Dissertations, Theses, and Masters Projects      Theses, Dissertations, & Master Projects

2001

# Virtual path bandwidth distribution and capacity allocation with bandwidth sharing

Karen Marie Anewalt
*College of William & Mary - Arts & Sciences*

Follow this and additional works at: https://scholarworks.wm.edu/etd

Part of the Computer Sciences Commons, and the Electrical and Computer Engineering Commons

# VIRTUAL PATH BANDWIDTH DISTRIBUTION AND CAPACITY ALLOCATION WITH BANDWIDTH SHARING

---

A Dissertation

Presented to

The Faculty of the Department of Computer Science

The College of William & Mary in Virginia

In Partial Fulfillment

Of the Requirements for the Degree of

Doctor of Philosophy

---

by

Karen Marie Anewalt

2001

UMI Number: 3026402

# UMI®

---

UMI Microform 3026402

Copyright 2001 by Bell & Howell Information and Learning Company.
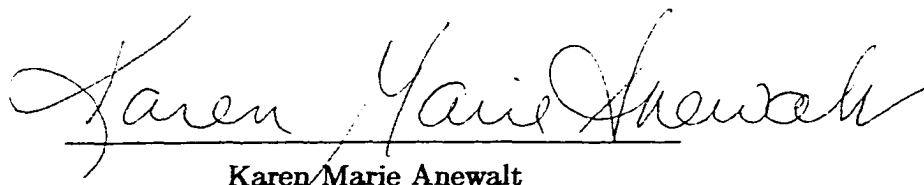
All rights reserved. This microform edition is protected against
unauthorized copying under Title 17, United States Code.

---

Bell & Howell Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

# APPROVAL SHEET

This dissertation is submitted in partial fulfillment of

the requirements for the degree of

Doctor of Philosophy

Karen Marie Anewalt

Approved, August 2001

Rahul Simha
Thesis Advisor

Weizhen Mao

William Bynum

Rex Kincaid
Department of Mathematics

ii

*To Michael...*

# Table of Contents
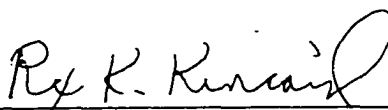
# ACKNOWLEDGMENTS

I would like to thank a multitude of people for the support that I received over the past five years. Foremost, I would like to thank Michael for his love, support and encouragement. He was there on the good days and the bad. I appreciated his presence on both.

I would like to thank my advisor Dr. Rahul Simha for his guidance throughout my career as a graduate student and especial for his assistance with my dissertation. I would like to thank the members of my committee - Dr. Weizhen Mao, Dr. William Bynum, and Dr. Rex Kincaid for their support and suggestions.

I would like to thank all of my family for their love and patience as I completed my degrees. I thank my friends from William and Mary and Allegheny College who provided companionship and made all of my college years fun. I also appreciate the support and patience of my colleagues at Mary Washington College during the past year.

Lastly, I would like to acknowledge the Virginia Space Grant Consortium for their financial support through a Virginia Space Grant Fellowship during three of my years at William and Mary.

# ABSTRACT

Broadband high-speed networks, such as B-ISDN, are expected to play a dominant role in the future of networking due to their capability to service a variety of traffic types with very different bandwidth requirements such as video, voice and data. To increase network efficiency in B-ISDN and other such connection oriented networks, the concept of a *virtual path* (VP) has been proposed and studied in the literature. A VP is a permanent or semi-permanent reservation of capacity between two nodes. Using VPs can potentially reduce call setup delays, simplify hardware, provide quality of service performance guarantees, and reduce disruption in the event of link or node failure.

In order to use VPs efficiently, two problems must be solved. With the objective of optimizing network performance, (1) the VPs must be placed within the network, and (2) network link capacity must be divided among the VPs. Most previous work aimed at solving these problems has focused on one problem in isolation of the other. At the same time, previous research efforts that have considered the joint solution of these problems have considered only restricted cases. In addition, these efforts have not explicitly considered the benefits of sharing bandwidth among VPs in the network.

We present a heuristic solution method for the joint problem of virtual path distribution and capacity allocation without many of the limitations found in previous studies. Our solution method considers the joint bandwidth allocation and VP placement problem and explicitly considers the benefits of shared bandwidth. We demonstrate that our algorithm out-performs previous algorithms in cases where network resources are limited. Because our algorithm provides shared bandwidth, solutions found by our algorithm will have a lower setup probability than a network that does not use VPs as well as a lower loss probability than provided by VPDBA solutions produced by previous algorithms. In addition, our algorithm provides fairness not found in solutions produced by other algorithms by guaranteeing that some service will be provided to each source-destination pair within the network.

# VIRTUAL PATH BANDWIDTH DISTRIBUTION AND CAPACITY ALLOCATION WITH BANDWIDTH SHARING

# Chapter 1

# Introduction

We consider the class of connection oriented networks in which *connections* or *calls* are requested between each source–destination pair. This class of networks includes several kinds of ATM networks, the telephone system, as well as many optical backbone systems. In this type of network, the user first establishes a connection, then uses the connection, and finally terminates the connection.

One example of a connection oriented network of increasing importance is the Broadband Integrated Services Digital Network (B-ISDN). B-ISDN has been designed to integrate voice and non–voice services on one network. The system was designed to enhance the existing worldwide telephone system by allowing it to handle modern communication needs such as data transmission and video, in addition to standard voice transmission [65]. B-ISDN is unique in that it can support many traffic classes, each with a different flow characteristic and desired quality of service. Asynchronous Transfer Mode (ATM) has been selected as the most promising transport technique that can be used to implement B-ISDN based on its efficiency and flexibility. Due to these positive qualities, the standards body CCITT has standardized ATM [16]. As ATM and B-ISDN networks become more common, performance issues related to connection oriented networks become increasingly important. Reducing

2

*call setup time* is one such issue and *virtual paths* are one way of reducing call setup times. Time spent processing call setup requests is overhead, making it desirable to minimize the amount of time spent processing such requests thereby enabling the network manager to concentrate on other tasks. VPs have been shown to have the potential to reduce processing costs associated with establishing connections by up to 90% [14].

To introduce the notion of a *virtual path*(VP), we begin with some definitions. A network can be thought of as a collection of switches and the wires or wireless links that connect these switches. The switches are referred to as *nodes* and the connecting wires are referred to as *links*. Each link has an associated *bandwidth*, or rate at which information can be transmitted on that link. In a connection oriented network, when a request to send information is received by a switch, an end–to–end path for the connection must be setup before data can be transmitted. During the setup procedure, the network searches for an available route from the source to the destination. The route is generally picked based on information about the location and available capacity of the various links in the network. The connection request must propagate all the way to the destination and be acknowledged before data transmission can begin. Once the setup procedure has been completed, there is little delay experienced by the data and no danger of congestion because adequate bandwidth along the route has been reserved for the connection [65]. In the case of a circuit–switched network, packets encounter no delay or loss, whereas in a packet–switched network, loss and delay characteristics are typically guaranteed to remain within specified tolerances. Consider the network in Figure 1.1. If switch A received a request to send a message to switch E, the message could take several routes. It could be sent on A-B-E, A-C-E, A-B-C-E, or A-C-B-E. In order to minimize the amount of time needed to send the
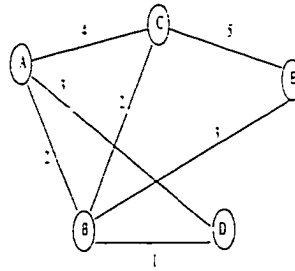
**Figure 1.1**: An example of a small computer network.

message, information about the speed and amount of bandwidth available on these links is used. Once the route is chosen, the bandwidth required is reserved on each of the links in the route.

Once the setup procedure concludes, a *connection* or *call* between the source and destination is initiated. The terms *call*, *connection*, and *session* are interchangeable in this context. We will use the term *call* to refer to such a connection. While the call is active, the source can transmit information to the destination. When the call is complete, the reserved bandwidth is released and can be used for other connections. If there is not enough bandwidth available on any route to process the call, then the call is said to be *lost*.

In this context, we now make a key observation that motivates the use of virtual paths: reserving and releasing the bandwidth required to establish a call takes time. If the number of links used between the source and destination node is large, or if the network offers many routes from the source to destination, or if the network is very busy, the amount of time required to find an available route may be substantial. For long routes, the process of releasing the previously reserved bandwidth may also be substantial. We can think of the combined time to reserve and release the bandwidth for a call as the *cost* of admitting the call. To minimize this cost, we would like to be able to permanently reserve a portion of the

available bandwidth along a specific route for all calls from a specific source to a specific destination node. To facilitate this type of cost reduction through permanent bandwidth reservation, the *virtual path* (VP) concept was developed.

A VP, shown as a dark line in Figure 1.2, is a reserved amount of bandwidth from a source node, $A$, to a destination node, $E$, that may only be used for calls from this source to destination. In the example, two units of bandwidth are reserved for calls from node $A$ to node $E$ and the remaining bandwidth can be used for all other calls. The VP route is setup once and is never torn down or is torn down only after long time intervals. Thus when a message arrives at node $A$ with the destination $E$, the message is sent on the predetermined route immediately, provided there is enough bandwidth available on the VP. No setup cost is incurred because checking to see if the VP has sufficient capacity for the call can be done locally at the node. The remaining bandwidth across the links from $A$ to $E$ can be used to service calls from $A$ to $E$ in the case in which the VP does not have the required amount of bandwidth available and can also be used to service calls for other source–destination pairs.
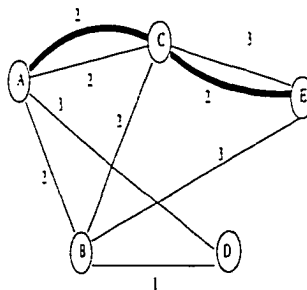


Figure 1.2: An example of a network containing a virtual path.

The key to using virtual paths effectively lies in the solution to an optimization problem involving the following trade-off. If a large amount of bandwidth is reserved for the VP,

then a large proportion of calls from the source to destination node will be able to utilize this path and the probability that a call does not incur a setup cost will be high. However in this scenario, any calls that arrive from nodes other than the source or calls that arrive for nodes other than the destination will be lost since all bandwidth is reserved for calls from the source to the destination. At the other extreme, one might design a system in which no bandwidth is reserved for calls from the source to the destination. In this case, the probability that a call will incur a setup cost will be high but the probability of a loss will be low. The optimization problem we formulate captures the trade-off between reserving too much and too little of the available bandwidth for the VP.

Two sub-problems must be solved to optimize the trade-off. First, the VPs must be placed (routed) within the network. Second, bandwidth must be assigned to the VPs. These two problems have been studied in the literature, most often in isolation. References [10, 23, 28, 69] consider the problem of determining the placement of VPs in the network, whereas references [4, 11, 13, 25, 26, 31, 32, 37, 44, 45, 46, 47, 50, 51, 52, 53, 55, 62, 61, 68] consider the problem of allocating bandwidth to the VPs. The solution to the VP placement problem affects the solution to the VP capacity assignment problem and vice versa. Thus the optimality of the solution depends heavily on the joint solution to these two problems. References [1, 7, 8, 9, 16, 17, 29, 34, 40] consider the joint VP placement and capacity assignment problem and offer solutions to restricted cases of these problems. We propose to expand the newly developed virtual path concept to more efficiently use network resources. As part of this research, we will use both analysis and simulation to study the trade-offs between increasing resource usage and maintaining the quality of service provided to the user.

## 1.1 Problem Statement

In this section, we describe the notation for and define the Virtual Path Distribution and Bandwidth Allocation (VPDBA) problem that we study in this thesis.

### 1.1.1 Notation

We are given an unweighted graph $G = (V, E)$ representing the topology of a communication network. The set $V$ represents the switches or nodes in the network, each capable of sending and receiving calls. The set $E$ represents the communication links found between the nodes. The size of the network is $N = |V|$.

A virtual path $p$ of length $n$ between any two vertices, $s$ and $d$ in the graph $G$ is a sequence $< v_0, v_1, v_2, \ldots, v_n >$ of vertices such that $s = v_0$ and $d = v_n$ and $(v_{i-1}, v_i) \in E$ for $i = 1, 2, \ldots, n$. If the virtual paths are labeled as $p_1, p_2, \ldots$, then the $i$-th virtual path can be uniquely described by the source of the path, $s_i$, the destination of the path, $d_i$, the route from $s_i$ to $d_i$, and the capacity allocated to the path, $c(p_i)$. Thus $p_i = (s_i, d_i, < s_i, v_j, v_{j+1}, \ldots, d_i >, c(p_i))$. Let $\wp = \{p_i, \forall i\}$ be the set of all virtual paths in the network. Although not written as a function, $\wp$ can be considered the "layout" function that provides the route for each virtual path.

Each link $e$ has a certain maximum capacity $C_e$: the bandwidth the link can support. Each virtual path $p_i$ containing link $e$ is allocated a portion of this bandwidth, $c(p_i)$. Let $\widehat{C_e} = \sum_{\forall p_i : e \in p_i} c(p_i)$ be the capacity on link $e$ assigned to VPs. The remaining unreserved bandwidth on a link that is not allocated to a particular virtual path is denoted $\overline{C_e} = C_e - \widehat{C_e}$. This bandwidth $\overline{C_e}$ can be used to handle local traffic as well as calls that cannot be handled

by the virtual path due to bandwidth constraints. We will call such unreserved bandwidth, *shared bandwidth.*
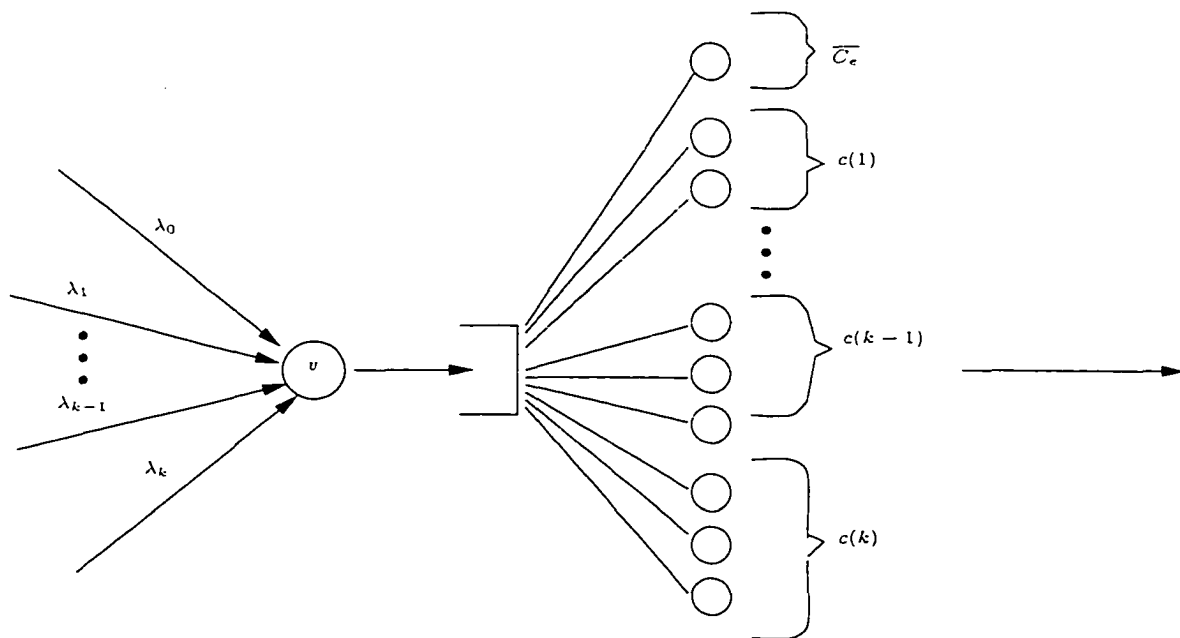


**Figure 1.3**: A single node in a network containing virtual paths.

Consider a single node, $v$, in the network. Assume that $v$ is the source node for some number, $k$, of virtual paths. Then $k + 1$ types of calls arrive at node $v$, one type for each of the $k$ virtual paths as well as a type for local calls. In general, call arrival times are unpredictable. To model each call arrival process, we use a probability model to approximate the real arrival process. Calls in connection oriented networks are typically modeled as a Poisson process [3]. The well-known characterization of a Poisson process is to assume that call interarrival times are exponentially distributed.

In our model, the $k$ virtual paths originating at node $v$ may be numbered from 1 to $k$. Node $v$ is depicted in Figure 1.3. Let $i$ be the virtual path number $1 \le i \le k$ as described

above. Then calls arriving at $v$ for the destination node of path $p_i$ arrive with a Poisson process with arrival rate $\lambda_i$. Local calls arrive with a Poisson process with arrival rate $\lambda_0$. In the general network, we can designate the call arrival rates as $\lambda_{s,d}^{vp}$ for the arrival rate of VP calls arriving at source $s$ for destination $d$ and $\lambda_{s,d}^{L}$ for the arrival rate of non-VP (local) calls arriving at source $s$ for destination $d$.

Each call that arrives to the network has an associated bandwidth requirement. In the telephone system, all calls are voice calls and require equal bandwidth. However, other types of calls may require different amounts of bandwidth. In the general case, we can assume that a different class of traffic exists for each possible bandwidth requirement. The traffic classes in such a general network can be numbered from 1 to $j$. Thus we can designate the call arrival rates as two vectors: $< \lambda_{s,d}^{vp_1}, \lambda_{s,d}^{vp_2}, \ldots, \lambda_{s,d}^{vp_j} >$ for the arrival rate of VP calls of each of the $j$ traffic classes arriving at source $s$ for destination $d$ and $< \lambda_{s,d}^{L_1}, \lambda_{s,d}^{L_2}, \ldots, \lambda_{s,d}^{L_j} >$ for the arrival rate of non-VP (local) calls of each of the $j$ traffic classes arriving at source $s$ for destination $d$.

In general, the duration of calls is also unpredictable. We call the duration of a call the *holding time* of the call because it represents the amount of time that the bandwidth is reserved or held by the call. Call holding times are taken to be exponentially distributed with parameter $\mu$, as is typically assumed in the literature.

Consider calls that arrive at node $v$ for some destination, $d \in V$. If there exists a virtual path, $p_i = (v, d, < v, v_1, \ldots, d >, c(p_i))$ then this call may use the capacity reserved for virtual path $i$ if sufficient capacity is available. If there is not sufficient capacity available on the virtual path, then this call can use the unreserved shared capacity, reserving the necessary capacity on each link at the time that the call arrives at node $v$. If reserved virtual

path capacity is not available and there is not sufficient shared capacity at each node on some route between $v$ and $d$, then the call is lost. We will allow for the possibility of not creating VPs for some source–destination pairs if that option is better for performance. In this case, calls must use only shared bandwidth and incur setup costs.

## 1.1.2 Performance Measures

As mentioned earlier, two key performance measures are the basis for our optimization problem. The first is the probability that a call incurs a setup cost (that is, is not able to use a VP). The second is the probability that a call is lost by the network. We define two events in the system. Define the events

$$loss \equiv a\ call\ is\ lost$$

and

$$setup \equiv a\ call\ is\ accepted\ and\ incurs\ a\ setup\ cost.$$

Then we are interested in the probability that each of these events will occur, $P(loss)$ and $P(setup)$. Ideally, we wish to minimize both the $P(loss)$ and the $P(setup)$ for the system.

## 1.1.3 Formal Problem Definition

Given a graph $G = (V, E)$, the available capacity on each link $C_e\ \forall e \in E$, the description of the classes of incoming traffic $\lambda_{s,d}^{vp}$ (or $< \lambda_{s,d}^{vp_1}, \lambda_{s,d}^{vp_2}, \ldots, \lambda_{s,d}^{vp_j} >$) and $\lambda_{s,d}^{L}$ (or $< \lambda_{s,d}^{L_1}, \lambda_{s,d}^{L_2}, \ldots, \lambda_{s,d}^{L_j} >$) for each source-destination pair and the holding time for calls $\mu$, the problem, informally, is to determine the optimal layout of virtual paths $\wp$ and the associated capacity assignments for each virtual path in the network such that the throughput of the network is maximized and the overall cost of establishing a call is minimized. More formally,

because we have two performance objectives, we may define three types of problems:

1. Given an upper bound $U$ for the call loss probability, find a VP layout $\wp^*$ and capacity function $c^*$ such that $P(loss) \leq U$ and $P(setup)$ is minimized (if such a layout and capacity function exists).

2. Given an upper bound $T$ on the setup probability, find a VP layout $\wp^*$ and capacity function $c^*$ such that $P(setup) \leq T$ and $P(loss)$ is minimized (if such a layout and a layout and capacity function exits).

3. Define a cost function $F = \alpha P(loss) + (1 - \alpha)P(setup)$ in which each cost measure is weighted by $\alpha$ or $(1 - \alpha)$. Find a VP layout $\wp^*$ and capacity function $c^*$ such that $F$ is minimized.

We will also refer to the VP Layout problem as the VP Distribution problem and the problem of determining capacities as the Bandwidth Allocation problem, giving rise to the acronym VPDBA for the joint problem. Initially, for simplicity, we will assume that all calls require a single unit of bandwidth. Thus only one traffic class exists and the call arrival rates can be specified as $\lambda_{s,d}^{vp}$ and $\lambda_{s,d}^{L}$.

## 1.1.4 The Objective Function

To determine the performance measures $P(loss)$ and $P(setup)$ from the given parameters $\lambda_{s,d}^{vp}$, $\lambda_{s,d}^{L}$ and $\mu$, we model a system using a continuous time Markov chain. First a state space is defined. Then the chain is solved for the state probabilities. Finally the state probabilities are used to calculate the theoretical probability that a call will be lost as well as the probability that no setup cost is incurred.
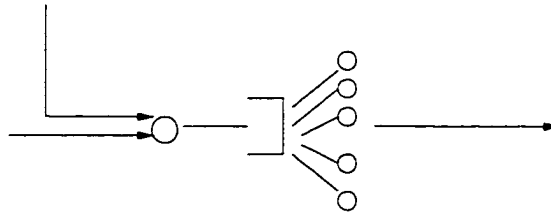
**Figure 1.4**: A network system containing 1 node with 5 servers.

Consider a simple system with one link shown in Figure 1.4. We can model this link using queueing theory. Each call can be thought of as a customer. We will assume that all calls require a single unit of bandwidth for service and model the total link bandwidth $C_e$ as $C_e$ servers. Our model assumes that the customers (calls) arrive using a Poisson process and that the service times (holding times) are exponentially distributed and there are $(k+1)$ servers available at each node. Thus a single network link can be modeled as an M/M/(k+1) queue [20].

In general, it is possible that a call could require more than one unit of bandwidth. In this case, each unit of bandwidth on a given link can still be modeled as a single server. Thus a link with bandwidth $C_e$ will be modeled as $C_e$ servers. However, in this case, some calls may require multiple servers based on their bandwidth requirements (traffic class).

In the queueing model, the amount of bandwidth reserved for the virtual path on an edge can be thought of as the number of *virtual path servers* and the amount of bandwidth unreserved can be thought of as the number of *shared servers*. Here one unit of bandwidth corresponds to one server and each call requires one unit of bandwidth for service. Suppose that two of the servers in our single link example are virtual path servers and three of the servers are shared servers. The state space for the system, $S$, can be defined as an ordered

pair of the number of virtual path servers and shared servers in use at a time $t$. Let

$$VP_t = \text{ the number of virtual path servers in use at time } t$$

then

$$S = \{(SH_t, VP_t), \ \forall t\}$$

$$= \{(0,0),(0,1),(0,2),(1,0),(1,1),(1,2),(2,0),(2,1),(2,2),(3,0),(3,1),(3,2)\}.$$

We can then write the Markov chain for the system by considering the transitions

between the states as in Figure 1.5. Then we can solve the chain for the state probabilities.
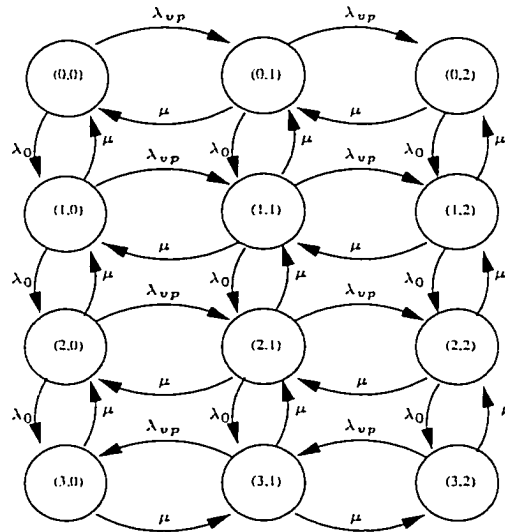


**Figure 1.5**: The Markov chain for a network containing 1 node with 5 servers.

These probabilities can then be used to calculate the probability that a call will be lost and

the probability that a call will incur a setup cost.

For large networks, solving the equivalent Markov chain becomes very complex. How-

ever, assumptions can be made to simplify such calculations. We will consider each network

node in isolation from the others. The solutions for the individual nodes will then be com-

bined to approximate the results that would be obtained by solving the Markov chain for

the larger system. Using the approximated results, we can calculate the performance measures, $P(loss)$ and $P(setup)$. These performance measures can then be used to design a VP layout and capacity assignment.

## 1.2 Related Research

VPs have been shown to be a useful mechanism in ATM networks because they provide increased network performance and reliability, reduced processing costs for call setup, increased network flexibility, and simplified network architecture [56]. Issues surrounding the solution to the joint VP distribution and capacity allocation problem have been studied in the literature. The formulation of the joint problem is known to be NP-Complete [2]. Thus solution methods given in the literature focus on solving part of problem (either VP capacity allocation or path distribution) or focus on simplifying assumptions that make the joint solution to the problem tractable.

Algorithms that solve the problem of VP distribution, VP bandwidth allocation, and the joint problem under certain simplifying assumptions are presented in this section. Studies presenting discussions of routing schemes that are appropriate for use in VP networks and studies discussing VP network restoration in response to link or node failures are also summarized. Other VP related issues such as switching, traffic estimation method, and call blocking probability calculation methods have also been studied and are summarized briefly.

We provide a detailed description of several related papers. Some description of the previous work done on VPs can also be found in [5]; we will defer to this survey where

appropriate to avoid repetition. In places we will point out which papers consider sharing, a key aspect of our problem formulation.

### 1.2.1   VP Distribution

VPs offer many benefits in terms of increased network efficiency. However, if the VP layout is not designed to maximize the benefits while minimizing the costs, the network performance will be negatively affected. A number of papers consider methods for distributing VPs optimally in a network [10, 23, 28, 69]. Summaries of the papers can be found in [5]. The papers discuss methods used to solve the VP Distribution problem but fail to consider how the bandwidth should be assigned to the VPs in the network. In addition the solution methods described do not consider the benefits of sharing, nor do they provide dynamic algorithms that can adapt to changes in traffic dynamics.

### 1.2.2   VP Bandwidth Allocation

A problem of equal importance to the VP distribution problem is the problem of assigning capacity to the VPs in a network. If capacity is not assigned to maximize the benefits of VPs while minimizing the costs of VPs, then network performance is negatively affected. Several papers seek to distribute capacity to the VPs in an optimal fashion [4, 11, 13, 25, 26, 31, 32, 37, 42, 43, 44, 45, 46, 47, 50, 51, 52, 53, 55, 62, 61, 68] . Summaries of the papers can be found in [5]. While these papers describe methods to solve the VP Bandwidth Allocation problem, they fail to consider how the VPs should be distributed in the network. Some of the papers listed above consider methods for sharing bandwidth to increase network throughput and decrease the call loss rate. However, these methods involve additional

overhead caused by frequent reallocation of bandwidth in the network or by requiring the network manager to record additional statistics about the calls or network. This overhead may result in a degradation of network performance that may be unacceptable to the user. None of the papers explicitly consider the issues surrounding shared bandwidth.

### 1.2.3 Joint VP Distribution and Bandwidth Allocation

The optimal solutions to the problems of VP distribution and bandwidth allocation are both important for optimal network performance. However, the formulation of this joint problem is NP–Complete [17]. Nonetheless, simplifying assumptions can be made that make the solution to the joint problem tractable. Solutions of this nature are found in [1, 7, 8, 9, 16, 17, 29, 34, 40]. Summaries of the papers that consider the joint problem can be found in [5]. Only one of these papers considers methods for sharing bandwidth in a network. Ahn *et al.* [1] consider the benefits of sharing bandwidth within the context of the joint problem solution. However, their solution does not consider many key sub-problems related to the effective solution of the joint problem such as how much bandwidth should remain unreserved on each route and how to control access to the unreserved bandwidth. In addition, the solution method presented by Ahn *et al.* is not dynamic.

### 1.2.4 Routing

The issue of routing is closely related to the optimal design of a VP network. The simplest routing policies allow a call to use a single route. If capacity is unavailable on this route, then the call is lost. Routing policies that seek to minimize call loss and/or minimize connection costs are considered in several papers [30, 41, 35, 33]. The policies use adaptive or alternate

routing to try to increase network performance while guaranteeing Quality of Service(QoS). A new routing policy using traffic profiles and available bandwidth information and a genetic algorithm is discussed in [24]. Slosiar *et al.* [64] discuss an algorithm for generating routing tables with primary and secondary paths. These routing tables could be used by upper level network layers to select VP placement. Summaries of the papers discussing routing issues can be found in [5].

### 1.2.5  Fast Restoration

All networks are vulnerable to failure. In B-ISDN networks that have the capability to handle high speed data, a link or node failure can result in the loss of a large volume of data. Thus it is important to make the interruption in service as short as possible. Restoration schemes for use in VP networks have been studied in the literature [27, 39, 19, 38, 48, 67]. Several of these schemes are summarized in [5].

### 1.2.6  Sharing Bandwidth

One way of more effectively using bandwidth with VPs is to allow calls to share bandwidth. Several papers have considered this idea using different methods to facilitate sharing. These methods are summarized below.

Habib *et al.* [31] allow calls of the same traffic class to share the bandwidth on a single VP. Liu *et al.* [45] allow calls of similar traffic classes to share VP bandwidth. These schemes choose to allow sharing only by similar classes to simplify network management. These sharing schemes improve throughput when compared to networks with no sharing scheme. However, only allowing similar traffic classes to share a VP is restrictive and will

result in a lower throughput than a scheme that allows all traffic classes to share bandwidth.

Wong *et al.* [68] allow several classes of traffic to use the same VP. Their network scheme specifies that two routes exist between each source–destination pair, a direct VP and a two hop route. The dynamic bandwidth allocation algorithm proposed causes the network to be redesigned frequently when calls cannot be accommodated by the assigned bandwidth. In some cases, calls are not even required to attempt to use both routes before deciding to re-allocate the bandwidth. Thus unnecessary network design overhead may be incurred.

Logothetis *et al.* [46] allow calls that are blocked on the assigned VP route due to a lack of bandwidth to borrow unused bandwidth from other VPs sharing the same path. This type of sharing is shown to lower the call blocking probability. However, this type of sharing must also raise the cost of network management functions. The network manager must redistribute VP bandwidths every time that a call is blocked from its attempted VP and some unused bandwidth is available in the network.

Frost *et al.* [25] also allows calls to borrow unused bandwidth from other VPs. In this scheme, when a call arrives and cannot be handled by the assigned VP, the call checks to see if there is bandwidth available on another VP following the same route. If unused bandwidth is available on another VP following the same route, then the bandwidth is borrowed for use by the call. In this scheme, the call is tagged to show that it has been admitted using bandwidth not reserved for its use. If congestion occurs, the tagged calls are discarded. This bandwidth sharing method also results in higher costs of network management. The bandwidth manager must search for unused bandwidth. Also additional processing is needed to tag calls.

Ahn *et al.* [1] note the benefits of allowing some bandwidth to remain unreserved in the

network. However, they do not consider how much bandwidth should remain unreserved or how to control access to the unreserved bandwidth. Aneroussis *et al.* [4] also recognize the benefits of shared bandwidth. Their bandwidth allocation scheme allows some bandwidth to remain unreserved and then be used in a prioritized alternate routing scheme. However, their algorithm fails to explicitly consider how much bandwidth should remain unreserved on each link. In some cases, all of the bandwidth on a particular link may be reserved for VP use and no bandwidth will remain unreserved. Gariglio *et al.* [26] claim to compare a pure VC, pure VP, and combination VP/VC network. However, the key issues of the combination scheme are not discussed. They do not consider how to determine the optimal number of VPs to establish between nodes, how to select which routes will be VPs and which will be VC routes, and there is no discussion about how to optimize the amount of bandwidth allocated to VPs in conjunction with the amount of bandwidth allocated to VCs.

These sharing schemes increase the amount of work that must be done by the network or bandwidth manager. This additional workload may cause increased setup delays or may slow down the dynamic reallocation of bandwidth in the network. These delays may be unacceptable to the user.

## 1.2.7 Other Work on VPs

Additional research has been done on issues related to the VP bandwidth distribution and capacity assignment problem. Although the research is tangential to the VPDBA problem formulation considered in this thesis, an overview of the previous research follows.

General information on VPs and the effect of using VPs in a network have been studied.

Bubenik *et al.* [12] look at whether VPs or VCs can better handle calls of different types. The need for both types of connections in a network is justified. Burgin *et al.* [14] present an overview of B-ISDN and VPs. Issues surrounding call setup, capacity allocation and update intervals are discussed. Some research related to the VP distribution problem is summarized in a survey paper [70].

Management issues concerning the actions that should be taken by the network controller have been studied. Sato *et al.* [59] explore an experimental transport system for ATM networks and the VP management tasks that are necessary to develop the described transport system. Hyman *et al.* [36] look at how to model resource allocation for VPs. They focus on how to control access to VPs to guarantee QoS. Sato *et al.* [57] propose a policing mechanism to ensure acceptable cell arrival rate and a method for calculating cell multiplexing delay in a ATM network allowing statistical multiplexing of cells on VPs. The proposed methods can be used to design a VP distribution and bandwidth allocation algorithm.

Methods to model different traffic streams have been studied. DeVeciana *et al.* [21] look at how to model and service VBR (variable bit rate) traffic in ATM networks. Dutkiewicz *et al.* [22] look at how to model arrival streams at the cell level as two state MMPPs. They also present an admission control scheme based on this queueing model. Sato *et al.* [58] look at how to model CBR (constant bit rate) traffic and how to evaluate the QoS for CBR traffic in ATM networks containing VPs.

Chan *et al.* [15] study the traffic interactions that result from multiplexing traffic classes on a VP. An algorithm is presented that determines the minimum bandwidth required to satisfy QoS requirements for various traffic classes that are multiplexed on a single VP.

Integrated and segregated traffic schemes are compared for homogeneous and heterogeneous traffic flow. In the case of homogeneous flow. it is shown that QoS alone may not be a sufficient indicator of whether to use an integrated or segregated traffic scheme. In the heterogeneous case. it is shown that the QoS determines which scheme is more suitable for a traffic class. A second algorithm is presented that approximates the optimal VP combination. The algorithm is evaluated using three traffic classes and between one and three VPs. It is shown that the second algorithm yields the optimal VP combination in most cases.

QoS issues are studied by Reiss *et al.* [54]. They explore how cells should be stored. buffered, and lost to assure QoS in ATM networks.

Several studies have considered methods for calculating or estimating parameter and performance measures. Zhang [71] looks at how to calculate cell loss in ATM networks due to cell level congestion. Siebenhaar [63] considers how to estimate call blocking probability in a multi-service ATM network. The network is assumed to use an alternate or adaptive call routing scheme. The network is reduced to an equivalent single path model to reduce the calculations with respect to routing. The method is shown to be fast and accurate.

Switching in ATM and VP networks has been studied by Veeraraghavan *et al.* [66] and Obara *et al.* [49].

Several authors consider the use of VPs in other types of networks. Chlamtac *et al.* [18] study the use of VPs in wireless networks. They give an algorithm to determine the VP routes to connect terminator pairs such that the maximum link load is minimized. The method used is very similar to the method presented by the authors in their earlier paper [17]. Aoyama *et al.* [6] consider issues related to extending ATM techniques to provide

cost effective and flexible multimedia ATM leased line services to interconnect private networks. They describe the VP transport network architecture that can be use to support such service, describe services that should be provided by the network, and discuss technical VP issues that must be resolved before the network can be realized.

