities, \( L_1, \ldots, L_J \):

\[
L_j = ER \left[ \sum_{k \in \mathcal{K}_j} \rho_k \prod_{i \in R_k \setminus \{j\}} (1 - L_i), C_j \right], j = 1, \ldots, J.
\]  

(1.4)
Once again invoking the link independence assumption, we have the following approximation for class-$k$ blocking probability:

$$B_k \approx 1 - \prod_{j \in R_k} (1 - L_j), \ k = 1, \ldots, K, \quad (1.5)$$

where $K$ is the total number of classes. Equations (1.4) and (1.5) constitute the reduce load approximation.

For some specific network topologies, the product form solution has been employed to develop efficient combinatorial algorithms to calculate the call blocking probability for all classes. Kaufman [22] and Roberts [30] have developed an algorithm to handle a single-link network with multirate connections. Ross and Tsang have developed algorithms to handle tree [36] and hierarchical tree [32] topologies with multirate connections. It appears difficult, however, to develop efficient combinatorial algorithms for more general topologies due to the complicated nature of the state space.

Mitra [26] has developed an asymptotic expansion of the normalization constant for (single rate) tree networks; however, no progress has been reported in this direction for more general topologies. Harvey and Hills [16] have considered Monte Carlo techniques exploit the product-form solution and can handle arbitrary topologies; however, convergence of the call blocking probabilities is typically slow [16].

In our research, we apply the reduced load approximation to develop an analytical model of Least Load Multicast Routing (LLMR), which has been described in the last section.

### 1.5 Organization of the Thesis

The thesis is organized into several chapters, where each chapter addresses a different problem. Chapter summaries are presented at the beginning of each
chapter. At the end of each chapter, conclusions and suggestions for further research related to that chapter are presented.

Chapter 2 presents the problem formulation of the state dependent multicast routing for single rate loss networks. The optimal solution of the problem is NP-complete and the computational complexity of an optimum solution is unreasonably high. Then a heuristic approach to solve the above problem is discussed with a small example.

Chapter 3 presents an analytical models of Least Load Multicast Routing (LLMR), which is the well-known multicast routing algorithm, for fully connected networks. The analytical models that we develop for calculating blocking probabilities are based on the Reduced Load Approximation (RLA) with the link independence assumption. For symmetrical networks, our analytical models include both state aggregation, i.e. ALLMR, and alternative routing for point-to-point communication. The agreements of simulation and analytical results in both scenarios are observed to be good and we find that the agreements are not affected significantly by the link independence assumption.

Chapter 4 presents three modified Least Load Multicast Routing (LLMR) algorithms which improve the performance of LLMR in terms of achieving different objectives: Aggregated Least Load Multicast Routing (ALLMR), which lumps several states into one aggregated state, can reduce the implementation complexity with only a slight performance degradation; Least Load Multicast Routing with Maximum Occupied Circuits (LLMRMOC) and Least Load Multicast Routing with Minimum Measured Blocking Time (LLMRMMBT), which consider secondary information to make a routing decision, not only produce a smaller network revenue loss, but also result in smaller call blocking probabilities for all classes of traffic. The moderate gain in the network performance comes only with a slight additional cost. Simulation results will be given to show
the performance of the three modified Least Load Multicast Routing (LLMR) algorithms.

Chapter 5 presents a new dynamic multicast routing algorithm called Maximum Mean Number of New Calls Accepted Before Blocking (MCBMR) multicast routing. The goal is to search for a connected tree with maximum mean number of new calls that can be accepted before blocking. Through simulations, we found that MCBMR always outperformed LLMR in all cases, in terms of achieving smaller network revenue loss. We also develop two approximation methods, state approximation and curve fitting, which can reduce the measurement complexity significantly with only a slight performance degradation.

Chapter 6 gives a brief conclusion for all our contribution and summarizes the direction of the future work which are mentioned in the earlier chapters.
Chapter 2

Problem Formulation

In this chapter, we will define the system model of our research interest in Section 2.1. Section 2.2 will introduce the performance measures of multicast routing for single rate loss networks. The global optimization problem and its limitation will be presented in Section 2.3. Based on these limitations, Section 2.4 will present a heuristic approach to handle the routing problem.

2.1 System Model

Multicast connection requests can be classified as Multipoint-to-Multipoint (MTM) or Point-to-Multipoint (PTM). In PTM connections, a single node transmits and the other nodes listen. However, in MTM connections, all nodes that participate in the connection are allowed to transmit information to all others, i.e. each member receives a copy of what each other member is sending. For the case where a network transports only MTM connections the underlying graph can be assumed to be undirected. However, in the presence of PTM connections, the underlying graph has to be assumed to be directed due to the asymmetric nature of the traffic. In view of the above observation, in this thesis, for notational simplicity we assume that all connection requests are MTM.

Before proceeding further, we would like to introduce necessary definitions and notations. Consider a single rate loss network to be an undirected graph
\( G = (V, E) \) where \( V \) and \( E \) are the set of nodes and links respectively. Consider a connection request \( c \) with destination set \( S(c) \) (each node \( s \in S(c) \) is referred to as a destination of \( c \)). Let \( d(c) \) be the number of nodes involved in the connection request \( c \), i.e., the size of the destination set. A connection \( c \) requires a connected, acyclic graph (tree) \( T(c) = (V(c), E(c)) \), where \( S(c) \subseteq V(c) \) is chosen and a unit bandwidth is reserved on each link \( e \in E(c) \). Depending on the routing algorithm, a connection may be either accepted or rejected. We assume that a connection \( c \) carried on the network produces \( r(c) \) units of revenue. From another point of view, the network losses \( r(c) \) units of revenue for each connection \( c \) that is rejected.

### 2.2 Performance Measures

We now introduce the parameter to measure the performance of a network in multicast routing. For point-to-point communication, a major performance measure in circuit switched networks is call blocking probability. Both end-to-end blocking and total network blocking are of interest. However, there are more than one class of call connection requests in multicast routing (a class here means a group of multicast call connections with the same destination size) and thus the above measure cannot be directly applied. In the past, performance of multicast routing algorithms has been evaluated by comparing the costs of the multicast trees found by the multicast routing algorithms to the that of the "optimum solution", i.e. the minimum cost of all possible multicast trees. However, it is inconsistent with the performance metric of traditional circuit-switched networks. Thus, to evaluate the performance of multicast routing algorithms, we use a performance measure called normalized revenue loss, which was proposed in [8]. Denote by \( \lambda_{d} \) and \( B(d) \), the call arrival rate of class \( d \) and the call blocking probability of class \( d \), respectively. By considering the revenue as \( r(c) \)
network resources, we set the corresponding revenue of a call connection request \( c \) with destination set of size \( d(c) \) to \( d(c) - 1 \) (i.e. the revenue is equal to the number of direct links used) and thus the expression of the normalized revenue loss is

\[
\frac{\sum_{d=2}^{D} (d - 1) \lambda_d B(d)}{\sum_{d=2}^{D} (d - 1) \lambda_d},
\]

(2.1)

where \( D \) is the maximum destination size of multicast call connections.

### 2.3 Global Optimization Problem and its Limitation

Once the performance metric is defined, the optimal routing problem can be formulated as follows:

**Given** the network topology, traffic matrix, and the different network state,

**Find** the optimal routing algorithm that maximizes the network revenue, or minimizes the normalized network revenue loss,

\[
E(\text{loss}) = \frac{\sum_{d=2}^{D} (d - 1) \lambda_d B(d)}{\sum_{d=2}^{D} (d - 1) \lambda_d},
\]

(2.2)

Three points need to be clarified. (i) We are interested in dynamic multicast call routing algorithms, i.e., those algorithms that use network state information to make routing decisions. We assume that global network state information is available whenever it is needed by the routing algorithms. (ii) Minimizing the normalized revenue loss is equivalent to maximizing the expected revenues produced by the network. (iii) The multicast call routing algorithm can be described as follows: when a multicast call connection request arrives, it will be either accepted or rejected; if it is accepted, a connected tree, \( T(c) = (V(c), E(c)) \), where \( S(c) \subseteq V(c) \) is chosen and a unit bandwidth is reserved on each link \( e \in E(c) \);
otherwise, the call is rejected and it will not retry. Note that a call connection request can be rejected even if there is enough network resources to carry it.

An optimum multicast call routing algorithm is as follows: When a multicast call connection request arrives, if accepting it will increase the normalized revenue loss in long term, the request will be rejected; otherwise, we accept it and choose a connected tee such that the tree will include all destination nodes of the call connection request and the normalized revenue loss in long term can be reduced.

An obvious approach to perform an optimum routing algorithm is to find an appropriate state descriptor for the network and model the evolution of the network state as a Markov process (after making suitable simplifying assumptions). Steady state performance metrics of interest can be obtained from the solution of the above Markov chain. The only problem with the above procedure, which is conceptually simple, is that it is computationally impossible to perform. Consider the case of a symmetrical network with 10 links in which each link can carry a maximum of 9 connections. Assume that the state descriptor used is the occupancy of each link (note that the above state descriptor does not uniquely specify the state of the network if connections traverse more than a single link). In this unreasonably small example, with an incomplete state descriptor, the size of the state space is \(10^{10}\), large enough to render the above mentioned approach useless.

A common heuristic approach to solve the problem is Steiner tree searching, which has been mentioned in Chapter 1. The approach is described as follow: the underlying network is modeled as a graph (directed or undirected) with links as edges and switches as node; and each edge is assigned a “cost” (for instance the cost could be a measure of expected delay). The multicast routing problem is then reduced to finding a “tree” that spans the nodes that wish to
participate in a given multicast connection and has minimal cost [21] [10] [35] [37] [38]. However, there are some problems to use the above approach. First, Steiner tree searching is known in the literature as an NP-complete problem [21] and thus it is computationally impossible. Moreover, there is no existing link cost function to exactly optimize our objective function based on this searching method.

### 2.4 Minimum Spanning Tree (MST) searching

Based on the above limitations, the following heuristic approach is employed: when a multicast connection request arrives, only the links that connect to the destination nodes will be considered. We assign a link cost function to each link and search for a minimum spanning tree for the connection request. If a minimum spanning tree is available, we accept the request and route the call using the tree; otherwise, we will allow one alternative node and restart the tree searching procedure. If a minimum spanning tree is still not available, the call request will be rejected; otherwise, we accept the request and route the call using the tree.

To illustrate the routing algorithm outlined above, we consider a 4-node network and a connection request with destination set \{r, a, b, c\} with node r as the source node. The steps in the connection establishment procedure are as shown in Figure 2.1 — the number next to a link denotes its link cost and links that are included in the set \(E(c)\) are drawn with “thick” lines. Node c is the first one to be added as link \((r, c)\) has a link cost of 4 that is smaller than the link costs of links \((r, a)\) and \((r, b)\) which is followed by node a (link \((c, a)\) is chosen due to its smaller link cost over links \((r, a)\), \((r, b)\) and \((c, b)\)). Finally, we choose link \((a, b)\) because the link cost of \((a, b)\) is the smallest compared with links \((r, b)\) and \((b, c)\). After iteration 4, the procedure is completed and a tree is
established for the connection request.

Figure 2.2 shows the operation of selecting an alternative node. Given that the link cost of \((r, a)\) is \(\infty\), i.e., this link cannot be used, two available two-link alternative routes are considered to connect nodes \(r\) and \(a\): \((r, b, a)\) and \((r, c, a)\). The cost of \((r, b, a)\) is 7, which is larger than that of \((r, c, a)\) (its cost is 5). Thus we choose node \(c\) as the alternative node and \((r, c, a)\) as the two-link alternative route.

![Diagram](image)

Figure 2.1: Example to illustrate the heuristic approach without alternative nodes

![Diagram](image)

Figure 2.2: Example to illustrate the heuristic approach with an alternative node
One of the major advantages to employ the above approach is its reasonable searching time, compared with Steiner tree searching. For fully connected network, consider a connection request with destination set of size \( d \). For direct routing (no alternative nodes are allowed), the total number of comparisons need for Minimum Spanning Tree (MST) searching is \( O(d^2 \log_2 d) \) (note that the number of comparisons needed equals to those needed to sort \( d(d - 1)/2 \) numbers). While, for alternative routing (at most one alternative node is allowed), it is \( O(Nd^2 \log_2 d) \), where \( N \) is the total number of nodes of a network.

Another major advantage is that we can reduce the global optimization problem into a much smaller problem of how to select a good link cost function to minimize the normalized revenue loss. In our research, we focus on selecting a suitable link cost function to optimize the performance of the multicast routing algorithm.

Some previous research work in the area of dynamic multicast routing schemes in single rate loss networks are reported in [17] and [8] as Least Load Multicast Routing (LLMR). The LLMR algorithm is to find a connected tree for the multicast call by applying minimum spanning tree searching with a link cost function defined as the negative value of the number of free circuits, i.e. the current residual capacity. Note that finding the minimum spanning tree is equivalent to finding a multicast connected tree with maximum number of free circuits. By considering the loading of links involved, LLMR chooses the links with the least loading to establish the connection.

There are two main contribution in our research. First, we develop analytical models of Least Load Multicast Routing (LLMR) for fully connected networks and symmetrical networks. Second, we propose four new link cost functions to improve the performance of the routing algorithm in different ways. The next chapter will concentrate on the development of the analytical models of LLMR,
while the different new link cost functions will be introduced in the subsequent chapters.
Chapter 3

Performance Analysis of LLMR Algorithm

In this chapter, we present the analytical model of Least Load Multicast Routing (LLMR) algorithm and its numerical results. In Section 3.1, we describe briefly the LLMR algorithm with a small example. Section 3.2 shows the system model and defines some symbolic notations. The performance modeling for fully connected networks is presented in Section 3.3. Section 3.4 gives an numerical example to determine the accuracy of the analytical model. Analytical models for symmetrical networks are presented in Section 3.5 and its numerical results are shown in Section 3.6.

3.1 Least Load Multicast Routing (LLMR) algorithm

In LLMR, the information used to make a routing decision is based on the states of links, i.e. the number of free circuits of links. To establish a connection, the LLMR algorithm tries to (i) use direct links to route the connection request, where direct links are links connected to two nodes in a destination set, and (ii) maximize the use of network capacity by accepting more connection requests. In order to achieve the above objectives, the LLMR locates a spanning tree
consisting of only direct links with the highest possible states. If no spanning tree is available, the connection request is denied.