Sethi [60] considers an alternate bandwidth reservation scheme called Virtual Trees(VTs) and demonstrates the improved performance of VTs over VPs. VTs have the advantage of allowing calls from the same source to different destinations to share bandwidth.

## 1.3 Thesis Outline

This thesis is divided into eight chapters. The first chapter introduces the virtual path concept, formally defines the problem and provides a brief overview of related research. The second chapter introduces the concept of shared bandwidth. We provide an overview of related research that considers the shared bandwidth concept. We introduce a new scheme for sharing bandwidth that improves the network performance in terms of loss probability. We show that capacity allocations using our shared bandwidth scheme have lower loss probability than capacity allocations that do not allow shared bandwidth. We show that as the number of traffic streams using the shared bandwidth pool increases, the benefits of sharing increase. We observe that the shared bandwidth scheme has other positive implications for the network.

The VPDBA problem is complex. Before developing a solution method for general networks, we consider solutions to the VPDBA for special case networks. In the third chapter, we consider a simple line model with a single VP. In order to solve the VPDBA problem, we

need a way to evaluate network performance. We use a network decomposition method to calculate performance measures without running a simulation or solving a complex Markov chain. We prove that the subsystems produced by the decomposition method are not independent which complicates the approximation method. We develop a method of combining performance results from the subsystems to produce an accurate approximation of the performance measures for the network. We show that our approximation method produces results that are similar to those produced through simulation.

In the fourth chapter, we consider a second special case. This case involves a single node that experiences arrivals from several VP streams and several non-VP streams. We expand our performance measure approximation method to apply to single node systems containing multiple VPs. Again we use a decomposition method to divide the network containing $i$ VPs into $i$ subsystems each containing a single VP arrival stream. We compare the performance measures obtained using our performance measure approximation method to performance measures obtained through simulation.

In the fifth chapter, we consider the general line model. We show an expanded performance measure approximation method for multiple node systems with the potential for housing multiple VPs. We use a decomposition method to calculate our performance measures. We compare the performance measures obtained by our solution method to those obtained through simulation and show that our approximation method is valid, effective and can be used in an algorithm to determine the optimal solution to the VPDBA problem.

In the sixth chapter, we present our algorithm for solving the VPDBA problem and explore it's effectiveness in general line networks. Solutions produced by our algorithm are guaranteed to contain shared bandwidth. We compare the network performance of the

solutions determined by our algorithm to the solutions determined by previous algorithms for the VPDBA problem. We show that our algorithm produces solutions with loss and setup probabilities comparable to those produced by previous algorithm. Our algorithm provides an increased level of fairness by providing service to a greater amount of the offered traffic than previous algorithms.

In the seventh chapter, we expand our solution method to apply to general network topologies. We extend our performance measure approximation method and algorithm to apply to general networks. We show that our algorithm performs well when compared to previous algorithms for solving the VPDBA problem and that our algorithm will outperform previous algorithms in the case in which network resources are limited. Our algorithm also produces solutions that are guaranteed to provide some level of service to all streams, thus providing a degree of fairness not found in previous algorithms.

Finally, in Chapter 8, we provide a summary of the contributions introduced in the seven previous chapters. In addition, we provide a description of several items that warrant future study.

# Chapter 2

# Shared Bandwidth

We believe that the existing VP distribution and bandwidth allocation schemes can be improved by explicitly considering the benefits of shared bandwidth. The previous VP work that considers shared bandwidth uses methods that increase the amount of work that must be done by the network or bandwidth manager. One method of sharing that involves very little management overhead is to force some of the bandwidth to remain unreserved along each route. Then any call that cannot be handled on its designated VP can attempt to use this unreserved bandwidth pool. We use this sharing scheme and explore the impact on a single network link.

In this section, we demonstrate the advantages of sharing bandwidth via examples. By simulating a single link, we show that a system that allows sharing will outperform (in terms of loss probability) a system that does not allow sharing. We also discuss other benefits of including shared bandwidth within a network. The observations that we make will later be applied to the algorithm that we develop to solve the VPDBA problem.

## 2.1 Shared Bandwidth Experiment

Consider a single link in a network with several VPs traversing the link. Assume that the link contains $B$ units of bandwidth. We will assume that all calls require a single unit of bandwidth for service and model the bandwidth as $B$ servers. Let $K$ be the number of traffic sources that send calls to the link. Assume that a VP exists for each traffic stream on the link. Calls from one traffic stream are not allowed to share bandwidth reserved for use by a different traffic stream. Assume that the same number of servers $S$ is assigned to each VP. Any servers not assigned to a VP are unreserved and can be shared. Such a link is shown in Figure 2.1.
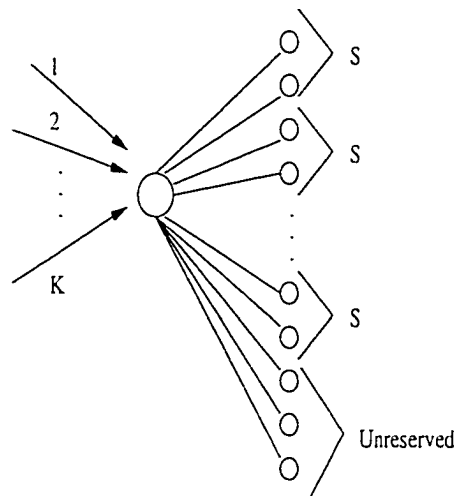
Figure **2.1**: A link with $K$ VPs and shared bandwidth.

Assume that when a call arrives from a source, it checks to see if there is enough capacity available on the VP assigned to it. If there is, then the call is established and no setup cost is incurred. If there is not, then the call checks to see if there is enough shared capacity available to handle it. If there is, then the call is established incurring a setup cost (for

using a non–VP route), otherwise the call is lost.

Intuitively, the lowest loss probability will occur when there are no VPs in the system and all servers are shared among all traffic sources. However, in this case, all calls will incur a setup cost. The highest probability of a loss will occur when all of the servers are reserved for VP traffic and no servers are unreserved for sharing. In this case the setup cost is minimized.

We study several partitions of the servers among the VPs and shared servers. To compare the performance of the system using the different partitions, we will calculate the probability of a loss and probability of setup in each case. Then we will calculate the percent difference in loss probability by finding the difference between the two loss probabilities, dividing this difference by the larger of the two probabilities and multiplying this quantity by 100%. Suppose that we wish to find the percent difference in loss probability $f$ for two server partitions. We will call the system with the first partition system $i$ and the system with the second partition system $j$.

$$percent\ loss = f = \frac{|P(loss\ in\ system\ i) - P(loss\ in\ system\ j)|}{\max\{P(loss\ in\ system\ i), P(loss\ in\ system\ j)\}} * 100\%$$

Similarly, the percent difference in call setup probability $g$ can be calculated by finding the difference between the two setup probabilities, dividing this difference by the larger of the two probabilities and multiplying this quantity by 100%.

$$percent\ setup = g = \frac{|P(setup\ in\ system\ i) - P(setup\ in\ system\ j)|}{\max\{P(setup\ in\ system\ i), P(setup\ in\ system\ j)\}} * 100\%$$

We also look at the performance of our objective function of the form

$$(\alpha P(loss)) + ((1 - \alpha)P(setup))$$

We can compute the percent difference for this objective function by calculating the value of $h$ in the two systems being considered.

$$h_i = (\alpha P(loss \ in \ system \ i)) + ((1 - \alpha)P(setup \ in \ system \ i))$$

$$h_j = (\alpha P(loss \ in \ system \ j)) + ((1 - \alpha)P(setup \ in \ system \ j))$$

Then the percent difference in the value of the objective function $h$ is

$$percent \ objective \ function = h = \frac{|h_i - h_j|}{\max\{h_i, h_j\}} * 100\%$$

### 2.1.1 Test Cases

The percent loss and percent setup results are compared for varying numbers of traffic sources in two network test cases. For each test case, the system in which all of the servers are equally divided among the VPs, thus resulting in no shared servers, is compared to systems in which some of the servers are shared.

We study two test cases. In the first, the overall arrival rate to the system is constant and equal to 60 calls/time unit. Thus if there are 30 traffic sources in the network, each has an arrival rate of 2 calls/time unit. If there are 2 traffic sources, then the arrival rate of each source is 30 calls/time unit. The parameter values for the systems considered in Test Case 1 with their corresponding number of shared servers are shown in Section A.1.1

of Appendix A Table A.1. A dash in a particular column indicates that the servers could not be evenly distributed among the VPs in this case.

We also consider a second test case. This second example network contains 48 servers. The overall arrival rate to the system is constant and equal to 48 calls/time unit. The parameter values for Test Case 2 are shown in Section A.1.1 of Appendix A Table A.2.

## 2.1.2 Results

A simulation was run for each test case and parameter value. For each test case the resulting loss probability

$$P(loss) = \frac{the\ number\ of\ calls\ lost}{the\ total\ number\ of\ calls}$$

and setup probability

$$P(setup) = \frac{the\ number\ of\ calls\ incurring\ a\ setup\ cost}{the\ number\ of\ calls\ accepted\ by\ the\ system}$$

was calculated.

The partitions of servers compared in each test case are shown in Table 2.1. When comparing the performance of these partitions, we will refer to each partition by the System number associated with it in Table 2.1.

| Test Case 1 – 60 total servers | |
|---|---|
| System 1 | 0 shared servers |
| System 2 | 30 shared servers |
| System 3 | 20 shared servers |
| System 4 | 12 shared servers |

| Test Case 2 – 48 total servers | |
|---|---|
| System 1 | 0 shared servers |
| System 2 | 24 shared servers |
| System 3 | 16 shared servers |
| System 4 | 12 shared servers |

**Table 2.1**: Summary of the server partitions for the sharing experiment.

The loss and setup probabilities for Test Case 1 are shown in Section A.1.2 of Appendix A in Tables A.3 and A.4. The loss and setup probabilities for Test Case 2 are shown in Appendix A Tables A.5 and A.6. The confidence intervals for the loss probabilities and setup probabilities expressed in the tables are based on 1,000 data points and represent a 95% level of confidence that the true mean of the probability lies in the indicated range. Each data point is the result of a simulation of 1,000,000 calls.

From the loss probability and setup probability values produced by the simulations, we calculate the values of $f$, $g$, and $h$ for each test case and set of parameter values. These values are shown for Test Case 1 in Section A.1.3 of Appendix A in Tables A.7, A.8, and A.9 and for Test Case 2 in Appendix A Tables A.10, A.11, and A.12.

The graphs in Figures 2.2, 2.3, and 2.4 show the percent difference for the loss probabilities $f$ and the percent difference for the setup probabilities $g$ for Test Case 1 comparing Systems 1, 2, 3, and 4. In addition, the percent difference for the objective function $h$ has been plotted for each comparison using several values of $\alpha$. Although the points plotted are discrete points, a line has been drawn connecting points for each function to show the trend of the data.

In Figure 2.2 the percent difference for each of the three functions is shown comparing System 1 and System 2. In this graph, we are comparing the case when no shared servers are available in the network and the case when 30 of the servers are shared. The line showing $f$, the percent difference for the probability of a loss, increases quickly as the number of traffic sources increases and then levels off. This implies that as the number of traffic sources that share servers increases, the number of losses that occur will increase more slowly when compared to a system with no shared servers. This trend is also seen
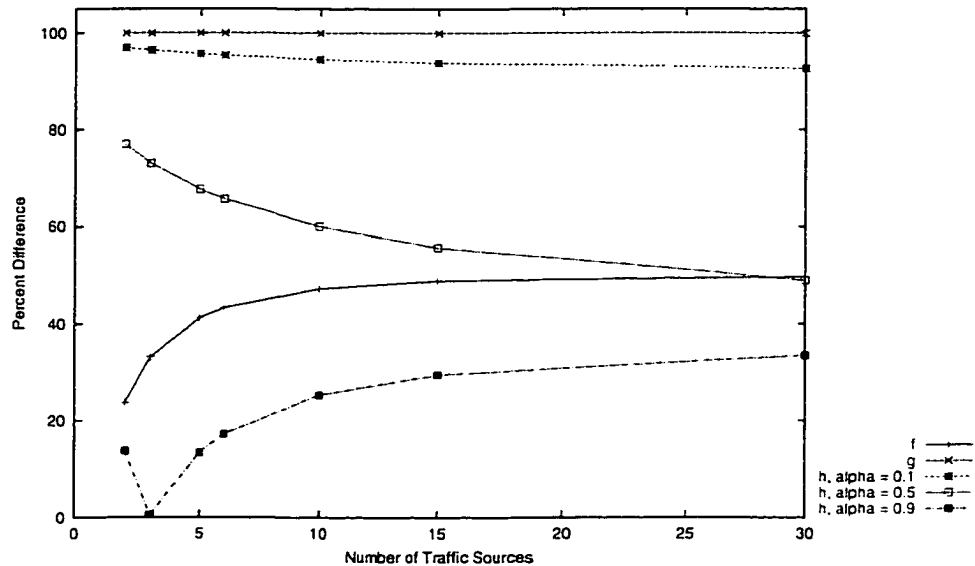
**Figure 2.2**: Results from Test Case 1 comparing System 1 and 2.

in Appendix A Table A.3. In Systems 1 and 2, when only two traffic sources enter the network, the $P(loss)$ values for these systems are relatively close in value. However, when 30 traffic sources enter the network, the network with shared servers experiences a much lower loss probability than the same network without shared servers.

The line showing the percent difference for the probability of setup $g$ is flat. Because there is never a setup cost when there are no shared servers (as in System 1), the percent difference in setup for Systems 1 and 2 is always 100%.

The lines representing the percent difference of the objective function $h$ vary with the value of $\alpha$. A low value of $\alpha$ emphasizes the importance of the setup cost. When a low value of $\alpha$ is used the resulting line mirrors the line for $g$. A high value of $\alpha$ emphasizes the importance of the loss probability. When a high value of $\alpha$ is used, the resulting line approximately mirrors the line for $f$. When a mid–range value of $\alpha$ is used, the loss

probability and the setup probability are given equal weight. The line for the mid–range value of $\alpha$ shown in Figure 2.2 lies between the lines for the objective function lines plotted with low and high values of $\alpha$.

Similar results can be seen in Figure 2.3 and Figure 2.4. These graphs compare Test Case 1, Systems 1 and 3 and Test Case 1, Systems 1 and 4 respectively. Again we see a slow increase in the percent difference for the loss probability in these graphs. The percent difference for the setup probability is constant. By varying the value of $\alpha$, emphasis can be placed on the percent difference in loss probability (resulting in a line that mirrors $f$), percent difference in setup probability (resulting in a line that mirrors $g$). When a mid–range value of $\alpha$ is used, equal importance is given to the percent difference in loss and percent difference in setup.
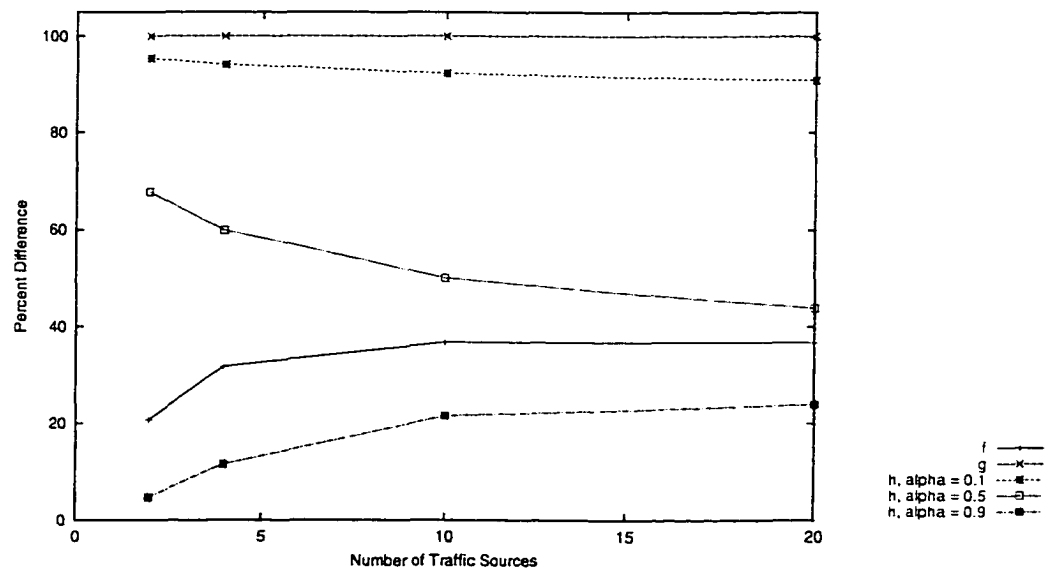


**Figure 2.3**: Results from Test Case 1 comparing System 1 and 3.
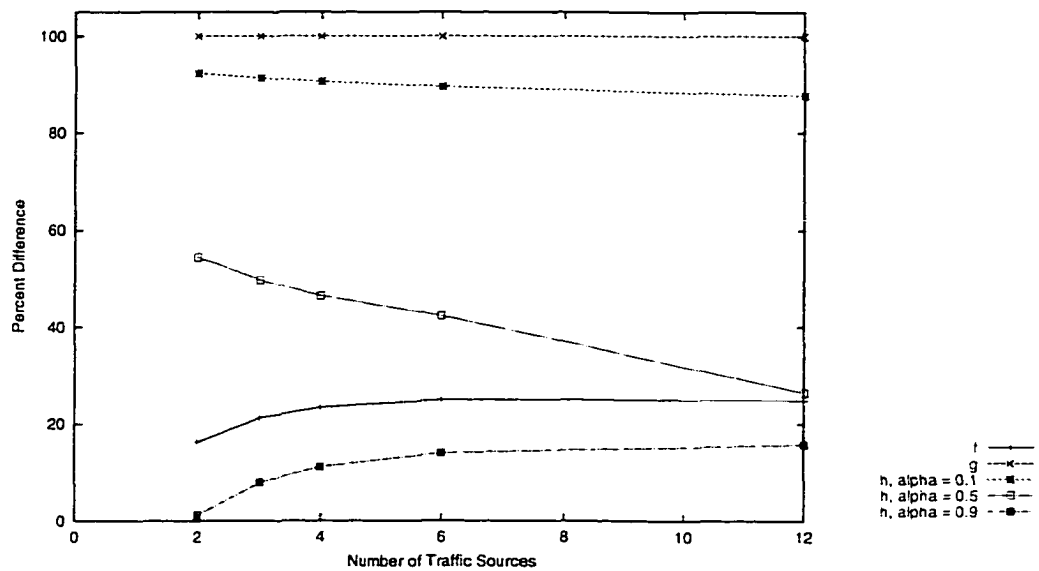
Figure 2.4: Results from Test Case 1 comparing System 1 and 4.

The graphs in Figures 2.5, 2.6 and 2.7 show $f$ the percent difference for the loss probability, $g$ the percent difference for the setup probability, and $h$ the percent difference for the objective function for the networks in Test Case 2. Again, each point in the graph in each figure represents a value of the percent difference for the systems described. Although the points plotted are discrete points, a line has been drawn connecting them to show the trend of the data.

The graphs for Test Case 2 display data with trends similar to the data in the graphs for Test Case 1. Again, we see that the line showing $f$ the percent difference for the probability of a loss increases quickly as the number of traffic sources increases. After a point, the line levels off. The graph shows that as the number of traffic sources increases, the benefits of sharing increase. As the number of traffic sources that share an unreserved (non-VP) pool of bandwidth increases, the number of losses that occur increase more slowly when
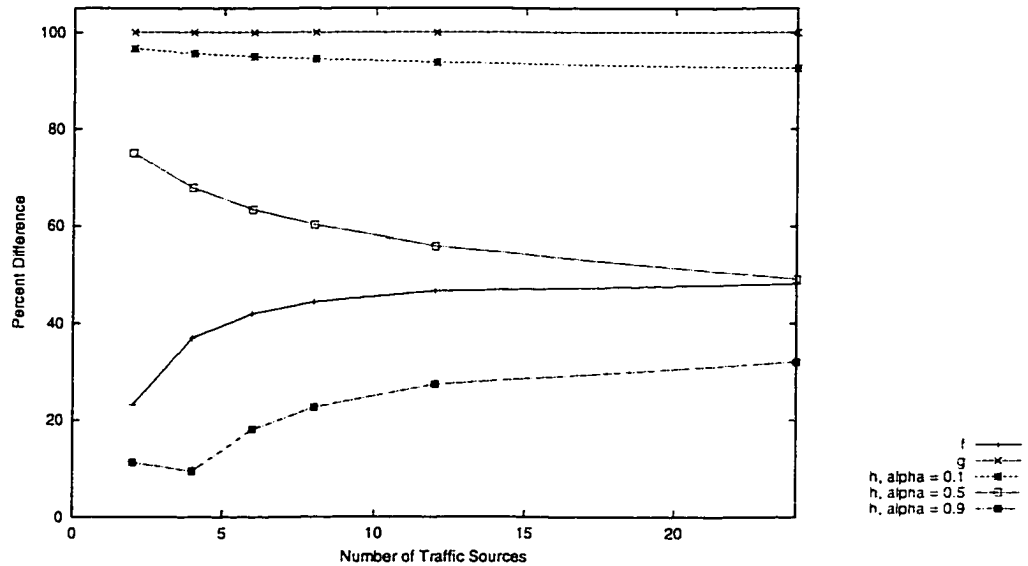
**Figure 2.5**: Results from Test Case 2 comparing System 1 and 2.

compared to a system that contains no shared bandwidth.

As in the graphs for Test Case 1, we see that the line for the percent difference in setup probability $g$ is flat. This is because we are always comparing a system with no shared bandwidth to a system with some shared bandwidth. The system without shared bandwidth will always have a $P(setup) = 0$ because all calls accepted in the network will use a VP and not incur a setup cost. Thus the percent difference for $g$ will always be 100%.

The line representing $h$ varies with the value of $\alpha$. When $\alpha$ is small, the contribution of the $P(setup)$ is emphasized in the objective function and the resulting line mirrors the line representing $g$. When $\alpha$ is large, the contribution of the $P(loss)$ is emphasized in the objective function and the resulting line mirrors the line representing $f$. When $\alpha$ has a moderate value, the contribution of the $P(loss)$ and the $P(setup)$ are equal.
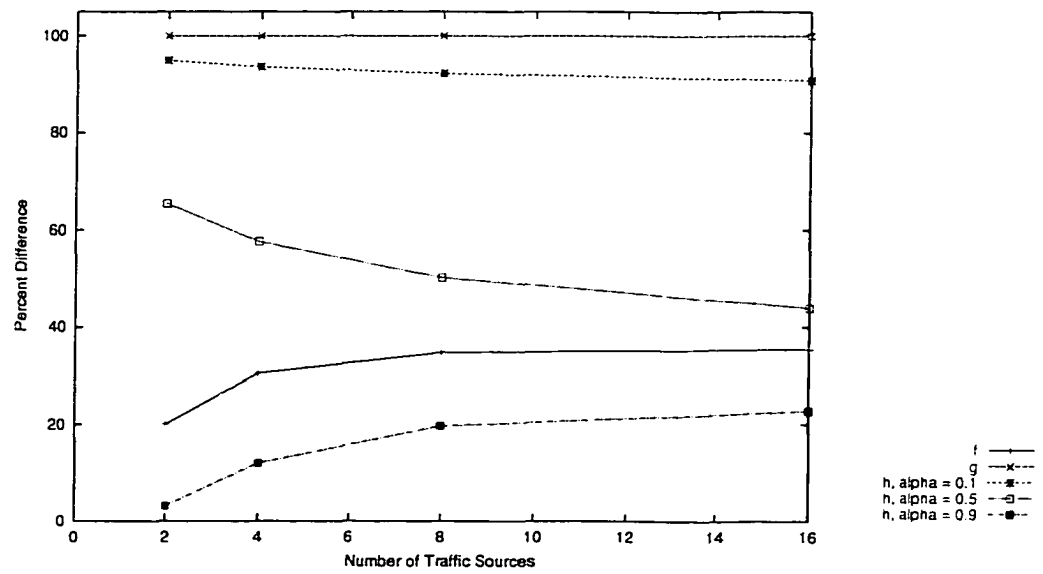
Figure 2.6: Results from Test Case 2 comparing System 1 and 3.
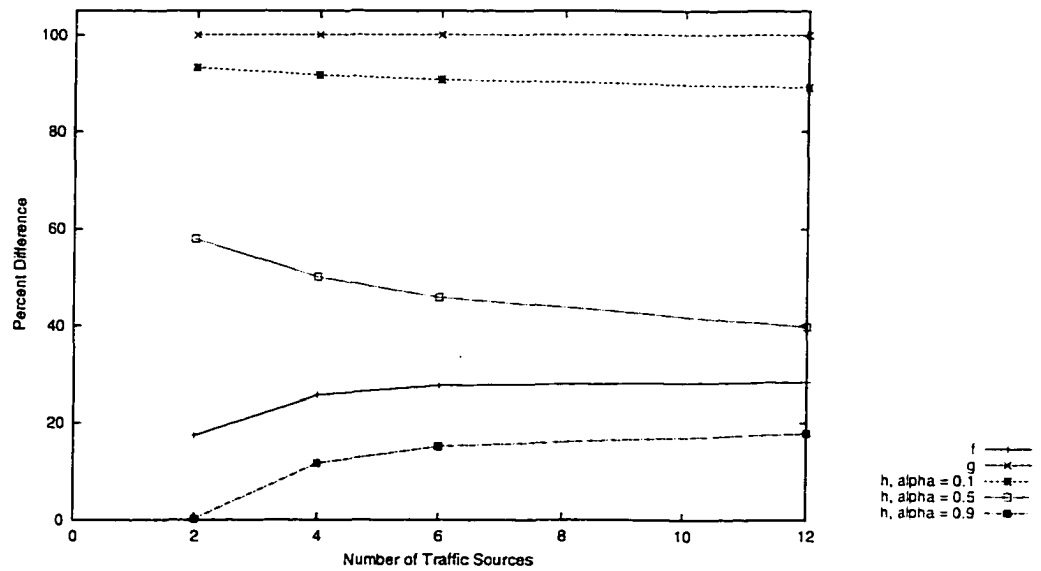


Figure 2.7: Results from Test Case 2 comparing System 1 and 4.

### 2.1.3 Conclusion

From the data collected, we conclude that a network that contains a shared pool of bandwidth will perform better in terms of loss probability when compared to a network in that all bandwidth is assigned to VPs. We further observe that as the number of traffic sources that share a common pool of bandwidth increase, the benefits of sharing increase. When more traffic sources use a shared bandwidth pool, the number of losses that occur increase more slowly than in a network that contains no shared bandwidth.

## 2.2 Other Benefits of Sharing

We have shown in the previous section that a network that allows shared bandwidth will have a lower overall call blocking probability than a network that does not share bandwidth. A small call blocking probability is a desirable characteristic in networks, thus making shared bandwidth desirable in a network layout. We argue that shared bandwidth is not only desirable but essential in some networks for a variety of reasons.

### 2.2.1 Expansion of the Solution Space

The use of shared bandwidth expands the solution space for some problem formulations. Suppose that the objective function being used to evaluate network performance is one in which there is an upper bound on the loss probability. When using this type of objective function, the goal is to minimize the probability of setup while maintaining a loss probability that is lower than the upper bound. We described this type of objective function in Section 1.1.3.

In a network in which no bandwidth is shared, there may be no VP layout and bandwidth allocation that maintains the loss probability below the upper bound. In this case, no solution can be found to the VPDBA problem. By allowing shared bandwidth, the overall loss probability can be decreased and the solution space of the VPDBA problem can be expanded.

## 2.2.2 Allows Local Control

The use of shared bandwidth provides additional local control. Consider a network in which all of the bandwidth in the network is reserved for VPs. Assume that several traffic sources are serviced by the network and that each traffic source has a dedicated VP. Suppose that one of the traffic sources experiences a burst. In a network configuration that does not allow shared bandwidth, this bursty traffic source will quickly use all of the servers assigned to it. Calls that arrive when all of the reserved servers are in use will be lost. If some of the network bandwidth is reserved for shared use, the amount of loss due to bursty traffic decreases. When some bandwidth is shared and a traffic source experiences a burst, the servers reserved for this source are used first. When all of the reserved servers are in use, the remaining calls in the traffic burst overflow the reserved servers and use the shared servers. Only when all of the reserved and shared servers are busy are the calls lost.

Shared servers can be used to handle bursts from any traffic source. There is no need to reconfigure the VP layout or bandwidth allocations of the network VPs, reducing network management costs.
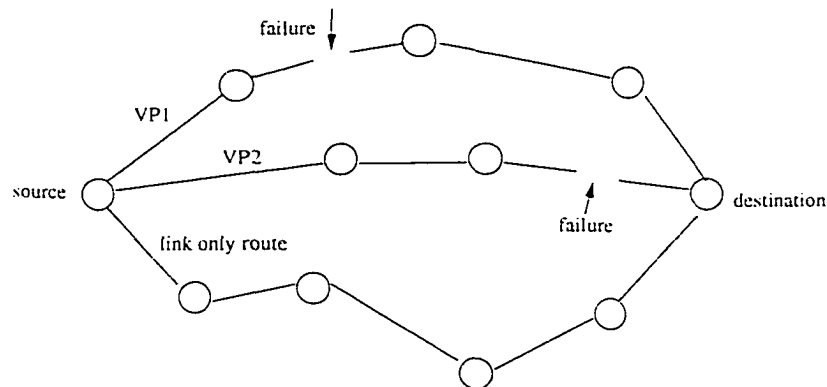
**Figure 2.8**: An example of a network with link failures.

### 2.2.3 Improved Connectivity

Sharing improves the connectivity of networks. Consider a network in which all bandwidth is assigned to VPs and only a few paths exist between a particular source-destination pair. Then a few link failures can cause severance of service. In a network that allows shared bandwidth, when a few failures occur, calls can still be serviced. If the VP assigned for a call experiences a link failure, the call can find a link only route using the shared bandwidth as shown in Figure 2.8. For a fairly connected network, a path will likely exist between the source and destination for the call.

## 2.3 Summary

We introduced a new scheme for sharing bandwidth that improves the network performance in terms of loss probability especially with a moderate to large number of sources. Our scheme does not increase the amount of work that must be done by the network or bandwidth manager and therefore has less potential to increase setup delays in a network.

Our scheme allows many streams to share a pool of unreserved bandwidth for calls that overflow from a VP assigned to an individual source. Thus our scheme reduces the need to reconfigure the VP layout and capacity allocation due to changes in traffic characteristics.

# Chapter 3

# Line Network Base Case

The VPDBA problem is NP-Complete as we discussed in Chapter 1. Because the solution to the distribution problem clearly affects the optimality of the solution to the allocation problem, we opt to solve these jointly by solving the problems in sequence iteratively. The algorithm is shown in Figure 3.1.

A key step in the solution of the VPDBA problem is the evaluation of the performance measures for the resulting layout and capacity assignment. We could estimate the performance measures using simulation. However the time needed to run a simulation of the system increases non-linearly with network size. To avoid lengthy simulations, we would like to be able to calculate performance measures theoretically by solving the equivalent Markov chain. However, as the size and complexity of the network increases, the difficulty of solving the Markov chain increases. We will propose a decomposition method that simplifies the calculation of performance measures. The performance measure approximation method proposed would be used in the second step of the algorithm shown in Figure 3.1. We investigate several methods for combining results from the decomposed networks to obtain a good approximation of the performance measures of interest.

Our goal is to provide a robust heuristic for solving the VPDBA problem. As a first

Initial parameter values

Generate layout and capacity assignments

Use layout and assignment to produce performance measures

Good enough?

No

Yes

Stop
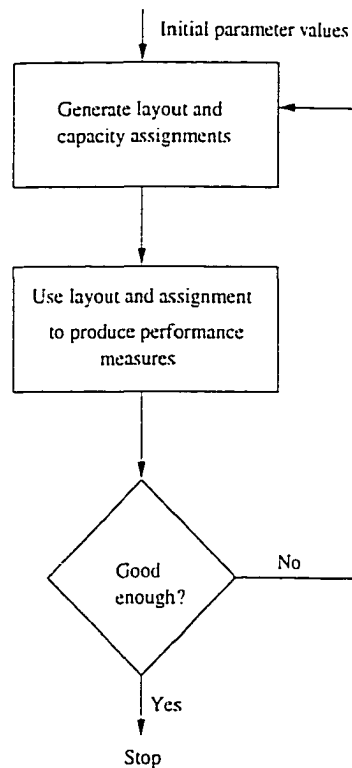
**Figure 3.1**: Algorithm for solving the VPDBA problem

step toward a solution for the VPDBA problem for a general network, we will consider the solution for a simple network with a line topology. Finding the optimal solution to the VPDBA problem for a line network is still a complex process. Therefore, we consider a simpler case initially. In this initial case, a single end–to–end VP has already been established in the line network. We develop an effective performance measure approximation method for this base case. Later we will use this base case as a sub-model for a general line network and the general network case.

## 3.1   Simplified Network Model Description

We describe the modified network model for the base case of a line network. In this model, a single end–to–end VP has been established within the network.

Consider a simple network of $N + 1$ nodes where node $i$ is connected to node $i + 1$ $\forall i, 0 \leq i \leq (N - 1)$ as shown in Figure 3.2. A VP has been established between node 0 and node $N$.
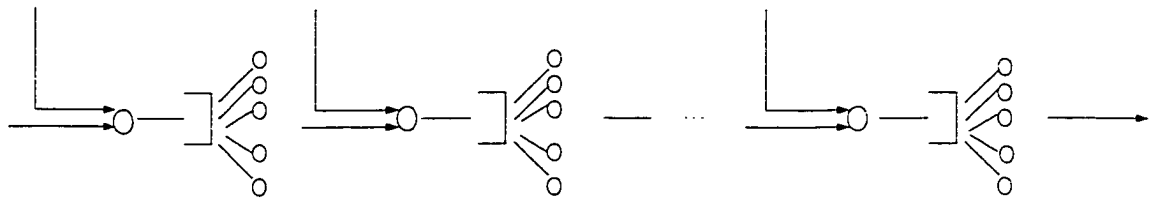


**Figure 3.2**: An example network.

- At each node $i$ there are $K_i$ servers. $V_i$ of these servers are VP servers that are reserved for use by calls from node 0 to node $N$. $SH_i$ of these servers are shared servers that can be used to handle local traffic as well as calls from node 0 to node $N$ that can not be handled by the VP servers.

- Two types of calls arrive in the network.

  1. Calls arrive at node 0 that are destined for node $N$. These arrivals are Poisson distributed with arrival rate $\lambda_{vp}$ and use the VP servers if one is available at each node between 0 and $N$ inclusive. If there is not a VP server available at each node, these calls can use the shared servers, reserving one shared server at each

      node at the time that the call arrives at node 0. If no server is available. the call

      is lost.

2. At each node $i$, $0 \leq i \leq (N-1)$, Poisson distributed local calls arrive with arrival
   rate $\lambda_i$. Local calls can only use shared servers. If no shared server is available
   at the node, then the call is lost.

- Services are exponentially distributed with parameter $\mu$.

We are then interested in the $P(loss)$, $P(setup)$ and value of the objective function of

the form $F = \alpha P(loss) + (1 - \alpha)P(setup)$. This objective function can be used to determine

the optimal VP capacity allocation for the single end–to–end VP established in the network.

## 3.2   Simulation

Before considering theoretical performance measure approximation methods, we develop a

simulation to validate our theoretical performance model. This simulation will be used to

study the performance of a simple network containing a single end–to–end VP as discussed

in Section 3.1. We present the pseudocode for the corresponding simulation and show the

validity of the simulation method.

### 3.2.1   Pseudocode

A next event simulation for the $N + 1$ node network described in Section 3.1 was written.

The pseudocode for this simulation follows.

```
generate an arrival at node 0 for the VP
for each node i
    generate a local arrival at node i

currentEvent = get the first event from the event list

while (number_of_calls_arrived < MAX_number_of_calls)
    currentTime = currentTime + currentEvent->time

    if (currentEvent->type == arrival)
        increment number_of_calls_arrived

        if (call is for VP)
            if (VP server is available)
                increment number_of_calls_requiring_no_setup
                mark one VP server used at each node i,  0 <= i <= N
                generate a service event
            else if (shared server is available at each node i, 0 <= i <= N)
                increment number_of_calls_requiring_setup
                mark one shared server used at each node i,  0 <= i <= N
                generate a service event
            else
                increment number_VP_Loss
            generate the next VP arrival

        else  // (call is local)
            if (shared server is available at this node)
                increment number_of_calls_requiring_setup
                mark one shared server used at this node
                generate a service event
            else
                increment number_Local_Loss[this node]
            generate the next local arrival at this node

    else // (call is service)
        if (call was local)
            unmark one shared server at this node
        else  // (call was VP)
            if (call was handled by VP servers)
                unmark one VP server at each node i, 0 <= i <= N
            else  // (call handled by shared servers)
                unmark one shared server at each node i, 0 <= i <= N

    currentEvent = get next event;
```

The simulation continues until the maximum number of arrivals have entered the system. The probability of a loss is calculated using the following formula:

$$P(loss) = \frac{(\sum_{i=0}^{(N-1)} number\_Local\_Loss[i]) + number\_VP\_Loss}{number\_of\_calls\_arrived}$$

and the probability of a setup is calculated using the following formula:

$$P(setup) = \frac{number\_of\_calls\_requiring\_setup}{number\_of\_calls\_requiring\_setup + number\_of\_calls\_requiring\_no\_setup}$$

## 3.2.2 The Validity of the Simulation

This section describes the network configuration studied and the solution of the equivalent Markov chain. Then the various test cases are listed and the results for the simulations and theoretical solutions are compared.

### 3.2.2.1 Markov Chain Solution Method

The following solution method is used to obtain the theoretical solution for the loss probability in each network test case. First the state space for the system is defined. Then the Markov chain is written for the system based on the state transition rates. Then we solve this chain for the state probabilities. Finally the state probabilities can be used to calculate the theoretical probability that a call will be lost.

Consider a system with 2 nodes. At each node there is one VP server and one shared server as in Figure 3.3. Then each state of the system can be uniquely described by a triple. Let

$$V_{used} = \text{the number of VP servers in use}$$
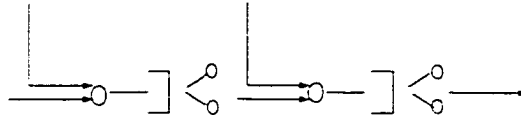
**Figure 3.3**: The system modeled in the Markov chain solution.

$SH_0$ = the number of shared servers in use at node 0

$SH_1$ = the number of shared servers in use at node 1

Then the state of the system can be written as a triple,

$$S = (V_{\text{used}}, SH_0, SH_1).$$

However, there are two transitions each with rate $\lambda_{vp}$ that can be made to state $(1,1,1)$. Suppose the system is in state $(1,0,0)$ and experiences a VP arrival. The VP server is already in use, but a shared server is available at each node. So the VP call will be admitted to the shared servers, causing the transition to state $(1,1,1)$. Suppose the system is in state $(0,1,1)$ and experiences a VP arrival. The VP server is free and so the call is admitted to this server causing the transition to state $(1,1,1)$. Because two transitions occur to the same state, $(1,1,1)$ with the same rate, we must have a method of differentiating between these two types of transitions. Otherwise when the service in state $(1,1,1)$ were completed, we wouldn't know to which state the next transition should occur. To solve this problem, a designation has been added to differentiate between a transition to state $(1,1,1)$ where a VP call uses shared servers, designated $(1,1,1,SH)$ and a transition to state $(1,1,1)$ where a VP call is using the VP server, designated $(1,1,1,VP)$.

We can then write the Markov chain for the system as shown in Figure 3.4 and solve for the probability that the system is in a given state. These state probability results can be combined to find the theoretical probability of a loss for a system. Each state probability
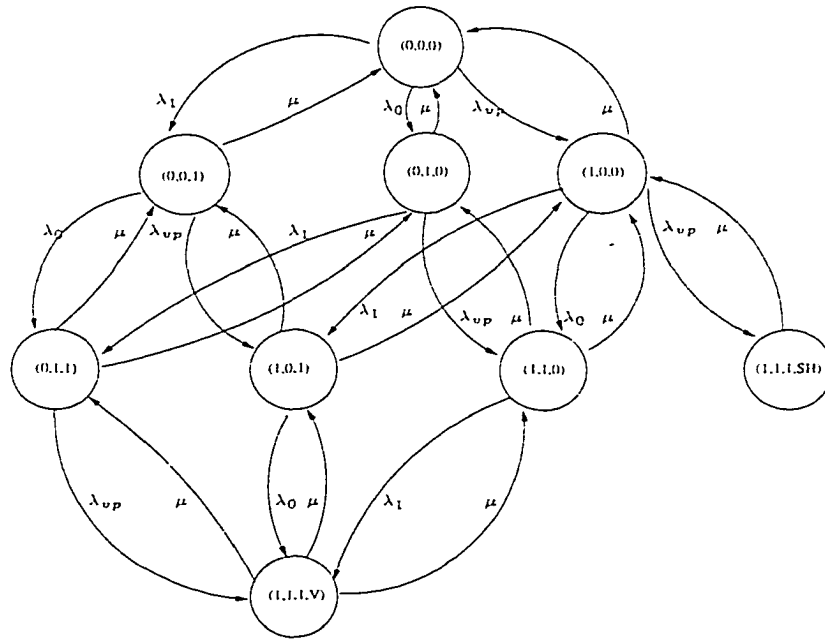
**Figure 3.4:** The Markov chain for the system in Figure 3.3.

is multiplied by the arrival rate to that state and summed. The sum is then divided by the total arrival rate to the system.

$$P(loss) = [\lambda_1 P(0,0,1) + \lambda_0 P(0,1,0) + (\lambda_0 + \lambda_1)P(0,1,1) + (\lambda_1 + \lambda_{vp})P(1,0,1)$$

$$+ (\lambda_0 + \lambda_{vp})P(1,1,0) + (\lambda_0 + \lambda_1 + \lambda_{vp})P(1,1,1,VP)$$

$$+ (\lambda_0 + \lambda_1 + \lambda_{vp})P(1,1,1,SH)]/[\lambda_0 + \lambda_1 + \lambda_{vp}]$$

#### 3.2.2.2 Test Cases

The theoretical solutions were found for the test cases listed in Section B.1.1.1 of Appendix B Table B.1. All test cases have a network as shown in Figure 3.3 and therefore can be modeled by the Markov chain shown in Figure 3.4.

### 3.2.2.3 State Probability Results

The Markov chain for each of the 10 test cases was solved for the steady state probabilities. These results are shown in Section B.1.1.2 of Appendix B Table B.2.

### 3.2.2.4 Results

Each simulation was run for 1,000,000 calls. The confidence intervals for the simulation results are based on batches of 1,000 data points and indicate a 95% confidence that the true mean lies within the indicated range. The percent difference between the two loss probability values obtained was calculated. Table 3.1 shows that the values are very close to one another in all test cases. The maximum percent difference for the test cases considered was less than 25%. It should be noted that loss probabilities must have a value between 0 and 1 by definition. While a difference of 25% represents a potential difference +/-0.25, in our test cases the actual difference between the simulation result and theoretical result that resulted in this percent difference was 0.04. In all but 2 test cases, the percent difference was less than 5%. Thus we conclude that valid performance measures can be obtained by using our simulation method.

## 3.3 Approximate Markov Model

We want to be able to calculate the overall loss probability for a given system without running a simulation. Clearly the results can be obtained by solving the equivalent Markov chain for the system but the state space becomes very large as the number of servers or number of nodes increase.

| Test Case | Theoretical P(loss) | Simulated P(loss) | Percent Difference |
|---|---|---|---|
| 1 | 0.3500753703 | 0.342678 +/- 0.000030 | 2.11308 |
| 2 | 0.3333333 | 0.319288 +/- 0.000031 | 4.21359 |
| 3 | 0.9009563858 | 0.871313 +/- 0.000023 | 3.29021 |
| 4 | 0.5938601909 | 0.659844 +/- 0.000037 | 9.99991 |
| 5 | 0.7211203632 | 0.719222 +/- 0.000031 | 0.263252 |
| 6 | 0.7503944727 | 0.720470 +/- 0.000031 | 3.98783 |
| 7 | 0.8663134572 | 0.865871 +/- 0.000022 | 0.0510736 |
| 8 | 0.1802141443 | 0.136488 +/- 0.000024 | 24.2634 |
| 9 | 0.9131143467 | 0.910919 +/- 0.000019 | 0.240424 |
| 10 | 0.47817245 | 0.472787 +/- 0.000033 | 1.12626 |

**Table 3.1**: Comparison of theoretical and simulation results for the test cases in Table B.1.

We would like to break the larger network, as shown in Figure 3.2, into a series of smaller systems, each with one node, as in Figure 3.5. These smaller systems can easily be solved by writing the corresponding Markov chain and solving the balance equations for each system. Then the results from these smaller chains can be combined to approximate the solution for the original larger system. Because of the dependence between the systems (which we will demonstrate), we need to find a good method for combining the results from the smaller systems.

### 3.3.1 Notation

We will be considering the solution to the base case network as described in Section 3.1. Subsequently, the following notation will be used.

- $\lambda_i$ refers to the arrival rate for local calls at node $i$.

- $\lambda_{vp}$ refers to the arrival rate for VP calls in the original system.

- $\lambda_{vp_i}'$ refers to the arrival rate for VP calls in the smaller system. $\lambda_{vp_i}'$ is calculated as
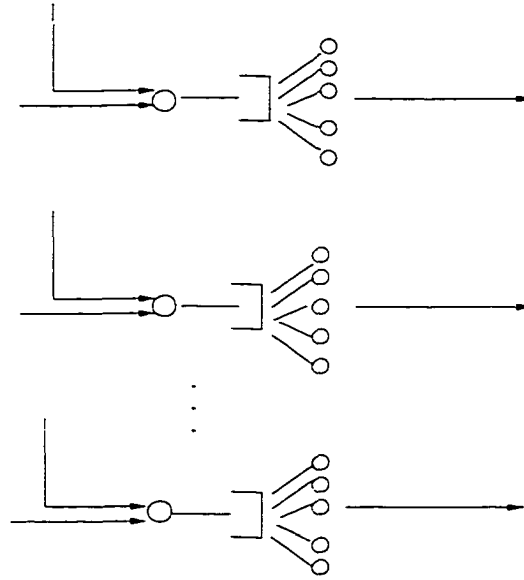
**Figure 3.5**: The example network broken into smaller systems.

follows:

$$(\lambda_{vp_i})' = (1 - (P(loss_{vp_{i-1}}))')(\lambda_{vp_{i-1}})' \ \forall i \geq 1$$

and

$$(\lambda_{vp_0})' = \lambda_{vp_0}$$

and

$$P(loss_{vp_0})' = P(loss_{vp_0})$$

- $P_i^v$ is the probability that the VP servers are all busy at node $i$ where $\lambda_{vp}$ is used as the arrival rate for the VP calls.

- $P_i^{v'}$ is the probability that the VP servers are all busy at node $i$ where $\lambda_{vp_i}'$ is used as the arrival rate for the VP calls.

- $P(loss_i)$ is the probability of a local loss at node $i$ in the smaller system containing

node $i$ where $\lambda_{vp}$ is used as the arrival rate for the VP calls.

- $P(loss_i)'$ is the probability of a local loss at node $i$ in the smaller system containing node $i$ where $\lambda_{vp_i}'$ is used as the arrival rate for the VP calls.

- $P(loss_{vp_i})$ is the probability of a VP loss in the smaller system containing node $i$ where $\lambda_{vp}$ is used as the arrival rate for the VP calls.

- $P(loss_{vp_i})'$ is the probability of a VP loss in the smaller system containing node $i$ where $\lambda_{vp_i}'$ is used as the arrival rate for the VP calls.

### 3.3.2 Proof of Subsystem Dependence

If the subsystems were independent, we would have a product–form network and would be able to derive an exact solution for the performance measures for the original system. We now show that the subsystems are not independent.

**Theorem 3.1** *Consider a line network containing $N$ nodes. Suppose that the network is separated into $N$ subsystems, each containing a single node, then these subsystems are not independent.*

**Proof:**

1. Consider a system with 2 nodes as in Figure 3.3. The network contains one shared server and one VP server at each node. Let $A$ be the event that no shared servers are in use at node 0. Let $B$ be the event that no shared servers are in use at node 1.

2. Assume that the subsystems are independent. The system could be split as shown in Figure 3.6.
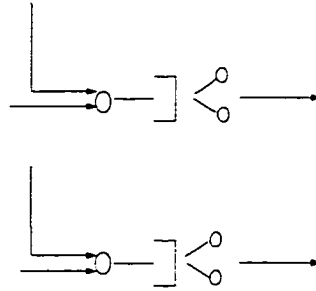
**Figure 3.6**: The network in Figure 3.3 broken into smaller systems.

3. Let $P_s(A \cap B)$ be the probability that there are no shared servers in use at node 0 and that there are no shared servers in use at node 1 in the original system. Let $P_0(A)$ be the probability that there are no shared servers in use at node 0 in the subsystem containing node 0. Let $P_1(B)$ be the probability that there are no shared servers in use at node 1 in the subsystem containing node 1.

4. Then $P_s(A \cap B) = P_0(A)P_1(B)$ if the events $A$ and $B$ are independent.

5. To prove that the subsystems are not independent, we need only show that for some system, $P_s(A \cap B) \neq P_0(A)P_1(B)$.

6. First consider system $s$. For the input parameters shown write and solve the corre-

$$\lambda_0 = 0.5 \quad V_0 = 1$$
$$\lambda_1 = 5.0 \quad V_1 = 1$$
$$\lambda_{vp} = 10.0 \quad SH_0 = 1$$
$$\mu = 1.0 \quad SH_1 = 1$$

sponding Markov chain to calculate the theoretical state probabilities.

$$
\begin{aligned}
P[0,0,0] &= 0.00502513 \\
P[0,0,1] &= 0.0251256 \\
P[0,1,0] &= 0.00251256 \\
P[0,1,1] &= 0.0125628 \\
P[1,0,0] &= 0.0502513 \\
P[1,0,1] &= 0.251256 \\
P[1,1,0] &= 0.0251256 \\
P[1,1,1,VP] &= 0.125628 \\
P[1,1,1,SH] &= 0.502513
\end{aligned}
$$

$$
\begin{aligned}
\text{So } P_s(A \cap B) &= P[0,0,0] + P[0,0,1] \\
&= 0.00502513 + 0.0251256 \\
&= 0.03015073
\end{aligned}
$$

7. Now write the Markov chain for each of the subsystems to obtain the theoretical state probability results.

| Subsystem 0 | |
|---|---|
| $\lambda_0$ | $= 0.5$ |
| $\lambda_{vp}$ | $= 10.0$ |
| $\mu$ | $= 1.0$ |
| $V_0$ | $= 1$ |
| $SH_0$ | $= 1$ |
| | |
| $P[0,0]_0$ | $= 0.0152931$ |
| $P[0,1]_0$ | $= 0.075616$ |
| $P[1,0]_0$ | $= 0.0849618$ |
| $P[1,1]_0$ | $= 0.824129$ |

| Subsystem 1 | |
|---|---|
| $\lambda_1$ | $= 5.0$ |
| $\lambda_{vp}$ | $= 10.0$ |
| $\mu$ | $= 1.0$ |
| $V_1$ | $= 1$ |
| $SH_1$ | $= 1$ |
| | |
| $P[0,0]_1$ | $= 0.00936849$ |
| $P[0,1]_1$ | $= 0.0815406$ |
| $P[1,0]_1$ | $= 0.0589868$ |
| $P[1,1]_1$ | $= 0.850104$ |

So

$$P_0(A) = P_0[0,0] + P_0[0,1]$$

$$= 0.0152931 + 0.0849618$$

$$= 0.1002549$$

And

$$P_1(B) = P_1[0,0] + P_1[0,1]$$

$$= 0.00936849 + 0.0589868$$

$$= 0.06835529$$

Therefore

$$P_0(A)P_1(B) = (0.1002549)(0.06835529)$$

$$= 0.0068529528$$

However

$$0.0068529528 \neq 0.03015073$$

So

$$P_s(A \cap B) \neq P_0(A)P_1(B)$$

Thus events $A$ and $B$ are not independent, implying that the subsystems are not independent. Hence the events at one node in the system effect the other nodes in the system.

■

Because the subsystems are not independent, we investigate several methods of approximating the performance measures of the original system using the subsystem results and compare the approximated performance measures to the performance measures found by simulation.

### 3.3.3 Test Cases

Before presenting our approximation methods and results, we describe the test cases used to test the approximations. We focus on the probability of loss in the system as a performance measure. Several networks with various parameter values were examined. Table B.3 in Section B.2.1 of Appendix B lists the parameter values for the 13 test cases considered. The test cases were designed with a variety of network sizes from 2 to 15 nodes and with various arrival rates to test the effect of these parameters on the overall loss probability.

### 3.3.4 Simulation Results

A simulation was run for each of the test cases in Table B.3. The resulting loss probability for each test case is shown in Appendix B Section B.2.2 Table B.4. All simulations were run for 1,000,000 calls. The confidence intervals indicated were calculated based on batches of 1,000 samples and represent a 95% confidence that the true mean lies within the indicated range of the given mean. We will later use these simulation results in a comparison with the performance measures calculated using our approximation method.

### 3.3.5 Subsystems with Original Arrival Rates

Each of the networks described in Section 3.3.3 was divided into a series of subsystems each containing a single node. Then the probability of loss for each of the subsystems was found by solving the Markov chain for each of the corresponding systems. If the original system given in Section 3.3.3 has $i$ nodes, then it was broken into $i$ subsystems. Each of these subsystems has two Poisson distributed call arrival streams having arrival rates $\lambda_i$ and $\lambda_{vp}$. The service rate in the subsystems is equal to the service rate in the original system $\mu$. The

subsystems each contain the same number of VP servers and shared servers as the original system.

The performance measures obtained by solving these system include the state probabilities, probability that the VP servers are busy at node 0, probability of loss occurring to a local call, and the probability of a loss occurring to a VP call. These performance measures are shown for each of the network test cases in Section B.2.3 of Appendix B Table B.5. Blank row entries appear in Table B.5 when the probabilities in these rows will not be used in the approximation calculations.

### 3.3.6 Subsystems with Thinned Arrival Rates

When the network is broken into subsystems, it was assumed that the VP input stream has the same rate at each node. However, in the original model when VP calls arrive at node 0 they are either accepted or lost based on the availability of servers at *all* nodes. So in the original system, the decision to accept or reject a VP call is made at node 0 only. Other nodes only experience arrivals that were accepted at node 0. Therefore, the other nodes in the original system will experience lower arrival rates for VP calls.

The following example illustrates the significance of this observation. Suppose the arrival rate of VP calls is high and the arrival rate of local calls at node 0 is high and the arrival rate of local calls at all other nodes is low. Then in the large system, many VP calls will be rejected because there is a high probability that all VP servers will be busy because the VP call arrival rate is high. There is also a high probability that a shared server will not be available at every node in the network because the arrival rate of local calls to node 0 is high. However, when this network is split into subsystems each with the same VP call

arrival rate, this same behavior is not modeled. The subsystem containing node 0 will still reject many VP calls since the VP arrival rate is high and the arrival rate of calls to node 0 is high, so there is a good chance that no server will be available for a call. However the other subsystems will admit a larger number of VP calls, because while the virtual path arrival rate is the same, the local call arrival rate is lower. Thus calls will be admitted to shared servers at these nodes that would have been rejected in the original system.

By assuming all subsystems have a VP input stream with rate $\lambda_{vp}$, we are over–estimating the arrival rate of VP calls to some of the nodes in the network. To compensate for this fact, we will thin the arrival rate of VP calls to each node in the network based on the following formula.

$$(\lambda_{vp_i})' = (1 - (P(loss_{vp_{i-1}}))')(\lambda_{vp_{i-1}})' \ \forall i \geq 1$$

and

$$(\lambda_{vp_0})' = \lambda_{vp_0}$$

and

$$P(loss_{vp_0})' = P(loss_{vp_0})$$

The intuition for this formula is as follows. At each node $i$, a number of VP calls will be lost. Thus at future nodes $j, j > i$, these VP calls will not arrive since they have already been lost at node $i$. The thinned loss rate formula reduces the VP call arrival rate at each node. To calculate the resulting thinned rate, we multiply the probability that a VP call is not rejected by the previous node $(1 - (Ploss_{vp_{i-1}}))'$ by the VP call arrival rate at the previous node $(\lambda_{vp_{i-1}})'$. This approximates the actual arrival rate of VP calls for a given node.

Table B.6 in Section B.2.4 of Appendix B shows the results obtained by solving the

Markov chain for each of the subsystems using the thinned arrival rates. The state proba-bilities, probability that the VP servers are all busy at node 0, probability of loss occurring to a local call, and the probability of a loss occurring to a VP call of each of the network test cases are shown.

### 3.3.7 Approximations

We now propose several approximation methods and then study their performance. The approximation methods are all based on the idea of using the steady state probabilities from the Markov chain solutions of the subsystems to calculate the loss rate for local calls and the loss rate for VP calls. These loss rates are summed and divided by the total rate at which calls enter the system. The intuition that we used to develop each approximation is explained as each approximation is defined.

**Approximation 1** As a first approximation, we break the larger system into a series of single node systems that are easy to solve using a Markov chain. In the series of smaller systems, there are $N$ streams of VP calls, one in each of the smaller systems. In the larger system there is only a single stream of VP calls. As a first approximation, we include all $N$ streams of VP calls in the formula.

$$P(loss) = \frac{\sum_{i=0}^{N-1} \lambda_i P(loss_i) + \sum_{i=0}^{N-1} \lambda_{vp} P(loss_{vp_i})}{(\sum_{i=0}^{N-1} \lambda_i) + (N)(\lambda_{vp})}$$

**Approximation 2** We hypothesized that the VP calls were getting too much emphasis in Approximation 1. The VP calls are counted $N$ times in Approximation 1. In Approximation 2, we tried to correct this by counting the VP calls just once in the numerator and once in the denominator. Because there are really $N$ loss probabilities for the VP calls, one for

each of the subsystems, we need a method of combining these results into a single measure. We used the straight average of the VP loss rate over the $N$ systems in the numerator and used $\lambda_{vp}$ in the denominator.

$$P(loss) = \frac{\sum_{i=0}^{N-1} \lambda_i P(loss_i) + (\lambda_{vp} \frac{1}{N} \sum_{i=0}^{N-1} P(loss_{vp_i}))}{(\sum_{i=0}^{N-1} \lambda_i) + \lambda_{vp}}$$

**Approximation 3** In Approximation 3, we use the idea of thinning the VP arrival rates based on the number of VP calls that were lost in the previous subsystem. This idea was discussed in Section 3.3.6. So if a VP call is lost in the subsystem containing node $i$, we decrease the average arrival rate to the subsystem containing node $i + 1$ accordingly. In other words

$$(\lambda_{vp_i})' = (1 - (P(loss_{vp_{i-1}}))')(\lambda_{vp_{i-1}})' \ \forall i \geq 1$$

and

$$(\lambda_{vp_0})' = \lambda_{vp_0}$$

and

$$P(loss_{vp_0})' = P(loss_{vp_0})$$

We then substitute the value of $(\lambda_{vp_i})'$ for $\lambda_{vp}$ and the substitute the value of $P(loss_{vp_i})'$ for $P(loss_{vp_i})$ in Approximation 1 and obtain Approximation 3.

$$P(loss) = \frac{\sum_{i=0}^{N-1} \lambda_i P(loss_i) + \sum_{i=0}^{N-1} (\lambda_{vp_i})'(P(loss_{vp_i})')}{(\sum_{i=0}^{N-1} \lambda_i) + \sum_{i=0}^{N-1} (\lambda_{vp_i})'}$$

**Approximation 4** In Approximation 4, we modify Approximation 3 to reflect the loss rates for the local calls for the subsystems with an arrival rate of $\lambda_{vp_i}'$ for VP calls.

$$P(loss) = \frac{\sum_{i=0}^{N-1} \lambda_i (P(loss_i)') + \sum_{i=0}^{N-1} (\lambda_{vp_i})'(P(loss_{vp_i})')}{(\sum_{i=0}^{N-1} \lambda_i) + \sum_{i=0}^{N-1} (\lambda_{vp_i})'}$$

**Approximation 5** In Approximation 5, we modify Approximation 2 by substituting the thinned rates and corresponding loss probabilities found using $\lambda_{vp_i}'$. An average of the $\lambda_{vp_i}'$ values has also been substituted for the value of $\lambda_{vp_i}$ in the denominator. The average is thought to better reflect the original system that contains a single VP arrival stream.

$$P(loss) = \frac{\sum_{i=0}^{N-1} \lambda_i (P(loss_i)') + \frac{\sum_{i=0}^{N-1} (\lambda_{vp_i})'(P(loss_{vp_i})')}{N}}{(\sum_{i=0}^{N-1} \lambda_i) + \frac{\sum_{i=0}^{N-1} (\lambda_{vp_i})'}{N}}$$

**Approximation 6** In Approximation 6, we attempt to weight the loss rates based on the arrival rates of the various VP streams. Rather than taking a straight average of the arrival rate multiplied by the probability of a loss for each subsystem in the numerator, we divide this quantity by the sum of the arrival rates. The denominator still contains a straight average of the thinned VP arrival rates.

$$P(loss) = \frac{\sum_{i=0}^{N-1} \lambda_i (P(loss_i)') + \frac{\sum_{i=0}^{N-1} (\lambda_{vp_i})'(P(loss_{vp_i})')}{\sum_{i=0}^{N-1} (\lambda_{vp_i})'}}{(\sum_{i=0}^{N-1} \lambda_i) + \frac{\sum_{i=0}^{N-1} (\lambda_{vp_i})'}{N}}$$

**Approximation 7** In Approximation 7, we experiment with a method of obtaining a better approximation for the VP arrival rate. We want to simplify the approximation and reflect the fact that the average $\lambda_{vp_i}'$ is very close to the original $\lambda_{vp}$.

$$P(loss) = \frac{\sum_{i=0}^{N-1} \lambda_i (P(loss_i)') + \sum_{i=0}^{N-1} (\lambda_{vp_i})'(P(loss_{vp_i})')}{(\sum_{i=0}^{N-1} \lambda_i) + \lambda_{vp}}$$

**Approximation 8** In Approximation 8, we attempt to improve the accuracy of the previous approximations by using true weighted averages of the VP loss rates. We multiply each VP loss rate by the arrival rate and divide by the sum of the arrival rates to obtain a weighted

average.

$$P(loss) = \frac{\sum_{i=0}^{N-1} \lambda_i (P(loss_i)') + \frac{\sum_{i=0}^{N-1} ((\lambda_{vp_i})')^2 (P(loss_{vp_i})')}{\sum_{i=0}^{N-1} (\lambda_{vp_i})'}}{(\sum_{i=0}^{N-1} \lambda_i) + \frac{\sum_{i=0}^{N-1} ((\lambda_{vp_i})')^2}{\sum_{i=0}^{N-1} (\lambda_{vp_i})'}}$$

**Approximation 9** Approximation 9 is a modification of Approximation 7. We use an approximation of the weighted average of the VP call loss rate.

$$P(loss) = \frac{\sum_{i=0}^{N-1} \lambda_i (P(loss_i)') + \lambda_{vp} \frac{\sum_{i=0}^{N-1} (\lambda_{vp_i})' (P(loss_{vp_i})')}{\sum_{i=0}^{N-1} (\lambda_{vp_i})'}}{(\sum_{i=0}^{N-1} \lambda_i) + \lambda_{vp}}$$

**Approximation 10** Approximation 10 uses a different method of calculating the probability of a loss of local and VP calls. In this approximation, let

$P_i^r$ = the probability that the shared servers are all busy at node i

$P_i^v$ = the probability that the VP servers are all busy at node i

Then let

$P^{v,r}$ = the probability that a VP call is rejected by the shared servers.

$$P^{v,r} = 1 - \prod_{i=0}^{N-1} (1 - P_i^r)$$

Note that a VP call is rejected by the shared servers when there is not a shared server available at each node. Since $P_i^r$ is the probability that the shared servers are all busy at node i, then $(1 - P_i^r)$ is the probability that there is a shared server available at node i. Then $(\prod_{i=0}^{N-1}(1 - P_i^r))$ is the probability that there is a shared server available at all nodes in the network. So $P^{v,r}$ is the probability that a shared server is not available at all nodes in the network.

Then let

$$P_{vp} = \text{probability that a VP call is rejected} = P_0^v \ P^{v,r}$$

Therefore the overall loss probability for the system is

$$P(Loss) = \frac{(\sum_{i=0}^{N-1} \lambda_i P_i^r) + \lambda_{vp} P_{vp}}{(\sum_{i=0}^{N-1} \lambda_i) + \lambda_{vp}}$$

**Approximation 11** Approximation 11 is a modification of Approximation 10 using the thinned rates and weighted averages.

$$P(Loss) = \frac{(\sum_{i=0}^{N-1} \lambda_i (P_i^r)') + \frac{\sum_{i=0}^{N-1} ((\lambda_{vp})')^2}{\sum_{i=0}^{N-1} (\lambda_{vp})'} (P_{vp})'}{(\sum_{i=0}^{N-1} \lambda_i) + \frac{\sum_{i=0}^{N-1} ((\lambda_{vp})')^2}{\sum_{i=0}^{N-1} (\lambda_{vp})'}}$$

### 3.3.8   Approximation Results

Table B.7 in Section B.2.5 of Appendix B displays the resulting loss probabilities calculated using the subsystem performance measures shown in Table B.5 and Table B.6 and the above approximation methods. For each test case, the loss probability for the simulation of the original system is shown for comparison. We omit the confidence intervals for the simulation results in Table B.7 because the intervals were presented previously in Table B.4 and were extremely small.

Table 3.2 shows the difference between the probability of a loss found by simulating the original system and approximate probability of a loss found using the approximation methods described. A negative (positive) difference indicates that the simulation result was smaller (larger) than the approximation result. In the table, an underline indicates that for the given test case, the approximation result in this column is closest to the simulation result. The sum of the absolute value of all the differences for each approximation method

was calculated and appears as the last row in the table. This sum represents the overall performance of the approximation method.

| test case | difference 1 | difference 2 | difference 3 | difference 4 | difference 5 | difference 6 |
|---|---|---|---|---|---|---|
| 1 | 0.027756 | 0.003073 | 0.027732 | 0.027773 | 0.003101 | 0.001228 |
| 2 | 0.005869 | 0.001401 | 0.005872 | 0.005910 | 0.001449 | 0.000269 |
| 3 | 0.008423 | 0.001689 | 0.008236 | 0.007751 | 0.000927 | 0.000140 |
| 4 | 0.039309 | 0.004959 | 0.038334 | 0.038410 | 0.004754 | 0.003276 |
| 5 | 0.029791 | 0.001133 | 0.023608 | 0.025621 | -0.000272 | -0.014354 |
| 6 | 0.058030 | -0.029472 | 0.035663 | 0.101149 | 0.023994 | 0.040806 |
| 7 | 0.035274 | -0.034289 | 0.029429 | 0.049612 | -0.012345 | -0.009440 |
| 8 | -0.038615 | -0.066470 | 0.111887 | 0.183751 | 0.187587 | 0.308739 |
| 9 | -0.002095 | -0.045636 | 0.052488 | 0.073370 | -0.002066 | 0.000742 |
| 10 | 0.002738 | -0.063659 | 0.074533 | 0.126093 | 0.042790 | 0.136802 |
| 11 | 0.079138 | -0.041674 | 0.108392 | 0.143217 | 0.031901 | 0.038188 |
| 12 | -0.076604 | -0.136654 | 0.088649 | 0.187374 | 0.129443 | 0.163490 |
| 13 | -0.008181 | -0.137250 | 0.052926 | 0.134134 | 0.017168 | 0.022021 |
| SUM | 0.411823 | 0.567359 | 0.657749 | 1.104165 | 0.457797 | 0.739495 |

| test case | difference 7 | difference 8 | difference 9 | difference 10 | difference 11 |
|---|---|---|---|---|---|
| 1 | -0.000580 | 0.003106 | 0.003144 | 0.000845 | 0.000857 |
| 2 | -0.000854 | 0.001449 | 0.001470 | 0.001149 | 0.001185 |
| 3 | -0.000618 | 0.000927 | 0.000942 | 0.001012 | 0.000290 |
| 4 | -0.000534 | 0.004749 | 0.005033 | -0.000127 | -0.000173 |
| 5 | -0.006283 | -0.000462 | 0.003323 | -0.005628 | -0.005885 |
| 6 | -0.068005 | 0.024792 | 0.037422 | -0.096350 | -0.030545 |
| 7 | -0.040041 | -0.008566 | -0.003938 | -0.049329 | -0.026461 |
| 8 | -0.084778 | 0.126266 | 0.185387 | -0.205076 | -0.002038 |
| 9 | -0.045306 | -0.002328 | 0.005727 | -0.060181 | -0.026366 |
| 10 | -0.084327 | 0.025340 | 0.086388 | 0.098066 | -0.039970 |
| 11 | -0.057308 | 0.031095 | 0.046515 | -0.029270 | -0.020197 |
| 12 | -0.079277 | 0.091908 | 0.158347 | -0.231263 | 0.010241 |
| 13 | -0.075235 | 0.014566 | 0.043888 | -0.193484 | -0.024838 |
| SUM | 0.543146 | 0.335554 | 0.581524 | 0.971780 | 0.189046 |

**Table 3.2**: The differences between the simulated and approximated loss probability values for each test case.

### 3.3.8.1 Best Individual Performances

Approximation 11 performed well on the average and also came closest to the simulation result in 4 of the 13 test cases. Approximations 1, 5, 6, 7 and 9 also performed well in some of the test cases. Approximation 6 was closest to the simulation result in 3 of the 13 test cases. Approximations 1 and 5 were closest to the simulation result in 2 of the 13 test cases. Approximations 7 and 9 were closest to the simulation result in 1 of the 13 test cases.

### 3.3.8.2 Best Overall Performance

Four of the approximations gave good results: 1, 5, 8, and 11. We do not know why Approximation 1 works well. We expected Approximation 5 to work well. It uses thinned rates for the arrival rates of the VP calls and an average of the thinned loss rates for the VP calls. We expected Approximation 8 to work well for the same reason. It also uses the thinned rates for the arrival rates of the VP calls and a weighted average of the thinned loss rates for the VP calls. Approximation 11 gave the smallest sum of the differences for the 13 test cases. Based on these test cases, it is the most accurate approximation method. Approximation 11 is based on finding the probability that the shared and VP servers are busy at each node and using these probabilities to calculate the probability of a loss. This approximation uses the thinned rates as well as a weighted value for $\lambda'_{vp}$ and is expected to perform well.

### 3.3.8.3 Upper and Lower Bounds

From the results it appears that Approximation 7 is an upper bound for the loss probability. Approximations 3 and 4 appear to be lower bounds for the loss probability, where

Approximation 3 gives a tighter bound in almost all of the test cases (the exception being test case 3).

We would expect the results from Approximation 7 to be larger than the actual loss probabilities because this approximation over–compensates for the losses due to the VP calls in the numerator only. Thus the numerator is larger than it should be and the resulting loss probability approximation is larger than the loss probability result in the simulation for all test cases.