To illustrate the routing algorithm outlined above, we consider a 4-node network and a connection request with destination set \( \{r, a, b, c\} \) with node \( r \) as the source node. The steps in the connection establishment procedure are as shown in Figure 3.1 — the number next to a link denotes its state and links that are included in the connection request are drawn with "thick" lines. Node \( c \) is the first one to be added as link \((r, c)\) has 3 free circuits that is larger than that of links \((r, a)\) and \((r, b)\) which is followed by node \( a \) (link \((c, a)\) is chosen due to its larger number of free circuits over links \((r, a), (r, b)\) and \((c, b)\)). Finally, we choose link \((a, b)\) because the number of free circuits of \((a, b)\) is the largest compared with links \((r, b)\) and \((b, c)\). After iteration 4, the procedure is completed and a tree is established for the connection request.

![Figure 3.1: Example to illustrate LLMR](image-url)
3.2 System Model and Symbolic Notations

Before proceeding further, we would like to introduce necessary definitions and notations. Consider a single rate loss network to be an undirected graph $G = (V, E)$ where $V$ and $E$ are the set of nodes and links respectively. Denote by $C(e)$, $e \in E$, the capacity or number of circuits allocated to link $e$. Consider a connection request $c$ with destination set $S(c)$ (each node $s \in S(c)$ is referred to as a destination of the connection request $c$). Let $d(c)$ be the number of nodes involved in the connection request $c$, i.e., the size of the destination set. The connection request $c$ requires a connected, acyclic graph (tree) $T(c) = (S(c), E(c))$, where $S(c) \subseteq V(c)$ is chosen and a unit bandwidth is reserved on each link $e \in E(c)$.

Consider a fully connected network with the set of nodes $V$ and links $E$. Connection requests (assumed for the discussion here to have identical bandwidth requirements of 1 bandwidth unit) with destination set, $S(c)$, arrive according to a Poisson process with rate $\lambda_c$. It is assumed that the holding times of connections are independent and identically distributed random variables with unit mean.

Throughout we denote by $\pi_i^{(e)}$, $i = 0, 1, 2, \ldots, C(e)$, and $e \in E$, the probability of the link occupancy of the link $e$ being $i$ in steady state. Note that, in LLMR algorithm, the link state represents the number of free circuits. In the analytical model, we denote instead the link state by the number of occupied circuits. We choose the link occupancy to represent the link state for convenience to develop the analytical model. Further, we denote the state dependent arrival rate of the link $e$ in state $i$ by $\Lambda_i^{(e)}$, $i = 0, \ldots, C(e) - 1$ and $e \in E$. We also denote $\Pi = \{\pi^{(e)} : e \in E\}$ and $\Lambda = \{\Lambda^{(e)} : e \in E\}$, where $\pi^{(e)} = (\pi_0^{(e)}, \ldots, \pi_{C(e)}^{(e)})$ and $\Lambda^{(e)} = (\Lambda_0^{(e)}, \ldots, \Lambda_{C(e)-1}^{(e)})$. 

20
3.3 Performance Modeling

The analytical models that we develop for calculating blocking probabilities are based on the *link independence assumption*, i.e. the random variables describing the state of each link are assumed to be independent. This assumption has been used with great success to analyze routing schemes for point-to-point connections (state dependent or otherwise) in a variety of networks; the studies [19] [25] [12] [18] are just a few examples.

The analytical model that we propose is based on the Reduced Load Approximation (RLA), which is an iterative procedure that alternates between the following two steps:

1. Evaluation of link occupancy distributions given the state dependent arrival rates and

2. Evaluation of state dependent arrival rates given the link occupancy distributions.

Once the consistent set of link occupancy distributions and state dependent arrival rates are obtained, the call blocking probability of each class (a class here means a group of multicast call connections with the same destination size) can be calculated with little effort.

Part 1 Evaluation of link occupancy distribution given the state dependent arrival rates

Given the state dependent arrival rates, $\Lambda^{(e)}_i, i = 0, \ldots, C(e) - 1$, the steady-state link occupancy distribution can be determined by the solution of a one-dimensional birth-death Markov process. Denote by $\gamma^{(e)}(j, l)$ the transition rate of the birth-death process from state with link occupancy $j$ to state with link occupancy $l$, where $j$ and $l$ are integers, $0 \leq j \leq C(e) - 1$ and $0 \leq l \leq C(e) - 1$. 

21
Thus we have
\[ \gamma^{(e)}(j, j + 1) = \gamma_{j}^{(e)}, \quad 0 \leq j < C(e) - 1; \] (3.1)
and recalling that the mean holding time of each connection is unity, it follows that
\[ \gamma^{(e)}(j, j - 1) = j, \quad 0 < j \leq C(e) - 1. \] (3.2)
Note that \( \gamma^{(e)}(j, l) = 0 \) if \( |j - l| > 1 \). Thus,
\[ \pi_{j}^{(e)} = f_{j}(\Lambda), \quad j = 0, \ldots, C(e) - 1, \] (3.3)
where \( \pi_{j}^{(e)}, j = 0, 1, 2, \ldots, C(e) - 1 \) and \( e \in E \), are the steady-state probabilities of the one-dimensional birth-death process with rates given by equations (3.1) and (3.2) \( (f_{j} \) is a function which is given by equations (3.1) and (3.2)).

Part 2 Evaluation of state dependent arrival rates given the link occupancy distributions

We now discuss the process of evaluating the state dependent arrival rates, \( \Lambda \), given the steady state link occupancy distribution \( \Pi \). Denote by \( Z(c) \) the set of direct links to be considered in a connection request \( c \) with destination set \( d(c) \) in a fully connected network and \( Z(c) = |Z(c)| \) (i.e. \( Z(c) = d(c)(d(c) - 1)/2 \)).

For \( e_{k} \in Z(c), 1 \leq k \leq Z(c) \), let \( i_{k} \) be its link state such that it is an integer and \( 0 \leq i_{k} \leq C(e_{k}) \), then the probability that the link \( e_{k} \) in state \( i \) which is selected and included in a connected tree for the call connection request \( c \) is
\[ \theta_{i}^{(e_{k})}(c) = \sum_{\forall l_{c} \in A_{c}(k, i)} P_{s}(e_{k}, C(e_{k}) - i, I_{c}) \prod_{j=1}^{Z(c)} \pi_{j}^{(e_{j})} \] (3.4)
where \( I_{c} = (C(e_{1}) - i_{1}, C(e_{2}) - i_{2}, \ldots, C(e_{Z(c)}) - i_{Z(c)}) \), \( A_{c}(k, i) = \{ I_{c} : i_{k} = i \text{ and } T(c) \neq \phi \} \), and \( P_{s}(e_{k}, C(e_{k}) - i, I_{c}) \) is the probability that the link \( e_{k} \) in state \( i \) with \( C(e_{k}) - i \) free circuits is selected when a connection request \( c \) arrives and the current state of the links involved is \( I_{c} \).
To illustrate the meaning of $P_k(e_k, C(e_k) - i, I_c)$, consider an example of the 3-node network shown in Figure 3.2. Denote by $R(e)$ the residual capacity of link $e$. There are three cases such that link $A$ is possible to be included in a connected tree:

- If $R(A)$ is not the smallest one among three links, $R(A) \neq R(B) \neq R(C)$, and there is at least one free circuits in links $B$ and $C$, then link $A$ must be included;

- If $R(A)$ is the smallest one among them and $R(A) = R(B)$ or $R(A) = R(C)$, then the probability that link $A$ is included in a connected tree is 0.5 because link $A$ is not selected or a second one to be selected;

- If $R(A) = R(B) = R(C)$, then each link has the same probability to be selected and there are two links to be selected in a connected tree. Hence the probability of link $A$ to be included in a connected tree is $2/3$. 
Thus, we have

\[
P_s(A, R(A), I_{(1,2,3)}) = \begin{cases} 
1 & \text{if } R(A) > R(B), R(A) > R(C) \\
0.5 & \text{if } (R(A) = R(B) \& R(C) > R(A)) \\
& \text{or } (R(A) = R(C) \& R(B) > R(A)) \\
\frac{2}{3} & \text{if } R(A) = R(B) = R(C) \\
0 & \text{otherwise}
\end{cases}
\]  

(3.5)

Further, by considering all call connection requests that can potentially be carried on link \( e \), the total traffic offered to link \( e \) in state \( i \) is

\[
\Lambda_i^{(e)} = \sum_{c \in D_e} \lambda_c \theta_i^{(c)}(c),
\]

(3.6)

where \( D_e = \{c : e \in Z(c)\} \) and \( \lambda_c \) is the arrival rate of the call connection request \( c \). Observe that equations (3.4) through (3.6) outline a procedure to obtain \( \Lambda_i^{(e)} \) from \( \Pi \); hence

\[
\Lambda_i^{(e)} = g_i(\Pi), \quad i = 0, \ldots, C(e) - 1 \quad \text{and} \quad e \in E,
\]

(3.7)

where \( g_i(\cdot) \) is a function that is given by equations (3.4) through (3.6).

The system of equations in (3.1) and (3.7) form the Reduced Load Approximations for \( \Pi \) and \( \Lambda \):

\[
\Pi = f(A) \\
\Lambda = g(\Pi)
\]

(3.8)

where \( f \) and \( g \) are the sets of functions \( f_j \) and \( g_i \), respectively.

The call blocking probability of call connection request \( c \) is then given by

\[
B(c) = \sum_{I_c \in \tilde{A}_c} \prod_{j=1}^{d(c)} \pi_{i_j}^{(c)}
\]

(3.9)

where \( \tilde{A}_c = \{I_c : T(c) = \phi\} \), and the call blocking probability of a class with destination set of size \( d \) is

\[
B(d) = \frac{\sum_{d(c) = d, \forall c} \lambda_c B(c)}{\sum_{d(c) = d, \forall c} \lambda_c}.
\]

(3.10)

Hence the normalized revenue loss can be expressed as,

\[
E(\text{loss}) = \frac{\sum_{d=2}^{D}(d-1)\lambda_d B(d)}{\sum_{d=2}^{D}(d-1)\lambda_d}.
\]

(3.11)
The computational requirements of the above approximation procedure are reasonable. Take our numerical results to be presented in Section 3.4 as an example. Each analytical result obtained requires only one or two minutes on a Sun Spare 5 machine, while a simulation run normally requires overnight on the same machine. The following algorithm outlines the procedure to obtain the values of $\Pi$ and $\Lambda$:

```
begin
  $k = 0$;
  $error = \epsilon + 1.0$;
  For each $e \in E$, $\Lambda_i^{(e)}(k) = C(e)$, $i = 0, 1, 2, \ldots, C(e) - 1$;
  while ($error > \epsilon$) do
    begin
      $error = 0$;
      For each $e \in E$,
        begin
          $\Pi_i^{(e)}(k + 1) = f(\Lambda^{(e)}(k))$;
          $\Lambda_i^{(e)}(k + 1) = g(\Pi_i^{(e)}(k + 1))$;
          $error = \max_{i=0}^{C(e)} (|\pi_i^{(e)}(k + 1) - \pi_i^{(e)}(k)|, error)$;
        end;
      $k \leftarrow k + 1$;
    end;
end,
```

where $\pi_i^{(e)}(n)$, $\Lambda_i^{(e)}(n)$, $\Pi^{(e)}(n)$ and $\Lambda^{(e)}(n)$ are the values of $\pi_i^{(e)}$, $\Lambda_i^{(e)}$, $\Pi^{(e)}$ and $\Lambda^{(e)}$ after $n$ iterations, respectively. $\epsilon$ is the upper bound of an acceptable error, which is set as $10^{-4}$ in the next section.
3.4 Numerical Results

In this section, we present numerical and simulation results to illustrate the performance of our proposed algorithm and verify the accuracy of the analytical models developed. We consider a fully connected network with 20 nodes. The capacity of each link is randomly generated between 10 and 20 bandwidth units or circuits. Three kinds of connection requests are considered and the destination sizes are 2, 3 and 4 (i.e. \( D = 4 \)).

We assume the loading of unicast connection is equal to the total loading of multicast connections, i.e. in our case, \( L_2 = L_3 + L_4 \) where \( L_d \) is defined as the loading of class \( d \) (\( L_d = (d - 1)\lambda_d \)). For convenience we define the normalized network load, \( L \), as the ratio of total offered load to total network capacity, i.e.,

\[
L = \frac{\sum_{d=2}^{D} L_d}{\sum_{e \in E} C(e)}.
\] (3.12)

For each simulation run, the simulation run was terminated after \( 10^6 \) connection requests had been generated and the initial 10% of each run was discarded to avoid the transient effect. The vertical lines about each point indicate the 95% percent confidence interval. For each analytical result, the iterative procedure will be terminated if the difference between the current and previous values of all \( \pi_i \)'s are less than \( 10^{-4} \), i.e. \( |\pi_i^{(e)}(k + 1) - \pi_i^{(e)}(k)| < 10^{-4} \) for each \( e \in E \) where \( \pi_i^{(e)}(k) \) is the value of \( \pi_i^{(e)} \) after \( k \) iterations. In our numerical examples, the iterative procedure always converges to the solution in less than 20 iterations.

Figure 3.3 shows the accuracy of the analytical models of LLMR. The agreement of simulation and analytical results is observed to be surprisingly good. The high accuracy of the call blocking probability of class 2 shows that the calculations of link occupancy distributions and state dependent arrival rates are not affected by the link independence assumption. We also find that the accuracy of the call blocking probability worsens if its destination size becomes larger. It
is because, from equation (3.9), we apply the link independence assumption in the calculation of the call blocking probability. However, the links that involved in a multicast call connection is depended. When the destination size increases, the number of links involved in a call connection increases and hence the link dependence becomes more significant.

Figure 3.4 shows that when the ratio of the loading of unicast connection to that of multicast connections increases, the error of call blocking probability of class 4 drops significantly and hence it indicates that the error of the calculation of the call blocking probability is due to the link independence assumption.
Figure 3.3: Call blocking probability versus network loading

Figure 3.4: % error in mean call blocking probability of class 4 versus the ratio of the loading of unicast connections to that of multicast connections
3.5 LLMR for Symmetrical Networks

For the development of the analytical model for fully connected networks in the previous section, it is difficult to give a closed form expression for the function \( P_s(e_k, C(e_k) - i, I_c) \). Moreover, the sets \( A_c(k, i) \) and \( A_c \) cannot be easily determined for general cases. However, for symmetrical networks, we can handle the above difficulties and thus we develop an analytical model for such networks. Note that a symmetrical network is defined as a network such that each link is identical and the call arrival rates of all possible combinations of each classes are the same.

For the analytical model applied to the case of symmetrical networks, we include alternative routings for point-to-point calls in our analytical model. The analytical models (models with and without alternative routings for point-to-point calls) that we propose are again based on the Reduced Load Approximation (RLA), which is an iterative procedure that alternates between the following two steps:

1. Evaluation of link occupancy distributions given the state dependent arrival rates and

2. Evaluation of state dependent arrival rates given the link occupancy distributions.

Once the consistent set of link occupancy distributions and state dependent arrival rates are obtained, the call blocking probability of each class (or any other performance measure for that matter) can be calculated with little effort.

Consider a fully connected symmetrical network with \( N \) nodes and each link having a capacity of \( C \) units. Requests for connections (assumed for the discussion here to have identical bandwidth requirements of 1 bandwidth unit) with destination set of size \( d, d > 1 \), arrive according to a Poisson process with
rate \( \lambda_d \). It is assumed that the holding times of connections are independent and identically distributed random variables with unit mean. Further, it is also assumed that the probability that a given node belongs to a destination set is the same for all nodes. Since the network we consider is symmetrical in all respects (the call arrival rates of all possible combinations of each classes are the same), the occupancy distributions for all links are identical. Henceforth, we refer to a link as being in state \( i, i = 0, \ldots, C \), if its occupied capacity is \( i \). Let \( \pi_i, i = 0, \ldots, C \), be the probability of the link occupancy being \( i \) in steady state. Further, let \( \Lambda_i, i = 0, \ldots, C - 1 \), denote the arrival rate of connections to a link in state \( i \). We begin by first considering the case of the LLMR no alternative nodes allowed and then extend the procedure to include alternative nodes.

### 3.5.1 LLMR with No Alternative Nodes

#### Part 1 Evaluation of link occupancy distribution given the state dependent arrival rates

Given the state dependent arrival rates, \( \Lambda_i, i = 0, \ldots, C - 1 \), the steady-state link occupancy distribution can be determined by the solution of a one-dimensional birth-death Markov process. Denote by \( \gamma(j, l) \) the transition rate of the birth-death process from state with link occupancy \( j \) to state with link occupancy \( l \), where \( j \) and \( l \) are integers, \( 0 \leq j \leq C - 1 \) and \( 0 \leq l \leq C - 1 \). Thus we have

\[
\gamma(j, j + 1) = \Lambda_j, \quad 0 \leq j < C - 1; \tag{3.13}
\]

and recalling that the mean holding time of each connection is unity, it follows that

\[
\gamma(j, j - 1) = j, \quad 0 < j \leq C - 1. \tag{3.14}
\]
Note that $\gamma(j,l) = 0$ if $|j - l| > 1$. Thus,

$$\pi_j = f_j(\Lambda), \quad j = 0, \ldots, C,$$

where $\pi_j, j = 0, 1, 2, \ldots, C$, are the steady state probabilities of the one-dimensional birth-death process with rates given by equations (3.13) and (3.14) ($f_j$ is a function which is given by equations (3.13) and (3.14)).

Part 2 Evaluation of state dependent arrival rates given the link occupancy distributions

We now discuss the process of evaluating the state dependent arrival rates, $\Lambda$, given the steady state link occupancy distribution $\Pi$. Before proceeding the evaluation, we need to introduce some additional concepts and definitions. Denote by $\Phi(i)$ the probability of a link being in state $i$ or higher, i.e.,

$$\Phi(i) = \sum_{j=i}^{C} \pi_j, \quad i = 0, \ldots, C.$$  \hfill (3.16)

Let $Q_i(l,m,n)$ be the probability that given $n - 1$ links, $l$ links are in the set of states $\{0, \ldots, i - 1\}$, $m - 1$ are in state $i$, and the remaining are in the set of states $\{i + 1, \ldots, C\}$. A routine combinatorial argument yields

$$Q_i(l,m,n) = \frac{(n-1)!}{l!(m-1)!(n-l-m)!} [1 - \Phi(i)]^{l} \pi_i^{m-1} \Phi(i+1)^{n-l-m}, \quad l + m \leq n.$$  \hfill (3.17)

Let $Q'_i(l,m,j,n)$ be the probability that given $n - 1$ links, $l$ links are in the set of states $\{0, \ldots, i - 1\}$, $m - 1$ are in state $i$, $j$ are in blocking state (i.e. all circuits are occupied), and the remaining are in the set of states $\{i + 1, \ldots, C - 1\}$. A routine combinatorial argument yields

$$Q_i'(l,m,j,n) = \frac{(n-1)!}{l!(m-1)!j!(n-l-m-j)!} [1 - \Phi(i)]^l \pi_i^{m-1} \pi_j^j [\Phi(i+1) - \pi_j]^{n-l-m-j},$$  \hfill (3.18)

where $l + m + j \leq n$. 

31
We now consider the following process for building a random graph. We start with an empty graph (i.e., with no edges) over \( n \) nodes to which edges are added randomly one at a time as described. The edges are chosen randomly (from the set of all possible edges) one at a time and considered for inclusion in the graph. An edge is discarded (and not considered again) if it forms a cycle, otherwise it is added to the graph. The procedure is stopped as soon as a spanning tree is obtained. Denote by \( Z(d) \) the total number of links to be considered in a connection request with destination set of size \( d \); for fully connected network, \( Z(d) = d(d - 1)/2 \). Let \( q(n, k), n > 0, Z(n) \geq k > 0, \) be the probability that the \( k \)th edge chosen is included in the graph. Similarly, we define \( s(n, k, l), n > 0, Z(n) \geq k > 0, Z(n) - k \geq l \geq 0, \) to be the probability that, given that the \( k \)th edge chosen is included in the graph, a graph over \( n \) nodes and having \( k + l \) randomly chosen links is not connected. Table 3.1 and Table 3.2 provide numerical values for \( s(n, k, l) \) and \( q(n, k) \) obtained through simulations for various graph sizes. It is important to observe that both \( s(n, k, l) \) and \( q(n, k) \) are functions of only the number of nodes and links and can therefore be computed “off-line” and stored in “look-up” tables.

It is instructive to note that the LLMR algorithm with no alternative nodes can be considered as one where links are considered in decreasing order of their residual capacity. A link is chosen for inclusion in the routing tree if it does not form a cycle with the previously included links. Now consider a connection request with a destination set of size \( d \). The probability that a link in state \( i \) is the \( k \)th, \( k = 1, \ldots, Z(d) \), one to be considered (recall, links can be assumed as being considered in LLMR for inclusion in the routing tree in order of increasing occupancy with any ties being broken randomly) is

\[
\Psi_i(k, d) = \sum_{l=0}^{k-1} \sum_{m=k-l}^{Z(d)-l} \frac{1}{m!} [Q_i(l, m, Z(d)) - \sum_{j=0}^{Z(d)-k} s(d, k, j)Q'_i(d, k, j, Z(d))],
\]  

(3.19)

where \( k = 1, \ldots, Z(d), \) \( d = 2, \ldots, D, \) and \( D, D \leq N, \) is the maximum destina-
<table>
<thead>
<tr>
<th>$k$</th>
<th>$k + l$</th>
<th>Nodes (n)</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>1,2</td>
<td>3</td>
<td>0.20000</td>
<td>1.00000</td>
<td>1.00000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.00000</td>
<td>0.38077</td>
<td>1.00000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td></td>
<td>0.10445</td>
<td>0.50712</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td></td>
<td>0.01949</td>
<td>0.20814</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td></td>
<td>0.00000</td>
<td>0.07828</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
<td>0.32549</td>
<td>1.00000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td></td>
<td>0.08664</td>
<td>0.48472</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td></td>
<td>0.01557</td>
<td>0.19546</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td></td>
<td>0.00000</td>
<td>0.07280</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td></td>
<td>0.11688</td>
<td>1.00000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td></td>
<td>0.03396</td>
<td>1.00000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td></td>
<td>0.00740</td>
<td>0.41691</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td></td>
<td>0.00000</td>
<td>0.16050</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.1: $s(n,k,l)$ as a function of $n$ and $k$.

tion size of any connection request.

The probability that a link in state $i$ which is the $k$th one to be considered and included in the routing tree for a connection request with destination set of size $d$ is

$$\theta_i(k,d) = q(d,k)\Psi_i(k,d)$$  \hspace{1cm} (3.20)

where $k = 1, \ldots, Z(d)$ and $d = 2, \ldots, D$. Further, since given a destination set size, all nodes have the same probability of being chosen as a destination, the total traffic offered to a link in state $i$ is

$$\Lambda_i = \sum_{d=2}^{D} \frac{Z(d)}{Z(N)} \lambda_d \sum_{k=1}^{Z(d)} \theta_i(k,d).$$ \hspace{1cm} (3.21)

Note that $\frac{Z(d)}{Z(N)}$ is the probability of a randomly chosen link being considered for routing a connection request with destination set of size $d$. (Observe that out of $Z(N)$ links in the network only $Z(d)$ can be considered for routing a connection request with destination set size $d$.)
<table>
<thead>
<tr>
<th>$k$</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.79995</td>
<td>0.91690</td>
<td>0.95602</td>
<td>0.97367</td>
<td>0.98295</td>
<td>0.98826</td>
<td>0.99153</td>
</tr>
<tr>
<td>4</td>
<td>0.20005</td>
<td>0.70236</td>
<td>0.84627</td>
<td>0.90945</td>
<td>0.94197</td>
<td>0.96031</td>
<td>0.97181</td>
</tr>
<tr>
<td>5</td>
<td>0.27763</td>
<td>0.66448</td>
<td>0.79964</td>
<td>0.87136</td>
<td>0.91239</td>
<td>0.93781</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.08411</td>
<td>0.32264</td>
<td>0.65092</td>
<td>0.77043</td>
<td>0.84248</td>
<td>0.88801</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.01900</td>
<td>0.13288</td>
<td>0.35876</td>
<td>0.64776</td>
<td>0.75218</td>
<td>0.82143</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.05184</td>
<td>0.16901</td>
<td>0.39008</td>
<td>0.64910</td>
<td>0.74103</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.01893</td>
<td>0.07668</td>
<td>0.20065</td>
<td>0.41848</td>
<td>0.65268</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.00582</td>
<td>0.03475</td>
<td>0.09812</td>
<td>0.22957</td>
<td>0.44302</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.00112</td>
<td>0.01580</td>
<td>0.04770</td>
<td>0.11825</td>
<td>0.25626</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0.00070</td>
<td>0.02396</td>
<td>0.06011</td>
<td>0.13756</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.00290</td>
<td>0.01230</td>
<td>0.03107</td>
<td>0.07201</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>0.00103</td>
<td>0.00639</td>
<td>0.01663</td>
<td>0.03807</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>0.00029</td>
<td>0.00330</td>
<td>0.00915</td>
<td>0.02047</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>0.00005</td>
<td>0.00168</td>
<td>0.00517</td>
<td>0.01147</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>0.00081</td>
<td>0.00300</td>
<td>0.00661</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>0.00036</td>
<td>0.00172</td>
<td>0.00393</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>0.00014</td>
<td>0.00100</td>
<td>0.00240</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>0.00005</td>
<td>0.00055</td>
<td>0.00149</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>0.00001</td>
<td>0.00030</td>
<td>0.00094</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>0.00016</td>
<td>0.00058</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>0.00007</td>
<td>0.00036</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>0.00004</td>
<td>0.00022</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>0.00014</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>0.00008</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>0.00005</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>0.00002</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>0.00001</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>0.00001</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.2: $q(n,k)$ as a function of $n$ and $k$. 
Observe that equations (3.16) through (3.22) outline a procedure to obtain
\( \Lambda_i \) from \( \Pi \); hence
\[
\Lambda_i = g_i(\Pi), \quad i = 0, \ldots, C - 1,
\]  
where \( g_i(\cdot) \) is a function that is given by equations (3.16) through (3.22).

The system of equations in (3.15) and (3.23) form the Fixed-Point Equations
for \( \Pi \) and \( \Lambda \):
\[
\Pi = f(\Lambda) \\
\Lambda = g(\Pi)
\]  
where \( f \) and \( g \) are the sets of functions \( f_j \) and \( g_i \), respectively.

### 3.5.2 LLMR with An Alternative Node Allowed for Class 2 only

Having discussed the formulation of the reduced load approximations for
the case with no alternative nodes, we now proceed with our discussion of the
procedure needed to obtain the reduced load approximations for LLMR with
alternative nodes. Denote by \( \Lambda'_i \) the state dependent arrival rate in state \( i \). The
evaluation of the steady state link occupancy probabilities is the same as that
for LLMR without alternative nodes and is obtained by simply replacing \( \Lambda_i \) by
\( \Lambda'_i \). Therefore, we concentrate on the evaluation of the state dependent arrival
rates.

Since we do not allow the use of alternative nodes for connection requests
with destination set size larger than 2, the connection arrival rates to a link in
state \( i, 0 \leq i \leq C - 1 \) due to connection requests with destination set size larger
than 2 remain unchanged. Hence, we now direct our efforts towards evaluating
the revised connection arrival rates due to point-to-point connection requests in
the presence of alternative routing. The approach we use to determine the above
state dependent arrival rates, \( \Lambda'_i, i = 0, \ldots, C - 1 \), is based on the discussion in
[25].
Denote by $\Phi^*(i)$ the probability that a given two link alternative path belongs to the set of states $\{i, \ldots, C-1\}$. It easily follows that

$$
\Phi^*(i) = [1 - \Phi(i + 1)]^2 - [1 - \Phi(i)]^2.
$$

(3.24)

For a given origin-destination pair, given that the least busy two link alternative route is in state $i$, $0 \leq i \leq t'$, where $t'$ is trunk reservation threshold (above which no alternative calls are allowed), the probability that there are $(n - 1)$, $n = 1, 2, \ldots, M$, other such least busy two-link routes is

$$
F(n|i) = \left( \frac{M - 1}{n - 1} \right) \Phi^*(i)^{n-1} \left[ \sum_{j=i+1}^{C-1} \Phi^*(j) \right]^{M-n},
$$

(3.25)

where $M$ is the number of available alternative nodes and is typically equal to $N - 2$. Hence, the rate of overflow traffic from a particular source-destination pair which is routed to a given two-link route in state $i$ is (recall, no overflow traffic can be routed on a link when its state exceeds the trunk reservation threshold)

$$
y_i = \frac{\lambda_2 \pi C}{Z(N)} \sum_{n=0}^{M} \frac{1}{n} F(n|i), \quad i = 0, \ldots, t'.
$$

(3.26)

The total rate of overflow traffic routed to a link in state $i$ from all source-destination pairs is

$$
v_i = 2M \sum_{j=0}^{t'} y_{\text{max}(i,j)} \pi_j, \quad i = 0, \ldots, t'
$$

$$
= 2M \left[ y_i \sum_{j=0}^{i} \pi_j + \sum_{j=i+1}^{t'} y_j \pi_j \right], \quad i = 0, \ldots, t',
$$

(3.27)

where

$$
y_{\text{max}(i,j)} = \begin{cases} 
y_i, & 0 \leq j \leq i \\
y_j, & \text{otherwise}
\end{cases}
$$

(3.23)

and

$$
v_i = y_i = 0 \quad i = t' + 1, \ldots, C - 1.
$$

(3.29)

Given a destination set size, all nodes have the same probability of being chosen as a destination. With the consideration of the alternative path for class
only, the total traffic offered to a link in state \( i \) is

\[
\hat{\Lambda}_i = \begin{cases} 
\Lambda_i + v_i, & 0 \leq i \leq t', \\
\Lambda_i, & \text{otherwise}
\end{cases}
\]  

(3.30)

where \( \Lambda_i \) is obtained from equation (3.22). The above process to obtain \( \hat{\Lambda}_i \) from

\[
\Pi \text{ can be expressed as:}
\]

\[
\hat{\Lambda}_i = \hat{g}_i(\Pi) \quad i = 0, \ldots, C - 1.
\]  

(3.31)

The system of equations in (3.15) and (3.32) form the Fixed-Point Equations

\[
\Pi = f(\hat{\Lambda})
\]

\[
\hat{\Lambda} = \hat{g}(\Pi)
\]  

(3.32)

where \( \hat{g} \) is the set of functions \( \hat{g}_i \).