Approximations 3 and 4 over–compensate for the losses due to the VP calls in both the numerator and denominator. This over–compensation results in a lower calculated loss probability than is seen in the simulation of the original system. It is interesting that Approximation 3 gives a tighter bound than Approximation 4. The only difference between the approximations is that Approximation 3 uses the values for $P(loss_i)$ from the system that uses $\lambda_{vp}$ whereas Approximation 4 uses the values for $P(loss_i)'$ from the system that uses the thinned arrival rates for VP calls $\lambda'_{vp}$. This suggests that the subsystem results that use original VP arrival rate may be more accurate than subsystem results that use the thinned VP arrival rates. We investigate the effect of $\lambda_{vp}$ and $\lambda_{vp_i}'$ on the loss probabilities in the next section.

### 3.3.9 The Effect of VP Arrival Rates on the Approximations

#### 3.3.9.1 The Effect of VP Arrival Rates on $P_i^r$

To gain further understanding of the results obtained with the approximation methods, we examined the quantities used in the approximations, paying special attention to those used in Approximations 10 and 11. Approximation 11 gave the best overall performance

for the test cases studied. However, Approximation 10 did not perform nearly as well in spite of the similarity between its formula and the formula used in Approximation 11. In order to understand why, we studied the results found in each test case. The individual values for $P_i^r = P(loss_i)$ and $P_{vp}$ were compared for the simulation of the original system, calculations using $\lambda_{vp}$ as the VP arrival rate, and calculations using $\lambda'_{vp_i}$ as the VP arrival rate. Table B.8 in Section B.2.6.1 of Appendix B shows the values for $P_i^r$ and $P_{vp}$ for each of the test cases in each of these situations. All simulations were run for 1,000,000 calls. The confidence intervals for the simulations are based on batches of 1,000 data points and represent a 95% confidence that the mean lies within the indicated range of the sample mean.

In many of the test cases, the value of $P_i^r$ found by solving the subsystem is larger than the corresponding value of $P_i^r$ found by simulation of the original system. In general the $P_0^r$ value in each subsystem as found using $\lambda'_{vp}$ as the VP arrival rate is closer to the $P_0^r$ value found for the original system than the $P_0^r$ value for the subsystem using $\lambda_{vp}$ as the VP arrival rate. However in several systems, the value of $P_i^r$ in the subsystem for a small $i$ value is not very close to the value obtained by simulating the original system, even when this quantity is calculated using the thinned arrival rate $\lambda'_{vp}$. Consider test case 13. The simulated value for $P_0^r$ is 0.378473, but the value for the subsystem using $\lambda'_{vp}$ is $P_0^r = 0.592215$.

We see that in the test cases in which the calculated value of $P_i^r$ is much too large for small $i$ values, as the value of $i$ increases, the original and subsystem $P_i^r$ values become closer. Again consider test case 13. The simulated value of $P_{14}^r$ is 0.004017 while the subsystem value using $\lambda'_{vp}$ is 0.00938642. These values are much closer than the values for $P_0^r$ in the same test case.

As seen in Table B.8, when using the presented estimation method, the simulation value of $P_i^r$ and the calculated value of $P_i^r$ used in the approximation calculation often differ greatly. We hypothesize that a better overall approximation of the loss probability is possible if a more accurate value for $\sum_{i=0}^{N-1} \lambda_i P(loss_i)'$ than the one used in our previous calculations is used. Specifically, we need a better method for calculating $P_i^r = P(loss_i)$.

### 3.3.9.2 Alternative Calculation Methods for $P_i^r$

In order to improve our approximation results, we attempt to find a more accurate method of calculating $P_i^r = P(loss_i)$ using a more accurate model of the system that we are solving. We consider systems with different values of $\lambda_{vp}$ and obtain $P_i^r$ by solving the Markov chains for these systems.

The first model of the system uses the theoretical values obtained using a system that thins the VP arrival rates through the entire sequence of subsystems.

The second model starts with the thinned rates. After the results for the last subsystem are obtained, the VP call arrival rate is set to the last value obtained for $\lambda_{vp_i}'$. Then the theoretical value for $P_i^r$ for each of the subsystems is re-calculated by solving the corresponding Markov chain using the new $\lambda_{vp}$ value as the VP call arrival rate in each subsystem.

The third model uses only the value obtained for $\lambda_{vp_i}'$ in the middle subsystem. We define the middle subsystem as the subsystem with index $\lceil (N-1)/2 \rceil$ where $(N+1)$ is the number of nodes in the system. Then the Markov chain for each of the subsystems is re-solved using this middle value of $\lambda_{vp_i}'$ as the VP call arrival rate. From the solutions obtained, the values of $P_i^r$ are re-calculated.

The final model uses the uses the average of all the theoretical $\lambda_{vp_i}'$ values obtained for

the entire sequence of subsystems as the VP call arrival rate. Then using this new VP call

arrival rate, the Markov chains for all of the subsystems are re-solved and the values of $P_i^r$

are re-calculated.

Appendix B Table B.9 (Section B.2.6.2) shows the resulting values of $\sum_{i=0}^{N-1} \lambda_i P(loss_i)'$

obtained by these four solution methods. Confidence intervals for the simulation values are

not presented because the confidence intervals for the systems presented previously were

very small and can be ignored.

The difference between the result obtained by each solution method and the value ob-

tained by simulating the original system is shown in Table 3.3. The underlined value is the

value found by a solution method that is closest to the simulation value for each test case.

The sum of the absolute values of the difference for each solution method is calculated and

displayed in the last row of the table. This sum represents the overall performance of the

solution method.

| test case | thin difference | final difference | middle difference | average difference |
|---|---|---|---|---|
| 1 | -0.000734 | <u>-0.000722</u> | -0.000885 | -0.000804 |
| 2 | -0.000744 | <u>-0.000642</u> | -0.000845 | -0.000743 |
| 3 | -0.000879 | <u>-0.000815</u> | -0.000900 | -0.000895 |
| 4 | -0.002666 | 0.009655 | -0.002610 | <u>-0.000283</u> |
| 5 | -0.022708 | <u>-0.008648</u> | -0.036789 | -0.022284 |
| 6 | -0.282710 | 0.216419 | <u>-0.148157</u> | -0.228450 |
| 7 | -0.528296 | <u>-0.196630</u> | -0.427808 | -0.510648 |
| 8 | <u>-0.011278</u> | 0.997223 | 0.449678 | -0.288625 |
| 9 | -0.385829 | <u>-0.110133</u> | -0.227111 | -0.338594 |
| 10 | -0.799889 | <u>0.140926</u> | -0.561001 | -0.958655 |
| 11 | -0.292439 | 0.293599 | <u>-0.253556</u> | -0.396022 |
| 12 | -0.603035 | 1.148769 | 0.367545 | <u>-0.346324</u> |
| 13 | -1.205071 | <u>0.201470</u> | -0.248909 | -0.820805 |
| SUM | 4.136278 | 3.325651 | <u>2.725794</u> | 3.913132 |

**Table 3.3**: Difference results for values in Table B.9 for the alternative $P_i^r$ calculation methods.

The table shows that using the final value of $\lambda_{vp_i}'$ as the VP call arrival rate for all of the subsystems gave the solution closest to the solution found by the simulation of the original system in the most test cases (8 of 13 test cases). However, the method of using the middle value of $\lambda_{vp_i}'$ gave the best overall performance based on the sum of the differences for the test cases considered.

### 3.3.9.3 Effect of $P_i^r$ on the Approximation Solution

Both the method of using the final value of $\lambda_{vp_i}'$ as the VP call arrival rate and the method of using the middle value of $\lambda_{vp_i}'$ as the VP call arrival rate produced values of $\sum \lambda_i P(loss_i)'$ that were closer to the corresponding value produced by simulation of the original system than the value produced by the method of thinning all of the VP arrival rates. We now use the improved values of $\sum \lambda_i P(loss_i)'$ in calculating the overall loss probability as found by Approximation 11. We consider only the results from Approximation 11 because it was the most accurate approximation method of those considered in Section 3.3.7. The results are shown in Appendix B Section B.2.6.3 Table B.10.

The difference between the loss probability value obtained by simulating the original system and the value obtained using each approximation of $\sum \lambda_i P(loss_i)'$ is shown in Table 3.4. The underlined value is the value found by a solution method that is closest to the simulation value for each test case. The sum of the absolute values of the differences for each solution method is calculated and displayed in the last row of the table. The sum represents the overall performance of the solution method.

The method of using the final value of $\lambda_{vp_i}'$ as the arrival rate for the VP calls in the subsystem models and the method of using the middle value of $\lambda_{vp_i}'$ as the arrival rate

| test case | difference thinned | difference final | difference middle |
|---|---|---|---|
| 1 | 0.000857 | 0.000862 | 0.000807 |
| 2 | 0.001185 | 0.001236 | 0.001134 |
| 3 | 0.000290 | 0.000315 | 0.000282 |
| 4 | -0.000173 | 0.001390 | -0.000166 |
| 5 | -0.005885 | -0.003742 | -0.008032 |
| 6 | -0.030545 | 0.016065 | -0.017980 |
| 7 | -0.026461 | -0.011021 | -0.021783 |
| 8 | -0.002038 | 0.079205 | 0.035096 |
| 9 | -0.026366 | -0.007713 | -0.015627 |
| 10 | -0.039970 | 0.009320 | -0.027454 |
| 11 | -0.020197 | 0.020336 | -0.017508 |
| 12 | 0.010241 | 0.103564 | 0.061946 |
| 13 | -0.024838 | 0.030578 | 0.012834 |
| SUM | 0.189046 | 0.285347 | 0.220649 |

Table 3.4: Differences between simulated P(loss) values and those found using the alternative methods for calculating $P_i^r$.

for the VP calls in the subsystem models improve the accuracy of the final loss probability result in several cases when compared to the loss probability result obtained by thinning the VP call arrival rates through the subsystems. However the method of thinning the VP call arrival rates for the subsystems performs better when you consider the overall performance measure.

The most accurate method of approximation for a given test case depends on the parameter values for that test case. The method of thinning the VP call arrival rates works best for test cases that have large local call arrival rates in the middle of the system as in test cases 8, and 12. The method of using the final $\lambda_{vp_i}'$ value as the VP call arrival rate works best for test cases in which all or most of the local call arrival rates are high. The method of using the middle $\lambda_{vp_i}'$ value as the VP call arrival rate works best for test cases in which the local call arrival rates toward the end of the system are large. Because the actual

distribution of arrival rates may not be known throughout a system, we conclude that it is best to use the method of thinning the VP call arrival rates throughout the subsystems to obtain the state probability results. This method of thinning the arrival rates gives the most accurate overall performance in our test cases.

### 3.3.10  Summary

We have developed a method of theoretically calculating the performance measures for a given network based on dividing the system into a sequence of subsystems that can be rapidly solved and then combined. We considered several methods of approximating the overall loss probability in a network of which the best was Approximation 11. Approximation 11 produced an approximate value for the probability of a loss that was within +/- 0.01 of the simulated value on average and within +/- 0.04 in all test cases.

## 3.4  Optimal VP Capacity Assignment in a Simple Network

We have developed an accurate approximation method for the probability of a loss in a line network. We now demonstrate that this approximate calculation can be used to determine the optimal capacity assignment for a single VP in a simple network. Later we will show that the performance measure approximation method can be used to determine the optimal location and capacity assignment for VPs in more general networks.

### 3.4.1  Problem Formulation

We consider the following problem. Given a graph $G = (V, E)$ of $N + 1$ nodes where node $i$ is connected to node $i + 1$ $\forall i, 0 \leq i \leq (N - 1)$ as shown in Figure 3.2. A single VP is

established between node 0 and node $N$. At each node there are $K$ servers, $V$ of which are VP servers that are reserved for use by calls from node 0 to node $N$, and $SH$ of which are shared servers that can be used to handle local traffic as well as calls from node 0 to node $N$ not carried by the VP servers. We wish to determine the optimal number of servers to be assigned to this VP.

Two types of calls arrive in the network. Calls arrive at node 0 destined for node $N$ with arrival rate $\lambda_{vp}$ and can use either a VP server if available or else a shared server. In addition, local calls arrive at each node with rate $\lambda_i$ and can only use shared servers.

Recall that we want to determine the capacity function $c^* = (V, SH)$ such that the cost function

$$F = \alpha P(loss) + (1 - \alpha) P(setup)$$

is minimized.

### 3.4.2 Solution Method

With $K$ servers at each node in the network, the optimal number of VP servers, $V$, will be in the range $[0, K]$. The optimal value $V$ can be determined by exhaustive search of the solution space. To determine the optimal number of VP servers $V$, we use our Markov Chain approximation method to calculate the probabilities of interest.

$$P(no\_setup) = \frac{(1 - P_0^v)\lambda_{vp}}{(\sum_{i=0}^{N-1} \lambda_i) + \lambda_{vp}}$$

We can then solve for the $P(setup)' = 1 - P(loss) - P(no\_setup)$ since each call must either be handled by a VP server (not incur a setup), handled by a shared server (incur a setup) or lost by the network.

Note that $P(setup)'$ will not necessarily be 1 when $V = 0$. When $V = 0$ none of the network servers are assigned to the VP. Thus all calls accepted in the network incur a setup cost. However not all of the calls are accepted by the network. Thus using the formulas above, the $P(setup)'$ need not equal 1 in this case. Instead we will define

$$P(setup) = \frac{P(setup)'}{P(setup)' + P(no\_setup)}$$

In the case where $V = K$, the method described above is not used to calculate $P(setup)$. When $V = K$ the $P(setup) = 0$ for all test cases because all servers are assigned to the VP. Thus no accepted call can incur a setup cost. After calculating the P(loss) and P(setup) for a given test case with a given capacity distribution, we will solve for $F = \alpha P(loss) + (1 - \alpha)P(setup)$, and find the value of $V$ that minimizes $F$.

### 3.4.3 Test Cases

We will consider the 13 test cases used in the previous sections. The parameter values are given in Section B.2.1 Appendix B Table B.3.

### 3.4.4 Results

Table B.11 in Appendix B (Section B.3.1.1) shows the resulting $P(loss)$ and $P(setup)$ values for each test case for each capacity distribution. The probabilities were calculated using the theoretical methods described above as well as by simulation. Both values are listed in the table for comparison. The simulation values were obtained by running the simulation for 1,000,000 calls. The confidence intervals for the simulations were obtained based on batches

of 1,000 data points and represent a 95% confidence that the true mean lies in the indicated interval.

The simulated and approximated probabilities were then used to calculate the value of $F$ for each test case with each possible capacity distribution. For each capacity distribution, $F$ was calculated using a range of $\alpha$ values, $\alpha \in 0.1, 0.2, 0.3, \ldots, 0.9$. The resulting $F$ values are shown in Appendix B Section B.3.1.2 Table B.12. The optimal capacity allocations as determined by the theoretical calculations and the simulation are underlined for each test case and $\alpha$ value. Table 3.5 shows the optimal VP capacity allocation as determined by the theoretical calculation and the simulation for each test case and $\alpha$ value.

| Test Case 1 | | | Test Case 2 | | | Test Case 3 | | |
|---|---|---|---|---|---|---|---|---|
| $\alpha$ | Theor-etical | Simula-tion | $\alpha$ | Theor-etical | Simula-tion | $\alpha$ | Theor-etical | Simul-tion |
| 0.1 | 5 | 5 | 0.1 | 5 | 5 | 0.1 | 5 | 5 |
| 0.2 | 5 | 5 | 0.2 | 5 | 5 | 0.2 | 5 | 5 |
| 0.3 | 5 | 5 | 0.3 | 5 | 5 | 0.3 | 5 | 5 |
| 0.4 | 5 | 5 | 0.4 | 5 | 5 | 0.4 | 5 | 5 |
| 0.5 | 5 | 5 | 0.5 | 5 | 5 | 0.5 | 5 | 5 |
| 0.6 | 1 | 1 | 0.6 | 2 | 2 | 0.6 | 2 | 2 |
| 0.7 | 1 | 1 | 0.7 | 2 | 2 | 0.7 | 2 | 2 |
| 0.8 | 1 | 1 | 0.8 | 2 | 2 | 0.8 | 1 | 1 |
| 0.9 | 0 | 0 | 0.9 | 1 | 1 | 0.9 | 1 | 1 |

| Test Case 4 | | | Test Case 5 | | | Test Case 6 | | |
|---|---|---|---|---|---|---|---|---|
| $\alpha$ | Theor-etical | Simula-tion | $\alpha$ | Theor-etical | Simula-tion | $\alpha$ | Theor-etical | Simul-tion |
| 0.1 | 6 | 6 | 0.1 | 3 | 3 | 0.1 | 5 | 5 |
| 0.2 | 6 | 6 | 0.2 | 3 | 3 | 0.2 | 5 | 5 |
| 0.3 | 6 | 6 | 0.3 | 3 | 3 | 0.3 | 5 | 5 |
| 0.4 | 6 | 6 | 0.4 | 3 | 3 | 0.4 | 5 | 5 |
| 0.5 | 6 | 6 | 0.5 | 3 | 3 | 0.5 | 5 | 5 |
| 0.6 | 0 | 0 | 0.6 | 3 | 3 | 0.6 | 5 | 5 |
| 0.7 | 0 | 0 | 0.7 | 0 | 0 | 0.7 | 2 | 2 |
| 0.8 | 0 | 0 | 0.8 | 0 | 0 | 0.8 | 1 | 1 |
| 0.9 | 0 | 0 | 0.9 | 0 | 0 | 0.9 | 0 | 0 |
| Test Case 7 | | | Test Case 8 | | | Test Case 9 | | |
| $\alpha$ | Theor-etical | Simula-tion | $\alpha$ | Theor-etical | Simula-tion | $\alpha$ | Theor-etical | Simul-tion |
| 0.1 | 5 | 5 | 0.1 | 5 | 5 | 0.1 | 5 | 5 |
| 0.2 | 5 | 5 | 0.2 | 5 | 5 | 0.2 | 5 | 5 |
| 0.3 | 5 | 5 | 0.3 | 5 | 5 | 0.3 | 5 | 5 |
| 0.4 | 5 | 5 | 0.4 | 5 | 5 | 0.4 | 5 | 5 |
| 0.5 | 5 | 5 | 0.5 | 5 | 5 | 0.5 | 5 | 5 |
| 0.6 | 5 | 5 | 0.6 | 5 | 5 | 0.6 | 5 | 1 |
| 0.7 | 0 | 0 | 0.7 | 5 | 5 | 0.7 | 0 | 0 |
| 0.8 | 0 | 0 | 0.8 | 5 | 4 | 0.8 | 0 | 0 |
| 0.9 | 0 | 0 | 0.9 | 2 | 2 | 0.9 | 0 | 0 |
| Test Case 10 | | | Test Case 11 | | | Test Case 12 | | |
| $\alpha$ | Theor-etical | Simula-tion | $\alpha$ | Theor-etical | Simula-tion | $\alpha$ | Theor-etical | Simul-tion |
| 0.1 | 5 | 5 | 0.1 | 5 | 5 | 0.1 | 5 | 5 |
| 0.2 | 5 | 5 | 0.2 | 5 | 5 | 0.2 | 5 | 5 |
| 0.3 | 5 | 5 | 0.3 | 5 | 5 | 0.3 | 5 | 5 |
| 0.4 | 5 | 5 | 0.4 | 5 | 5 | 0.4 | 5 | 5 |
| 0.5 | 5 | 5 | 0.5 | 5 | 5 | 0.5 | 5 | 5 |
| 0.6 | 5 | 5 | 0.6 | 5 | 5 | 0.6 | 5 | 5 |
| 0.7 | 5 | 5 | 0.7 | 1 | 1 | 0.7 | 5 | 5 |
| 0.8 | 5 | 5 | 0.8 | 0 | 0 | 0.8 | 1 | 2 |
| 0.9 | 0 | 0 | 0.9 | 0 | 0 | 0.9 | 1 | 1 |

| Test Case 13 | | |
|---|---|---|
| $\alpha$ | Theoretical | Simulation |
| 0.1 | 5 | 5 |
| 0.2 | 5 | 5 |
| 0.3 | 5 | 5 |
| 0.4 | 5 | 5 |
| 0.5 | 5 | 5 |
| 0.6 | 5 | 5 |
| 0.7 | 0 | 0 |
| 0.8 | 0 | 0 |
| 0.9 | 0 | 0 |

Table 3.5: Optimal VP capacity allocation as determined using the theoretical method and through simulation.

In almost all test cases, for all values of $V$ and $\alpha$, the optimal capacity distribution found through simulation and the optimal capacity distribution found using the theoretical calculations are identical. The instances where the optimal capacity distributions found by the two solution methods are not identical are Test Case 8 with $\alpha = 0.8$, Test Case 9 with $\alpha = 0.6$ and Test Case 12 with $\alpha = 0.8$. In these three test cases, we see that if the optimal capacity distribution were chosen by the theoretical calculation, the optimality of the solution is affected only slightly. For example in Test Case 8 with $\alpha = 0.8$, the theoretically calculated optimal capacity distribution is $V = 5$ with $F = 0.583443$. The simulated $F$ for this test case, $\alpha$, and $V = 5$ is $0.583422$, which differs only slightly from the optimal value chosen by the simulation results, $V = 4$, $F = 0.580552$.

### 3.4.5 Summary

We have shown that our performance measure approximation method that calculates the $P(loss)$ can be extended to calculate our other performance measure $P(setup)$. We then show that the optimal solution to the capacity function $c^* = (V, SH)$ can be determined

using our performance measure approximations. The solutions obtained using the performance measure approximation method were compared to the solutions obtained through simulation for several test cases. In all cases, the solutions obtained using the performance measure approximation method were equivalent to those obtained through simulation. Thus we conclude that our approximation method is accurate and can be used in an algorithm to determine the optimal capacity distribution for a given network.

## 3.5   Optimal VPDBA Solution for a Simple Network

We now show that our performance measure approximation method can be expanded to apply to a more general class of line networks.

### 3.5.1   Problem Formulation
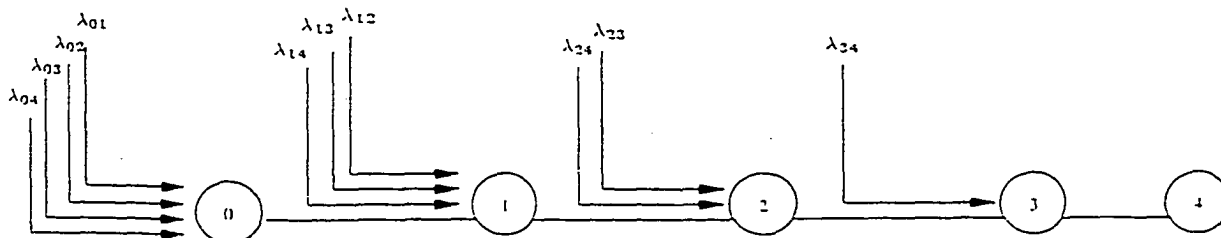


Figure 3.7: A 5 node network.

We consider the following problem. A simple network of $(N+1)$ nodes exists in which each node $i$ is connected to node $i+1$ $\forall i$, $0 \leq i \leq (N-1)$. An example of such a network with 5 nodes is pictured in Figure 3.7. Calls arrive at each node in the network. A stream of traffic exists between each node pair in the network $i, j$ where $0 \leq i \leq (N-1)$, $(i+1) \leq j \leq N$. We wish to find the optimal placement and capacity allocation for a

single VP in this network. Thus depending on the placement of the VP, the stream that is designated to use the VP will change.

At each node there are $K$ servers. Along all nodes traversed by the VP, $V$ of these servers are reserved for use for VP calls. All other servers are shared severs and can be used to handle non-VP as well as VP calls.

## 3.5.2   Expanded Performance Measure Approximation Method

Our performance measure approximation method must be modified to apply to this system because in the previous case our system always contained a single stream of VP traffic and a single stream of non-VP traffic at each node, whereas the current system there may exist several streams of non-VP traffic and one stream or no stream of VP traffic.

As in the previous approximation, we will divide the system into a series of single node systems, solve the Markov chains for each, and combine the results to approximate the loss probability for the original system.

We will use the following ideas in the calculation of the loss probability:

- We let

    $P_i^r$ = the probability that the shared servers are all busy at node i

    $P_i^v$ = the probability that the VP servers are all busy at node i

- In the previous approximation, we thinned the rate of the VP calls at each node to account for losses incurred at previous nodes in the network. In our expanded approximation, we will thin the rate of all traffic streams that traverse more nodes than the source node and destination node. So in a 5 node network, we would thin

$\lambda_{02}$, $\lambda_{03}$, $\lambda_{04}$, $\lambda_{13}$, $\lambda_{14}$, and $\lambda_{24}$. We let $\lambda_{ij}^a$ be the thinned arrival rate of calls from source node $i$ to destination node $j$ at node $a$ on the route from $i$ to $j$. In general, we thin the stream $\lambda_{ij}$ as follows.

if $(j = (i+1))$ or if $(i = 0)$ then $(\lambda_{ij}^i)' = \lambda_{ij}$

else $(\lambda_{ij}^a)' = (1 - (P_{a-1}^v)')(\lambda_{ij}^{a-1})'$

Otherwise

if $(j = (i+1))$ or if $(i = 0)$ then $(\lambda_{ij}^i)' = \lambda_{ij}$

else $(\lambda_{ij}^a)' = (1 - (P_{a-1}^r)')(\lambda_{ij}^{a-1})'$

- When a stream is thinned, we use the weighted average of the $\lambda$ values for the stream in the loss probability formula.

$$avg(\lambda_{ij}) = \frac{\sum_{a=i}^{(j-1)}((\lambda_{ij}^a)')^2}{\sum_{a=i}^{(j-1)}(\lambda_{ij}^a)'}$$

- We calculate the probability of a loss for each non-VP stream using a method similar to that used to calculate the VP loss rate in the previous approximation method.

We know that a call arriving at node $i$ with a destination of node $j$ that is not a VP call will be rejected by the shared servers when there is not a shared server available at each node. Since $P_a^r$ is the probability that the shared servers are all busy at node $a$, $(1 - P_a^r)$ is the probability that there is a shared server available at node $a$. Then $P^{R\ ij} = (\prod_{a=i}^{j-1}(1 - P_a^r))$ is the probability that a shared server is not available at all nodes on the route between source node $i$ and destination node $j$. Note that when $j = i + 1$, then $P^{R\ ij} = P_i^r$.

- We calculate the loss probability using a method similar to that used above to calculate the probability of a loss from a non-VP stream.

We know that a VP call is rejected when there is no VP server available for its use on the VP and there is not a shared server available at each node along the VP.

We let

$$P^{R\ ij} = \text{the probability that all shared servers are busy on path } ij.$$

Then,

$$P_{vp} = \text{the probability that a VP call is rejected} = P^v_{vp\_begin} P^{R\ ij}$$

Then using these loss probabilities for the individual streams, we can calculate the overall probability of a loss in the network.

$$P(loss) = \frac{\sum_{\forall i, \forall j, ij \neq vp} avg(\lambda_{ij}) P^{Rij} + avg(\lambda_{vp}) P_{vp}}{\sum_{i=0}^{(N-1)} \sum_{j=(i+1)}^{N} avg(\lambda_{ij})}$$

From the state probabilities and input values, we can calculate the $P(no\_setup)$. In the new network model, this value cannot be calculated exactly, because in this case $P^v_{vp\_begin}$ represents the probability that the VP servers are all busy at the source node of the VP. Therefore, this quantity will be affected by the behavior of the nodes that precede the VP source node in the network.

$$P(no\_setup) = \frac{(1 - P^v_{vp\_begin})\lambda_{vp}}{\sum_{\forall i, \forall j, ij \neq vp} \lambda_{ij} + \lambda_{vp}}$$

As in the previous approximation method, we observe that all call are either lost, serviced by a VP server or serviced by a shared server. Thus the probability of being lost and handled

by a server must sum to 1.

$$P(loss) + P(no\_setup) + P(setup)' = 1$$

To calculate the $P(setup)$ for the system, we want to determine the probability that a call that is accepted by the network uses a shared server. To calculate this $P(setup)$ value, we use the following formula.

$$P(setup) = \frac{P(setup)'}{P(no\_setup) + P(setup)'}$$

We can then calculate the value of the objective function $F = \alpha P(loss) + (1-\alpha)P(setup)$. By using an exhaustive search method that examines all possible VP placements and capacity allocations, we are guaranteed to find the position and capacity assignment that minimizes this objective function.

### 3.5.3   Test Cases

We use the 10 test cases listed in Table B.13 in Appendix B (Section B.4.1) to evaluate the performance of our expanded performance measure estimation method.

### 3.5.4   Results

An exhaustive search of all possible VP placement and capacity allocations was performed for each test case, using both our performance measure estimation method and simulation. Simulations were each run for 1,000,000 calls. Confidence intervals for all values for $P(loss)$ and $P(setup)$ were calculated based on batches of 1,000 data points. All confidence intervals

were found to be very small. The results from the simulation were considered to be equally optimal if the difference in their objective function values was smaller than the largest of the confidence intervals (approximately 0.00001). For the results listed in Table 3.6, the optimal solution was defined to be the solution that minimized the objective function $F = 0.5P(loss) + 0.5P(setup)$.

| Test Case | Approximation | Simulation |
|---|---|---|
| 1 | VP: 0-3, 5 servers | VP: 0-3, 5 servers |
| 2 | VP: 3-4, 2 servers | VP: 3-4, 2 servers |
| 3 | VP: 0-4, 5 servers | VP: 0-4, 5 servers |
| 4 | VP: 0-2, 3 servers | VP: 0-2, 3 servers |
| 5 | VP: 1-4, 2 servers | VP: 1-4, 2 servers |
| 6 | VP: 0-2, 2 servers | VP: 2-4, 2 servers or VP: 0-2, 2 servers |
| 7 | VP: 2-3, 3 servers | VP: 2-3, 3 servers |
| 8 | VP: 0-1, 3 servers | VP: 0-1, 3 servers |
| 9 | VP: 0-4, 3 servers | VP: 0-4, 3 servers |
| 10 | VP: 3-7, 3 servers | VP: 3-7, 3 servers |

**Table 3.6**: Comparison of approximation method results and simulation results for the test cases in Table B.13.

In all test cases, both the optimal placement *and* capacity distribution found through simulation and the optimal placement and capacity distribution found using the theoretical approximation method are identical. In Test Case 6, the simulation found two optimal solutions (within 0.00001 of each other) however the theoretical method only found one of these solutions to be optimal. Clearly selecting the one solution found using the theoretical method would not adversely affect the network performance because it was found to be optimal through simulation as well. We conclude that the generalized performance measure approximation method is accurate. We have shown that the approximation method can be used in an algorithm to calculate performance measures that can then be used to find the

optimal placement and capacity allocation for a single VP.

### 3.5.5 Summary

We have extended the performance measure approximation method presented in Section 3.4 to apply to a more general class of line networks, in which a traffic stream exists between each node pair in the network. We then showed that the optimal placement and capacity allocation for a single VP can be accurately determined using our performance measure approximation method.

## 3.6 Summary of Base Case

We have considered several concepts that are key to solving the VPDBA problem. To avoid the complexity of solving a large Markov chain equivalent to the network system, we developed an efficient method for estimating the probability of a loss in a network with a given layout and traffic parameters. We propose a method of decomposing the original system into a sequence of subsystems, solving the subsystems, and then combining the results to approximate the results of the original system. Because the subsystems are not independent, we considered several approximations for approximating the overall loss probability in a network. We then showed that the performance measure approximation method can be used to determine the optimal capacity allocation for a simple line network, and extended our performance measure approximation method to apply to a more general class of networks. We have shown that the solutions obtained using this extended performance measure approximation method are equivalent to those obtained through simulation for all test cases considered. We conclude that our extended approximation method is accurate

and can be used in an algorithm to determine the optimal placement and capacity allocation

for a single VP in a given network.

# Chapter 4

# Single Node Network Base Case

Before generating our model for the line network, we first consider a simple single node model containing multiple VPs. We show that an expanded version of our original approximation method (for a single node) provides accurate results for this new model.

## 4.1 Problem Formulation

We consider the following problem. A single node experiences arrivals from multiple VP streams as well as multiple non–VP streams. In general there are $i$ VP streams and $j$ non–VP streams that traverse the node. The node has a number of servers $K$ associated with it. $V_a$ of these servers are associated with the $a$-th VP passing through the node, $0 \leq a \leq i$. All other servers are shared and can be used by the non-VP traffic streams as well as by VP calls that cannot be handled by the VP servers due to insufficient bandwidth.

For a node with $i$ VP arrival streams, the Poisson arrival rate at each stream will be denoted $\lambda_{vp_a}$ $\forall a$, $1 \leq a \leq i$. Similarly the node will have associated with it $j$ non-VP arrival streams each with Poisson arrival rate $\lambda_b$ $\forall b$, $1 \leq b \leq j$. The average holding time of all calls, as usual, is denoted by $\mu$. This single node model is depicted in Figure 4.1.

85
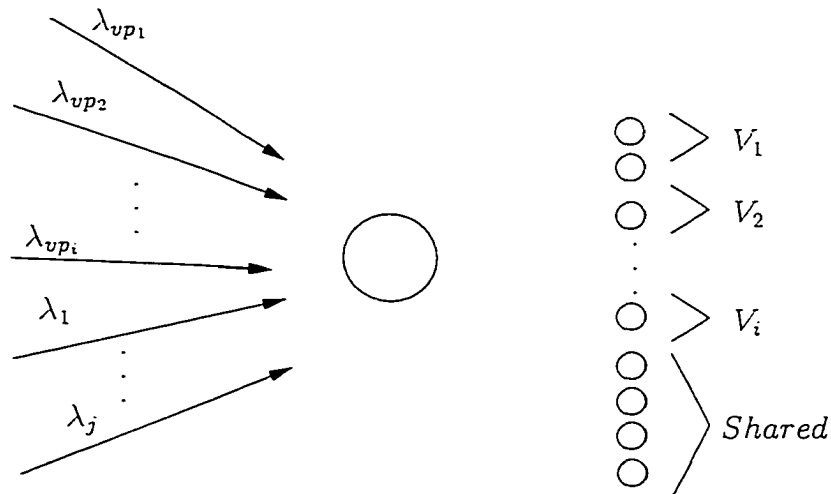
**Figure 4.1**: The single node model.

## 4.2 Expanded Approximation Method

Our previous solution methods assumed that only a single VP existed in the network. In this single VP model, we could generate the statistics about the system by keeping track of the number of VP servers in use at each node and the number of shared servers in use at each node. Thus the Markov chain modeling the server usage at each node was simple and two dimensional.

However, as we expand our model to consider systems with multiple VPs, we notice that directly applying our previous approximation method complicates the Markov chain substantially. For a node with $i$ VPs traversing a node, we need to keep track of the number of VP servers of each type $a$, $1 \leq a \leq i$. We also need to keep track of the number of shared servers in use at this node. Thus for a node with $i$ VP streams and $j$ non-VP streams the exact Markov chain model for this node will have $i + 1$ dimensions. For $i > 2$ it becomes very difficult to construct and work with a Markov chain of such dimension. Thus we need

to consider an extension of our earlier approximation method that does not involve such Markov chain complexity.

We use the following steps to approximate the overall performance measures $P(loss)$ and $P(setup)$ for the single node system.

1. Consider all $j$ of the non-VP arrival streams as a single arrival stream having an overall arrival rate of $\lambda = \sum_{b=1}^{j} \lambda_b$. Grouping these arrivals as a single stream has no impact on the model since all $j$ of these non-VP arrival streams only have access to the shared servers.

2. Break the model of the single node with $i$ VPs into $i$ systems each with one VP arrival stream and one non-VP arrival stream. Thus each single VP system can be modeled using a two dimensional Markov chain that can be easily solved. Solve each single VP system with VP arrival stream $\lambda_{vp_a}$ for $P^{v_a}$, the probability that the $VP_a$ servers are all busy.

3. For each single VP system, calculate a new non-VP arrival rate based on the probability that VP calls are lost at the other nodes. In other words, for the system containing $VP_a$, calculate the "true" non-VP arrival rate $s_a = \lambda + \sum_{\forall c, c \neq a} \lambda_{vp_c} P^{v_c}$. This step is necessary because if we ignore the impact of the overflow calls from all of the VPs, we will be significantly under–estimating the arrival rate to the shared servers. This under-estimation could have a negative impact on the calculation of the system performance measures.