A few comments about the approximation procedure are now in order. The approximation procedure depends on one-time simulations and analytical approximations to evaluate the performance measures of interest. It is important to observe that since \( q(n, k) \) and \( s(n, k, l) \) do not depend on any time varying parameter, they need to be evaluated only once, and therefore can be precomputed and stored in lookup tables. We know that the accuracy of the simulation results depend on the length of the simulation runs. Since these two functions can be obtained “off-line”, we can run the simulations as long as possible and thus the error of the simulation results can be made as small as possible. In Tables 3.1 and 3.2, the 95% confidence interval is within \( 10^{-5} \). Further, the computational requirements of the approximation procedure are reasonable. Take our numerical results in Section 3.6 as an example. Each analytical result obtained requires only a few seconds on a Sun Spare 5 machine, while a simulation run normally requires overnight on the same machine. Finally, for the case that no alternative nodes are allowed, the following algorithm is proposed to obtain the values of \( \Pi \) and \( \Lambda \):
begin

\[ k = 0; \]

\[ error = \epsilon + 1.0; \]

\[ \Lambda_i(k) = C, \quad i = 0, 1, 2, \ldots, C - 1; \]

while (error > \epsilon) do

begin

\[ \Pi(k + 1) = f(\Lambda(k)); \]

\[ \Lambda(k + 1) = g(\Pi(k + 1)); \]

\[ error = \max_{i=0}^{C}(\|\pi_i(k + 1) - \pi_i(k)\|); \]

\[ k \leftarrow k + 1; \]

end;

end,

where \( \pi_i(n) \) is the value of \( \pi_i \) after \( n \) iterations and \( \epsilon \) is the upper bound of an acceptable error. For the case that an alternative node is allowed for class 2 only, we can apply the above algorithm except the function \( g \) is replaced by \( \hat{g} \) and \( \Lambda \) by \( \hat{\Lambda} \).

### 3.5.3 Evaluation of Performance Metrics

In [11], Gilbert gave a nice recursive function to determine the probability of connectivity of a random graph and we can apply the result for the evaluation of performance metrics. For the case of no alternative nodes allowed, the cell blocking probability of a connection request of destination set size \( d \) is given by

\[ B(d) = 1 - p_d \quad (3.33) \]

where

\[ 1 - p_d = \sum_{k=1}^{d-1} \binom{d-1}{k-1} \pi_k \pi_{C}^{k(d-k)}, \quad (3.31) \]
for $d > 1$ and $p_l = 1$. Note that $\pi_C$ is obtained from the solution of (3.23). For the case of an alternative node allowed for class 2 only, the blocking probabilities for class $d, d > 2$, remain the same as in equation (3.33), while the blocking probability for connection request of destination set size 2 is changed to
\[
\pi_C[1 - (1 - \Phi(t + 1))^2]^M
\]

where $M = N - 2$ and $\Phi(t)$ is given by (3.16). To calculate the network performance, again we make the performance metric of normalized revenue loss defined in equation (2.1).

### 3.5.4 Numerical Results

In this section, we present numerical and simulation results to illustrate the performance of our proposed algorithm and verify the accuracy of the analytical models developed. We consider a fully connected network with $N$ nodes and each link has a capacity of $C$ units. Three kinds of connection requests are considered and their destination sizes are 2, 3 and 4 (i.e. $D = 4$).

We assume the loading of unicast connection is equal to the total loading of multicast connections, i.e. in our case, $L_2 = L_3 + L_4$ where $L_d$ is defined as the loading of class $d$ ($L_d = (d - 1)\lambda_d$). For convenience we define the normalized network load, $L$, as the ratio of total offered load to total network capacity, i.e.,
\[
L = \frac{\sum_{d=2}^{D} L_d}{Z(N)C}
\]

For each simulation run, the simulation run was terminated after $10^6$ connection requests had been generated and the initial 10% of each run was discarded to avoid the transient effect. The vertical lines about each point indicate the 95% percent confidence interval. For each analytical result, the iterative procedure will be terminated if the difference between the current and previous values of all $\pi_i$s are less than $10^{-4}$, i.e. $|\pi_i(n) - \pi_i(n-1)| < 10^{-4}$ for $i = 0, \ldots, C$.
where \( \pi_i(n) \) is the value of \( \pi_i \) after \( n \) iterations. In our numerical examples, the iterative procedure always converges to the solution in less than 20 iterations.

Figures 3.5 and 3.6 show the accuracy of the analytical models of LLMR. The agreement of simulation and analytical results is observed to be surprisingly good. The high accuracy of the call blocking probability of class 2 shows that the calculations of link occupancy distributions and state dependent arrival rates are not affected by the link independence assumption. We also find that the accuracy of the call blocking probability worsens if its destination size becomes larger. It is because, from equations (3.33), (3.34) and (3.35), the calculation of the call blocking probability with destination set of size \( d \) is a polynomial of \( \pi_C \) in degree \( Z(d) \). Hence a small error in \( \pi_C \) will result in a big discrepancy in the call blocking probability. To compare two figures, we find that the blocking probability of class 2 in Figure 3.6 is smaller than that in Figure 3.5 because alternative routing is allowed in Figure 3.6. Based on the same reason, there are some circuits used for the alternative routing for class 2 and hence the blocking probabilities of classes 3 and 4 in Figure 3.6 are larger than that in Figure 3.5.
Figure 3.5: Call blocking probability versus network loading (N = 30, C = 20, D = 4, and no alternative node is allowed)

Figure 3.6: Call blocking probability versus network loading (N = 30, C = 20, D = 4, trunk reservation = 4 free circuits and an alternative node is allowed for class 2)
Chapter 4

Modified Least Load Multicast Routing (MLLMR)

In this chapter, we present three MLLMR algorithms with performance comparison. Section 4.1 will present the Aggregated Least Load Multicast Routing (ALLMR) with analytical models. LLMR with the consideration of secondary parameters are shown in Section 4.2.

4.1 Aggregated Least Load Multicast Routing (ALLMR)

In this section, we study the effect of state aggregation for the purpose of simplifying the implementation of LLMR. Consider the following example: there is a 30-node fully connected network and each link has a capacity of 20 circuits. Three kinds of connection request are considered and their destination sizes are 2, 3 and 4. All classes of traffic are symmetrical in all respects. Least Load Multicast Routing (LLMR) is applied in this network and Figures 4.1 and 4.2 show the state dependent arrival rates and occupancy distributions versus link state with 90% network loading.
Figure 4.1: The state dependent arrival rate versus link state (occupied circuits)

Figure 4.2: The state occupancy distribution versus link state (occupied circuits)

It is easy to observe that there is a large transition over the last few states (i.e. when the link is almost full) in both figures. In Figure 4.1, the state dependent arrival rates are high at the beginning and decrease rapidly in the final few states. This implies LLMR avoids to select a link whose loading is high. The bell shape curve in Figure 4.2 shows that the probabilities of a link being in the low states are very small since LLMR always chooses those links. When the loading of a link is high, LLMR avoids to choose the link and hence the state probabilities in the final few states are dominant.

This observation suggests that there is no difference in choosing which links
to be a part of the connected tree when they are in the low states. However, when they are in the high states, LLMR will need to make use of the exact state information of links to establish the connected tree. Aggregation here means that the detailed state information is not used; instead, a coarse representation is used. At any given time each link belongs to an aggregated state in \( \{0, 1, \ldots, K - 1\} \). A link with capacity \( C \) belongs to aggregated state \( k, 0 \leq k \leq K - 1 \), if its occupied capacity \( x \) is between integers \( A_k \) and \( A_{k+1} \) (i.e. \( A_k \leq x < A_{k+1} \)), where \( 0 = A_0 < A_1 < \ldots < A_K = C + 1 \). For example, if the capacity or the number of circuits of a link is 20, we can aggregate the first 16 states into one aggregated state and the rest of the states will not be aggregated, i.e. \( A_0 = 0, A_1 = 16, A_2 = 17, A_3 = 18, A_4 = 19, A_5 = 20 \) and \( A_6 = 21 \). Note that we will not aggregate the final few states since they are significant to the performance of LLMR.

### 4.1.1 The System Model and The Algorithm

Before proceeding further, we would like to introduce necessary definitions and notations. Consider a single rate loss network to be an undirected graph \( G = (V, E) \) where \( V \) and \( E \) are the set of nodes and links respectively. Denote by \( C(e), e \in E \), the capacity or number of circuits allocated to link \( e \). At any given time each link belongs to an aggregated state in \( \{0, 1, \ldots, K - 1\} \). A link with capacity \( C(e) \) belongs to aggregated state \( k, 0 \leq k \leq K - 1 \) if its occupied capacity \( x \) is between integers \( A_k(e) \) and \( A_{k+1}(e) \), i.e. \( A_k(e) \leq x < A_{k+1}(e) \), where \( 0 = A_0(e) < A_1(e) < \ldots < A_K(e) = C(e) + 1 \). In the trivial case, a link with capacity \( C(e) \) would have states in the set \( \{0, 1, \ldots, C(e)\} \).

Consider a connection request \( c \) with destination set \( S(c) \) (each node \( s \in S(c) \) is referred to as a destination of \( c \)). Let \( d(c) \) be the number of nodes involved in a connection request \( c \), i.e., destination size. Let \( s_r \in S(c) \) be the destination
node initiating the connection request; henceforth referred to as the root node. A connection $c$ requires a connected, acyclic graph (tree) $T(c) = (V(c), E(c))$, where $S(c) \subseteq V(c)$ and a unit bandwidth is reserved on each link $e \in E(c)$. Let $L(e)$ be the link cost function and $L(e) = K - 1 - k$, where $k$ is the aggregated state of the link $e$.

4.1.2 Numerical Results

We consider a fully connected network with 8 nodes. The link capacity of each link is randomly generated between 40 and 60 bandwidth units or circuits. There are three kinds of call connection requests in the network and their destination sizes are 2, 3 and 4. The traffic pattern is randomly generated but the loadings of the three classes are equal. Each class of calls generated in the network follows a Poisson process. All call holding times are exponentially distributed with unit mean. All processing time, including call setup and release time, are negligible. When a call connection request is rejected, it will not retry and is cleared immediately from the network. For each set of simulation point, the length of each run is $10^6$ units of mean interarrival time of connection request and the initial 10% is discarded to avoid the effect of transient state. The vertical lines about each point represent the 95% confidence intervals. We define the network loading as the ratio of the total offered load to the total network capacity and set 80% as the engineered load (i.e. 0% overload).

Table 4.1 shows the effect of state aggregation in LLMR where the overloading is 10%. As mentioned in the beginning of Section 4.1, the routing decision of LLMR is important only when the occupancy of a link is nearly full. The final few states are thus not aggregated and we aggregate only the low states. We find that when the number of aggregated states increases, the difference between ALLMR and LLMR decreases rapidly. There is almost no difference
when $K = 10$, i.e. the final nine states are not aggregated and we aggregate only the rest of the low states into one aggregated state. Note that the case of $K = C(e) + 1$ corresponds to no aggregation (i.e. LLMR). Table 4.2 shows the effect of the network loading on the performance of ALLMR when the number of aggregated state is 8. We find that when the overloading increases, the difference between ALLMR and LLMR decreases. It indicates that when the network loading is high, ALLMR performs as well as LLMR. Tables 4.3 and 4.4 are the case of allowing at most one alternative node and the result is similar to Tables 4.1 and 4.2 respectively. Note that, to minimize the performance degradation, the number of the aggregated states required is different for different network environment, but will be much less than the total number of states without aggregation. The advantage of state aggregation is not only to reduce the computation complexity of determining the network performance, but more importantly is its simpler implementation, lower signaling traffic for establishing connection request and lower sensitivity to the design parameters.