4. Re–solve each single VP system using $\lambda_{vp_a}$ as the arrival rate for the VP calls and $s_a$ as the arrival rate for the non-VP calls. Again, this results in an easily–solved two

dimensional Markov chain. Solve the system for:

$P^{r_a}$ = the probability that all of the shared servers are busy in system $a$

$P^{r,vp_a}$ = the probability that all shared and $VP_a$ servers are busy in system $a$

$P^{v_a}$ = the probability that all $VP_a$ servers are busy in system $a$

5. Use the probabilities calculated in each system to calculate the approximate performance measures for the original system.

$$P(loss) = \frac{\lambda \frac{\sum_{a=1}^{i} P^{r_a}}{i} + \sum_{a=1}^{i} \lambda_{vp_a} P^{r,vp_a}}{\lambda + \sum_{a=1}^{i} \lambda_{vp_a}}$$

$$P(no\ setup) = \frac{\sum_{a=1}^{i} (1-P^{v_a})\lambda_{vp_a}}{\lambda + \sum_{a=1}^{i} \lambda_{vp_a}}$$

$$P(setup)' = 1 - P(loss) - P(no\ setup)$$

$$P(setup) = \frac{P(setup)'}{P(setup)' + P(no\ setup)}$$

## 4.3 Simulation

A next–event simulation was written to model this single node network. It was tested for validity and shown to be valid. The single node, multiple VP network simulation was run for various networks containing a single node and single VP stream. The resulting performance measures were compared to the performance measures obtained using the multiple node, single VP network simulation for the same network test cases. The results matched in all test cases. In addition, the performance measures obtained by the single node, multiple VP network simulation were compared to those obtained through the equivalent Markov chain solution for networks containing a single VP. The results were very close in all test cases. Thus we conclude that our single node, multiple VP simulation is valid.

## 4.4  Test Cases

Ten test cases were used to compare the resulting performance measures found through the approximation method to those obtained by simulation. Each test case was a single node system containing between 2 and 10 VPs. The specifics of the input parameters for each test case are shown in Appendix C Table C.1.

## 4.5  Results

The results obtained by the performance measure approximation method and simulation are shown in Table 4.1. The confidence intervals reported in the tables were calculated using batches of 1,000 data points and represent a 95% level of confidence that the true mean lies within the indicated range. All simulations were run for 100,000 calls. In all test cases, the results found using the performance measure approximation method and those found through simulation were very close in value. On average, the values for $P(loss)$ had a percent difference of 2.0%. The values for $P(setup)$ had a percent difference of 3.1%. The values for $P(no\ setup)$ differed by 0.6%. We note that even in the test cases resulting in the observed worst case behavior, the difference between the approximated performance measure and the simulated performance measure was approximately 0.05, which is a relatively small difference. In addition, the test cases exhibiting the worst behavior were the test cases with parameters representing a heavy traffic load that is unlikely to occur in practice.

| p(loss) | | | |
|---|---|---|---|
| test case | approximation | simulation | percent difference |
| 1 | 0.5947505421 | 0.590759 +/- 0.000125 | 0.671129 |
| 2 | 0.6759527727 | 0.625322 +/- 0.000124 | 7.490280 |
| 3 | 0.4265988939 | 0.421527 +/- 0.000117 | 1.188910 |
| 4 | 0.5431407826 | 0.538770 +/- 0.000123 | 0.804724 |
| 5 | 0.5006049809 | 0.494078 +/- 0.000130 | 1.303820 |
| 6 | 0.4798931500 | 0.477005 +/- 0.000130 | 0.601832 |
| 7 | 0.4995906333 | 0.496564 +/- 0.000140 | 0.605823 |
| 8 | 0.5885366111 | 0.553998 +/- 0.000131 | 5.868560 |
| 9 | 0.6469980714 | 0.644736 +/- 0.000106 | 0.349626 |
| 10 | 0.5241362710 | 0.519302 +/- 0.000120 | 0.922331 |
| average percent difference | | | 1.980704 |

| p(setup) | | | |
|---|---|---|---|
| test case | approximation | simulation | percent difference |
| 1 | 0.2120798316 | 0.219488 +/- 0.000110 | 3.375200 |
| 2 | 0.5885369684 | 0.644059 +/- 0.000127 | 8.620640 |
| 3 | 0.1587503518 | 0.166129 +/- 0.000080 | 4.441520 |
| 4 | 0.3222564558 | 0.328427 +/- 0.000116 | 1.878820 |
| 5 | 0.3644481323 | 0.372348 +/- 0.000110 | 2.121640 |
| 6 | 0.5888652966 | 0.591145 +/- 0.000103 | 0.385642 |
| 7 | 0.6498103266 | 0.651764 +/- 0.000109 | 0.299752 |
| 8 | 0.5218059769 | 0.531412 +/- 0.000131 | 1.807640 |
| 9 | 0.2816716704 | 0.286201 +/- 0.000134 | 1.582570 |
| 10 | 0.1257380549 | 0.134445 +/- 0.000089 | 6.476210 |
| average percent difference | | | 3.098963 |

| p(no setup) | | | |
|---|---|---|---|
| test case | approximation | simulation | percent difference |
| 1 | 0.3193042210 | 0.319417 +/- 0.000105 | 0.0353078 |
| 2 | 0.1333334545 | 0.133363 +/- 0.000063 | 0.0221542 |
| 3 | 0.4823734788 | 0.482372 +/- 0.000101 | 0.0003066 |
| 4 | 0.3096333852 | 0.309749 +/- 0.000092 | 0.0373253 |
| 5 | 0.3173914371 | 0.317543 +/- 0.000093 | 0.0477299 |
| 6 | 0.2138339755 | 0.213828 +/- 0.000069 | 0.0027945 |
| 7 | 0.1752381927 | 0.175314 +/- 0.000068 | 0.0432409 |
| 8 | 0.1967593333 | 0.208991 +/- 0.000083 | 5.8527200 |
| 9 | 0.2535712857 | 0.253587 +/- 0.000081 | 0.0061968 |
| 10 | 0.4160295493 | 0.416070 +/- 0.000106 | 0.0097221 |
| average percent difference | | | 0.605750 |

Table 4.1: The simulated and approximated results for the single node, multiple VP test cases.

## 4.6 Summary

We have expanded the previous approximation method to apply to single node systems containing multiple VPs. To simplify the calculations necessary for solution, we decompose the system containing $i$ VPs into $i$ subsystems each containing a single VP arrival stream. To improve the accuracy of our solution method, we calculate the probability of VP overflow from each stream, and then use these probabilities to estimate the actual arrival rate to the shared servers in each subsystem. We have compared the performance measures obtained by our solution method to those obtained through simulation. Our approximation method produces results very close to those obtained through simulation. We conclude that our approximation method is valid and effective for calculating performance measures in a single node, multiple VP network.

# Chapter 5

# Approximation Method for

# General Line Networks

We now consider a line network containing multiple nodes and multiple VPs. In order to quickly calculate performance measures for our generalized network model, we extend our previous performance measure approximation methods. Our expanded approximation method combines the method that we used to calculate performance measures in the line network containing multiple nodes and a single VP with the method used to calculate performance measures in the single node network containing multiple VP and non-VP arrival streams. We show that our expanded performance measure approximation method is accurate. We provide evidence that our approximation method can be used in an algorithm to determine an effective placement and bandwidth allocation to VPs within a network.

## 5.1 Problem Formulation

We consider a line network with multiple nodes and arrivals from multiple streams. Each node has a number of servers $K$ associated with it. We would like to determine the optimal

92

number of VPs for the network. the optimal placement of VPs within the network. and the optimal number of servers that should be assigned to each VP. As before, servers not assigned to any VP are shared and can be used by both non-VP and VP calls.

We assume that a single traffic stream exists between each source and each destination node. The Poisson arrival rate of a stream from source node $s$ to destination node $d$ will be denoted $\lambda_{sd}$. An example of a this type of network model containing 5 nodes is depicted in Figure 5.1 below.
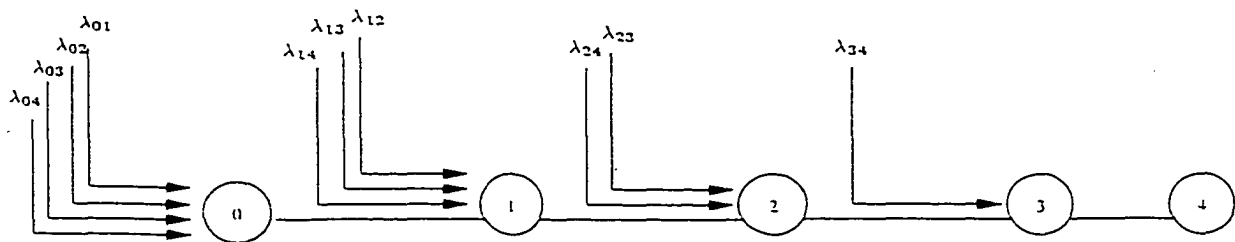


Figure 5.1: A 5 node network.

## 5.2 Expanded Approximation Method

Consider a network of $N$ nodes in which each node $v$, $0 \leq v < N$, has some number $p_v$ VPs traversing it. Suppose that each node has $K$ servers associated with it. In order to solve for the performance measures in this network, we need to keep track of the number of servers $k_{q_v}$ $0 \leq k_{q_v} \leq K$ assigned to each VP $q_v$ $0 \leq q_v < p_v$ at each node $v$ $0 \leq v < N$. We also need to keep track of the number of shared servers $s_v$ $0 \leq s_v \leq K$ in use at each node. Thus for a node with $p_v$ VP streams and $w_v$ non-VP streams, the exact Markov chain model for this node is impractical because it has $p_v + 1$ dimensions. So as we discussed in

the previous model, all non-VP streams can be considered jointly as a single stream. We now expand the previous solution methods to apply to a general multiple node, multiple VP line network.

Our approximation method uses the following steps to solve for the $P(loss)$ and $P(setup)$ in a multiple node, multiple VP network.

1. Decompose the multi-node system with $N$ nodes into a series of $N$ single node systems. At each node $v$ the actual arrival rates will be approximated by thinning each stream using the $P(loss)$ for that stream at the previous node.

   • The actual arrival rate $\lambda_{s,d_v}^{vp_q}$ for the $q^{th}$ VP arrival stream from node $s$ to node $d$ at node $v$ where $0 \leq q < p_v$ is estimated using the following formula.

$$(\lambda_{s,d_v}^{vp_q})' = (1 - P(loss_{s,d_{v-1}}^{vp_q})')(\lambda_{s,d_{v-1}}^{vp_q})' \ \forall v \geq 1$$

$$(\lambda_{s,d_0}^{vp_q})' = \lambda_{s,d_0}^{vp_q}$$

$$P(loss_{s,d_0}^{vp_q})' = P(loss_{s,d_0}^{vp_q})$$

   where $P(loss_{s,d_v}^{vp_q})$ is the probability that the all of the servers assigned to the $q^{th}$ VP and all of the shared servers are busy at node $v$.

   • The actual arrival rate $\lambda_{s,d_v}^{L}$ for a non-VP arrival stream from node $s$ to node $d$ at node $v$ is estimated using the following formulas.

$$(\lambda_{s,d_v}^{L})' = (1 - (P^{r_{v-1}}))'(\lambda_{s,d_{v-1}}^{L})' \ \forall v \geq 1$$

$$(\lambda_{s,d_0}^{L})' = \lambda_{s,d_0}^{L}$$

$$(P^{r_0})' = P^{r_0}$$

   where $P^{r_v}$ is the probability that all of the shared servers are busy at node $v$.

2. Begin with the lowest numbered non-solved single node system. Solve for the loss

probability of each individual stream in this single node system using the following method.

- Suppose that the node under consideration has $w$ non-VP arrival streams. Consider all $w$ of the non-VP arrival streams as a single arrival stream having an overall arrival rate of $\lambda_v^{L'} = \sum_{b=1}^{w} \lambda_{b_v}^{L}{}'$ where $\lambda_{b_v}^{L}{}'$ is the thinned arrival rate of each non-VP arrival stream as described above. Grouping these arrivals as a single stream has no impact on the model since all $w$ of these non-VP arrival streams only have access to the shared servers.

- Break the model of the single node with $p > 1$ VPs into $p$ systems each with one VP arrival stream and one non-VP arrival stream. Number these systems $0 \ldots (p-1)$. The non-VP arrival stream in each single-VP system will be equal to the non-VP arrival stream in the single node, multiple VP system $\lambda_v^{L}$. Thus each single VP system can be modeled using a two dimensional Markov chain that can be easily solved. Solve each single VP system with VP arrival stream $\lambda_v^{vp_q}{}'$ for $P^{v_q}$ = the probability that the VP servers assigned to VP $q$ $0 \leq q < p_v$ are all busy. This probability will be used to estimate the true non-VP arrival rate for each single VP system.

- For each single VP system, calculate a new non-VP arrival rate based on the probability that VP calls are lost in the other single VP systems at this node. In other words, for the system containing $VP_q$, calculate the "true" non-VP arrival rate

$$s_q = \lambda_v^{L'} + \sum_{\forall c,\ c \neq q} \lambda_v^{vp_c}{}' P^{v_c}$$

where $0 \leq c < p_v$. This step is necessary because if we ignore the impact of the overflow calls from all the other VPs at this node, we will be significantly under-estimating the arrival rate to the shared servers.

- Re–solve each single VP system still using $\lambda_v^{vp_q\,\prime}$ as the arrival rate for the VP calls but now using $s_q$ as the arrival rate for the non-VP calls. Again, this results in an easily solved two dimensional Markov chain. Solve the system for:

$P_v^{r_q}$ = probability that all of the shared servers are busy in system $q$

$P_v^{r,vp_q}$ = probability that all shared and $VP_q$ servers are busy in system $q$

$P_v^{v_q}$ = probability that all $VP_q$ servers are busy in system $q$

- The loss probabilities of each single VP system are then averaged to find the overall loss probability for non-VP arrival streams at the node $v$.

$$P_v^r = \frac{\sum_{\forall q} P_v^{r_q}}{p_v}$$

3. After the system is solved at node $v$, the performance measures at that node are used to estimate the actual call arrival rate at node $(v + 1)$. In this way, the actual arrival rates for each stream at each node are estimated based on the $P(loss)$ at the previous node.

4. After the probabilities have been calculated at each node, we use these results to calculate the approximate performance measures for the original system. First a weighted average of all of the $\lambda$ values for each particular source–destination pair is calculated.

$$\Lambda_{s,d} = \frac{\sum_{i=s}^{d-1} ((\lambda_{s,d}^i)\prime)^2}{\sum_{i=s}^{d-1} (\lambda_{s,d}^i)\prime}$$

$$P(loss) = \frac{\sum_{\forall i,j} \Lambda'_{ij} P(loss_{ij})}{\sum_{\forall i,j} \Lambda_{ij}'}$$

where $P(loss_{ij}) = P^{r_i}$ if $\lambda_{ij}$ is a non-VP stream

$\qquad = P^{r,vp_a}$ if $\lambda_{ij}$ is the $a^{th}$ VP at the node and $i$ is the source node of

the corresponding VP

$$P(no\ setup) = \frac{\sum_{a=1}^{p} (1 - P^{v_a}) \Lambda'_{vp_a}}{\sum_{\forall i,j} \Lambda'_{i,j}}$$

where $(1 - P^{v_a})$ is the probability that all of the servers assigned to the $a^{th}$ VP are

busy at the source node of the VP.

$$P(setup)' = 1 - P(loss) - P(no\ setup)$$

$$P(setup) = \frac{P(setup)'}{P(setup)' + P(no\ setup)}$$

## 5.3 Using the Performance Measure Approximation Method

To show the validity of our approximation method, we compare the performance measures found using simulation to the performance measures found using our approximation method for various network test cases. In addition, we show that our approximation method can be used to find the optimal number and layout of VPs and optimal capacity allocation to VPs within a network. We present several network test cases and the corresponding optimal solutions found through exhaustive search of all simulation solutions as well as through exhaustive search of all solutions obtained through approximation. The details of these experiments are described in this section.

### 5.3.1 Validity of the Simulation

A next event simulation was written for the multiple node, multiple VP network. To verify the validity of our simulation, we considered several bases cases containing a single VP. The performance measures for the network were calculated using our new simulation. These results were compared to results obtained by running the same network using the previously verified simulation for the multiple node, single VP network. The performance measures matched in all test cases. We conclude that the simulation method is valid.

### 5.3.2 Validity of Performance Measure Approximation Method

In order to verify that the expanded approximation method produces results have the same level of accuracy as those obtained through simulation, several test cases were run. In these cases, a specific network layout was given. Then the performance measures for the network were calculated using our approximation method and by running the corresponding simulation. The resulting values of the equation

$$F = \alpha P(loss) + (1 - \alpha)P(setup)$$

were then compared for each network test case.

The test cases are described in Appendix D Table D.1 (Section D.1.1.1). All test cases contain between 5 and 10 nodes, 2 and 7 VPs and each node within each network has 3 to 10 servers associated with it. There are 10 network scenarios considered, each with unique parameters. Within each network scenario, the performance measures for 5 different VP layouts and capacity assignments were calculated. These test cases then form 10 groups with 5 layouts per group for a total of 50 test cases considered. For each group, the network parameters are given. The lambda values are listed in increasing $i$, $j$ or-

der $(\lambda_{01}, \lambda_{02}, \ldots, \lambda_{12}, \lambda_{13}, \ldots)$. The VP layout for each test case is listed as $(SD - V)_0$, $(SD - V)_1, \ldots$ where $S$ is the source node of the VP, $D$ is the destination node of the VP and $V$ is the number of servers assigned to that VP. Any streams not listed in this layout are not assigned VP servers and only transmit information using the shared servers within the network.

The performance measure results are listed in Section D.1.1.2 of Appendix D Tables D.2, D.3 and D.4. All simulation results were obtained by running each test case for 10,000 calls. Each simulation solution listed represents the mean of 500 data points with a 95% level of confidence. We note that all confidence intervals are very small and will be ignored hereafter. The percent difference between the simulated performance measure and approximated performance are listed in Table 5.1.

In all cases, the simulation and approximation results corresponded very closely with one another. The average percent difference observed was 9.5% for loss probabilities, 4.2% for setup probability and 6.7% for the probability that a setup does not occur. Table 5.2 shows the number of results that lie within a given percent difference range for each performance measure calculated. We observe that most of the approximated loss probabilities were less than 20% different from the simulated values. The percent differences for the setup probabilities were much smaller with most of the approximated values falling less than 5% different than the simulated values. The percent differences for the probability of no setup cost being incurred was also small in most cases with the majority of the approximated solutions having less than a 10% difference from the simulated solutions.

We note that in some cases, our approximation method performs much worse than the average case behavior. Particularly the loss probabilities calculated for test cases 10, 36 and

| TC | loss percent difference | setup percent difference | nosetup percent difference | TC | loss percent difference | setup percent difference | nosetup percent difference |
|---|---|---|---|---|---|---|---|
| 1 | 0.906915 | 1.7665 | 1.79302 | 26 | 1.54077 | 0.875373 | 8.86527 |
| 2 | 16.7492 | 8.75514 | 34.6914 | 27 | 3.61394 | 1.26576 | 3.34854 |
| 3 | 19.4097 | 7.47135 | 26.8381 | 28 | 2.59789 | 2.06589 | 0.499491 |
| 4 | 8.078400 | 3.16155 | 1.49655 | 29 | 4.18389 | 4.32667 | 0.397401 |
| 5 | 15.764000 | 47.411 | 5.10816 | 30 | 0.51494 | 1.15537 | 1.91084 |
| 6 | 18.0281 | 1.11418 | 1.93236 | 31 | 10.7074 | 2.14298 | 2.61875 |
| 7 | 19.7043 | 0.756051 | 2.48039 | 32 | 16.141 | 3.96701 | 7.01199 |
| 8 | 18.1140 | 0.684974 | 2.41963 | 33 | 8.20764 | 0.631104 | 0.209282 |
| 9 | 12.1374 | 1.00943 | 5.51082 | 34 | 7.42545 | 1.23672 | 2.71038 |
| 10 | 20.5236 | 1.91486 | 4.62471 | 35 | 16.2539 | 2.97237 | 6.51716 |
| 11 | 11.7854 | 1.3438 | 0.668521 | 36 | 20.6511 | 0.603225 | 0.826 |
| 12 | 11.2095 | 2.78064 | 3.39451 | 37 | 15.5685 | 0.985493 | 1.9724 |
| 13 | 1.40529 | 1.0908 | 1.15046 | 38 | 24.7158 | 1.46641 | 1.38424 |
| 14 | 2.08405 | 8.45171 | 8.24939 | 39 | 13.6341 | 1.86565 | 4.9354 |
| 15 | 16.3655 | 4.42288 | 7.352 | 40 | 17.8135 | 4.81993 | 16.1415 |
| 16 | 3.84745 | 2.46243 | 4.02201 | 41 | 4.77104 | 8.84431 | 19.2363 |
| 17 | 2.04618 | 0.3931 | 0.522315 | 42 | 9.4235 | 17.1961 | 35.4234 |
| 18 | 18.5193 | 2.19027 | 5.60588 | 43 | 5.63397 | 1.71012 | 1.29412 |
| 19 | 3.13829 | 3.94996 | 10.6006 | 44 | 8.36749 | 11.9389 | 19.895 |
| 20 | 7.38361 | 8.19146 | 6.61535 | 45 | 6.20177 | 3.53495 | 3.76311 |
| 21 | 12.6898 | 1.34306 | 6.16523 | 46 | 0.917468 | 3.29029 | 8.44586 |
| 22 | 5.09787 | 3.6932 | 7.17308 | 47 | 0.563805 | 2.24416 | 8.8131 |
| 23 | 6.72258 | 0.32182 | 2.83409 | 48 | 1.90346 | 1.56457 | 4.67725 |
| 24 | 0.70572 | 0.279624 | 1.11649 | 49 | 10.0496 | 13.9582 | 16.0561 |
| 25 | 6.02397 | 1.56295 | 3.66128 | 50 | 3.21397 | 0.32086 | 1.52562 |

Table 5.1: Difference in simulated and approximated loss, setup, and nosetup probabilities for test cases in Table D.1.

38, setup probability for test case 42, and probability of no setup for test cases 2, 3 and 42. In the networks associated with these test cases, the traffic parameters indicate that the network is heavily loaded. For example in the first network (that corresponds to test cases 2 and 3), the network is heavily loaded by $\lambda_{03} = 0.75$, $\lambda_{04} = 4.0$ and $\lambda_{05} = 5.0$. Such a heavily loaded situation is unlikely to occur in practice. Even in these extreme situations, out approximation method performs fairly accurately. The purpose of the approximation method is to provide an approximate value for the performance measures associated with

| Percent Difference Range | Number of Losses | Number of Setups | Number of No Setups |
|---|---|---|---|
| x < 1% | 5 | 10 | 7 |
| 1 ≤ x < 5% | 12 | 31 | 22 |
| 5 ≤ x < 10% | 11 | 5 | 13 |
| 10 ≤ x < 15% | 7 | 2 | 1 |
| 15 ≤ x < 20% | 12 | 1 | 4 |
| 20 ≤ x < 25% | 3 | 0 | 0 |
| 25 ≤ x < 30% | 0 | 0 | 1 |
| 30 ≤ x < 40% | 0 | 0 | 2 |
| 40 ≤ x < 50% | 0 | 1 | 0 |

Table 5.2: Number of percent difference results for each statistic falling in the indicated ranges.

a particular network that can be used in an algorithm for solving the VPDBA problem. An average percent difference of 10% can be tolerated in such an application. We conclude that our approximation method is reasonably accurate and is appropriate for determining performance measures in multiple node, multiple VP networks.

### 5.3.3 VPDBA Solution Experiment

We now verify that our approximation method can be used effectively to find the optimal solution to the VPDBA problem for a specific network test case. We consider the optimal solution to be the solution that minimizes $F = \alpha P(loss) + (1 - \alpha)P(setup)$ for the network. The optimal solution includes $V^*$ the number of VPs, $\wp^*$ the placement of the VPs, and $c^*$ the bandwidth allocation to the VPs.

In each test case, all possible solutions $(V^*, \wp^*, c^*)$ were enumerated. Then the performance measures were obtained for each possible solution using both simulation and the approximation method. The optimal solution in each case was the solution that minimized the overall network performance measure $F$ for an $\alpha$ value of 0.5. This value of $\alpha$ was

chosen because it places equal emphasis on the $P(loss)$ and $P(setup)$.

### 5.3.3.1 Test Cases

We considered small networks with 5 nodes. Each node had 4-5 servers associated with it and each network had 10 arrival streams. While these networks test cases are small, the number of possible solution examined in the exhaustive search of all possible solutions for each network ranged from 13,000 to 52,000. The specifics of the input parameters for each test case are shown in Appendix D Table D.5 (Section D.1.2.1).

These test cases were designed to consider various network situations. In test case 1, the first node in the network receives the majority of the traffic. The heavily loaded streams run the length of the network. In test case 2, the heavily loaded stream arrives in the middle of the network and the path followed is much shorter. In both test cases 1 and 2, it is fairly obvious that the network will perform well when the streams that make significant contributions to the overall network load are assigned VP servers. However the optimal capacity assignment is not necessarily clear. In the third test case, no single stream is the obvious dominant source of incoming traffic. In this case, the optimal solution to the VP layout and capacity allocation problem is unclear to the user.

### 5.3.3.2 Results

Table 5.3 shows the optimal $F$ value and layout and capacity solution as determined by exhaustive search of the simulated network performance measures and the network performance measures approximated using our approximation method in each test case. The layout and capacity are expressed as $SD(V)$ where $S$ is the source node of the VP, $D$ is

| test case | simulation optimal solution | simulation F value | approximation optimal solution | approximation F value |
|---|---|---|---|---|
| 1 | 5: 02(1), 03(2), 04(2), 23(1), 34(3) | 0.206659 | 7: 01(1), 02(1), 03(2), 04(1), 14(1), 23(1), 34(3) | 0.208561 |
| 2 | 6: 01(2), 02(2), 13(2), 23(1), 24(1), 34(3) | 0.200047 | 6: 01(2), 02(2), 13(2), 23(1), 24(1), 34(3) | 0.202435 |
| 3 | 8: 01(2), 02(1), 03(1), 12(1), 13(1), 23(1), 24(1), 34(3) | 0.125871 | 8: 01(2), 02(1), 03(1) 12(1), 13(1), 23(1), 24(1), 34(3) | 0.125289 |

Table 5.3: Resulting optimal number, placement and capacity for VPs in each network based on simulated values and approximated values.

the destination node of the VP and $V$ is the number of servers assigned to this VP. Tables 5.4, 5.5 and 5.6 give more detailed information about the solutions found for each test case. In Tables 5.4, 5.5 and 5.6 the number of equivalent solutions has been noted. This represents the number of solutions within +/- 0.025 of the optimal $F$ value. We consider these essentially equivalent solutions to be equally optimal.

### 5.3.3.3 Conclusion

For each network test case, thousands of VP configurations were considered. In all test cases, the optimal solution found using our approximation method had an overall performance measure $F$ that was very close to the optimal $F$ value found through simulation. In test cases 2 and 3, the solution found using the performance measure approximation method and the solution found through simulation were identical. In addition, the overall network performance measure $F$ calculated via simulation and approximation were extremely close (+/- 0.002338 for test case 2 and +/- 0.000582 for test case 3) representing a 1.2% difference for test case 2 and 0.5% difference in test case 3. In test case 1, the layouts selected as

| test case 1 | | | | | | | |
|---|---|---|---|---|---|---|---|
| simulation | | | | | approximation method | | |
| number of vps | optimal solution | F value | number of equivalent | | optimal solution | F value | number of equivalent |
| 0 | | 0.602758 | 1 | | | 0.649445 | 1 |
| 1 | 03(5) | 0.327057 | 1 | | 03(5) | 0.331764 | 1 |
| 2 | 03(3), 04(2) | 0.251921 | 3 | | 03(3), 04(2) | 0.273550 | 8 |
| 3 | 03(3), 04(2), 34(3) | 0.213920 | 9 | | 03(3), 04(2), 34(3) | 0.228140 | 9 |
| 4 | 02(1), 03(2), 04(2), 34(3) | 0.223457 | 23 | | 02(1), 03(3), 04(1), 34(4) | 0.226276 | 13 |
| 5 | 02(1), 03(2), 04(2), 23(1) 34(3) | 0.206659 | 21 | | 02(1), 03(3), 04(1), 23(1), 34(4) | 0.208810 | 20 |
| 6 | 02(2), 03(2), 04(1), 23(1), 24(1), 34(3) | 0.221936 | 63 | | 02(2), 03(2), 04(1), 23(1), 24(1), 34(3) | 0.221224 | 32 |
| 7 | 01(1), 02(1), 03(2), 04(1), 14(1), 23(1), 24(3) | 0.210783 | 30 | | 01(1), 02(1), 03(2), 04(1), 14(1), 23(1), 34(3) | 0.208561 | 22 |
| 8 | 01(1), 02(1), 03(2), 04(1), 12(1), 23(1), 24(1), 34(3) | 0.215937 | 5 | | 01(1), 02(1), 03(2), 04(1) 12(1), 23(1) 24(1), 34(3) | 0.214060 | 15 |
| 9 | 01(2), 02(1), 03(1), 04(1), 12(1), 14(1), 23(1), 24(1), 34(2) | 0.246461 | 8 | | 01(1), 02(1), 03(1), 04(1), 12(1), 14(1), 23(1), 24(1), 34(2) | 0.219963 | 5 |

Table 5.4: Optimal VP layout and capacity for each possible number of VPs for the network in Test Case 1.

optimal via simulation and via our performance method had overall network performance measures ($F$ values) with a 0.9% difference or +/- 0.025 from each other. The approximated $F$ value for the optimal layout found through simulation (01(4), 12(2), 13(2), 23(2), 34(4)) was 0.215057 while the approximated $F$ value for the optimal layout found through

| test case 2 | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| simulation | | | | approximation method | | | |
| number of vps | optimal solution | F value | number of equivalent | | optimal solution | F value | number of equivalent |
| 0 | | 0.603874 | 1 | | | 0.614286 | 1 |
| 1 | 04(4) | 0.478747 | 4 | | 04(4) | 0.478726 | 3 |
| 2 | 01(4), 13(4) | 0.316777 | 3 | | 01(3), 13(4) | 0.311112 | 3 |
| 3 | 01(4), 13(2), 24(2) | 0.285116 | 20 | | 01(3), 13(3), 24(1) | 0.281117 | 21 |
| 4 | 01(2), 02(2), 13(2), 24(2) | 0.257768 | 7 | | 01(2), 02(2), 13(2), 24(2) | 0.238121 | 12 |
| 5 | 01(2), 02(2), 13(2), 23(1), 24(1) | 0.239428 | 18 | | 01(2), 02(2), 13(2), 24(2), 34(2) | 0.218600 | 33 |
| 6 | 01(2), 02(2), 13(2), 23(1), 24(1), 34(3) | 0.200047 | 20 | | 01(2), 02(2), 13(2), 23(1), 24(1), 34(3) | 0.202435 | 25 |
| 7 | 01(2), 02(1), 03(1), 12(1), 13(1), 24(2), 34(2) | 0.220909 | 30 | | 01(2), 02(1), 03(1), 12(1), 13(1), 24(2), 34(2) | 0.221853 | 32 |
| 8 | 01(2), 02(1), 03(1), 12(1), 13(1), 23(1), 24(1), 34(3) | 0.206427 | 8 | | 01(2), 02(1), 03(1), 12(1), 13(1), 23(1), 24(1), 34(3) | 0.204837 | 10 |

Table 5.5: Optimal VP layout and capacity for each possible number of VPs for the network in Test Case 2.

approximation (01(1), 02(1), 03(2), 04(1), 14(1), 24(1), 34(3)) was 0.208561. The difference between these two approximated values is 0.006496. We claim that the overall network performance in these two test cases is essentially the same and that the solutions can be considered equally optimal.

We also observe that in the intermediate layout and capacity results obtained for the test cases, many of the overall performance measures for specific numbers of VPs are very close to one another. In many cases the optimal layout and capacity assignment found

| test case 3 | | | | | | |
|---|---|---|---|---|---|---|
| simulation | | | | approximation method | | |
| number of vps | optimal solution | F value | number of equivalent | optimal solution | F value | number of equivalent |
| 0 | | 0.643511 | 1 | | 0.528922 | 1 |
| 1 | 34(2) | 0.436760 | 4 | 34(2) | 0.440528 | 4 |
| 2 | 03(4), 34(4) | 0.367041 | 12 | 03(4), 34(4) | 0.366966 | 10 |
| 3 | 01(4), 13(4), 34(4) | 0.267016 | 8 | 01(3), 13(4), 34(4) | 0.264087 | 9 |
| 4 | 01(4), 13(2), 23(2), 34(4) | 0.225085 | 61 | 01(3), 13(2), 23(2), 34(4) | 0.219653 | 69 |
| 5 | 01(4), 12(2), 13(2), 23(2), 34(4) | 0.174702 | 73 | 01(3), 12(3), 23(2), 24(2), 34(2) | 0.168676 | 67 |
| 6 | 01(4), 12(3), 13(1), 23(2), 24(1), 34(2) | 0.143855 | 47 | 01(3), 12(2), 13(1), 23(2), 24(1), 34(3) | 0.135016 | 39 |
| 7 | 01(4), 12(2), 13(1), 14(1), 23(1), 24(1) 34(2) | 0.140576 | 17 | 01(3), 12(2), 13(1), 14(1), 23(1), 24(1) 34(2) | 0.135016 | 13 |
| 8 | 01(2), 02(1), 03(1), 12(1), 13(1), 233(1) 24(1), 34(3) | 0.125871 | 6 | 01(2), 02(1), 03(1), 12(1), 13(1), 23(1), 24(1), 34(3) | 0.125289 | 7 |

Table 5.6: Optimal VP layout and capacity for each possible number of VPs for the network in Test Case 3.

via simulation and our approximation method for a specific number of VPs within a given network were identical.

We conclude that our approximation method is valid and can be used to accurately evaluate network performance without using time-consuming simulation. Thus our performance measure approximation method is appropriate for use in a heuristic algorithm to determine the optimal placement of and capacity allocation to VPs with in a general line network.

## 5.4 Summary

We have expanded the previous approximation methods to apply to multiple node systems containing multiple VPs. Our approximation method produces results very close to those obtained through simulation. In addition, the optimal VPDBA solution identified through our approximation method and the optimal solution identified by simulation were identical in all test cases considered. We conclude that our approximation method is effective and can be used in an algorithm to determine the optimal solution to the VPDBA problem.

# Chapter 6

# Algorithm for the VPDBA

# Problem

We have shown that using shared bandwidth in a network containing VPs can provide many benefits. We have also developed a method for approximating network performance measures in a network containing shared bandwidth and have shown that our approximation method is accurate. We now present our heuristic algorithm for VP Distribution and Bandwidth Allocation that explicitly considers shared bandwidth. The goal of the algorithm is to find a good solution to the NP-Complete VPDBA problem including the placement of VPs and capacity allocation to VPs within the network.

The algorithm that we have developed uses a greedy strategy to quickly determine a good solution to the VPDBA problem for a given network and set of network parameters. Our algorithm guarantees that each VP layout produced will contain shared bandwidth. By using shared bandwidth, our algorithm produces solutions that provide improved performance in several respects when compared to the solutions produced by algorithms that do not consider shared bandwidth. Specifically, our algorithm produces VP layouts and capacity assignments that exhibit a degree of fairness not seen in VP layouts created with

108

previous VPDBA algorithms. Our algorithm guarantees that in the resulting network, no traffic stream will be completely denied service even if not allocated VP capacity. Using shared bandwidth, some losses are still possible for each VP and non-VP stream, however no stream will be forced to loose all of its traffic so that better service may be provided to other streams within the network. While providing improved fairness, our algorithm produces solutions with network–wide performance measures comparable to those produced by other algorithms.