<table>
<thead>
<tr>
<th>$K$</th>
<th>$d = 2$</th>
<th>$d = 3$</th>
<th>$d = 4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$1.86 \times 10^{-1}$</td>
<td>$6.52 \times 10^{-2}$</td>
<td>$4.11 \times 10^{-2}$</td>
</tr>
<tr>
<td>4</td>
<td>$9.24 \times 10^{-2}$</td>
<td>$2.14 \times 10^{-2}$</td>
<td>$9.62 \times 10^{-3}$</td>
</tr>
<tr>
<td>6</td>
<td>$5.99 \times 10^{-2}$</td>
<td>$1.14 \times 10^{-2}$</td>
<td>$4.94 \times 10^{-3}$</td>
</tr>
<tr>
<td>8</td>
<td>$4.93 \times 10^{-2}$</td>
<td>$8.82 \times 10^{-3}$</td>
<td>$3.84 \times 10^{-3}$</td>
</tr>
<tr>
<td>10</td>
<td>$4.66 \times 10^{-2}$</td>
<td>$8.46 \times 10^{-3}$</td>
<td>$3.73 \times 10^{-3}$</td>
</tr>
<tr>
<td>$C(e) + 1$</td>
<td>$4.63 \times 10^{-2}$</td>
<td>$8.38 \times 10^{-3}$</td>
<td>$3.63 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

Table 4.1: The effect of the number of aggregated states, $K$, on mean call blocking probabilities (10% overload, no alternative nodes are allowed)
<table>
<thead>
<tr>
<th>% Overload</th>
<th>% Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$d = 2$</td>
</tr>
<tr>
<td>0</td>
<td>82.49</td>
</tr>
<tr>
<td>5</td>
<td>22.81</td>
</tr>
<tr>
<td>10</td>
<td>6.48</td>
</tr>
<tr>
<td>15</td>
<td>1.48</td>
</tr>
<tr>
<td>20</td>
<td>0.84</td>
</tr>
</tbody>
</table>

Table 4.2: The effect of the network loading on mean call blocking probabilities comparing ALLMR ($K = 8$) to LLMR ($K = C(e) + 1$) (no alternative nodes are allowed)

<table>
<thead>
<tr>
<th>$K$</th>
<th>Mean call blocking probability</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$d = 2$</td>
</tr>
<tr>
<td>6</td>
<td>$3.37 \times 10^{-2}$</td>
</tr>
<tr>
<td>8</td>
<td>$2.99 \times 10^{-2}$</td>
</tr>
<tr>
<td>10</td>
<td>$2.89 \times 10^{-2}$</td>
</tr>
<tr>
<td>12</td>
<td>$2.86 \times 10^{-2}$</td>
</tr>
<tr>
<td>$C(e) + 1$</td>
<td>$2.82 \times 10^{-2}$</td>
</tr>
</tbody>
</table>

Table 4.3: The effect of the number of aggregated states, $K$, on mean call blocking probabilities (10% overload, trunk reservation = 5, at most one alternative node is allowed)
<table>
<thead>
<tr>
<th>% Overload</th>
<th>% Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$d = 2$</td>
</tr>
<tr>
<td>0</td>
<td>64.21</td>
</tr>
<tr>
<td>5</td>
<td>18.50</td>
</tr>
<tr>
<td>10</td>
<td>6.02</td>
</tr>
<tr>
<td>15</td>
<td>1.41</td>
</tr>
<tr>
<td>20</td>
<td>0.26</td>
</tr>
</tbody>
</table>

Table 4.4: The effect of the network loading on mean call blocking probabilities comparing ALLMR ($K = 8$) to LLMR ($K = C(e) + 1$) (trunk reservation = 5, at most one alternative node is allowed)

4.1.3 Analytical Models of ALLMR for Symmetrical Networks

In this section, we will show how to modify the analytical model of LLMR for fully symmetrical networks to handle ALLMR($K$) where $K$ is the number of aggregated states. Throughout we denote by $\pi_j$, $j = 0, 1, 2, \ldots, C$, the probability of the link occupancy being $j$ in steady state and let $\Pi = (\pi_0, \pi_1, \pi_2, \ldots, \pi_C)$. Note that, in the ALLMR algorithm, the link state is the aggregated state representing the number of free circuits. In the analytical model, we denote instead the link state by the aggregated state representing the number of occupied circuits. We choose the link occupancy to represent the link state for convenience to develop the analytical model. A link with capacity $C(e)$ belongs to state $k, 0 \leq k \leq K$ if its occupied capacity $x'$ is between integers $A'_k(e)$ and $A'_{k+1}(e)$ (more specifically, $A'_k(e) \leq x' < A'_{k+1}(e)$), where $0 = A'_0(e) < A'_1(e) < \ldots < A'_K(e) = C(e) + 1$. Further, we denote the arrival rate of connections to a link in aggregated state $i$ by $\Lambda_i, i = 0, \ldots, K - 1$ and let $\Lambda = (\Lambda_0, \ldots, \Lambda_{K-1})$. We begin by first considering a special case of the ALLMR routing algorithm where no al-
ternative nodes are allowed and then extend the procedure to include alternative
nodes for class 2 only.

**ALLMR(K) with No Alternative Nodes**

**Part 1** Evaluation of link occupancy distribution given the state dependent arrival rates

Given the state dependent arrival rates, \( \Lambda_i, i = 0, \ldots, K - 1 \), the steady-state link occupancy distribution can be determined by the solution of a state dependent one dimensional birth-death Markov process. Denote by \( \gamma(j, l) \) the transition rate of the birth-death process from state with link occupancy \( j \) to state with link occupancy \( l \), where \( j \) and \( l \) are integers, \( 0 \leq j \leq C \) and \( 0 \leq l \leq C \). Thus we have

\[
\gamma(j, j + 1) = \Lambda_i, \quad A_i^j \leq j < A_{i+1}^j, \quad 0 \leq i < K - 1; \quad (4.1)
\]

and recalling that the mean holding time of each connection is unity, it follows that

\[
\gamma(j, j - 1) = j, \quad 0 < j \leq C. \quad (4.2)
\]

Note that \( \gamma(j, l) = 0 \) if \( |j - l| > 1 \).

Thus,

\[
\pi_j = f_j(\Lambda), \quad j = 0, \ldots, C, \quad (4.3)
\]

where \( \pi_j, j = 0, 1, 2, \ldots, C \), are the state probabilities of the one-dimensional birth-death process with rates given by equations (4.1) and (4.2).

**Part 2** Evaluation of state dependent arrival rates given the link occupancy distributions

We now discuss the process of evaluating the state dependent arrival rates,
Λ, given the steady state link occupancy distribution Π. Before proceeding
the evaluation, we need to introduce some additional concepts and definitions
Denote by \( P(i) \) the stationary probability that a link is in aggregated state \( i \),
\[
P(i) = \sum_{j=A_i}^{A_{i+1}-1} \pi_j, \quad i = 0, \ldots, K - 1. \tag{4.4}
\]
Denote by \( \Phi(i) \) the probability of a link being in aggregated state \( i \) or higher,
i.e.,
\[
\Phi(i) = \sum_{k=i}^{K-1} P(k), \quad i = 0, \ldots, K - 1. \tag{4.5}
\]
Then the equations (3.17) and (3.18) are changed to be
\[
Q_i(l, m, n) = \frac{(n-1)!}{l!(m-1)!(n-l-m)!} [1 - \Phi(i)]^l P(i)^{m-1} \Phi(i+1)^{n-l-m}, \quad l+m \leq n. \tag{4.6}
\]
and
\[
Q'_i(l, m, j, n) = \frac{(n-1)!}{l!(m-1)!j!(n-l-m-j)!} [1 - \Phi(i)]^l P(i)^{m-1} \pi_j \Phi(i+1)^{n-l-m-j}, \tag{4.7}
\]
where \( l + m + j \leq n \), respectively. Then the rest of the evaluation are the
same as the analytical model of LLMR with no alternative nodes. Observe that:
equations (3.16) through (3.22) with the replacement of (3.16) by (4.5), (3.17),
by (4.6), and (3.18) by (4.7) outline a procedure to obtain \( \Lambda_i \) from \( \Pi \); hence
\[
\Lambda_i = g_i(\Pi), \quad i = 0, \ldots, K - 1, \tag{4.8}
\]
where \( g_i(\cdot) \) is a function that is given by equations (4.4) through (4.7), and (3.19);
through (3.22).

**LLMR with An Alternative Node Allowed for Class 2 only**

The analytical model of LLMR(K) is the same as that of LLMR in this
case, except we replace the variable \( C \) (link capacity) by \( K - 1 \) (where \( K \) is the
number of aggregated states) and the equation (3.27) is changed to be

\[ v_i = 2M \sum_{j=0}^{t'} y_{\text{max}(i,j)} P(j), \quad i = 0, \ldots, t' \]

\[ = 2M \left[ y_i \sum_{j=0}^{i} P(j) + \sum_{j=i+1}^{t'} y_{j} P(j) \right], \quad i = 0, \ldots, t'. \quad (4.9) \]

The numerical example used is the same as the one in Section 3.6 of Chapter 3. In Table 4.5 for the case of no alternative node allowed, we see that the aggregated result for \( K = 5 \) is approximately the same as that of ALLMR(C+1), i.e. no aggregation. It implies that state aggregation with \( K = 5 \) is good enough to make the routing decision, compared with no aggregation. In Table 4.6 for the case of an alternative node allowed for class 2 only, the result is similar. The two examples show that state aggregation can reduce the computational complexity significantly.
<table>
<thead>
<tr>
<th>ALLMR(K)</th>
<th>Call blocking probability</th>
<th>Aggregated states</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$d = 2$</td>
<td>$d = 3$</td>
</tr>
<tr>
<td>$K = 2$</td>
<td>$1.35 \times 10^{-1}$</td>
<td>$5.00 \times 10^{-2}$</td>
</tr>
<tr>
<td>$K = 3$</td>
<td>$1.11 \times 10^{-1}$</td>
<td>$3.46 \times 10^{-2}$</td>
</tr>
<tr>
<td>$K = 4$</td>
<td>$1.00 \times 10^{-1}$</td>
<td>$2.82 \times 10^{-2}$</td>
</tr>
<tr>
<td>$K = 5$</td>
<td>$9.66 \times 10^{-2}$</td>
<td>$2.62 \times 10^{-2}$</td>
</tr>
<tr>
<td>$K = C + 1$</td>
<td>$9.55 \times 10^{-2}$</td>
<td>$2.56 \times 10^{-2}$</td>
</tr>
</tbody>
</table>

Table 4.5: The effect of number of aggregated states ($N = 30$, $C = 20$, $D = 4$, no alternative node is allowed and $L = 0.9$) (Note that the number inside the braces represents the actual number of free circuits)

<table>
<thead>
<tr>
<th>ALLMR(K)</th>
<th>Call blocking probability</th>
<th>Aggregated states</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$d = 2$</td>
<td>$d = 3$</td>
</tr>
<tr>
<td>$K = 3$</td>
<td>$8.68 \times 10^{-2}$</td>
<td>$6.55 \times 10^{-2}$</td>
</tr>
<tr>
<td>$K = 4$</td>
<td>$7.89 \times 10^{-2}$</td>
<td>$4.89 \times 10^{-2}$</td>
</tr>
<tr>
<td>$K = 5$</td>
<td>$7.68 \times 10^{-2}$</td>
<td>$4.34 \times 10^{-2}$</td>
</tr>
<tr>
<td>$K = C + 1$</td>
<td>$7.63 \times 10^{-2}$</td>
<td>$4.23 \times 10^{-2}$</td>
</tr>
</tbody>
</table>

Table 4.6: The effect of number of aggregated states ($N = 30$, $C = 20$, $D = 4$, an alternative node is allowed for class 2, trunk reservation = 4 free circuits and $L = 0.9$) (Note that the number inside the braces represents the actual number of free circuits)
4.2 Least Load Multicast Routing with the Consideration of Secondary Parameters

In Chapter 2, we define the normalized revenue loss as

\[
\frac{\sum_{d=2}^{D} (d - 1)\lambda_d B_d}{\sum_{d=2}^{D} (d - 1)\lambda_d}.
\] (4.10)

From the above equation, in order to maximize the normalized network revenue or minimize the normalized revenue loss, we need to minimize the call blocking probabilities of all classes. The intuitive reason of Least Load Multicast Routing (LLMR) to consider the number of free circuits as the link cost function is that this cost can reflect the current loading or resources of a link. The number of free circuits of a link indicates currently how many circuits are available and, based on this information, LLMR can evenly distribute the loading of a network to minimize the call blocking probability. Note that LLMR attempts to keep every link from becoming full so as to avoid call blocking. However, when more than one link have the same number of free circuits, LLMR cannot identify which one is the best choice. Hence, under this situation, LLMR will randomly choose one from them and thus it cannot make the best routing decision to establish the call connection. We propose two secondary parameters to help LLMR make a better decision.

1. Least Load Multicast Routing with Maximum Occupied Circuits (LLMR-MOC): To identify which link should be selected as a part of a connected tree, the number of occupied circuits is one of suitable secondary parameters. The link service rate is directly proportional to the number of occupied circuits of a link. Hence a larger number of occupied circuits of a link intuitively implies a shorter time to release a free circuit. Therefore when more than one link are in the least load state (i.e. having the same
maximum number of free circuits), a link with the largest number of occupied circuits is the best choice among them to be selected as a part of a connected tree.

2. Least Load Multicast Routing with Minimum Measured Blocking Time (LLMRMMBT): Link blocking probability is also a suitable secondary parameter to represent the loading of a link. Higher link blocking probability intuitively implies higher link loading. The link blocking probability can be measured by the ratio of the link blocking time to the total elapsed time. Since the total elapsed time of each link is the same, we measure the link blocking time instead. Thus, we choose a link with the smallest link blocking time when more than one link are in the least load state (i.e. having the same maximum number of free circuits).

4.2.1 The Network Model and The Proposed Routing Algorithm

Before proceeding further, we would like to introduce necessary definitions and notations. We consider a single rate loss network to be an undirected graph $G = (V, E)$ where $V$ and $E$ are the set of nodes and links respectively. Throughout we denote by $R(e), e \in E$, the number of free circuits, or residual capacity of link $e$. Denote by $M(e), e \in E$, the secondary parameter of link $e$. For LLMRMC, $M(e)$ is the number of occupied circuits, while, for LLMRMMBT, $M(e)$ is the negative value of the measured link blocking time (the value is negative since we shall choose the link with the minimum measured blocking time or the maximum value of $M(e)$).

Consider a connection request $c$ with destination set $S(c)$ (each node $s \in S(c)$ is referred to as a destination of $c$). Let $d(c)$ be the number of nodes involved
in a connection request $c$, i.e. destination size. A connection $c$ requires a connected, acyclic graph (tree) $T(c) = (V(c), E(c))$, where $S(c) \subseteq V(c)$ and a unit bandwidth is reserved on each link $e \in E(c)$. Nodes $s \in V(c) - S(c)$ are referred to as alternative nodes. A link, $e \in E(c)$, is referred to as an alternative link if it connects to an alternative node. Let $t(e)$ denote the reservation threshold for link $e$, i.e., link $e$ cannot be used as an alternative link if its current state $R(e)$ does not exceed the reservation threshold $t(e)$.

As direct links are always tried first, our proposed algorithms will attempt to build a connected tree $T(c)$ which contains all nodes in $S(c)$ and its cost $\alpha(T(c)) = \sum_{e \in E(c)} \alpha(e)$ is minimum compared with all possible connected trees. The cost function of link $e$, $\alpha(e)$, is defined as the negative value of $R(e)$, i.e. finding a minimum spanning tree is equivalent to finding a connected tree with maximum number of free circuits. When more than one link have the same minimum cost, we choose a link $e$ with the maximum value of $M(e)$. If a connected tree cannot be established (i.e. all the destination nodes cannot be connected by direct links only), our proposed algorithms will attempt to build a connected tree with one additional node, i.e. an alternative node, and the cost function of link $e$ (either direct or alternative) is defined as

$$\alpha(e) = \begin{cases} -R(e) & R(e) > t(e) \\ 0 & \text{otherwise}. \end{cases}$$ (4.11)

Note that in the selection of an alternative node, we still consider the secondary parameter $M(e)$ when more than one link have the same minimum cost. We select a connected tree with the minimum cost among all possible cases of one alternative node. If a connected tree still cannot be established, the call connection request will be rejected. Our proposed algorithm is given in Figure 4.3.
begin
Construct a minimum spanning tree \( T(c) = (S(c), E(c)) \) with \( \alpha(e) = -R(e) \).
When more than one link have the minimum link cost,
choose the one with maximum \( M(e) \).
if a minimum spanning tree cannot be established,
set \( \alpha(T_{\text{best}}) = 0 \).
for each \( s \in V(c) - S(c) \) (alternative node),
construct a minimum spanning tree \( T(c) = (S(c) + s, E(c)) \),
with \( \alpha(e) = \begin{cases} 
- R(e) & \text{if } R(e) > t(e) \\
0 & \text{otherwise.} 
\end{cases} \)
When more than one link have the minimum link cost,
choose the one with maximum \( M(e) \).
if a minimum spanning tree is established and \( \alpha(T_{\text{best}}) > \alpha(T(c)) \),
set \( \alpha(T_{\text{best}}) = \alpha(T(c)) \) and \( T_{\text{best}} = T(c) \).
endif
end for
if a minimum spanning tree cannot be established,
the call connection request is rejected;
else
the tree is \( T_{\text{best}} \) and done;
endif
else
done;
endif
end

Figure 4.3: MLLMR algorithms

Four points need to be clarified. (i) A link cannot be selected if it does not have any free circuits. (ii) A connected tree cannot be established if there exists some unconnected destination nodes. (iii) We allow at most one alternative node when no connected tree with direct links only is found because each additional alternative node requires one additional link to form a connected tree. This wastes resources and the routing complexity also increases exponentially as the number of alternative nodes increases. (iv) The dynamic add/drop procedures are not shown in the above algorithm because they are quite straightforward. If a node requests to join a connected tree, it should choose a link with minimum
link cost to connect from the node to the tree. For simplicity, if there is no direct link available, the request will be denied. For the procedure to drop a node, if there is only one link connected from the node to its connected tree, the link will be released; otherwise, nothing is changed.

4.2.2 Illustrative Example

To illustrate the routing algorithm outlined above, we consider a 4-node network and a connection request with destination set \{r, a, b, c\} with node r as the source node. The steps in the connection establishment procedure are as shown in Figure 4.4 — the number next to a link denotes its link cost, i.e. the negative value of residual capacity, and the number in parentheses next to a link denotes its secondary parameter, \(M(e)\). The secondary parameter is the number of occupied circuits in the LLMRMO algorithm while in the LLMRMBT algorithm it is the negative value of the measured blocking time. The links that are included in the set \(E(c)\) are drawn with “thick” lines. When the tree searching procedure starts from node r, we find that links \((r, b)\) and \((r, c)\) have the same minimum link cost of -2. Since the value of the secondary parameter of link \((r, b)\) is less than that of link \((r, c)\), node c is the first one to be added which is followed by node a (link \((c, a)\) is chosen due to its smaller link cost over links \((r, a), (r, b)\) and \((c, b)\)). Finally, we choose link \((a, b)\) because the link cost of \((a, b)\) is the smallest compared with links \((r, b)\) and \((b, c)\). After iteration 4 the procedure is completed and a tree is established for the connection request.

To select an alternative node, we randomly pick an alternative node and treat it as an additional destination node. The procedure of searching a minimum spanning tree is restarted. Finally, we choose the tree with the minimum cost among all possible cases of one alternative node.
Figure 4.4: Example to illustrate our proposed algorithm without alternative nodes

4.2.3 The Implementation Issues

Before proceeding to present the performance of the two proposed algorithms (which will be done in Section 4.2.4), we would like to discuss about the implementation issues.

LLMRMOC

The secondary parameter of LLMRMOC is the number of occupied circuits, which is equal to the difference between the link capacity and the number of free circuits, both of which can be easily obtained. To implement the algorithm, we need to consider the link cost and sometimes the secondary parameter. To minimize the change of implementation from LLMR to LLMRMOC, we combine the original link cost and the secondary parameter as a new link cost function which is defined as:

\[ \alpha'(e) = -(R(e) + \frac{M(e)}{R(e) + M(e)}) \]  

(4.12)
Note that, for links $e_1$ and $e_2$, if $R(e_1) = R(e_2) = R(e)$ and $M(e_1) > M(e_2)$, we have the following relationship:

$$-(R(e) - 1) > \alpha'(e_2) > \alpha'(e_1) > -(R(e) + 1).$$  \hfill (4.13)

Therefore, comparing this new cost function directly is equivalent to taking into account the secondary parameter when needed. Based on this relationship, we can simply replace $\alpha(e)$ by $\alpha'(e)$ to upgrade the routing algorithm from LLMR to LLMRMOC without any other changes.

**LLMRMMBT**

The secondary parameter of LLMRMMBT is the negative value of the measured blocking time $B(e)$, which can be obtained by setting a counter to record the time elapsed when a link is in the blocking state. Similar to LLMRMOC a new link cost function is proposed to combine the original link cost and the secondary parameter. The new link cost function is defined as:

$$\alpha'(e) = -(R(e) + (1 - \frac{B(e)}{T}))),$$  \hfill (4.14)

where $T$ is the total measurement time. For links $e_1$ and $e_2$, if $R(e_1) = R(e_2) = R(e)$ and $M(e_1) > M(e_2)$ (i.e. $B(e_1) < B(e_2)$), we have the following relationship:

$$-(R(e) - 1) > \alpha'(e_2) > \alpha'(e_1) > -(R(e) + 1).$$  \hfill (4.15)

Therefore, comparing this new cost function directly is also equivalent to taking the secondary parameter into account. Again, we can replace $\alpha(e)$ by $\alpha'(e)$ to upgrade the routing algorithm from LLMR to LLMRMMBT without any additional changes.
4.2.4 Numerical Results

In this section, we give numerical results to illustrate the performance of the two Modified Least Load Multicasting Routing (MLLMR) algorithms, LLMR with Maximum Occupied Circuits (LLMRMOC) and LLMR with Minimum Measured Blocking Time (LLMRMMBT), when compared with LLMR proposed in [17] and [8]. We consider a fully connected network with 8 nodes. The capacity of each link is randomly generated between 30 and 70 bandwidth units or circuits. Multicast connection requests are fully symmetrical and identically distributed. There are three classes of call connection requests in the network and their destination sizes are 2, 3 and 4. The loadings of the three classes are equal and the call arrivals of each class follow a Poisson process. All call holding times are exponentially distributed with unit mean. All processing time, including call setup and release time, are negligible. When a call connection request is rejected, it will not retry and is cleared immediately from the network. For each set of simulation point, the length of each run is $10^6$ units of mean interarrival time of connection request and the initial 10% is discarded to avoid the effect of transient state. The vertical lines about each point represent the 95% confidence intervals. We define the network loading as the ratio of the total offered load to the total network capacity and set 80% as the engineered load (i.e. 0% overload). For alternative routing, we set the trunk reservation threshold of each link to 5 circuits. We use a performance measure called normalized revenue loss defined in equation (4.9).

Figures 4.5 and 4.6 show the mean call blocking probabilities for LLMR and MLLMR under the same range of overload conditions for the cases of no alternative nodes and at most one alternative node allowed, respectively. We find that the two MLLMR algorithms, LLMRMOC and LLMRMMBT, always outperform LLMR in terms of achieving lower call blocking probabilities for
all three classes. Note that the call blocking probabilities in the figures are plotted in logarithm scale. The performance of two MLLMR algorithms is better because they make use of additional information (the secondary parameters) to forecast the change of the number of free circuits. Moreover, the performance of LLMRMMBT is better than LLMRMOC. It is because, for LLMRMMBT, the secondary parameter can more accurately predict the change in the number of free circuits. For example, if the call arrival rate of a link is sufficiently high, even the number of occupied circuits is large, the number of free circuits may decrease in the near future. For LLMRMMBT, larger link arrival rate will likely increase the measured blocking time and thus reduce the probability of the link being selected. Thus, the secondary parameter of LLMRMMBT is better than that of LLMRMOC to represent the change in the number of free circuits and hence the performance of LLMRMMBT is better.

The relative improvement of the normalized revenue loss of MLLMR over LLMR are shown in Figures 4.7 and 4.8. The relative improvement is up to 14% because the MLLMR algorithms can take advantage over LLMR only when more than one link have the same maximum number of free circuits and this may not happen frequently. Although the relative improvement is moderate, the main contribution of the MLLMR algorithms is that the gain in performance only comes with minimal additional cost. For LLMRMOC, there is no additional cost at all, while for LLMRMMBT we only require one additional counter for each link to measure the mean blocking time. Moreover, we also find that the relative improvement decreases when the amount of overload increases. It is because when the network utilization becomes saturated, the network resources are not able to accept more incoming calls and hence no algorithm can improve the network performance under this condition.
Figure 4.5: Mean call blocking probabilities of MLLMR and LLMR (no alternative node is allowed)

Figure 4.6: Mean call blocking probabilities of MLLMR and LLMR (at most one alternative node is allowed)
Figure 4.7: Relative improvement of revenue loss of MLLMR over LLMR (no alternative node is allowed)

Figure 4.8: Relative improvement of revenue loss of MLLMR over LLMR (at most one alternative node is allowed)
Chapter 5

Maximum Mean Number of New Calls Accepted Before Blocking Multicast Routing (MCBMR)

In this chapter, we propose a new multicast routing algorithm called Maximum Mean Number of New Calls Accepted Before Blocking Multicast Routing (MCBMR), which can more accurately capture the current and future loading of a network. Simulation results show that this algorithm, compared with LLMR, not only has a smaller network revenue loss, but also results in smaller call blocking probabilities for all classes of traffic. We also discuss the implementation issues of our proposed algorithm and develop two approximation methods, state approximation and curve fitting, which can reduce the measurement complexity significantly with only a slight performance degradation.

5.1 The Mean Number of New Calls Accepted Before Blocking (MCB)

We now introduce the parameter to measure the performance of a network in multicast routing. In point-to-point communication, the end-to-end call blocking probability is the most common measure of quality of a network. However,
there are more than one class of call connection requests (a class here means a group of multicast call connections with the same destination size) and thus the above measure cannot be directly applied in multicast routing. For evaluating the performance of multicast routing algorithms, we again use the performance measure called normalized revenue loss as defined by equation (2.1).

From the equation to maximize the normalized network revenue or minimize the normalized revenue loss, we need to minimize the call blocking probabilities of all classes. The intuitive reason of Least Load Multicast Routing (LLMR) to consider the number of free circuits as the link cost function is that this cost can reflect the current loading or resources of a link. The number of free circuits of a link indicates currently how many circuits are available and, based on this information, LLMR can evenly distribute the loading of a network to minimize the call blocking probability. However, the LLMR approach does not take into account the future call arrivals of the network and thus sometimes cannot make a good decision. In this paper, we propose a new link cost function, the mean number of new calls accepted before blocking, which can more accurately reflect the current and future loading of a link. Its value reflects more accurately the actual available network resources that one can make use of before blocking.

Consider the following example: a multicast call request arrives to a single rate-loss network and, during the routing decision, a link will be chosen from two links whose information are shown in Table 5.1. Both the call interarrival time and the call holding time of the two links are exponentially distributed. The current link offered loads of link 1 and link 2 are 45 circuits/sec. and 40 circuits/sec. respectively. Based on the number of free circuits available, LLMR will choose link 1 because the number of free circuits of link 1 (5 circuits) is more than that of link 2 (2 circuits). However, by considering the value of MCB (the mean number of new calls accepted before blocking), the new link cost function shows
that link 1 can accept 60.4 new calls before blocking while link 2 can accept 75.5 new calls before blocking. Therefore, based on the values of MCB, link 2 will be chosen instead of link 1. This example shows that the number of free circuits alone cannot accurately reflect the actual available resources of a link.

5.1.1 The Network Model and The Proposed Routing Algorithm

Before proceeding further, we would like to introduce necessary definitions and notations. We consider a single rate loss network to be an undirected graph $G = (V, E)$ where $V$ and $E$ are the set of nodes and links respectively. Throughout we denote by $R(e), e \in E$, the number of free circuits, or residual capacity, of link $e$. Denote by $\hat{M}(e), e \in E$, the mean number of new calls accepted before blocking for link $e$.

Consider a connection request $c$ with destination set $S(c)$ (each node $s \in S(c)$ is referred to as a destination of $c$). Let $d(c)$ be the number of nodes involved in a connection request $c$, i.e. destination size. A connection $c$ requires a connected, acyclic graph (tree) $T(c) = (V(c), E(c))$, where $S(c) \subseteq V(c)$ and a uni-link bandwidth is reserved on each link $e \in E(c)$. Nodes $s \in V(c) - S(c)$ are referred to as alternative nodes. A link, $e \in E(c)$, is referred to as an alternative link:

<table>
<thead>
<tr>
<th>link information</th>
<th>link 1</th>
<th>link 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>link capacity</td>
<td>50 circuits</td>
<td>50 circuits</td>
</tr>
<tr>
<td>call holding time</td>
<td>1 sec.</td>
<td>1 sec.</td>
</tr>
<tr>
<td>current link offered load</td>
<td>45 circuits/sec.</td>
<td>40 circuits/sec.</td>
</tr>
<tr>
<td>number of free circuits</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>$MCB$</td>
<td>60.4</td>
<td>75.5</td>
</tr>
</tbody>
</table>

Table 5.1: The link information of two links
if it connects to an alternative node. Let \( t(e) \) denote the reservation threshold for link \( e \), i.e., link \( e \) can only be used as an alternative link if \( R(e) \) exceeds the reservation threshold \( t(e) \).

As direct links are always tried first, our proposed algorithm will attempt to build a connected tree \( T(c) \) which contains all nodes in \( S(c) \) and its cost \( \alpha(T(c)) = \sum_{e \in E(c)} \alpha(e) \) is minimum compared with all possible connected trees. The cost function of link \( e \), \( \alpha(e) \), is defined as the negative value of \( \dot{M}(e) \). Finding a minimum spanning tree is thus equivalent to finding a connected tree with maximum value of MCB. If a connected tree cannot be established (i.e. all the destination nodes cannot be connected by direct links only), our proposed algorithm will attempt to build a connected tree with one additional node (i.e. an alternative node) and the cost function of link \( e \) (either direct or alternative) is now defined as

\[
\alpha(e) = \begin{cases} 
-\dot{M}(e) & R(e) > t(e) \\
0 & \text{otherwise.}
\end{cases}
\] (5.1)

We select a connected tree with minimum cost from all possible cases. If a connected tree still cannot be established, the call connection request will be rejected. Our proposed algorithm is given in Figure 5.1.