We begin by summarizing previous algorithms for the VPDBA problem. Then we present our algorithm for the VPDBA problem. In this chapter, we restrict our network topology to be a simple line. Later, we will discuss extensions of our algorithm that apply to more general networks. We present several test cases and show that our algorithm produces results that are in many ways superior to those produced by the previously developed algorithms.

## 6.1 Previous VPDBA Algorithms

The problem of VPDBA has been studied previously in the literature and heuristic algorithms for VP distribution and capacity allocation have been presented [34, 18, 17, 1, 16, 7, 8, 40, 29, 4]. However, previous studies have failed to explicitly consider the idea of shared bandwidth. In addition, the majority of these works consider network models and problem formulations that differ greatly from the model and problem formulation used in our approach. The algorithms presented in [17, 18, 1, 40, 29] use methods of combining existing VPs to form new VP layouts. These algorithms focus on network topology data

rather than performance measures. The work done by [34] and [8] explores the effect of parameters on network performance but does not present an algorithm for VP Distribution and Bandwidth Allocation.

We focus on the three previous algorithms that use models closely related to our network model. In these algorithms, VPs are assigned and then capacities are adjusted in an attempt to optimize some measure of network performance. However, in these algorithms there is no guarantee that every incoming traffic stream in the network will be allocated capacity on a VP. If five streams share a single link, the previous algorithms allow all of the capacity to be allocated to a few of the streams. Thus, some streams suffer complete loss of service in order to improve the performance of other streams within the network.

Before presenting our heuristic algorithm for VPDBA, we summarize these algorithms for solving the VPDBA problem in closely related network models. Later the resulting solution for these algorithms will be compared to the solution determined by our algorithm for several test cases.

## 6.1.1 Gain/Loss Ratio Method

Arvidsson [7] presents a greedy algorithm for VP Distribution and Bandwidth Allocation. A high, initially acceptable call loss level is assigned to each traffic stream. In [7] the initial loss levels are assigned to be 0.50. In other words, half of the calls on each stream can be lost initially. This parameter value seems to be arbitrary as no justification or intuition is provided for the choice of initial loss level value. When running this algorithm, we used the value of 0.50 as the initial loss level.

After the initial loss levels have been assigned, the shortest path is calculated for each

source-destination pair within the network. Then each traffic stream is considered and the network performance gain achieved by adding one unit of capacity to the VP associated with the traffic stream is calculated. Similarly, the network performance loss achieved by adding one unit of capacity to the VP associated with the traffic stream is calculated. These performance measures are calculated using the Erlang B formula. After the gain and loss have been calculated for each traffic stream, the gain/loss ratio is calculated for each traffic stream. The traffic stream with the highest gain/loss ratio is selected to receive an additional unit of capacity. If there is no gain achieved by adding capacity to the VP associated with any stream then the algorithm determines whether the loss levels have been achieved. If they have not, then loss levels are reduced and the algorithm proceeds. If they have, then the algorithm terminates with the VP distribution and capacity assignment as the final solution. The pseudocode for this algorithm is shown in Appendix E.

### 6.1.2  Overall Blocking Rate Method

Cheng and Lin [16] present a greedy algorithm for the VPDBA problem. In their algorithm, all capacity is initially assigned to the one hop paths within the network. Thus initially, the traffic streams corresponding to these one hop paths are the only VPs within the network. All other traffic streams are completely lost since they do not have any capacity assigned to handle their calls.

As the algorithm proceeds, the capacity is adjusted by the following sequence of steps. First the current overall blocking rate is calculated using the Erlang B formula. Increasing the capacity allocated to each traffic source by one unit is considered. A new blocking rate is calculated assuming that one unit of capacity is added to the traffic source under

consideration and subtracted from each of the corresponding one hop routes on the same physical path. If the overall blocking rate is lowered by one or more of these new capacity assignments, then the assignment with the lowest overall blocking rate is selected as the new VP layout and capacity assignment and the algorithm continues to try to improve the layout and capacity assignment. If no new capacity assignment results in a lower blocking rate, then the algorithm terminates with the current layout and capacity assignment. The pseudocode for this algorithm is shown in Appendix E.

### 6.1.3 Blocking Drift Method

Aneroussis and Lazar [4] present a greedy algorithm for VP capacity assignment. Their algorithm does not address how the layout of the VPs should be chosen for a network. So that we may compare the performance of this algorithm with the other algorithms discussed as well as our own algorithm for VP distribution and capacity assignment, we add a step to the algorithm in which VPs are distributed. We assume that all streams are potentially VPs with assigned capacities of zero. Then we proceed to the capacity assignment adjustment algorithm presented by Aneroussis and Lazar.

The capacity assignment algorithm uses three phases. In the first phase, all source–destination pairs for which the blocking constraints are not satisfied are considered. For each of these source–destination pairs, the capacity for this traffic source is increased while holding the capacity of all other VPs constant. Then the blocking drift is calculated for the new capacity assignment. The blocking drift $D_b$ is defined to be

$$D_b = \sum_{\forall w \in W, k \in K} max(0, P_w^k - \beta_w^k)$$

where $W$ is the set of all source–destination pairs, $K$ is the set of all traffic classes, $P_w^k$ is

the percentage of call attempts for traffic class $k$ for the source–destination pair $w$ that are denied service due to the unavailability of resources, and $\beta_w^k$ is the blocking constraint for traffic class $k$ from source–destination pair $w$. The blocking probability is calculated using the Erlang B formula. If one of the new assignments results in a lower blocking drift, then the assignment with the lowest blocking drift is selected as the new capacity assignment.

In the second phase, signaling violations are considered. Our model does not consider signaling violations. Therefore, we ignore this phase of the algorithm and assume that no signaling constraints are violated. Phase one and two are repeated until no blocking violations or signaling violations are incurred.

After all blocking violations and signaling violations are satisfied, then the algorithm proceeds to phase three that attempts to further optimize the VP capacity assignment. In this phase, the network throughput is calculated for each source–destination pair using the Erlang B formula. Then the network revenue is calculated. Network revenue is sum of the network throughput for each source–destination pair multiplied by the revenue obtained by accepting one call from this source–destination pair. In our experiments, we assume that each traffic source produces equal network revenue. Every VP whose capacity can be increased by one unit is considered. The new network revenue is calculated for each of these new capacity assignments. If none of the new capacity assignments results in a higher network revenue, then the algorithm terminates, otherwise the capacity assignment with the highest revenue is selected as the new network layout and capacity assignment and phase three is repeated. The pseudocode for this algorithm is presented in Appendix E.

## 6.2   VPDBA Algorithm with Bandwidth Sharing

We present our algorithm, compare the performance of our algorithm with the optimal solution found by exhaustive search of the solution space for several networks, present test cases used to compare our algorithm to the other algorithms discussed in this chapter, and compare the results of our algorithm to the results produced by other algorithms.

### 6.2.1   Our Algorithm

We use a greedy strategy to distribute VPs and assign capacity to them while ensuring that the resulting network contains shared bandwidth. As an initial step, we force one unit of bandwidth to remain unreserved or shared within the network. Additional units of bandwidth may be shared as determined by our bandwidth allocation process. By forcing this shared bandwidth to remain unreserved within the network, we provide a degree of fairness to all traffic sources within the network. It is never the case that all network capacity on a particular link is assigned completely to one or more VPs. In this way, our algorithm allocates capacity to VPs but does not force the loss of all incoming non-VP traffic. The presence of shared bandwidth also provides greater network flexibility and reduces the need for re-distributing capacity when traffic patterns change in the network.

After the single unit of shared bandwidth is set aside, the algorithm begins to determine the VP distribution and capacity allocation. At each step of the algorithm, the stream that contributes the most significantly to overall call loss probability experienced by the network is given one additional unit of reserved VP capacity. Our algorithm continues to assign capacity in this fashion until all capacity available for VPs has been distributed or until the

network performance ceases to improve with the addition of capacity to any VP. Pseudocode for this algorithm is shown below.

1. Calculate the current overall network performance using out network performance measure approximation method. The overall network performance is defined to be

   $F = \alpha P(loss) + (1 - \alpha)P(setup)$

2. On each link in the network, one unit of capacity is set aside. This capacity is guaranteed to be shared bandwidth.

3. While there is capacity remaining that has not been allocated (either to a VP or in step 2 above) and the network performance is improving

   (a) Consider all traffic sources that can have an increased amount of capacity allocated to them. Calculate the amount of traffic currently lost from each of these streams.

   (b) Select the traffic stream that has the highest amount of traffic loss associated with it. The VP capacity of the selected stream is increased by one unit.

   (c) The network performance $F$ is re-calculated

   (d) If the network performance of the new VP layout and capacity assignment has not improved when compared to the previous network performance, then return to the previous VP layout and capacity assignment and stop, otherwise continue to try to improve network performance.

## 6.3   Our Algorithm vs. Exhaustive Search

We considered three network test cases for which an exhaustive search method was used to calculate the optimal VP layout and capacity allocation. For each of these test cases, we ran our algorithm and compared the resulting VP layout and capacity allocation to that found by the exhaustive search. We present the layout and capacity assignment as well as the overall network performance for each test case using each solution method. The test cases are shown in the Appendix D in Table D.5.

The resulting layout and capacity allocation and corresponding network performance value for each of the three test cases is shown in Table 6.1. In all test cases considered, the performance measure for the resulting layout and capacity allocation produced by our algorithm are within 0.15 of the performance measures for the true optimal solution as found by exhaustive search.

| test case | exhaustive search layout | network performance $\alpha = 0.5$ | our algorithm layout | network performance $\alpha = 0.5$ |
|---|---|---|---|---|
| 1 | 01(1),02(1),03(2) 04(1), 14(1), 23(1), 34(3) | 0.208561 | 03(3), 04(1), 34(3) | 0.286351 |
| 2 | 01(2), 02(2), 13(2), 23(1), 24(1), 34(3) | 0.202435 | 01(1), 02(1), 03(1), 13(1), 24(1) | 0.340908 |
| 3 | 01(2), 02(1), 03(1) 12(1), 13(1), 23(1) 24(1), 34(3) | 0.125289 | 12(1), 13(1), 23(1), 24(1), 34(2) | 0.263817 |

Table 6.1: The resulting layout and capacity assignment as produced by our algorithm and exhaustive search.

Our algorithm always produces a VPDBA solution that includes shared bandwidth. Because of this fact, the probability of call setup will never be zero for our algorithm. Therefore, when the network performance is calculated, our network will always have a probability of

call setup that contributes to the final network statistic, while this is not necessarily true of the optimal solution found through exhaustive search. In the test cases considered, the VPDBA solution found through exhaustive search contained no shared bandwidth. Thus none of the accepted calls incurred a call setup cost. So the probability of a call setup was zero for these optimal solutions. As such, we would expect the overall network performance of the VPDBA solutions produced by our algorithm to be higher than those found through exhaustive search. Table 6.2 shows the loss probability and setup probability for each of the solutions described in the table above. As a result of including shared bandwidth in our solution, we have some traffic that incurs a setup cost and contributes to the final network performance. We expect our algorithm to perform fairly well, but recognize that it will not always be capable of producing solutions with overall network performance measures that mimic the optimal because our algorithm includes shared bandwidth. The shared bandwidth helps ensure that no single traffic stream will be completely denied service in order to improve the performance for other streams. We conclude that our algorithm produces fairly good VP layouts and capacity assignments.

| Network | Algorithm | Loss | Setup |
|---------|-----------|------|-------|
| 1 | Anewalt | 0.343228 | 0.229474 |
| 1 | Exhaustive Search | 0.430113 | 0 |
| 2 | Anewalt | 0.327516 | 0.354300 |
| 2 | Exhaustive Search | 0.404870 | 0 |
| 3 | Anewalt | 0.145929 | 0.381000 |
| 3 | Exhaustive Search | 0.250578 | 0 |

Table 6.2: The Loss and Setup Probabilities for our algorithm and exhaustive search.

## 6.4 Comparing Our Algorithm to Previous Algorithms

We have shown that our algorithm produces results that compare favorably with the solutions found through exhaustive search. We now present 10 test cases and use our algorithm as well as the other algorithms discussed to produce a VP layout and capacity assignment for each test case. Then we calculate network performance measures including the loss probability, setup probability, fraction of traffic handled and the fraction of streams handled for each resulting VP layout and capacity assignment. We show that our algorithm produces VPDBA solutions that provide superior network performance in some respects without sacrificing overall network performance.

### 6.4.1 Performance Measures

To compare the performance of the VP layout and capacity assignments produced by our algorithm to those produced by the previously studied algorithms, we use several performance measures. We compare the overall network performance $F = \alpha P(loss) + (1 - \alpha)P(setup)$ using a range of $\alpha$ values. In addition, we individually compare the loss probability and setup up probability for each layout. We also identify the number of traffic streams that are able to transmit some information as a fraction of the total number of streams in the network. Finally, we look at the fraction of the total traffic carried in each case. This is the sum of the estimated amount of each traffic source carried divided by the sum of the total amount of traffic offered to the network. Our performance measure approximation method is used to calculate these performance measures.

## 6.4.2   Assumptions

We restrict our attention to line networks at this time.

The algorithm proposed by Arvidsson assigns initially acceptable loss levels as the first step in the algorithm. In their paper a value of 0.50 is used for all traffic. We use the same parameter value as no intuition or reason for selecting this value is given.

The algorithm presented by Aneroussis and Lazar only considers capacity assignment and assumes that the location of VPs has been previously established. In order to compare the performance of their algorithm to our algorithm, we assume that each traffic source is assigned a VP with an initial capacity of zero. Then we use their capacity assignment algorithm to adjust these capacities. In this algorithm, a maximum blocking probability is initially assigned to each traffic source. No intuition is given for choosing a good value for this quantity. We assume that the maximum blocking probability is 0 for all traffic streams to try to produce the best possible layout. This algorithm also uses a revenue value for each traffic source. We assume that all traffic sources produce equal revenue and use a value of one in for the revenue for all traffic sources.

## 6.4.3   Test Cases

A set of ten test cases are considered in this experiment. Each test case contains between 4 and 6 nodes and between 3 and 5 servers on each link. The $\lambda$ values for each test case are listed in Appendix E Table E.1 (Section E.2.1).

In order to present a full range of network parameter values, the test cases were considered with a network load parameter $\gamma$. The value $\gamma$ varied between 0.1 and 10.0. For $\gamma$ values between 0.1 and 1.0, the values varied by a step size of 0.1. For $\gamma$ values greater

than 1.0, the values varied by a step size of 1.0. This allowed a more in depth study of the algorithms' performance for lightly loaded networks and heavily loaded networks.

### 6.4.4 Results

Each of the four algorithms was run for each test case. The value of $\alpha$ was varied over the range of possible values $0 \leq \alpha \leq 1.0$. The load of the network $\gamma$ was varied over a range of possible values $0 \leq \gamma \leq 10.0$. After the VP layout and capacity assignment was determined by each algorithm, the network performance measures were calculated for each network. We calculate the network performance measures, overall network performance, probability of loss, probability of setup, fraction of traffic handled, and fraction of the streams handled for each solution. Our performance measure approximation method was used to produce estimates of the loss probability, setup probability, overall network performance and fraction of traffic carried for each network.

The calculated performance measures for a single test case are shown in Figures 6.1, 6.2, 6.3 and 6.4 . Similar results were obtained for other test cases and are shown in Section E.2.2 of Appendix E.

In figure 6.1, we see that our algorithm provides a lower loss probability for all $\gamma$ values less than 2.0. For a $\gamma$ value of 0.1, the loss probability for our algorithm is 0.014 compared to the loss probabilities of 0.297, 0.411, 0.744 for the other algorithms). For $\gamma$ values greater than 2.0, our algorithm performs comparably to the best of the other algorithms.

In figure 6.2, we see that our algorithm produces networks that have a higher resulting probability of call setup. However, the call setup probability for the networks produced by our algorithm is still significantly lower than the setup probability for networks that do
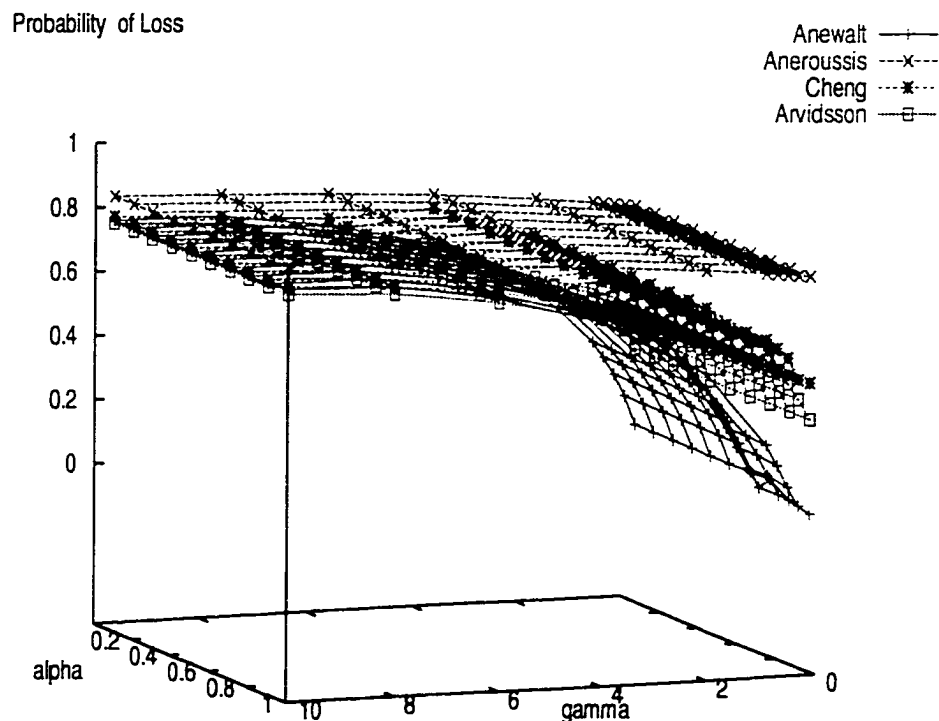
**Figure 6.1**: Comparing the Probability of Loss for the layout produced from our algorithm and the layouts produced from the three other algorithms.

not contain VPs. The other algorithms only carry traffic on VPs - all traffic that is not assigned to a VP is lost. As a result their call setup probability is always essentially 0. Our algorithm only services a portion of traffic on VPs. This increases the probability of call setup, but also increases the number of streams handled by the network.

In figure 6.3, we see that the fraction of traffic carried by the network produced by our algorithm is much higher than the fraction of traffic carried by the networks produced by
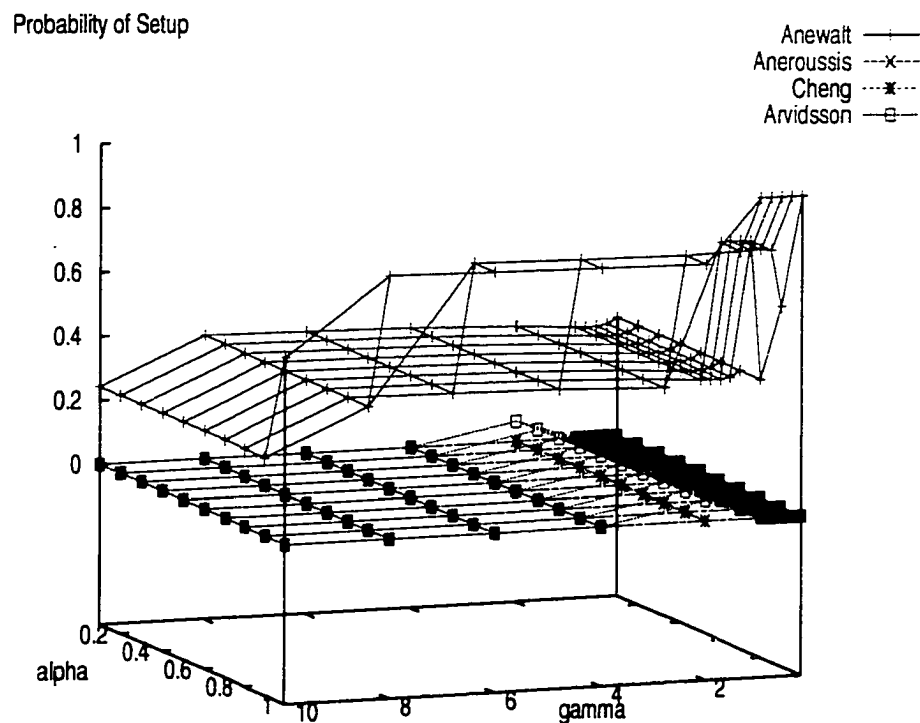
**Figure 6.2**: Comparing the Probability of Setup for the layout produced from our algorithm and the layouts produced from the three other algorithms.

the other algorithms for $\gamma$ values less than 2.0. It is highly undesirable for networks to provide no service to a large amount of the traffic offered to the network. For $\gamma$ values less than 2.0, our algorithm produces networks that handle greater than or equal to 20% more of the offered traffic than the networks produced by the other algorithms considered. Thus the network produced by our algorithm provides a more desirable level of service to the network traffic.
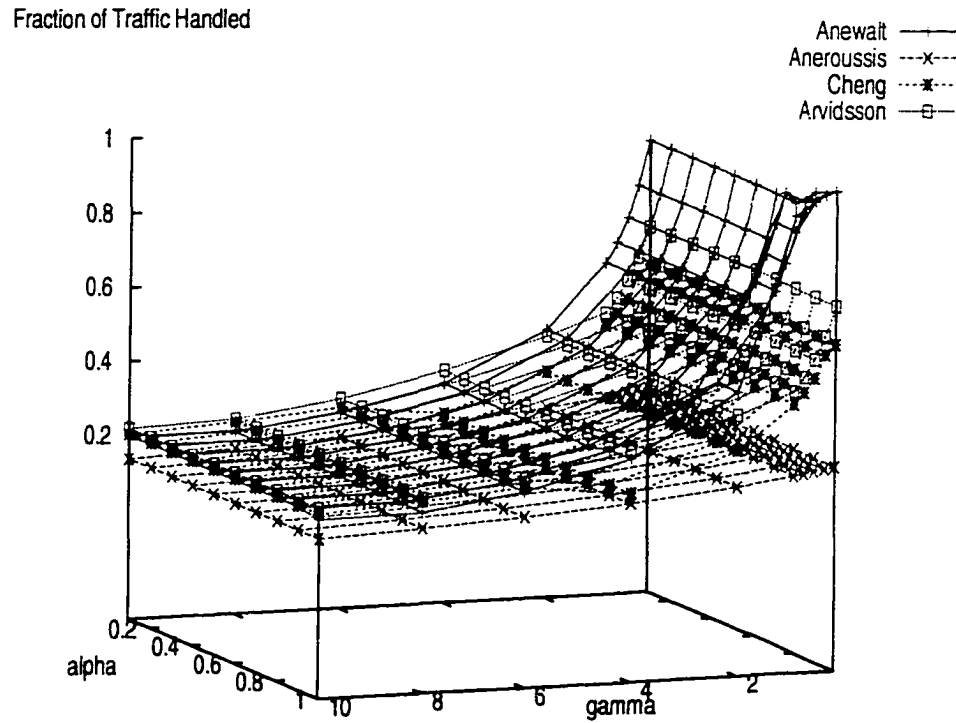
**Figure 6.3**: Comparing the Fraction of Traffic handled by the layout produced from our algorithm and the layouts produced from the three other algorithms.

The fraction of streams handled by the network is shown in Figure 6.4. The fraction of streams handled by the network is not affected by the value of $\alpha$. In addition, the value of the parameter is not greatly affected by the $\gamma$ parameter. The fraction of streams handled is almost constant through the entire range of $\alpha$ and $\gamma$ values with only slight variances.

The fraction of streams handled is always higher for networks produced by our algorithm. Our algorithm guarantees that 100% of the traffic streams will receive some level of service.
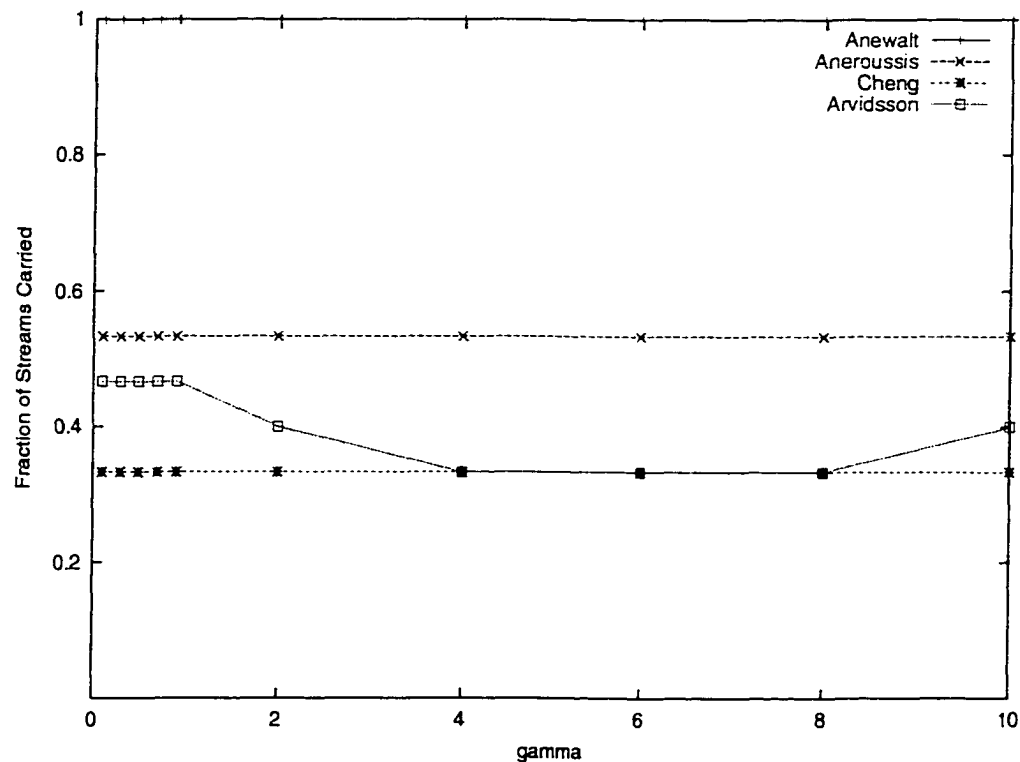
**Figure 6.4**: Comparing the Fraction of Streams handled by the layout produced from our algorithm and the layouts produced from the three other algorithms.

None of the other algorithms for the VPDBA problem provide this type of guarantee. In many cases, this guarantee produces significant effects on the traffic dynamics. For example in the test case results shown, our algorithm produces a network with 100% of the streams handled. However the other algorithms produce networks in which 40%, 30%, 50% (approximately) of the traffic is handled. Clearly a VP layout that is only capable of handling calls from 50% or less of the traffic streams offered to the network is undesirable. In this way our algorithm produces solutions to the VPDBA problem that offer significant network benefits. Our algorithm guarantees that all streams will receive some level of service. No stream is completely denied service in order to improve the service given to

other streams.

The other test cases produced similar results. Our algorithm produces results that have the same or lower loss probabilities as the other algorithms. Our algorithm always produces VPDBA results that have higher setup probabilities since some traffic will be handled without VPs. Our algorithm handled more traffic and a much larger fraction of traffic streams in all test cases. These results are shown in Appendix E Section E.2.2.

## 6.5  Summary

We have shown earlier that shared bandwidth can provide many benefits within a VP network. We developed a performance measure approximation method for networks containing shared bandwidth. We have presented an algorithm for VP Distribution and Bandwidth Allocation that explicitly considers shared bandwidth. The algorithm ensures that the resulting VP layout and capacity assignment will contain some shared bandwidth. Thus total loss of non-VP streams is avoided. Our algorithm is simple and efficient. It uses a greedy strategy to select the best stream to add more VP capacity at each iteration of the algorithm. Our algorithm halts when the network performance ceases to improve when more VP capacity is added. In this way, all capacity is not assigned to VPs if it does not help to improve network performance. Rather we keep as much of the bandwidth shared as possible.

We have compared the network performance of the VP layouts and capacities produced by our algorithm to those produced by the other algorithms and have shown that our algorithm produces VPDBA solutions that provide 100% of the offered traffic streams with

some level of service. We have shown that solutions determined by our algorithm provide service to a greater amount of the offered traffic than solutions determined by the previous algorithms, resulting in a greater level of fairness in the resulting network. In addition, the resulting network can more easily adapt to changing traffic patterns by reducing the need for re-configuring the network over time. In addition to providing this increased level of fairness, our algorithm produces networks with loss probabilities and setup probabilities that are comparable to those produced by previous algorithms.

# Chapter 7

# The General Network

We now show that our greedy algorithm can be used in a network with a general topology to produce results similar to those produced for the line network.

## 7.1 VPDBA Algorithm with Bandwidth Sharing for a General Network

### 7.1.1 Our Algorithm for a General Network

Before beginning the VPDBA process, we find the shortest path for each traffic stream using Dijkstra's algorithm. We assume that this shortest path will always be used for the purposes of our algorithm and performance approximation calculations. We then use a greedy strategy to distribute VPs and assign capacity to the links while ensuring that the resulting network contains shared bandwidth. As in the line case, we force one unit of bandwidth to remain unreserved or shared within the network; additional units of bandwidth may be shared as determined by our bandwidth allocation process.

After the single unit of shared bandwidth is set aside, the algorithm begins to determine the VP distribution and capacity allocation. At each step of the algorithm, the stream that

127

contributes the most significantly to overall call loss probability experienced by the network is given one additional unit of reserved VP capacity. Our algorithm continues to assign capacity in this fashion until all capacity available for VPs has been distributed or until the network performance ceases to improve with the addition of capacity to the VP under consideration. Pseudocode for this algorithm is shown below.

1. Determine the shortest path associated with each traffic stream.

2. Calculate the current overall network performance using out network performance measure approximation method. The overall network performance is defined to be

   $F = \alpha P(loss) + (1 - \alpha)P(setup)$

3. On each link in the network, one unit of capacity is set aside. This capacity is guaranteed to be shared bandwidth.

4. While there is capacity remaining that has not been allocated (either to a VP or in step 2 above) and the network performance is improving

   (a) Consider all traffic sources that can have an increased amount of capacity allocated to them. Calculate the amount of traffic currently lost from each of these streams.

   (b) Select the traffic stream that has the highest amount of traffic loss associated with it. The VP capacity of the selected stream is increased by one unit.

   (c) The network performance $F$ is re-calculated

   (d) If the network performance of the new VP layout and capacity assignment has not improved when compared to the previous network performance, then return

to the previous VP layout and capacity assignment and stop, otherwise continue to try to improve network performance.

In order to calculate the performance of the general network quickly, we use a modification to the performance measure approximation used in the line networks previously considered. In our modified approach, we consider one traffic stream at a time. For each stream considered, we create a sub-network associated with the stream from the general network. The sub-network for each traffic stream has a line topology and contains all the nodes associated with the path of that stream and all of the traffic within the network that passes through the nodes associated with the stream. We then find the performance measures for the stream using the performance approximation methods detailed in Section 5.2 applied to the line network associated with the stream under consideration. The loss probability and setup probability associated with the stream of consideration are recorded and then the next stream in the general network is considered. After the individual performance of each stream is calculated, these measures are combined into an overall network performance measure using the formulas given in Section 5.2.

$$P(loss) = \frac{\sum_{\forall i,j} \Lambda'_{ij} P(loss_{ij})}{\sum_{\forall i,j} \Lambda_{ij}'}$$

$$P(no\ setup) = \frac{\sum_{a=1}^{p} (1-P^{v_a}) \Lambda'_{v p_a}}{\sum_{\forall i,j} \Lambda'_{i,j}}$$

$$P(setup)' = 1 - P(loss) - P(no\ setup)$$

$$P(setup) = \frac{P(setup)'}{P(setup)' + P(no\ setup)}$$

The overall network loss probability and setup probability are used to calculate the overall network performance for the general network under consideration using the formula

$$F = \alpha P(loss) + (1 - \alpha)P(setup)$$

## 7.2 Evaluating Algorithm Performance

We evaluate the performance of our algorithm by comparing the performance results of general networks with VPDBA solutions determined by our algorithm to the performance of the same networks without VP assignments. In addition to overall network performance, we compare the loss probability and setup probability for each layout.

### 7.2.1 Assumptions

We assume that all links within the network are bi-directional and that the same amount of capacity has been allocated to the link in each direction. Thus if the link between node 0 and node 1 has 10 units of capacity, we assume that 5 units are used for traffic from node 0 to node 1 and 5 units are used for traffic from node 1 to node 0.

### 7.2.2 Test Cases

Three network topologies were considered. Each Test Case represents a unique topology. Several traffic distributions were considered for each network topology. Connectivity and arrival rates were selected at random. Tables showing the connectivity and arrival rates for each network can be found in Appendix F Table F.1 (Section F.1.1).