Four points need to be clarified. (i) A link cannot be selected if it does not have any free circuits. Note that the value of \( \dot{M}(e) \) will be zero if link \( e \) does not have any free circuits. (ii) A connected tree has not been established if there exists some unconnected destination nodes. (iii) We allow at most one alternative node when no connected tree with direct links only is found. It is because each additional alternative node requires one additional link to form a connected tree and this wastes network resources. In addition, the routing complexity increases exponentially when the number of alternative nodes increases. (iv) The dynamic add/drop procedures are not shown in the above algorithm because they are quite straightforward. If a node requests to join a connectec
begin
Construct a minimum spanning tree \( T(c) = (S(c), E(c)) \) with \( \alpha(e) = -\hat{M}(e) \).
if a minimum spanning tree cannot be established,
set \( \alpha(T_{\text{best}}) = 0 \).
for each \( s \in V(c) - S(c) \) (alternative node),
construct a minimum spanning tree \( T(c) = (S(c) + s, E(c)) \),
with \( \alpha(e) = \begin{cases} 
-\hat{M}(e) & \text{if } R(e) > t(e) \\
0 & \text{otherwise.} 
\end{cases} \)
if a minimum spanning tree is established and \( \alpha(T_{\text{best}}) > \alpha(T(c)) \),
set \( \alpha(T_{\text{best}}) = \alpha(T(c)) \) and \( T_{\text{best}} = T(c) \).
end
end for
if a minimum spanning tree cannot be established,
the call connection request is rejected;
else
the tree is \( T_{\text{best}} \) and done;
else
done;
end
end

Figure 5.1: The MCBMR algorithm

tree, it should choose a link with minimum link cost to connect from the node to the tree. For simplicity, if there is no direct link available, the request will be denied. For the procedure to drop a node, if there is only one link connecting from the node to its connected tree, the link will be released; otherwise, nothing is changed.

5.1.2 Illustrative Example

To illustrate the routing algorithm outlined above, we consider a 4-node network and a connection request with destination set \( \{r, a, b, c\} \) with node \( r \) as the source node. The steps in the connection establishment procedure are as shown in Figure 5.2 — the number next to a link denotes its link cost, the negative value of MCB, and links that are included in the set \( E(c) \) are drawn
with “thick” lines. Node $c$ is the first one to be added as link $(r, c)$ has a link cost of -3 that is smaller than the link costs of links $(r, a)$ and $(r, b)$ which is followed by node $a$ (link $(c, a)$ is chosen due to its smaller link cost over links $(r, a)$, $(r, b)$ and $(c, b)$). Finally, we choose link $(a, b)$ because the link cost of $(a, b)$ is the smallest compared with links $(r, b)$ and $(b, c)$. After iteration 4, the procedure is completed and a tree is established for the connection request.

![Diagram showing iterations of the algorithm with nodes and link costs](image)

**Figure 5.2:** Example to illustrate our proposed algorithm without alternative nodes

Figure 5.3 shows the operation of selecting an alternative node. Given that each link’s residual capacity exceeds the trunk reservation threshold, except link $(r, a)$. Since there are no free circuits on link $(r, a)$ (assume a link in state 0 is full), two available two-link alternative routes are considered to connect nodes $r$ and $a$: $(r, b, a)$ and $(r, c, a)$. The cost of $(r, b, a)$ is -7, which is larger than that of $(r, c, a)$ (its cost is -9). Thus we choose node $c$ as the alternative node and $(r, c, a)$ as the two-link alternative route.
Figure 5.3: Example to illustrate our proposed algorithm with an alternative node

5.2 The Implementation Issues and Approximation Methods

To obtain the value of the mean number of new calls accepted before blocking (MCB), the obvious approach is to measure the value directly. However, the method of direct measurement cannot frequently update the MCB values and can update them only when a link is blocked. Thus this method cannot accurately determine the current MCB values. In this section, we introduce an indirect measurement method to obtain the MCB values more accurately and propose two approximation methods to reduce the measurement complexity.

5.2.1 Indirect Measurement

Consider a fully connected single rate loss network. When a multicast call request arrives, a set of direct links will be considered and, if available, a connected tree is established among them for establishing the call. Again we assume all multicast call connection requests arrive according to a Poisson process and the holding times of accepted calls are exponentially distributed. With the link independence assumption, the queueing model of a link can be modeled as a general M/M/N/N queue as shown in Figure 5.4, where 1/μ is the mean holding time of calls and N is the total number of circuits of the link. In addition,
given that there are \( k \) occupied circuits in the link, we define \( \lambda_k \) and \( M_k \) as the state dependent arrival rate at state \( k \) and the mean number of new calls that can be accepted starting from state \( k \) before reaching state \( k + 1 \), respectively.

![Diagram of M/M/N/N queueing model](image)

Figure 5.4: General M/M/N/N queueing model

By examining the possible state transitions from state \( k \), it is not difficult to see that \( M_k \) satisfies the first-order difference equation

\[
M_k = \frac{\lambda_k}{\lambda_k + k\mu} + \frac{k\mu}{\lambda_k + k\mu} [M_{k-1} + M_k] \tag{5.2}
\]

with the initial condition \( M_0 = 1 \). Equation (5.2) can be simplified as

\[
M_k = \frac{k\mu}{\lambda_k} M_{k-1} + 1 \tag{5.3}
\]

and thus \( M_k \) can be easily obtained from the above recursive equation. Let \( \hat{M} \) be the mean number of new calls that can be accepted starting from state \( i \) before blocking, i.e. the mean number of new calls accepted starting from state \( i \) before reaching state \( N \). It is easy to observe that \( \hat{M}_i \) equals to the sum of \( M_k, k = i, i + 1, \ldots, N - 1 \), i.e.

\[
\hat{M}_i = \sum_{k=i}^{N-1} M_k. \tag{5.4}
\]

Therefore, we can compute the link cost by knowing the state dependent arrival rates, which can be determined by measuring the mean state sojourn time.

In the calculation of the mean number of new calls accepted before blocking, we make use of the state dependent arrival and departure rates of links, which
can be obtained by real-time measurements during network operations. For the
state dependent departure rates, we can measure the mean call holding time and
hence obtain the call service rate. The state dependent departure rates simply
equal to the number of occupied circuits multiplied by the call service rate.
However, for the state dependent arrival rates, they are much more difficult to
measure since they are all different and do not have any obvious relationships
among them. Direct measurement of the state dependent arrival rates, which
directly measures the number of call arrivals and the mean state sojourn time, is
one possible way to obtain $\lambda_k$. Instead of using the direct approach, we propose
two simpler ways to obtain $\lambda_k$:

1. \textit{To measure the mean state sojourn time:} From Figure 5.4, it is easy to
   observe that the mean state sojourn time at state $k$, $T_k$, is

   $$\frac{1}{\lambda_k + k\mu}. \quad (5.5)$$

   Since $\mu$ can be obtained by measuring the mean call holding time, we can
determine $\lambda_k$ by using the following expression

   $$\lambda_k = \frac{1}{T_k} - k\mu \quad (5.6)$$

   with the measured value of the mean state sojourn time, $T_k$.

2. \textit{To measure the state transition probability:} From Figure 5.4, the state
   transition probabilities, $p_k^{(u)}$, from state $k$ to state $k + 1$ and, $p_k^{(d)}$, from
   state $k$ to state $k - 1$ are, respectively,

   $$\frac{\lambda_k}{\lambda_k + k\mu} \quad (5.7)$$

   and

   $$\frac{k\mu}{\lambda_k + k\mu}. \quad (5.8)$$
We can measure either $p_k^{(u)}$ or $p_k^{(d)}$ to determine $\lambda_k$ by using one of the following expressions:

$$\lambda_k = \frac{p_k^{(u)} k \mu}{1 - p_k^{(u)}}$$

(5.9)

and

$$\lambda_k = \frac{(1 - p_k^{(d)}) k \mu}{p_k^{(d)}}.$$  

(5.10)

Note that sometimes, especially in the first few states (with a small number of occupied circuits), the measured $p_k^{(u)}$ may be equal to 1 or the measured $p_k^{(d)}$ may be equal to 0. Under this situation, the above expressions for $\lambda_k$ are undefined. To overcome this problem, we can assign a sufficiently large value as the state dependent arrival rates for these first few states and this almost does not affect the calculation of the mean number of new calls accepted since a large value of $\lambda_k$ simply makes the first term of the right hand side of Equation (5.4) become very small, compared with the second term.

Although the indirect measurement method can update the current MCB value more frequently, the number of data points of a link to be measured is still large and the measurement complexity is high. The following two approximation methods are proposed to reduce the number of data points (i.e. measurement complexity) significantly with only a slight degradation in performance.

### 5.2.2 Approximation Method 1: State Approximation

As we have mentioned above, when the state of a link, say $k$, is low (i.e. the number of occupied circuits is small), the value of $\lambda_k$ will be large and thus,

$$M_k = \frac{k \mu}{\lambda_k} M_{k-1} + 1 \approx 1 \quad if \quad \lambda_k \gg \mu.$$  

(5.11)

Based on this observation, we propose to develop an approximation of the computation of the link cost function: If the number of measurement points of a
link is $K$ and the link’s total number of circuits is $N$, then we approximate $M_k$ by $M'_{k}$, for $k = 0, 1, \ldots, N$, where

$$M'_{k} = \begin{cases} 
1 & k = 0, \ldots, N - K - 1 \\
\frac{k}\lambda_k M'_{k-1} + 1 & k = N - K, \ldots, N - 1.
\end{cases}$$ (5.12)

We assume when $k < N - K$, the state dependent arrival rates are sufficiently large such that $M_k$, $0 \leq k \leq N - K - 1$, can be approximated by 1. Thus we can reduce the measurement complexity from measuring $N$ data points to $K$ data points, for $\lambda_k$, $k = N - K, \ldots, N - 1$.

### 5.2.3 Approximation Method 2: Curve Fitting

By observing the shape of the curve of state dependent arrival rates for a link under MCBMR routing (the graph will be shown in Section 5.3), we find that the function of the state dependent arrival rate is monotonic decreasing as the state increases (or the number of free circuits decreases). Moreover, the state dependent arrival rate equals zero when the number of free circuit is zero (i.e. blocking state). Since the shape of the curve for each link is similar, based on the above information, we propose to apply the curve fitting technique [27] to estimate the state dependent arrival rates of all states. To satisfy the two main properties of the curve of state dependent arrival rates, the following curve fitting function is proposed:

$$y = a(N - x)^b$$ (5.13)

where $x$ is the state (the number of occupied circuits), $y$ is the state dependent arrival rate, $N$ is the total number of circuits, $a$ and $b$ are positive real constants to be determined. We apply Least Square Error Minimization to find the values of $a$ and $b$. The approximation of the computation of the link cost function is proposed as follows: If the number of measurement points of a link is $K$ and the link’s total number of circuits is $N$, we approximate $\lambda_k$ by $\lambda'_k$, $k = 0, 1, \ldots, N - 1$. 
such that
\[ \lambda'_k = \begin{cases} a(N-k)^b & k = 0, \ldots, N-K-1 \\ \lambda_k & k = N-K, \ldots, N-1 \end{cases} \]  
(5.14)
where
\[ b = \frac{K \sum_{j=1}^{K} \log j \lambda_{N-j} - (\sum_{j=1}^{K} \log j)(\sum_{j=1}^{K} \log \lambda_{N-j})}{K \sum_{j=1}^{K} \log^2 j - (\sum_{j=1}^{K} \log j)^2} \]  
(5.15)
and
\[ a = \exp\left(\frac{1}{K} \left(\sum_{j=1}^{K} \log \lambda_{N-j} - b \sum_{j=1}^{K} \log j\right)\right). \]  
(5.16)

Again we can reduce the measurement complexity from \( N \) data points to \( K \) data points.

5.3 Numerical Results

In this section, we give numerical results to illustrate the performance of the MCBMR algorithm when compared with Least Load Multicast Routing (LLMR) proposed in [17] and [8]. We consider a fully connected network with 8 nodes. The capacity of each link is randomly generated between 40 and 60 bandwidth units or circuits. There are three classes of call connection requests in the network and their destination sizes are 2, 3 and 4. The loadings of the three classes are equal and the call arrivals of each class follow a Poisson Process. The traffic matrix of multicast connection requests is randomly generated. All call holding times are exponentially distributed with unit mean. All processing time, including call setup and release time, are negligible. When a call connection request is rejected, it will not retry and is cleared immediately from the network. For each set of simulation point, the length of each run is \( 10^6 \) units of mean interarrival time of connection request and the initial 10% is discarded to avoid the effect of transient state. The vertical lines about each point represent the 95% confidence intervals. We define the network loading as the ratio of the total offered load to the total network capacity and set 80% as the engineered
load (i.e. 0% overload). For alternative routing, we set the trunk reservation threshold of each link as 5 circuits. We use the performance measure called normalized revenue loss introduced in Section 2.

5.3.1 Performance of MCBMR Algorithm

In this section, the performance of our proposed MCBMR algorithm is compared with that of LLMR. The new link cost (i.e. the MCB value) is computed from Equations (5.4) and (5.5), where all state dependent arrival rates, \( \lambda_k \), are obtained using the indirect measurement method. Figures 5.5 and 5.6 show the mean call blocking probabilities for LLMR and MCBMR under the same range of overload conditions for the cases of no alternative nodes and at most one alternative node respectively. We find that MCBMR outperforms LLMR in terms of achieving lower call blocking probabilities for all three classes in most cases. It is because LLMR just captures the current information (the current loading of links) but MCBMR makes use of all information (the current state of the network and the state dependent arrival rates of links) to forecast the future congestion.

The relative improvement of the normalized revenue loss of MCBMR over LLMR are shown in Figures 5.7 and 5.8. We find that the relative improvement can be up to 50% and, as we expect, it decreases when the overload increases. It is because when the network utilization becomes saturated, the network resources are insufficient to accept incoming calls and hence no algorithm can improve the network performance under this condition.

Figure 5.9 shows the simulation traces of the normalized revenue loss for MCBMR algorithm as a function of time under different loading conditions. It shows that, without a priori knowledge of the required traffic parameters, the MCBMR algorithm can learn how to adjust the required parameters through
real-time measurement without any oscillation problem.

5.3.2 Performance of the Two Approximation Methods

Figure 5.10 shows the values of MCB, $\hat{M}_k$, of a link against the link state (the number of occupied circuits). The solid curve is obtained by using all $\lambda_k$, $k = 0, \ldots, N - 1$, obtained from indirect measurement, while the others are obtained by measuring only 10 data points of $\lambda_k$, $k = N - 10, \ldots, N - 1$, using the two approximation methods discussed earlier. The 10 data points are the state dependent arrival rates of the last 10 states. We observe that the difference between the simulation curve and the approximation curves is large when the link state is low. However, from another point of view, when the link state is low, there are many free circuits available and thus the routing decision does not significantly affect the network performance. We need only the relative link cost for comparison purpose.

Figure 5.11 shows the the state dependent arrival rate, $\lambda_k$, against the link state. We see the curve fitting approximation is close to the simulation curve when the link state is high. From the tend of the fitting curve, we observe that the difference between the fitting curve and the simulation result will be large when the link state is low. However, based on the similar argument in the previous paragraph, this difference does not significantly affect the network performance.

Tables 5.2 and 5.3 show the performance of the two approximation methods. We find that a few data points are good enough for both approximation methods to obtain good results close to the simulation result. When the number of data points is up to 6, there are almost no difference between the simulation result and the approximation one, but the measurement complexity is reduced as much as 88%.
Tables 5.4 and 5.5 show how the network loading affect the performance of the two approximation methods. We find that when the network loading is low, the difference between the exact result and the approximation results is slightly larger than that when the network loading is high. It implies that more data points are required to produce a good approximation of the network performance when the network loading is low. When the amount of overload is above 5%, a reasonably small number of data points is good enough for both approximation methods to produce good results. Moreover, when the network loading is low, the performance of the curve fitting method is better than that of the state approximation method.

Figure 5.5: Mean call blocking probabilities of MCBMR and LLMR (no alternative node is allowed)
Figure 5.6: Mean call blocking probabilities of MCBMR and LLMR (at most one alternative node is allowed)

Figure 5.7: Relative improvement of revenue loss of MCBMR over LLMR (no alternative node is allowed)
Figure 5.8: Relative improvement of revenue loss of MCBMR over LLMR (at most one alternative node is allowed)

Figure 5.9: Normalized revenue loss vs. time (at most one alternative node is allowed)
Figure 5.10: MCB value vs. link state (no alternative node is allowed)

Figure 5.11: State dependent arrival rate vs. link state (no alternative node is allowed)
<table>
<thead>
<tr>
<th>Number of data points</th>
<th>Class</th>
<th>Call blocking probability</th>
<th>Curve fitting</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>State approximation</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>$3.66 \times 10^{-2}$</td>
<td>$5.12 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$8.60 \times 10^{-3}$</td>
<td>$9.16 \times 10^{-3}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$3.96 \times 10^{-3}$</td>
<td>$4.12 \times 10^{-3}$</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>$3.45 \times 10^{-2}$</td>
<td>$3.78 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$8.74 \times 10^{-3}$</td>
<td>$7.41 \times 10^{-3}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$3.80 \times 10^{-3}$</td>
<td>$3.36 \times 10^{-3}$</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>$3.47 \times 10^{-2}$</td>
<td>$3.60 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$8.17 \times 10^{-3}$</td>
<td>$8.60 \times 10^{-3}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$3.56 \times 10^{-3}$</td>
<td>$3.96 \times 10^{-3}$</td>
</tr>
<tr>
<td>(N) (Exact Method)</td>
<td>2</td>
<td>$3.53 \times 10^{-2}$</td>
<td>$3.53 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$8.22 \times 10^{-3}$</td>
<td>$8.22 \times 10^{-3}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$3.49 \times 10^{-3}$</td>
<td>$3.49 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

Table 5.2: The effect of the number of measured data points on the performance of the two approximation methods (10% overload and no alternative nodes are allowed)
<table>
<thead>
<tr>
<th>Number of data points</th>
<th>Call blocking probability</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class</td>
<td>State approximation</td>
<td>Curve fitting</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>$2.31 \times 10^{-2}$</td>
<td>$2.87 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$8.56 \times 10^{-3}$</td>
<td>$8.78 \times 10^{-3}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$4.38 \times 10^{-3}$</td>
<td>$4.24 \times 10^{-3}$</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>$2.24 \times 10^{-2}$</td>
<td>$2.36 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$8.79 \times 10^{-3}$</td>
<td>$7.84 \times 10^{-3}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$4.22 \times 10^{-3}$</td>
<td>$3.73 \times 10^{-3}$</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>$2.26 \times 10^{-2}$</td>
<td>$2.27 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$8.27 \times 10^{-3}$</td>
<td>$8.18 \times 10^{-3}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$4.13 \times 10^{-3}$</td>
<td>$3.98 \times 10^{-3}$</td>
</tr>
<tr>
<td>N</td>
<td>2</td>
<td>$2.26 \times 10^{-2}$</td>
<td>$2.26 \times 10^{-2}$</td>
</tr>
<tr>
<td>(Exact Method)</td>
<td>3</td>
<td>$8.36 \times 10^{-3}$</td>
<td>$8.36 \times 10^{-3}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$3.95 \times 10^{-3}$</td>
<td>$3.95 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

Table 5.3: The effect of the number of measured data points on the performance of the two approximation methods (10% overload, at most one alternative node is allowed and trunk reservation is 5 circuits)
<table>
<thead>
<tr>
<th>% Overload</th>
<th>Class</th>
<th>Exact method</th>
<th>State approximation</th>
<th>Curve fitting</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>$3.19 \times 10^{-3}$</td>
<td>$5.68 \times 10^{-3}$</td>
<td>$6.62 \times 10^{-3}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$2.80 \times 10^{-4}$</td>
<td>$9.07 \times 10^{-4}$</td>
<td>$3.44 \times 10^{-4}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$6.05 \times 10^{-5}$</td>
<td>$2.02 \times 10^{-4}$</td>
<td>$9.10 \times 10^{-5}$</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>$1.27 \times 10^{-2}$</td>
<td>$1.31 \times 10^{-2}$</td>
<td>$1.48 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$2.04 \times 10^{-3}$</td>
<td>$2.47 \times 10^{-3}$</td>
<td>$1.78 \times 10^{-3}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$7.19 \times 10^{-4}$</td>
<td>$7.95 \times 10^{-4}$</td>
<td>$6.77 \times 10^{-4}$</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>$3.53 \times 10^{-2}$</td>
<td>$3.47 \times 10^{-2}$</td>
<td>$3.60 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$8.21 \times 10^{-3}$</td>
<td>$8.17 \times 10^{-3}$</td>
<td>$7.83 \times 10^{-3}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$3.49 \times 10^{-3}$</td>
<td>$3.56 \times 10^{-3}$</td>
<td>$3.36 \times 10^{-4}$</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>$6.84 \times 10^{-2}$</td>
<td>$6.83 \times 10^{-2}$</td>
<td>$6.91 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$2.08 \times 10^{-2}$</td>
<td>$2.05 \times 10^{-2}$</td>
<td>$2.04 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$1.04 \times 10^{-2}$</td>
<td>$1.06 \times 10^{-2}$</td>
<td>$1.05 \times 10^{-2}$</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>$1.07 \times 10^{-1}$</td>
<td>$1.07 \times 10^{-1}$</td>
<td>$1.07 \times 10^{-1}$</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>$3.94 \times 10^{-2}$</td>
<td>$3.89 \times 10^{-2}$</td>
<td>$3.90 \times 10^{-2}$</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>$2.22 \times 10^{-2}$</td>
<td>$2.28 \times 10^{-2}$</td>
<td>$2.23 \times 10^{-2}$</td>
</tr>
</tbody>
</table>

Table 5.4: The effect of overloading on the performance of the two approximation methods (the number of measured data points is 6 and no alternative nodes are allowed)
| % Overload | Call blocking probability |  |
|-----------|---------------------------|--|--|--|--|--|
|           | Class | Exact method | State approximation | Curve fitting |
| 0         | 2     | $6.27 \times 10^{-4}$ | $1.10 \times 10^{-3}$ | $1.04 \times 10^{-3}$ |
|           | 3     | $1.72 \times 10^{-4}$ | $3.83 \times 10^{-4}$ | $1.76 \times 10^{-4}$ |
|           | 4     | $6.66 \times 10^{-5}$ | $1.20 \times 10^{-4}$ | $5.55 \times 10^{-5}$ |
| 5         | 2     | $5.44 \times 10^{-3}$ | $5.57 \times 10^{-3}$ | $5.77 \times 10^{-3}$ |
|           | 3     | $1.73 \times 10^{-3}$ | $1.96 \times 10^{-3}$ | $1.56 \times 10^{-3}$ |
|           | 4     | $7.38 \times 10^{-4}$ | $7.49 \times 10^{-4}$ | $6.43 \times 10^{-4}$ |
| 10        | 2     | $2.26 \times 10^{-2}$ | $2.26 \times 10^{-2}$ | $2.29 \times 10^{-2}$ |
|           | 3     | $8.36 \times 10^{-3}$ | $8.27 \times 10^{-3}$ | $8.18 \times 10^{-3}$ |
|           | 4     | $3.95 \times 10^{-3}$ | $4.13 \times 10^{-3}$ | $3.98 \times 10^{-3}$ |
| 15        | 2     | $5.39 \times 10^{-2}$ | $5.32 \times 10^{-2}$ | $5.41 \times 10^{-2}$ |
|           | 3     | $2.27 \times 10^{-2}$ | $2.23 \times 10^{-2}$ | $2.24 \times 10^{-2}$ |
|           | 4     | $1.22 \times 10^{-2}$ | $1.21 \times 10^{-2}$ | $1.20 \times 10^{-2}$ |
| 20        | 2     | $9.15 \times 10^{-2}$ | $9.14 \times 10^{-2}$ | $9.17 \times 10^{-2}$ |
|           | 3     | $4.31 \times 10^{-2}$ | $4.25 \times 10^{-2}$ | $4.29 \times 10^{-2}$ |
|           | 4     | $2.53 \times 10^{-2}$ | $2.54 \times 10^{-2}$ | $2.53 \times 10^{-2}$ |

Table 5.5: The effect of overloading on the performance of the two approximation methods (the number of measured data points is 6, at most one alternative node is allowed and trunk reservation is 5 circuits)
Chapter 6

Conclusions and Future Work

6.1 Conclusions

Multicasting refers to the ability of a set of more than two nodes or end-users in a communication network to communicate simultaneously with each other. Applications that require multicast capability (either point-to-multipoint (PTM) as in distributional video or multipoint-to-multipoint (MTM) as in video conferencing, online collaboration and others) will be an integral part of future broadband services. Given the popularity of multicast end-user services and applications, we studied the problem of state dependent multicast call routing for single rate loss networks.

We formulated the problem of state dependent multicast call routing for single rate loss networks and discussed the difficulties to implement the optimum solution. Because of the difficulties, a heuristic approach was proposed: we applied Minimum Spanning Tree (MST) searching with a suitable link cost function to search a connected tree for a multicast call connection request such that its normalized revenue loss can be reduced. If there is no connected tree available, the call connection request will be rejected.

We developed analytical models for Least Load Multicast Routing (LLMR) which is the well-known multicast routing algorithm, for fully connected net-
works. The analytical models that we developed for calculating blocking probabilities are based on (i) the link independence assumption, i.e. the random variables describing the state of each link are assumed to be independent, and (ii) the Reduced Load Approximation (RLA), which is an iterative procedure that alternates between the following two steps:

1. Evaluation of link occupancy distributions given the state dependent arrival rates and

2. Evaluation of state dependent arrival rates given the link occupancy distributions.

Once the consistent set of link occupancy distributions and state dependent arrival rates are obtained, the call blocking probability of each class (or any other performance measure for that matter) can be calculated with little effort. The analytical results can be used to engineer routing policies to achieve design objectives and to dimension networks.

For symmetrical networks, our analytical models included alternative routing for point-to-point communication. The agreements of simulation and analytical results in both scenarios were observed to be good and we find that the agreements were not affected significantly by the link independence assumption.

Four new link cost functions were proposed to improve the network performance in different ways:

ALLMR. We proposed a new algorithm called Aggregated Least Load Multicast Routing (ALLMR), which is modified from the well-known multicast routing algorithm called Least Load Multicast Routing (LLMR). The link cost function is an aggregated number of free circuits of a link. The main contribution of ALLMR is its simpler implementation, lower signaling traffic for establishing connection requests and lower sensitivity to the design
parameters, compared with LLMR. We also modify the analytical model of LLMR for symmetrical networks to handle ALLMR and the numerical results showed that the state aggregation can reduce the computational complexity.

**LLMRMOC** This is a modified LLMR algorithm called Least Load Multicast Routing with Maximum Occupied Circuits (LLMRMOC). The link cost function includes both the number of free circuits and the number of occupied circuits of a link. Compared with LLMR, the main contribution of LLMRMOC is its moderate improvement with almost no additional costs. We have also discussed its implementation issues.

**LLMRMMBT** This is a modified LLMR algorithm called Least Load Multicast Routing with Minimum Measured Blocking Time (LLMRMMBT). The link cost function also includes both the number of free circuits of a link and its measured blocking probability. Compared with LLMR, the main contribution of LLMRMMBT is its moderate improvement, which is better than that of LLMRMOC, with minimum additional cost. We have also discussed its implementation issues.

**MCBMR** We also proposed a new routing algorithm called Maximum Mean Number of New Calls Accepted Before Blocking Multicast Routing (MCBMR), which can more accurately capture the current and future loading of a network. Simulation results show that this algorithm, compared with LLMR, not only has a smaller network revenue loss, but also results in smaller call blocking probabilities for all classes of traffic. We also discussed the implementation issues of our proposed algorithm and developed two approximation methods, state approximation and curve fitting, which can reduce the measurement complexity significantly with only a slight perfor-
mance degradation.

6.2 Future Works

In this thesis, our research mainly focused multicast applications on fully connected single rate loss networks. In order to extend the research work to all multicast applications, we should generalize the scenario to partially connected multi-rate loss networks. The following three problems should be considered as possible future research:

1. For the new scenario, the corresponding revenue of a call connection request should be reconsidered. For fully connected single rate loss networks, the corresponding revenue can be easily set as the number of circuits allocated for a call connection request. However, for partially connected multi-rate loss networks, the corresponding revenue should be related to the bandwidth required and the number of links used for a call connection request which is difficult to determine.

2. For partially connected networks, Minimum Spanning Tree (MST) is no longer a suitable method to search a connected tree for multicast call connection requests because, for some kind of requests, there may not be any existing connected trees to be selected by using the MST method and thus the requests will never be established. Instead of MST, we can consider Steiner tree searching with heuristic methods. To select a link cost function, we should investigate the trade off between the number of links involved in a call connection request and the cost of a connected tree for the call connection request.

3. For single rate loss networks, the link cost function is highly related to the call blocking probability. However, for multi-rate loss networks, the
link cost function should also be related to the bandwidth required for a call connection request. Currently in single rate loss network, link cost functions do not consider the bandwidth requirement. Thus, new approach should be applied to related the bandwidth requirement and a link cost function to reduce the normalized revenue loss.

4. Our research works are concentrated on the call admission and the routing of multicast calls. Future works can be made for a decision based on the number of non-connectable destinations in certain multicast applications.
REFERENCES