### 7.2.3 Results

Our algorithm was run for each network test case and traffic distribution. A VP layout and capacity assignment was determined by the algorithm and the network performance measures were calculated for each network. For each network test case, we compare the approximated performance measures for the network with no VP assignments to the ap-

proximated performance measures for the network with the VPDBA determined by our algorithm. The comparisons are shown in the following tables. For each test case and traffic distribution, we show the resulting performance measures using a variety of $\alpha$ values. In addition, we highlight the improvement in network performance for the $\alpha$ value of 0.6. This value was chosen as a moderate value of $\alpha$ that does not overwhelmingly favor loss or setup probability. For the highlighted performance measures, we show the difference between the performance measures for a network with no VPs and the performance measures for the network with the VPDBA solution found using our algorithm. The actual VPDBA determined by our algorithm for each test case for all values of $\alpha$ are shown in Appendix F Section F.1.2.2.

| Test Case 1 - Traffic Distribution 1 | | | | | | |
|---|---|---|---|---|---|---|
| | no VP | | | VPDBA algorithm result | | |
| alpha | loss probability | setup probability | overall perform-ance | loss probability | setup probability | overall perform-ance |
| 0.1 | 0.490016 | 1.0 | 0.9490016 | 0.433869 | 0.272687 | 0.288805 |
| 0.4 | 0.490016 | 1.0 | 0.7960064 | 0.433869 | 0.272687 | 0.337160 |
| 0.6 | 0.490016 | 1.0 | 0.6940096 | 0.433869 | 0.272687 | 0.369396 |
| 0.9 | 0.490016 | 1.0 | 0.5410144 | 0.480098 | 0.793688 | 0.511457 |

| Test Case 1 - Traffic Distribution 1 - Alpha = 0.6 | | | | |
|---|---|---|---|---|
| | no VP | VPDBA algorithm | difference | percent difference |
| loss: | 0.490016 | 0.433869 | 0.056147 | 11.4582 |
| setup: | 1.0 | 0.272687 | 0.727313 | 72.7313 |
| performance: | 0.6940096 | 0.369396 | 0.324614 | 46.7736 |

| Test Case 1 - Traffic Distribution 2 | | | | | | |
|---|---|---|---|---|---|---|
| | no VP | | | VPDBA algorithm result | | |
| alpha | loss probability | setup probability | overall perform-ance | loss probability | setup probability | overall perform-ance |
| 0.1 | 0.370847 | 1.0 | 0.9370847 | 0.327637 | 0.262876 | 0.269352 |
| 0.4 | 0.370847 | 1.0 | 0.7483388 | 0.327637 | 0.262876 | 0.292666 |
| 0.6 | 0.370847 | 1.0 | 0.6225082 | 0.323947 | 0.267227 | 0.301259 |
| 0.9 | 0.370847 | 1.0 | 0.4337623 | 0.323744 | 0.738935 | 0.365263 |

| Test Case 1 - Traffic Distribution 2 - Alpha = 0.6 | | | | |
|---|---|---|---|---|
| | no VP | VPDBA algorithm | difference | percent difference |
| loss: | 0.370847 | 0.323947 | 0.046900 | 12.6467 |
| setup: | 1.0 | 0.267227 | 0.732773 | 73.2773 |
| network performance: | 0.6225082 | 0.301259 | 0.321249 | 51.6056 |

| Test Case 2 - Traffic Distribution 1 | | | | | | |
|---|---|---|---|---|---|---|
| | no VP | | | VPDBA algorithm result | | |
| alpha | loss probability | setup probability | overall perform-ance | loss probability | setup probability | overall perform-ance |
| 0.1 | 0.0476037 | 1.0 | 0.904760 | 0.0887154 | 0.134103 | 0.129564 |
| 0.4 | 0.0476037 | 1.0 | 0.619041 | 0.0887154 | 0.134103 | 0.115948 |
| 0.6 | 0.0476037 | 1.0 | 0.428562 | 0.0887154 | 0.134103 | 0.106870 |
| 0.9 | 0.0476037 | 1.0 | 0.142843 | 0.0542722 | 0.755723 | 0.124417 |

| Test Case 2 - Traffic Distribution 1 - Alpha = 0.6 | | | | |
|---|---|---|---|---|
| | no VP | VPDBA algorithm | difference | percent difference |
| loss: | 0.0476037 | 0.0887154 | 0.041112 | 46.3411 |
| setup: | 1.0 | 0.134103 | 0.865894 | 86.5894 |
| network performance: | 0.428562 | 0.106870 | 0.321692 | 75.0631 |

| Test Case 2 - Traffic Distribution 2 | | | | | | |
|---|---|---|---|---|---|---|
| | no VP | | | VPDBA algorithm result | | |
| alpha | loss probability | setup probability ance | overall perform- | loss probability | setup probability | overall perform- ance |
| 0.1 | 0.297206 | 1.0 | 0.9297206 | 0.270126 | 0.238546 | 0.241704 |
| 0.4 | 0.297206 | 1.0 | 0.7188824 | 0.270126 | 0.238546 | 0.251178 |
| 0.6 | 0.297206 | 1.0 | 0.5783236 | 0.270126 | 0.238546 | 0.257494 |
| 0.9 | 0.297206 | 1.0 | 0.3674854 | 0.294774 | 0.960976 | 0.361394 |

| Test Case 2 - Traffic Distribution 2 - Alpha = 0.6 | | | | |
|---|---|---|---|---|
| | no VP | VPDBA algorithm | difference | percent difference |
| loss: | 0.297206 | 0.270126 | 0.02708 | 9.1115 |
| setup: | 1.0 | 0.238546 | 0.761454 | 76.1454 |
| network performance: | 0.5783236 | 0.257494 | 0.320830 | 55.4758 |

| Test Case 3 - Traffic Distribution 1 | | | | | | |
|---|---|---|---|---|---|---|
| | no VP | | | VPDBA algorithm result | | |
| alpha | loss probability | setup probability ance | overall perform- | loss probability | setup probability | overall perform- ance |
| 0.1 | 0.426907 | 1.0 | 0.9426907 | 0.445535 | 0.340384 | 0.350899 |
| 0.4 | 0.426907 | 1.0 | 0.7707628 | 0.443863 | 0.365446 | 0.396813 |
| 0.6 | 0.426907 | 1.0 | 0.6561442 | 0.443863 | 0.365446 | 0.412496 |
| 0.9 | 0.426907 | 1.0 | 0.4842163 | 0.399459 | 0.864811 | 0.445994 |

| Test Case 3 - Traffic Distribution 1 - Alpha = 0.6 | | | | |
|---|---|---|---|---|
| | no VP | VPDBA algorithm | difference | percent difference |
| loss: | 0.426907 | 0.443863 | 0.016956 | 3.820097 |
| setup: | 1.0 | 0.365446 | 0.634554 | 63.4554 |
| network performance: | 0.6561442 | 0.412496 | 0.243648 | 37.13333 |

| Test Case 3 - Traffic Distribution 2 | | | | | | |
|---|---|---|---|---|---|---|
| | no VP | | | VPDBA algorithm result | | |
| alpha | loss probability | setup probability | overall perform-ance | loss probability | setup probability | overall perform-ance |
| 0.1 | 0.002301 | 1.0 | 0.902301 | 0.0665417 | 0.226174 | 0.210211 |
| 0.4 | 0.002301 | 1.0 | 0.600920 | 0.0636039 | 0.233324 | 0.165436 |
| 0.6 | 0.002301 | 1.0 | 0.401381 | 0.0286559 | 0.48189 | 0.20995 |
| 0.9 | 0.002301 | 1.0 | 0.102071 | 0.0113238 | 0.586639 | 0.0688554 |

| Test Case 3 - Traffic Distribution 2 - Alpha = 0.6 | | | | |
|---|---|---|---|---|
| | no VP | VPDBA algorithm | difference | percent difference |
| loss: | 0.002301 | 0.0286559 | 0.0263549 | 91.9702 |
| setup: | 1.0 | 0.48189 | 0.51811 | 51.8110 |
| network performance: | 0.401381 | 0.20995 | 0.191431 | 47.6931 |

Table 7.1: The approximated performance measures and percent differences for the loss probability, setup probability and network performance for each test case

## 7.2.4 Conclusion

In all test cases considered the probability of setup was greatly reduced by including VPs when compared to networks that did not include VPs. The VPDBA solutions found by our algorithm greatly improved the cost of call setup in all cases considered. An improvement of 50% to 80% was observed in the probability of call setup in the test cases. We would

expect the networks containing VPs to have smaller setup probabilities when compared to networks without VPs since calls using the VPs will not incur setup costs.

In addition, the VPDBA solutions found using our algorithm did not have significantly different probabilities of call loss when compared to the networks without VPs. In two of the six cases, we saw that the solution including VPs had slightly higher probabilities of call loss. We would expect that networks using VPs would have slightly higher probability of call loss than networks that contained no VPs. Networks that contain no VPs assign resources as they are requested. In such a network there is never a time in which a call is denied service when resources are actually available to carry the call. In contrast, networks containing VPs may deny service to an incoming call if it does not have a VP associated with it and all shared resources are in use or if the VP associated with the call is currently being used to capacity and all shared resources are in use. In four of the six test cases, we observed a small improvement in call loss probability when using the VPDBA solution found by our algorithm. We would not expect that the probability of call loss would be reduced by including VPs in the network. The improvements are very small and we attribute them to the approximation method. Our approximation method only finds approximate solutions for performance measures associated with a network. Thus the loss probabilities for the network without VPs and the VPDBA solution found by our algorithm are likely very close in value. It is likely that the VPDBA solution actually has a slightly higher rather than lower probability of call loss. Because our approximation method is only intended to enable optimization, we can tolerate these small variations.

In all cases, the overall network performance for the network using the VPDBA solution was better than the overall network performance for the network without VPs. We would

expect this improvement because the reduction in call setup was so significant for the test cases, it is not surprising that the network performance was an improvement for all test cases and all alpha values.

We have shown that with our algorithm the probability of call setup can be significantly reduced while maintaining the probability of call loss similar to networks without VPs (the minimum possible value).

## 7.3 Comparison of Our Algorithm to Other Algorithms

We now compare the performance of VPDBA solutions produced by our algorithm to those produced by the three algorithms described in Chapter 6, using the same performance metric $F = \alpha P(loss) + (1 - \alpha)P(setup)$. We also compare the loss probability, setup probability, fraction of traffic carried by each solution and the fraction of streams carried by each solution.

### 7.3.1 Test Cases

Two traffic distributions were considered for each of three network topologies. Connectivity and arrival rates were selected at random, shown in Table F.1 in Appendix F (Section F.1.1).

### 7.3.2 Results

We compare the overall network performance, probability of loss, and probability of setup with various $\alpha$ values for each network test case for our algorithm and the other algorithms. We also compare the fraction of traffic carried, the sum of the estimated amount of each traffic source carried divided by the sum of the total amount of traffic offered to the network.

In addition, we also compare the number of streams that are able to transmit some information divided by the total number of streams in the network. Each following graph represents the results for a single performance measure for a single network test case as labeled. Results for other test cases are shown in Figures F.1- F.25 in Appendix F Section F.1.2.1. The VPDBA results are given in tabular form in Section F.1.2.2 of Appendix F.
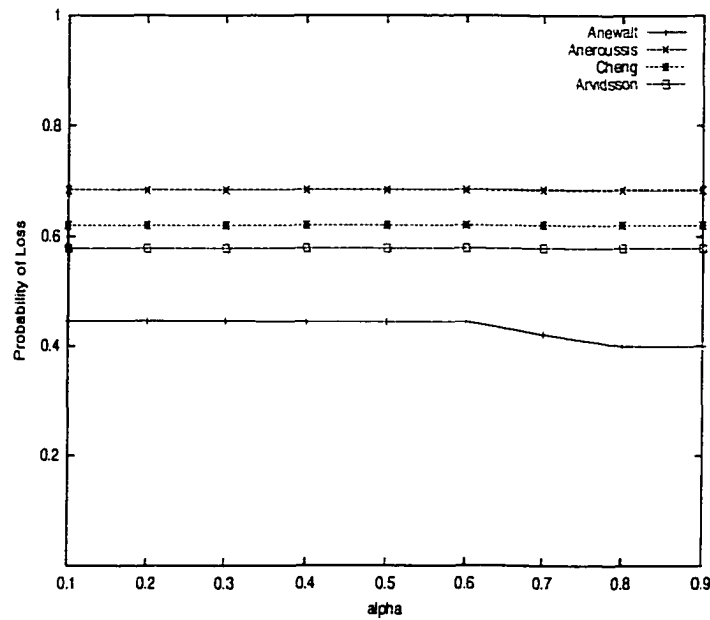


**Figure 7.1**: Comparing the Probability of Loss for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 3, Traffic Distribution 1.
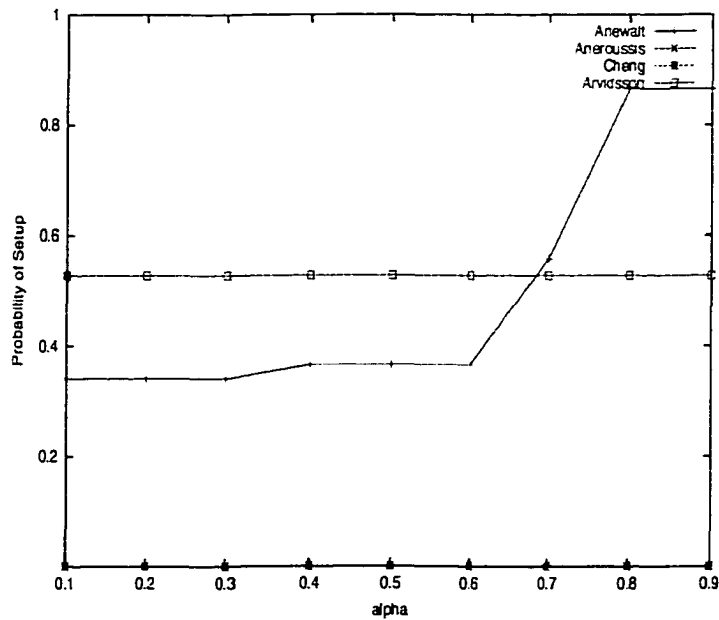
**Figure 7.2**: Comparing the Probability of Setup for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 3, Traffic Distribution 1.
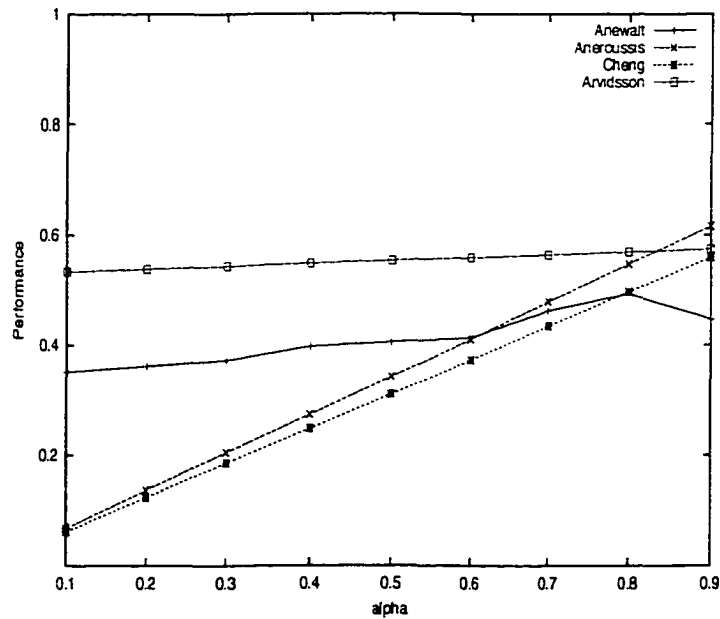


**Figure 7.3**: Comparing the Overall Performance of the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 3, Traffic Distribution 1.
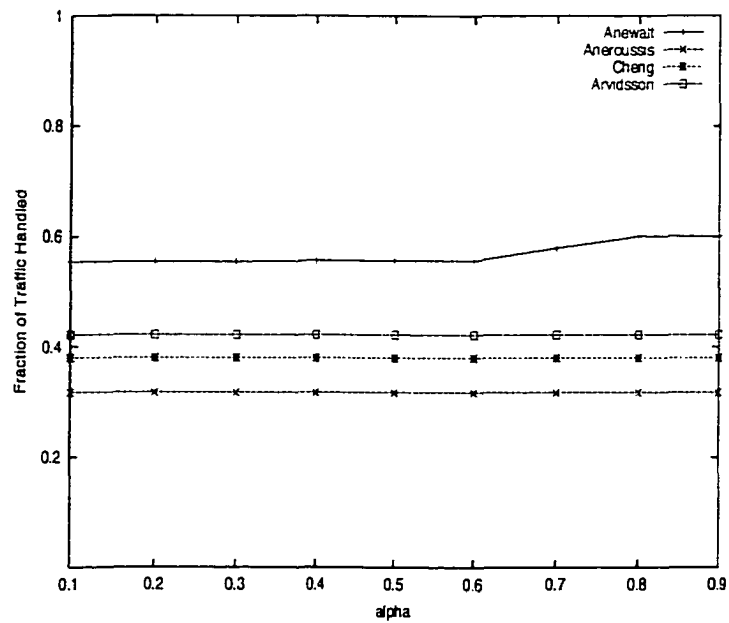
**Figure 7.4**: Comparing the Fraction of Traffic carried by the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 3, Traffic Distribution 1.
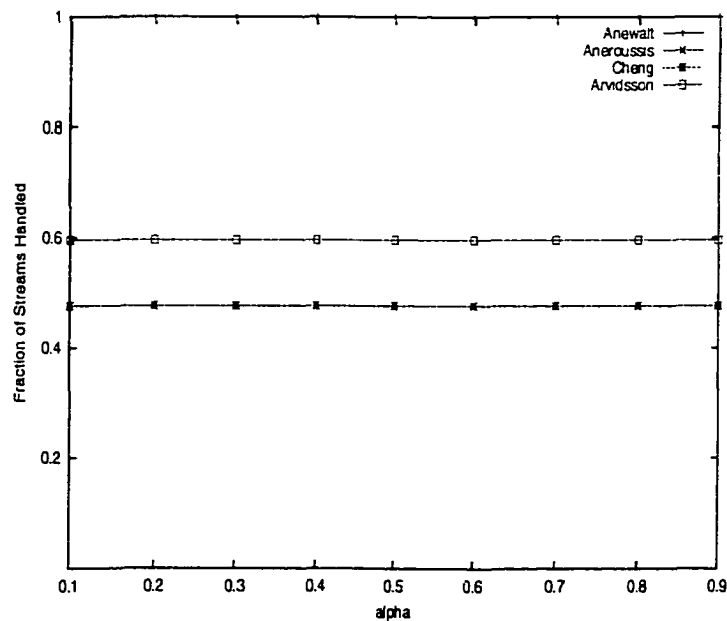


**Figure 7.5**: Comparing the Fraction of Streams carried by the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 3, Traffic Distribution 1.

## 7.3.3 Conclusions

We discuss the results for each of the performance measures individually and then make general conclusions about the results.

### 7.3.3.1 Loss Probability

In all test cases, we observe that the VPDBA solution produced by our algorithm has a lower loss probability for all $\alpha$ values than the layout produced by all other algorithms, and this value is close to the optimal value found in non-VP solutions.

We also observe that in contrast to other algorithms, loss probabilities associated with our algorithm change with the value of $\alpha$ because our algorithm uses $\alpha$ in determining layout. As such we would expect that the loss probability will decrease as the value of $\alpha$ increases. We observe the expected behavior in four of the six test cases, the exceptions being Test Case 1, Traffic Distribution 1 and Test Case 2, Traffic Distribution 2. We suspect that that the slight increase in loss probability as the value of $\alpha$ increases in these test cases is a result of the estimated nature of our performance measures. Another possible explanation for the unexpected behavior is that the heuristic VPDBA algorithm finds a local optimal solution for this particular $\alpha$ value. This issue will be considered in future study.

### 7.3.3.2 Setup Probability

In all cases the setup probability associated with the solution produced by our algorithm increases with the value of $\alpha$, as the setup component of the overall network performance measure is weighted less. The other algorithms do not consider the value of $\alpha$ in their algorithm, thus the probability of setup does not vary with $\alpha$.

Except for cases of large $\alpha$ values, the setup probability associated with the layout produced by our algorithm lies between the best and worst probability of call setup associated with the layouts produced by the other algorithms in all test cases. The algorithm developed by Cheng [16] consistently has the lowest setup probability, a value of zero in all test cases because the capacity was completely allocated to VPs. The same is true of Aneroussis' algorithm [4]; in all cases considered except Test Case 1, Traffic Distribution 2, this algorithm terminated only when all resources had been allocated to VPs leaving no shared bandwidth in the network to incur a setup cost. This type of solution will have a low setup probability; however as we have argued, the choice to allocate all resources to VPs has a negative impact on the loss probability and fairness of the network.

The layouts produced by the algorithm developed by Arvidsson [7] always had a setup probability that exceeded the setup probability associated with our solution except in cases of large $\alpha$ values. When $\alpha$ is large, our algorithm places less emphasis on the setup probability and greater emphasis on the probability of loss. Thus, layouts produced by our algorithm with large $\alpha$ values will have larger probability of call setup and smaller probability of call loss than layouts produced with moderate or small $\alpha$ values. In all test cases, we observe a steep rise in the setup probability as the value of $\alpha$ increases and the weight placed on the setup measure of performance decreases. Note that the use of the $\alpha$ parameter allows the network manager to have an additional level of control over the VPDBA layout produced by the algorithm by specifying the importance of each of the performance measures in the overall network design.

All of the algorithms provide a reduction in call setup probability when compared to a network that does not contain VPs and must have a setup probability of 100%. The

differences in setup probability observed by the four algorithms can be attributed to the algorithms themselves. Each uses a particular starting point for the VPDBA solution and a particular statistic to determine when to stop searching for a better VPDBA solution. It appears that the algorithm developed by Arvidsson emphasizes reducing the loss probability of the final layout, while the algorithms developed by Cheng and Aneroussis emphasize reducing the setup probability of the final layout. Our algorithm offers the most flexibility by allowing the network manager to choose whether the setup probability or loss probability should be minimized at a particular time. The manager can also choose to use a moderate value of $\alpha$ that will produce a solution that takes both measures into account.

### 7.3.3.3 Overall Performance

Recall that the overall network performance is calculated using the equation

$$F = \alpha P(loss) + (1 - \alpha)P(setup)$$

For five of the six network topology and traffic distributions considered, there exists an $\alpha$ value at which our algorithm will yield the best overall network performance. In Test Case 2, Traffic Distribution 2 our algorithm did not produce the best overall network performance, but did come very close to the best network performance observed for several $\alpha$ values for this test case and traffic distribution. We note that in general, our algorithm produces results that are better in terms of overall performance when the value of $\alpha$ is high. We would expect this because a large value of $\alpha$ places greater emphasis on loss probability and less emphasis on setup probability. Our algorithm always produced layouts with lower loss probability than the other algorithms because our layouts contain shared bandwidth. The overall performance measure results show that there is a $\alpha$ value at which our network

will provide better overall performance in terms of combined loss and setup probability for many network situations.

VPDBA solutions found using our algorithm contains shared bandwidth, thus it is unlikely that the setup probability for a layout produced by our algorithm would ever be zero. Some calls would necessarily be carried by the shared bandwidth and incur a setup cost unless the amount of resources available to the network were so large as to allow every stream to be designated a VP with enough bandwidth to handle all calls. In this case the shared bandwidth would be present, but unused. It is unrealistic to assume that a network will contain such an overabundance of resources.

In four of the six test cases considered, the line representing the overall network performance contains a small increase followed by a small decrease as the value of $\alpha$ increases. This occurs because the VPDBA solution itself is discrete. The solution must contain a number of VPs, their placement and the capacity allocated to each VP. As such, we would not expect the curve representing the performance of the solutions over a range of $\alpha$ values to be completely smooth. While the probability of loss is decreasing as $\alpha$ increases and the probability of setup is decreasing as $\alpha$ increases, at particular positions in the graph it happens that the overall network performance is slightly higher at some $\alpha$ value. Again, this is due to the nature of the VPDBA solution. We note that the increases are small rather than dramatic.

### 7.3.3.4 Fraction of Traffic

The *fraction of traffic* is the sum of the estimated amount of each traffic source carried divided by the sum of the total amount of traffic offered to the network. In all cases, the

layout produced by our algorithm carries a larger fraction of traffic than that carried by the layouts produced by the other algorithms. This is expected as our algorithm produces layouts with guaranteed shared bandwidth that can be used to carry calls not allocated VP resources or calls for full VPs. Two of the other algorithms, Cheng [16] and Aneroussis [4], produced layouts in which all resources were distributed among certain traffic streams in the network. This allowed the layouts to provide very good service (with low loss probability and setup probability) for certain traffic streams, but also prohibited them from providing service to other streams in the network. A network that only provides through VPs is inflexible and impractical. Such a network divides resources among the streams that have the largest arrival rates and ignores all other network traffic completely. A network that provides a measure of fairness to the network traffic, allowing all source-destination pairs to receive some service is more practical and useful.

### 7.3.3.5   Fraction of Streams

The *fraction of streams* is the number of streams that are able to transmit some information divided by the total number of streams in the network. This performance measure gives us a another measure of the fairness of the network.

Layouts produced by our algorithm always provide some level of service to 100% of the streams within the network. In Test Case 2, Traffic Distributions 1 and 2, the layouts produced by the algorithm developed by Aneroussis also provide service to all streams within the network. We have already seen that our algorithm produced results that were better in terms of fraction of traffic carried and loss probability when compared to the layouts produced by Aneroussis' algorithm in these cases. We have also seen that in the

other test cases considered, the layout produced by Aneroussis' algorithm did not provide service to 100% of the traffic streams. For example in Test Case 3, Traffic Distribution 1, the layout produced by Aneroussis' algorithm handled approximately 50% fewer streams than the layout produced by our algorithm. The algorithm developed by Aneroussis does not guarantee that 100% of the traffic will receive some level of service, while our algorithm ensures that this will always be the case by including shared bandwidth.

The other algorithms produce layouts in which many streams are not allocated resources as VPs. If the layout does not contain shared bandwidth, then all streams not designated as VPs and provided resources, will be completely denied service. Again, a network that only provides service through VPs divides resources among the streams that have the largest arrival rates and ignores all other network traffic completely.

### 7.3.3.6 Observations

We have observed that if there are sufficient resources available so that every (or almost every) stream can be assigned enough bandwidth so as to handle all incoming calls for that stream, then the previous algorithms stand to produce a layout that will outperform the layout produced by our algorithm by allocating all resources to VPs. The resulting layout would have a setup probability of zero and the loss probability would be modest because the previous algorithms do a good job of allocating resources in a proportional manner, giving the most bandwidth to the most heavily loaded streams. On the other hand, our algorithm would yield a solution with a setup probability greater than zero because it must contain shared bandwidth. In terms of traffic and streams carried, there would be little or no difference observed between the layout produced by our algorithm and the layouts

produced by the previous algorithms. This is because enough resources exist so as to handle all calls from all streams at all times. In this situation, it is hard to argue that our algorithm would provide superior performance.

As mentioned previously, the more interesting scenario has limited resources. In this case, it is interesting and useful to determine how and where the resources should be distributed, a task accomplished by our algorithm. When the resources are limited, our algorithm will out perform the others, as it uses shared bandwidth to provide a lower loss probability and some service to each source-destination pair while still using VPs to provide a reduced setup probability compared to a solution with no VPs. The ability to guarantee some service to every stream is important considering that in some cases other algorithms provided service to 50% fewer streams than our algorithm.

The observations made in studying the general network case support the results of our earlier experiments with line networks in Chapter 2 and Chapter 6.

## 7.4  Summary

In this chapter, we have shown that our algorithm for finding good solutions to the VPDBA problem in a line network can be extended and applied to general networks. We considered several test cases and showed that the VPDBA solutions produced by our algorithm had better overall network performance and reduced costs of call setup when compared to networks containing no VPs. At the same time the VPDBA solutions produced by our algorithm had similar probabilities of call loss when compared to networks containing no VPs.

We have also demonstrated that our algorithm finds VPDBA solutions that lead to improved network performance when compared to VPDBA solutions produced by previous algorithms. Our algorithm produces layouts that are fair and provide service to a greater amount of network traffic and provide some level of service to all network streams. In addition, our algorithm produces layouts that have a lower probability of call loss. The explicit inclusion of shared bandwidth by our algorithm is used to achieve these positive characteristics and other positive attributes for the network.

We conclude that VPDBA solutions containing shared bandwidth strike a balance between efficiency in terms of loss probability and setup probability and at the same time provide other network benefits. Our algorithm is guaranteed to produce VPDBA solutions that contain shared bandwidth and therefore solutions to the VPDBA problem produced by our algorithm have these positive qualities.

# Chapter 8

# Summary and Conclusion

Broadband high-speed networks, such as B-ISDN, are expected to play a dominant role in the future of networking due to their capability to service a variety of traffic types with very different bandwidth requirements such as video, voice and data. To increase network efficiency in B-ISDN and other such connection oriented networks, the concept of a *virtual path* (VP) has been proposed in the literature. Using VPs can potentially reduce call setup delays, simplify hardware, provide network flexibility, and reduce loss in the event of link or node failure.

In order to use VPs efficiently, two problems must be solved. The VPs must be placed within the network such that network performance is optimized and the network link capacity must be divided among the VPs such that the network performance is optimized. Most of the previous work aimed at solving these problems has focused on one problem in isolation of the other. At the same time, previous research efforts that have considered the joint solution of these problems have considered only restricted cases. In addition, these efforts have not explicitly considered the benefits of sharing bandwidth among VPs in the network.

148

## 8.1   Research Summary

We have formally studied the issue of shared bandwidth and made the following contributions to the study of the VPDBA problem.

- **We introduced a new scheme for sharing bandwidth that improves the network performance in terms of loss probability.** We have shown that capacity allocations using our shared bandwidth scheme have lower loss probability than capacity allocations that do not allow shared bandwidth. We have shown that as the number of traffic streams using the shared bandwidth pool increases, the benefits of sharing increase. We discussed other positive implications of shared bandwidth including expanding the solution space to the VPDBA problem, improving performance for networks with bursty traffic and improving connectivity.

- **We created a network model and VPDBA problem formulation using two separate performance measures that results in additional flexibility.** Our network model can be used to define the Virtual Path Distribution and Bandwidth Allocation (VPDBA) problem. Within our problem definition, we consider a weighted combination of two separate performance measures, the cost of setup and the probability of call loss. By adjusting the weight, a network manager can specify which component should be given greater emphasis during the design process.

- **We developed a new performance measure approximation method that facilitates the simple calculation of the network performance measures of interest.** An approximation method such as this is necessary because finding the exact performance measures using the non-product form Markov chain equivalent to

the network is impractical as the size and complexity of the network increases. Using a simulation to determine performance measures for networks has similar drawbacks in terms of time requirements and complexity. We use a decomposition method in which we divide the original system into a series of smaller systems, each of which can be easily solved by writing the corresponding Markov chain and solving the balance equations for the smaller system. Our approximation method then combines the results from the smaller chains to obtain an approximate solution for the original larger system. We have shown that the performance measure approximation method yields good results for the types of networks studied.

- **We studied the VPDBA problem in the context of simple network topologies, gained intuition for the problem, and demonstrated that our approximation method could be used to successfully calculate network performance and determine a good VPDBA solution.**

  1. As a starting point, we studied networks with a line topology containing a single VP. We showed that the best capacity allocation found using our performance measure approximation method and the best capacity allocation found through simulation were equivalent in all test cases considered. In additional experiments using simple line networks, we showed that the approximation method and simulations chose the same optimal VPDBA solution for such a network.

  2. We also considered a single node with multiple VP and non-VP streams and showed that our performance measure approximation method could be used to find a good solution to the VPDBA problem.

3. We then combined these two special case models into a more general line network model. In the general model, each node may have many VP and non-VP streams passing through it. We compared the performance measure results obtained through our approximation method to those obtained through simulation and showed that our approximation method is effective. We also demonstrated that our performance measure approximation method is accurate enough to be successfully used in an algorithm for solving the VPDBA problem.

• **We developed an algorithm for solving the VPDBA problem and demonstrated its effectiveness in general line networks.** Our VPDBA algorithm guarantees that all solutions will contain shared bandwidth. In the previous algorithms, there was no guarantee that each stream would be given some level of service. Streams could be completely denied service if the final solution produced by the algorithm assigned all of the available capacity on one or more links along their path to other VPs. We have discussed the advantages of including shared bandwidth in a VPDBA solution. We compared the network performance our the solutions determined by our algorithm to solutions determined by previous VPDBA algorithms. We showed that our algorithm produces solution with loss and setup probabilities comparable to those produced by previous algorithms. Our algorithm provides an increased level of fairness by providing service to a greater amount of the offered traffic than previous algorithms. Previous algorithms make no guarantee about the presence of shared bandwidth in the solution. In the cases considered, the other algorithms only carried traffic on VPs - all traffic that was not assigned to a VP was lost. As a result their

call setup probability was always essentially 0. Our algorithm only services a portion of traffic on VPs. This increases the probability of call setup, but also increases the number of streams handled by the network.

- **We expanded our algorithm to apply to general network topologies and demonstrated the effectiveness of our algorithm.** We have shown that our algorithm finds VPDBA solutions that lead to improved network performance when compared to VPDBA solutions produced by previous algorithms. Our algorithm produces layouts that are fair and provide service to a greater amount of network traffic and provide some level of service to all network streams. The benefits produced by our algorithm are particularly dramatic when network resources are limited.

  1. Our algorithm produces layouts that have a lower probability of call loss. We observed that VPDBA solutions found by our algorithm had very similar probabilities of call loss when compared to layouts with no VPs. This implies that the probability of call loss is near-optimal for the VPDBA solutions produced by our algorithm. The explicit inclusion of shared bandwidth by our algorithm is used to achieve these positive characteristics and other positive attributes for the network.

  2. We have shown that our algorithm will outperform the previous algorithms in cases where resources are limited. This is accomplished through the guarantee that layouts produced by our algorithm will include shared bandwidth, thus providing a lower loss probability and guarantee of service to each source-destination pair while still using VPs to provide a reduced setup probability compared to

a solution with no VPs. Other algorithms can not make such claims about the solutions produced.

3. We have confirmed the findings from our earlier experiments by studying the behavior of general networks containing VPs and shared bandwidth. In situations in which there is competition for resources, the concept of shared bandwidth is valuable and results in a VPDBA layout with lower loss probability and a degree of fairness by allowing all streams to receive some service. These improvements are made at the sacrifice of increased probability of call setup.

4. We observed that even when shared bandwidth is used, thus elevating the setup probability for the VPDBA solution, the observed setup is still smaller than it would be for a network containing no VPs. We conclude that VPDBA solutions containing shared bandwidth strike a balance between efficiency in terms of loss probability and setup probability and at the same time provide other network benefits. Our algorithm is guaranteed to produce VPDBA solutions that contain shared bandwidth and therefore solutions to the VPDBA problem produced by our algorithm have these positive qualities.

## 8.2 Future Research Directions

Some of the directions for future research are described below.

- **Experiments with Bursty Traffic**

    In Chapter 2 we discussed the positive qualities associated with networks containing shared bandwidth. We noted that networks containing bursty traffic would partic-

ularly benefit from the presence of shared bandwidth. A network containing bursty traffic could typically require frequent redistribution of VP bandwidth to adapt to the changes in the bursty traffic if a VPDBA algorithm that did not guarantee shared bandwidth were used. However, in a VP solution containing shared bandwidth, fewer redistributions of bandwidth should be required because bursts could potentially be handled by shared bandwidth rather than individual VPs. The use of VPDBA solutions in networks with bursty traffic should reduce the necessity of frequent VP reconfigurations. Experiments using our VPDBA algorithm and networks with bursty traffic distributions could be done to confirm this hypothesis.

- **Further Experiments With Call Loss Probability**

  In Chapter 7 we observed that in two of the test cases considered (Test Case 1, Traffic Distribution 1 and Test Case 2, Traffic Distribution 2) the probability of call loss increased as the value of $\alpha$ increased. This result was unexpected because the value of $\alpha$ represents the weight or importance of the call loss in the calculation of overall network performance. We would expect that as $\alpha$ increases, the value of the loss probability would decrease. Further study may provide intuition or explanation of why the loss probability increased with the value of $\alpha$ in these two cases.

- **Further Experiments with Fairness Schemes**

  We have shown that our algorithm for solving the VPDBA problem produces solutions with a degree of fairness when compared to solutions produced by previous algorithms. Our algorithm does not currently impose a completely fair distribution of resources. Some streams may lose more calls than other depending on their arrival rates and the

resource distribution. Further study could modify our VPDBA algorithm to provide increased fairness by evenly distributing losses among traffic streams throughout the network.

- **On-The-Fly Dynamic VPDBA Algorithm**

  Our algorithm for solving the VPDBA problem allows some flexibility for changes in the traffic distribution and network layout through shared bandwidth. If the arrival rate of a particular traffic stream increases unexpectedly, the calls will likely overflow the capacity assigned to the associated VP, but will be handled by the shared bandwidth. If drastic changes occur in the arrival rates of many streams simultaneously or for an extended period of time, then the VPDBA algorithm will need to be reapplied to the network in order to update the position of the VPs and resource distribution to the VPs. It may be possible to extend our algorithm to make intelligent decisions about the allocation of bandwidth to the VPs at the time at which the changes in traffic occur, rather than waiting for the algorithm to process the network information from scratch.

- **Study of Alternate Performance Measures**

  We have developed one performance measure approximation method that provides an estimate of network performance measures of interest. We have noted that our performance measure is not perfect, but does provide enough accuracy for use in our heuristic algorithm. Further study may yield performance measure approximation methods that are more accurate.

## 8.3   Conclusion

Networks are playing a more dominant role in every day life. As network use increases, the study of types of networks including B-ISDN becomes increasingly important. B-ISDN networks have two main performance measures of interest, the cost of call setup and the probability of call loss. One way of reducing the cost of call setup in B-ISDN networks is to include VPs within the network. In order to use VPs efficiently, the VPs must be placed within the network and capacity must be allocated to the VPs. Most previous algorithms focus on doing one of these tasks rather than considering the solution to the joint problem. In addition the previous work has not explicitly considered the benefits of including shared bandwidth in the VPDBA solution. We have formally studied the VPDBA problem and explicitly considered the benefits of shared bandwidth. We have developed a heuristic algorithm for solving the joint problem of VP distribution and bandwidth allocation that guarantees that the VPDBA solution produced will contain shared bandwidth. We have demonstrated the effectiveness of our algorithm in comparison to previous algorithms. We have also explicitly shown that including shared bandwidth in the VPDBA solution can have many positive implications for the performance of the network.

# Appendix A

# Shared Bandwidth

## A.1    Shared Bandwidth Experiment

### A.1.1    Test Cases

| number traffic sources | arrival rate | System 1 0 shared servers number of servers per VP | System 2 30 shared servers number of servers per VP | System 3 20 shared servers number of servers per VP | System 4 12 shared servers number of servers per VP |
|---|---|---|---|---|---|
| 2 | 30 | 30 | 15 | 20 | 24 |
| 3 | 20 | 20 | 10 | – | 16 |
| 4 | 15 | 15 | – | 10 | 12 |
| 5 | 12 | 12 | 6 | – | – |
| 6 | 10 | 10 | 5 | – | 8 |
| 10 | 6 | 6 | 3 | 4 | – |
| 12 | 5 | 5 | – | – | 4 |
| 15 | 4 | 4 | 2 | – | – |
| 20 | 3 | 3 | – | 2 | – |
| 30 | 2 | 2 | 1 | – | – |

Table A.1: Description of parameter data for Test Case 1.

157

| | | System 1 0 shared servers | System 2 24 shared servers | System 3 16 shared servers | System 4 12 shared servers |
|---|---|---|---|---|---|
| number of traffic sources | arrival rate | number of servers per VP | number of servers per VP | number of servers per VP | number of servers per VP |
| 2 | 24 | 24 | 12 | 16 | 18 |
| 4 | 12 | 12 | 6 | 12 | 9 |
| 6 | 8 | 8 | 4 | – | 6 |
| 8 | 6 | 6 | 3 | 4 | – |
| 12 | 4 | 4 | 2 | – | 3 |
| 16 | 3 | 3 | – | 2 | – |
| 24 | 2 | 2 | 1 | – | – |

Table A.2: Description of parameter data for Test Case 2.

## A.1.2    Sharing Experiment Loss and Setup Probability Results

| number of VPs | P(loss system 1) | P(loss system 2) | P(loss system 3) | P(loss system 4) |
|---|---|---|---|---|
| 2 | 0.132418 +/- 0.000053 | 0.101004 +/- 0.000054 | 0.104897 +/- 0.000055 | 0.110864 +/- 0.000054 |
| 3 | 0.158861 +/- 0.000050 | 0.106194 +/- 0.000053 | – | 0.125006 +/- 0.000056 |
| 4 | 0.180281 +/- 0.000049 | – | 0.122920 +/- 0.000056 | 0.137973 +/- 0.000055 |
| 5 | 0.198546 +/- 0.000048 | 0.116597 +/- 0.000054 | – | – |
| 6 | 0.214627 +/- 0.000046 | 0.121607 +/- 0.000055 | – | 0.160992 +/- 0.000054 |
| 10 | 0.264907 +/- 0.000044 | 0.139619 +/- 0.000055 | 0.167402 +/- 0.000053 | – |
| 12 | 0.284898 +/- 0.000043 | – | – | 0.213836 +/- 0.000052 |
| 15 | 0.310671 +/- 0.000039 | 0.158737 +/- 0.000057 | – | – |
| 20 | 0.346164 +/- 0.000037 | – | 0.219051 +/- 0.000054 | – |
| 30 | 0.400018 +/- 0.000034 | 0.201171 +/- 0.000056 | – | – |

Table A.3: P(loss) results for Test Case 1.

| number of VPs | P(loss system 1) | P(loss system 2) | P(loss system 3) | P(loss system 4) |
|---|---|---|---|---|
| 2 | 0 +/- 0.000000 | 0.474041 +/- 0.000035 | 0.307318 +/- 0.000033 | 0.179902 +/- 0.000026 |
| 3 | 0 +/- 0.000000 | 0.482978 +/- 0.000033 | – | 0.190856 +/- 0.000025 |
| 4 | 0 +/- 0.000000 | – | 0.327614 +/- 0.000030 | 0.199035 +/- 0.000025 |
| 5 | 0 +/- 0.000000 | 0.497446 +/- 0.000032 | – | – |
| 6 | 0 +/- 0.000000 | 0.503562 +/- 0.000031 | – | 0.211284 +/- 0.000024 |
| 10 | 0 +/- 0.000000 | 0.523621 +/- 0.000029 | 0.362918 +/- +/- 0.000027 | – |
| 12 | 0 +/- 0.000000 | – | – | 0.234644 +/- 0.000026 |
| 15 | 0 +/- 0.000000 | 0.542792 +/- 0.000029 | – | – |
| 20 | 0 +/- 0.000000 | – | 0.397412 +/- 0.000030 | – |
| 30 | 0 +/- 0.000000 | 0.582743 +/- 0.000027 | – | – |

Table A.4: P(setup) results for Test Case 1.

| number of VPs | P(loss system 1) | P(loss system 2) | P(loss system 3) | P(loss system 4) |
|---|---|---|---|---|
| 2 | 0.146455 +/- 0.000051 | 0.112478 +/- 0.000054 | 0.117125 +/- 0.000053 | 0.120891 +/- 0.000054 |
| 4 | 0.198529 +/- 0.000048 | 0.124921 +/- 0.000053 | 0.138015 +/- 0.000053 | 0.147567 +/- 0.000053 |
| 6 | 0.235510 +/- 0.000045 | 0.136507 +/- 0.000056 | – | 0.170334 +/- 0.000051 |
| 8 | 0.264929 +/- 0.000041 | 0.147191 +/- 0.000053 | 0.172968 +/- 0.000053 | – |
| 12 | 0.310717 +/- 0.000039 | 0.165724 +/- 0.000055 | – | 0.222603 +/- 0.000051 |
| 16 | 0.346174 +/- 0.000037 | – | 0.223558 +/- 0.000053 | – |
| 24 | 0.400009 +/- 0.000035 | 0.207302 +/- 0.000054 | – | – |

Table A.5: P(loss) for Test Case 2.

| number of VPs | P(loss system 1) | P(loss system 2) | P(loss system 3) | P(loss system 4) |
|---|---|---|---|---|
| 2 | 0 +/- 0.000000 | 0.473453 +/- 0.000035 | 0.307363 +/- 0.000032 | 0.227317 +/- 0.000029 |
| 4 | 0 +/- 0.000000 | 0.492692 +/- 0.000033 | 0.330143 +/- 0.000029 | 0.249676 +/- 0.000027 |
| 6 | 0 +/- 0.000000 | 0.507337 +/- 0.000030 | – | 0.264438 +/- 0.000027 |
| 8 | 0 +/- 0.000000 | 0.519403 +/- 0.000030 | 0.358604 +/- 0.000028 | – |
| 12 | 0 +/- 0.000000 | 0.538933 +/- 0.000029 | – | 0.293415 +/- 0.000028 |
| 16 | 0 +/- 0.000000 | – | 0.393925 +/- 0.000028 | – |
| 24 | 0 +/- 0.000000 | 0.579512 +/- 0.000029 | – | – |

Table A.6: P(setup) for Test Case 2.

## A.1.3  Sharing Experiment Percent Difference Values

Test Case 1 - System 1 and 2

| number of VPs | f | g | h - $\alpha = 0.1$ | h - $\alpha = 0.5$ | h - $\alpha = 0.9$ |
|---|---|---|---|---|---|
| 2 | 23.723361 | 100.000000 | 96.968017 | 76.972585 | 13.832563 |
| 3 | 33.152882 | 100.000000 | 96.432492 | 73.036567 | 0.623817 |
| 5 | 41.274566 | 100.000000 | 95.677779 | 67.665782 | 13.436293 |
| 6 | 43.340307 | 100.000000 | 95.388001 | 65.668963 | 17.271204 |
| 10 | 47.295088 | 100.000000 | 94.540485 | 60.058651 | 25.332622 |
| 15 | 48.905112 | 100.000000 | 93.840616 | 55.715159 | 29.492221 |
| 30 | 49.709513 | 100.000000 | 92.654638 | 48.971698 | 33.522936 |

**Table A.7**: Percent difference values for f, g, and h for Test Case 1, System 1 and 2

Test Case 1 - System 1 and 3

| number of VPs | f | g | h - $\alpha = 0.1$ | h - $\alpha = 0.5$ | h - $\alpha = 0.9$ |
|---|---|---|---|---|---|
| 2 | 20.783428 | 100.000000 | 95.387352 | 67.876472 | 4.765017 |
| 4 | 31.817551 | 100.000000 | 94.130419 | 59.985040 | 11.625986 |
| 10 | 36.807257 | 100.000000 | 92.285005 | 50.047707 | 21.585227 |
| 20 | 36.720456 | 100.000000 | 90.880243 | 43.846752 | 23.964396 |

**Table A.8**: Percent difference values for f, g, and h for Test Case 1, System 1 and 3

Test Case 1 - System 1 and 4

| number of VPs | f | g | h - $\alpha = 0.1$ | h - $\alpha = 0.5$ | h - $\alpha = 0.9$ |
|---|---|---|---|---|---|
| 2 | 16.277243 | 100.000000 | 92.345701 | 54.458912 | 1.181780 |
| 3 | 21.311083 | 100.000000 | 91.378947 | 49.705568 | 7.962167 |
| 4 | 23.467809 | 100.000000 | 90.655568 | 46.505424 | 11.200848 |
| 6 | 24.989866 | 100.000000 | 89.594085 | 42.347344 | 14.051820 |
| 12 | 24.942962 | 100.000000 | 87.749653 | 36.474759 | 15.791773 |

**Table A.9**: Percent difference values for f, g, and h for Test Case 1, System 1 and 4

Test Case 2 - System 1 and 2

| number of VPs | f | g | h - $\alpha = 0.1$ | h - $\alpha = 0.5$ | h - $\alpha = 0.9$ |
|---|---|---|---|---|---|
| 2 | 23.199618 | 100.000000 | 96.651351 | 75.004736 | 11.284498 |
| 4 | 37.076699 | 100.000000 | 95.645481 | 67.855437 | 9.502110 |
| 6 | 42.037705 | 100.000000 | 94.991855 | 63.421264 | 18.102086 |
| 8 | 44.441341 | 100.000000 | 94.505620 | 60.256318 | 22.657601 |
| 12 | 46.664006 | 100.000000 | 93.805632 | 55.905213 | 27.391985 |
| 24 | 48.175666 | 100.000000 | 92.623720 | 49.160920 | 32.078473 |

Table A.10: Percent difference values for f, g, and h for Test Case 2, System 1 and 2

Test Case 2 - System 1 and 3

| number of VPs | f | g | h - $\alpha = 0.1$ | h - $\alpha = 0.5$ | h - $\alpha = 0.9$ |
|---|---|---|---|---|---|
| 2 | 20.019801 | 100.000000 | 94.920757 | 65.499249 | 3.193574 |
| 4 | 30.481189 | 100.000000 | 93.614998 | 57.593590 | 12.004012 |
| 8 | 34.711564 | 100.000000 | 92.208896 | 50.161220 | 19.671728 |
| 16 | 35.420338 | 100.000000 | 90.814944 | 43.937890 | 22.776568 |

Table A.11: Percent difference values for f, g, and h for Test Case 2, System 1 and 3

Test Case 2 - System 1 and 4

| number of VPs | f | g | h - $\alpha = 0.1$ | h - $\alpha = 0.5$ | h - $\alpha = 0.9$ |
|---|---|---|---|---|---|
| 2 | 17.455191 | 100.000000 | 93.240780 | 57.940369 | 0.209317 |
| 4 | 25.669801 | 100.000000 | 91.709481 | 50.023285 | 11.696136 |
| 6 | 27.674409 | 100.000000 | 90.765313 | 45.831378 | 15.198505 |
| 12 | 28.358281 | 100.000000 | 89.148434 | 39.785628 | 17.865882 |

Table A.12: Percent difference values for f, g, and h for Test Case 2, System 1 and 4

# Appendix B

# Line Network Base Case

## B.1   Simulation

### B.1.1   Simulation Validity Experiment

#### B.1.1.1   Test Cases

| Test Case 1 | Test Case 2 | Test Case 3 | Test Case 4 | Test Case 5 |
|---|---|---|---|---|
| $\lambda_0 = 0.6666666667$ | $\lambda_0 = 0.5$ | $\lambda_0 = 0.5$ | $\lambda_0 = 0.5$ | $\lambda_0 = 4.0$ |
| $\lambda_1 = 0.5$ | $\lambda_1 = 0.5$ | $\lambda_1 = 5.0$ | $\lambda_1 = 0.25$ | $\lambda_1 = 0.25$ |
| $\lambda_{vp} = 0.4$ | $\lambda_{vp} = 0.5$ | $\lambda_{vp} = 10.0$ | $\lambda_{vp} = 2.0$ | $\lambda_{vp} = 0.5$ |
| $\mu = 1.0$ | $\mu = 1.0$ | $\mu = 1.0$ | $\mu = 1.0$ | $\mu = 1.0$ |
| Test Case 6 | Test Case 7 | Test Case 8 | Test Case 9 | Test Case 10 |
| $\lambda_0 = 3.0$ | $\lambda_0 = 0.25$ | $\lambda_0 = 0.1$ | $\lambda_0 = 10.0$ | $\lambda_0 = 0.75$ |
| $\lambda_1 = 0.6$ | $\lambda_1 = 10.0$ | $\lambda_1 = 0.1$ | $\lambda_1 = 10.0$ | $\lambda_1 = 1.25$ |
| $\lambda_{vp} = 3.0$ | $\lambda_{vp} = 0.5$ | $\lambda_{vp} = 0.5$ | $\lambda_{vp} = 10.0$ | $\lambda_{vp} = 0.5$ |
| $\mu = 1.0$ | $\mu = 1.0$ | $\mu = 1.0$ | $\mu = 1.0$ | $\mu = 1.0$ |

Table B.1: Test cases to establish validity of simulation.

163

## B.1.1.2  State Probability Results

|  | Test Case 1 | Test Case 2 | Test Case 3 | Test Case 4 | Test Case 5 |
|---|---|---|---|---|---|
| P[0,0,0] | 0.273224 | 0.275862 | 0.00502513 | 0.103896 | 0.103896 |
| P[0,0,1] | 0.136612 | 0.137931 | 0.0251256 | 0.025974 | 0.025974 |
| P[0,1,0] | 0.182149 | 0.137931 | 0.00251256 | 0.0519481 | 0.415584 |
| P[0,1,1] | 0.0910747 | 0.0689655 | 0.0125628 | 0.012987 | 0.103896 |
| P[1,0,0] | 0.10929 | 0.137931 | 0.0502513 | 0.207792 | 0.0519481 |
| P[1,0,1] | 0.0546448 | 0.0689655 | 0.251256 | 0.0519481 | 0.012987 |
| P[1,1,0] | 0.0728597 | 0.0689655 | 0.0251256 | 0.103896 | 0.207792 |
| P[1,1,1,VP] | 0.0364299 | 0.0344828 | 0.125628 | 0.025974 | 0.0519481 |
| P[1,1,1,SH] | 0.0437158 | 0.0689655 | 0.502513 | 0.415584 | 0.025974 |

|  | Test Case 6 | Test Case 7 | Test Case 8 | Test Case 9 | Test Case 10 |
|---|---|---|---|---|---|
| P[0,0,0] | 0.0289017 | 0.0479042 | 0.484262 | 0.000698812 | 0.162437 |
| P[0,0,1] | 0.017341 | 0.479042 | 0.0484262 | 0.00698812 | 0.203046 |
| P[0,1,0] | 0.0867052 | 0.011976 | 0.0484262 | 0.00698812 | 0.121827 |
| P[0,1,1] | 0.0520231 | 0.11976 | 0.00484262 | 0.0698812 | 0.152284 |
| P[1,0,0] | 0.0867052 | 0.0239521 | 0.242131 | 0.00698812 | 0.0812183 |
| P[1,0,1] | 0.0520231 | 0.239521 | 0.0242131 | 0.0698812 | 0.101523 |
| P[1,1,0] | 0.260116 | 0.00598802 | 0.0242131 | 0.0698812 | 0.0609137 |
| P[1,1,1,VP] | 0.156069 | 0.0598802 | 0.00242131 | 0.698812 | 0.0761421 |
| P[1,1,1,SH] | 0.260116 | 0.011976 | 0.121065 | 0.0698812 | 0.0406091 |

Table B.2: Markov Chain results for the test cases in Table B.1

# B.2   Approximate Markov Model

## B.2.1   Test Cases

| Test Case 1 | Test Case 2 |
|---|---|
| *number of nodes* $= 2$ | *number of nodes* $= 2$ |
| $\lambda_0 = 1/3 = 0.3333333333$ | $\lambda_0 = 2/3 = 0.6666666667$ |
| $\lambda_1 = 2.0$ | $\lambda_1 = 2/3 = 0.6666666667$ |
| $\lambda_{vp} = 2/3 = 0.6666666667$ | $\lambda_{vp} = 2/3 = 0.6666666667$ |
| $V_0 = 2$ | $V_0 = 2$ |
| $V_1 = 2$ | $V_1 = 2$ |
| $SH_0 = 3$ | $SH_0 = 3$ |
| $SH_1 = 3$ | $SH_1 = 3$ |
| **Test Case 3** | **Test Case 4** |
| *number of nodes* $= 3$ | *number of nodes* $= 5$ |
| $\lambda_0 = 1/1.7 = 0.5882352941$ | $\lambda_0 = 1/0.8 = 1.25$ |
| $\lambda_1 = 1/1.5 = 0.6666666667$ | $\lambda_1 = 1/1.5 = 0.6666666667$ |
| $\lambda_2 = 1/1.2 = 0.8333333333$ | $\lambda_2 = 1/3.0 = 0.3333333333$ |
| $\lambda_{vp} = 1/2 = 0.5$ | $\lambda_3 = 1/0.2 = 5.0$ |
| $\mu = 1.0$ | $\lambda_4 = 1/7.0 = 0.1428571429$ |
|  | $\lambda_{vp} = 1/2 = 0.5$ |
|  | $\mu = 1.0$ |
| $V_0 = 2$ | $V_0 = 1$ |
| $V_1 = 2$ | $V_1 = 1$ |
| $V_2 = 2$ | $V_2 = 1$ |
| $SH_0 = 3$ | $V_3 = 1$ |
| $SH_1 = 3$ | $V_4 = 1$ |
| $SH_2 = 3$ | $SH_0 = 5$ |
|  | $SH_1 = 5$ |
|  | $SH_2 = 5$ |
|  | $SH_3 = 5$ |
|  | $SH_4 = 5$ |

| Test Case 5 | Test Case 6 |
|---|---|
| number of nodes $= 2$ | number of nodes $= 5$ |
| $\lambda_0 = 1/0.2 = 5.0$ | $\lambda_0 = 2.0$ |
| $\lambda_1 = 1/0.9 = 1.1111111111$ | $\lambda_1 = 1/0.9 = 1.1111111111$ |
| $\lambda_{vp} = 1/2 = 0.5$ | $\lambda_2 = 1/1.9 = 0.5263157895$ |
| $\mu = 1.0$ | $\lambda_3 = 4.0$ |
| $V_0 = 1$ | $\lambda_4 = 1/1.2 = 0.83333333333$ |
| $V_1 = 1$ | $\lambda_{vp} = 3.0$ |
| $SH_0 = 2$ | $\mu = 1.0$ |
| $SH_1 = 2$ | $V_0 = 2$ |
| | $V_1 = 2$ |
| | $V_2 = 2$ |
| | $V_3 = 2$ |
| | $V_4 = 2$ |
| | $SH_0 = 3$ |
| | $SH_1 = 3$ |
| | $SH_2 = 3$ |
| | $SH_3 = 3$ |
| | $SH_4 = 3$ |
| **Test Case 7** | **Test Case 8** |
| number of nodes $= 5$ | number of nodes $= 5$ |
| $\lambda_0 = 4.0$ | $\lambda_0 = 1/0.75 = 1.3333333333$ |
| $\lambda_1 = 4.0$ | $\lambda_1 = 1/0.9 = 1.1111111111$ |
| $\lambda_2 = 4.0$ | $\lambda_2 = 1/0.82 = 1.219512195$ |
| $\lambda_3 = 4.0$ | $\lambda_3 = 1/0.75 = 1.3333333333$ |
| $\lambda_4 = 4.0$ | $\lambda_4 = 1/0.9 = 1.1111111111$ |
| $\lambda_{vp} = 2.0$ | $\lambda_{vp} = 1/0.1 = 10.0$ |
| $\mu = 1.0$ | $\mu = 1.0$ |
| $V_0 = 2$ | $V_0 = 2$ |
| $V_1 = 2$ | $V_1 = 2$ |
| $V_2 = 2$ | $V_2 = 2$ |
| $V_3 = 2$ | $V_3 = 2$ |
| $V_4 = 2$ | $V_4 = 2$ |
| $SH_0 = 3$ | $SH_0 = 3$ |
| $SH_1 = 3$ | $SH_1 = 3$ |
| $SH_2 = 3$ | $SH_2 = 3$ |
| $SH_3 = 3$ | $SH_3 = 3$ |
| $SH_4 = 3$ | $SH_4 = 3$ |

| Test Case 9 | Test Case 10 |
|---|---|
| *number of nodes = 7* | *number of nodes = 4* |
| $\lambda_0 = 2.0$ | $\lambda_0 = 4.0$ |
| $\lambda_1 = 4.0$ | $\lambda_1 = 2.0$ |
| $\lambda_2 = 4/3 = 1.33333333333$ | $\lambda_2 = 4/3 = 1.3333333333$ |
| $\lambda_3 = 1/1.5 = 0.66666666667$ | $\lambda_3 = 1/0.2 = 5.0$ |
| $\lambda_4 = 1/1.25 = 0.8$ | $\lambda_{vp} = 1/0.1 = 10.0$ |
| $\lambda_5 = 3/2 = 1.5$ | $\mu = 1.0$ |
| $\lambda_6 = 3.0$ | $V_0 = 3$ |
| $\lambda_{vp} = 2.0$ | $V_1 = 3$ |
| $\mu = 1.0$ | $V_2 = 3$ |
| $V_0 = 2$ | $V_3 = 3$ |
| $V_1 = 2$ | $SH_0 = 2$ |
| $V_2 = 2$ | $SH_1 = 2$ |
| $V_3 = 2$ | $SH_2 = 2$ |
| $V_4 = 2$ | $SH_3 = 2$ |
| $V_5 = 2$ | |
| $V_6 = 2$ | |
| $SH_0 = 3$ | |
| $SH_1 = 3$ | |
| $SH_2 = 3$ | |
| $SH_3 = 3$ | |
| $SH_4 = 3$ | |
| $SH_5 = 3$ | |
| $SH_6 = 3$ | |

| Test Case 11 | Test Case 12 |
|---|---|
| *number of nodes* $= 10$ | *number of nodes* $= 10$ |
| $\lambda_0 = 0.25$ | $\lambda_0 = 3.0$ |
| $\lambda_1 = 0.5$ | $\lambda_1 = 2.0$ |
| $\lambda_2 = 0.8$ | $\lambda_2 = 0.5$ |
| $\lambda_3 = 2.0$ | $\lambda_3 = 0.25$ |
| $\lambda_4 = 0.75$ | $\lambda_4 = 1.25$ |
| $\lambda_5 = 0.25$ | $\lambda_5 = 4.0$ |
| $\lambda_6 = 0.6$ | $\lambda_6 = 0.25$ |
| $\lambda_7 = 3.0$ | $\lambda_7 = 0.5$ |
| $\lambda_8 = 0.25$ | $\lambda_8 = 1.25$ |
| $\lambda_9 = 4.0$ | $\lambda_9 = 0.75$ |
| $\lambda_{vp} = 3.0$ | $\lambda_{vp} = 10.0$ |
| $\mu = 1.0$ | $\mu = 1.0$ |
| $V_0 = 2$ | $V_0 = 2$ |
| $V_1 = 2$ | $V_1 = 2$ |
| $V_2 = 2$ | $V_2 = 2$ |
| $V_3 = 2$ | $V_3 = 2$ |
| $V_4 = 2$ | $V_4 = 2$ |
| $V_5 = 2$ | $V_5 = 2$ |
| $V_6 = 2$ | $V_6 = 2$ |
| $V_7 = 2$ | $V_7 = 2$ |
| $V_8 = 2$ | $V_8 = 2$ |
| $V_9 = 2$ | $V_9 = 2$ |
| $SH_0 = 3$ | $SH_0 = 3$ |
| $SH_1 = 3$ | $SH_1 = 3$ |
| $SH_2 = 3$ | $SH_2 = 3$ |
| $SH_3 = 3$ | $SH_3 = 3$ |
| $SH_4 = 3$ | $SH_4 = 3$ |
| $SH_5 = 3$ | $SH_5 = 3$ |
| $SH_6 = 3$ | $SH_6 = 3$ |
| $SH_7 = 3$ | $SH_7 = 3$ |
| $SH_8 = 3$ | $SH_8 = 3$ |
| $SH_9 = 3$ | $SH_9 = 3$ |

| Test Case 13 | | |
|---|---|---|
| *number of nodes* $= 15$ | | |
| $\lambda_0 = 3.0$ | $V_0 = 2$ | $SH_0 = 3$ |
| $\lambda_1 = 2.0$ | $V_1 = 2$ | $SH_1 = 3$ |
| $\lambda_2 = 0.5$ | $V_2 = 2$ | $SH_2 = 3$ |
| $\lambda_3 = 0.25$ | $V_3 = 2$ | $SH_3 = 3$ |
| $\lambda_4 = 1.25$ | $V_4 = 2$ | $SH_4 = 3$ |
| $\lambda_5 = 4.0$ | $V_5 = 2$ | $SH_5 = 3$ |
| $\lambda_6 = 0.25$ | $V_6 = 2$ | $SH_6 = 3$ |
| $\lambda_7 = 0.5$ | $V_7 = 2$ | $SH_7 = 3$ |
| $\lambda_8 = 1.25$ | $V_8 = 2$ | $SH_8 = 3$ |
| $\lambda_9 = 0.75$ | $V_9 = 2$ | $SH_9 = 3$ |
| $\lambda_{10} = 0.5$ | $V_{10} = 2$ | $SH_{10} = 3$ |
| $\lambda_{11} = 0.6$ | $V_{11} = 2$ | $SH_{11} = 3$ |
| $\lambda_{12} = 2.0$ | $V_{12} = 2$ | $SH_{12} = 3$ |
| $\lambda_{13} = 6.0$ | $V_{13} = 2$ | $SH_{13} = 3$ |
| $\lambda_{14} = 0.2$ | $V_{14} = 2$ | $SH_{14} = 3$ |
| $\lambda_{vp} = 5.0$ | | |
| $\mu = 1.0$ | | |

Table B.3: Test cases for approximation results.

## B.2.2 Simulation Results

| Test Case | overall P(loss) |
|---|---|
| 1 | 0.155723 +/- 0.000031 |
| 2 | 0.026349 +/- 0.000013 |
| 3 | 0.029519 +/- 0.000013 |
| 4 | 0.197246 +/- 0.000037 |
| 5 | 0.574883 +/- 0.000037 |
| 6 | 0.357552 +/- 0.000042 |
| 7 | 0.449853 +/- 0.000040 |
| 8 | 0.566914 +/- 0.000044 |
| 9 | 0.301966 +/- 0.000040 |
| 10 | 0.659053 +/- 0.000038 |
| 11 | 0.336325 +/- 0.000040 |
| 12 | 0.534212 +/- 0.000041 |
| 13 | 0.403717 +/- 0.000040 |

**Table B.4**: Simulation results for the test cases in Table B.3.

## B.2.3 Approximation Method Results - Subsystems with Original Arrival Rates

| Test Case 1 | Subsystem 0 | Subsystem 1 |
|---|---|---|
| $p[state2]$ | 0.0648661 | |
| $p[state5]$ | 0.0379788 | |
| $p[state8]$ | 0.011991 | |
| $p[state9]$ | 0.003088 | 0.112946 |
| $p[state10]$ | 0.00309695 | 0.0779166 |
| $p[state11]$ | 0.00281114 | 0.0309805 |
| $P_0^v$ | 0.11764704 | |
| $p(loss_i)$ | 0.00899609 | 0.2218431 |
| $p(loss_{up})$ | 0.00281114 | 0.0309805 |

| Test Case 2 | | |
|---|---|---|
| | Subsystem 0 | Subsystem 1 |
| $p[state2]$ | 0.0468319 | |
| $p[state5]$ | 0.0431208 | |
| $p[state8]$ | 0.0206157 | |
| $p[state9]$ | 0.0149447 | 0.0149447 |
| $p[state10]$ | 0.0118586 | 0.0118586 |
| $p[state11]$ | 0.00707866 | 0.00707866 |
| $P_0^v$ | 0.11764706 | |
| $p(loss_i)$ | 0.03388196 | 0.03388196 |
| $p(loss_{vp})$ | 0.00707866 | 0.00707866 |

| Test Case 3 | | | |
|---|---|---|---|
| | Subsystem 0 | Subsystem 1 | Subsystem 2 |
| $p[state2]$ | 0.0355065 | | |
| $p[state5]$ | 0.0274267 | | |
| $p[state8]$ | 0.0109114 | | |
| $p[state9]$ | 0.0123274 | 0.0159321 | 0.026971 |
| $p[state10]$ | 0.0070354 | 0.00826424 | 0.0146417 |
| $p[state11]$ | 0.00307838 | 0.0024931 | 0.00543009 |
| $P_0^v$ | 0.07692298 | | |
| $p(loss_i)$ | 0.02244118 | 0.02668944 | 0.04704279 |
| $p(loss_{vp})$ | 0.00307838 | 0.0024931 | 0.00543009 |

| Test Case 4 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $p[state1]$ | 0.0716426 | | | | |
| $p[state3]$ | 0.109821 | | | | |
| $p[state5]$ | 0.0848404 | | | | |
| $p[state7]$ | 0.0440654 | | | | |
| $p[state9]$ | 0.0173598 | | | | |
| $p[state10]$ | 0.00649288 | 0.00075568 | 0.0000855279 | 0.192994 | 0.0000137713 |
| $p[state11]$ | 0.00560436 | 0.00100697 | 0.000197402 | 0.105027 | 0.0000512148 |
| $P_0^v$ | 0.33333356 | | | | |
| $p(loss_i)$ | 0.01209724 | 0.00176265 | 0.0002829299 | 0.298021 | 0.0000649861 |
| $p(loss_{vp})$ | 0.00560436 | 0.00100697 | 0.000197402 | 0.105027 | 0.0000512148 |

| Test Case 5 | | |
|---|---|---|
| | Subsystem 0 | Subsystem 1 |
| $p[state1]$ | 0.0158814 | |
| $p[state3]$ | 0.0854714 | |
| $p[state4]$ | 0.451698 | 0.162108 |
| $p[state5]$ | 0.231981 | 0.100369 |
| $P_0^v$ | 0.3333338 | |
| $p(loss_i)$ | 0.683679 | 0.262477 |
| $p(loss_{vp})$ | 0.231981 | 0.100369 |

| Test Case 6 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $p[state2]$ | 0.0246787 | | | | |
| $p[state5]$ | 0.0892361 | | | | |
| $p[state8]$ | 0.171965 | | | | |
| $p[state9]$ | 0.0335396 | 0.0203 | 0.0117592 | 0.0565717 | 0.0161235 |
| $p[state10]$ | 0.119279 | 0.0818453 | 0.0556031 | 0.18139 | 0.069349 |
| $p[state11]$ | 0.243532 | 0.192094 | 0.150861 | 0.319704 | 0.173205 |
| $P_0^v$ | 0.5294118 | | | | |
| $p(loss_i)$ | 0.3963506 | 0.2942393 | 0.2182233 | 0.5576657 | 0.2586775 |
| $p(loss_{vp})$ | 0.243532 | 0.192094 | 0.150861 | 0.319704 | 0.173205 |

| Test Case 7 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $p[state2]$ | 0.00909703 | | | | |
| $p[state5]$ | 0.0462568 | | | | |
| $p[state8]$ | 0.121363 | | | | |
| $p[state9]$ | 0.0928268 | 0.0928268 | 0.0928268 | 0.0928268 | 0.0928268 |
| $p[state10]$ | 0.194118 | 0.194118 | 0.194118 | 0.194118 | 0.194118 |
| $p[state11]$ | 0.223283 | 0.223283 | 0.223283 | 0.223283 | 0.223283 |
| $P_0^v$ | 0.39999983 | | | | |
| $p(loss_i)$ | 0.5102278 | 0.5102278 | 0.5102278 | 0.5102278 | 0.510227 |
| $p(loss_{vp})$ | 0.223283 | 0.223283 | 0.223283 | 0.223283 | 0.223283 |

| Test Case 8 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $p[state2]$ | 0.00470939 | | | | |
| $p[state5]$ | 0.0394083 | | | | |
| $p[state8]$ | 0.178797 | | | | |
| $p[state9]$ | 0.00795959 | 0.0077442 | 0.00785017 | 0.00795959 | 0.0077442 |
| $p[state10]$ | 0.0957427 | 0.0941479 | 0.0949331 | 0.0957427 | 0.0941479 |
| $p[state11]$ | 0.596758 | 0.591879 | 0.594285 | 0.596758 | 0.591879 |
| $P_0^v$ | 0.81967269 | | | | |
| $p(loss_i)$ | 0.70046029 | 0.6937711 | 0.69706827 | 0.70046029 | 0.6937711 |
| $p(loss_{vp})$ | 0.596758 | 0.591879 | 0.594285 | 0.596758 | 0.591879 |

| Test Case 9 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $p[state2]$ | 0.0286281 | | | | |
| $p[state5]$ | 0.0854074 | | | | |
| $p[state8]$ | 0.133848 | | | | |
| $p[state9]$ | 0.0492607 | 0.0928268 | 0.0306952 | 0.0132124 | 0.0163754 |
| $p[state10]$ | 0.112594 | 0.194118 | 0.0765808 | 0.0402096 | 0.0471363 |
| $p[state11]$ | 0.152116 | 0.223283 | 0.11718 | 0.0767313 | 0.0850919 |
| $P_0^v$ | 0.3999995 | | | | |
| $p(loss_i)$ | 0.3139707 | 0.5102278 | 0.224456 | 0.1301533 | 0.1486036 |
| $p(loss_{vp})$ | 0.152116 | 0.223283 | 0.11718 | 0.0767313 | 0.0850919 |
| | Subsystem 5 | Subsystem 6 | | | |
| $p[state9]$ | 0.0353956 | 0.0736786 | | | |
| $p[state10]$ | 0.0858519 | 0.158509 | | | |
| $p[state11]$ | 0.126512 | 0.193079 | | | |
| $p(loss_i)$ | 0.2477595 | 0.4252666 | | | |
| $p(loss_{vp})$ | 0.126512 | 0.193079 | | | |

| Test Case 10 | | | | |
|---|---|---|---|---|
| | Subsystem 0 | Subsystem 1 | Subsystem 2 | Subsystem 3 |
| $p[state3]$ | 0.00923877 | | | |
| $p[state7]$ | 0.101439 | | | |
| $p[state8]$ | 0.00302275 | 0.00258254 | 0.00235148 | 0.00317972 |
| $p[state9]$ | 0.0315837 | 0.0280531 | 0.0260794 | 0.0328724 |
| $p[state10]$ | 0.168679 | 0.156187 | 0.148095 | 0.173345 |
| $p[state11]$ | 0.621387 | 0.598897 | 0.577381 | 0.629879 |
| $P_0^v$ | 0.73206477 | | | |
| $p(loss_i)$ | 0.82467245 | 0.78571964 | 0.75390688 | 0.83927612 |
| $p(loss_{vp})$ | 0.621387 | 0.598897 | 0.577381 | 0.629879 |

| Test Case 11 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $p[state2]$ | 0.0865361 | | | | |
| $p[state5]$ | 0.154033 | | | | |
| $p[state8]$ | 0.159149 | | | | |
| $p[state9]$ | 0.00826996 | 0.0114054 | 0.0156329 | 0.0335396 | 0.0149032 |
| $p[state10]$ | 0.0437441 | 0.0544446 | 0.0678453 | 0.119279 | 0.0655919 |
| $p[state11]$ | 0.129694 | 0.148882 | 0.17085 | 0.243532 | 0.167284 |
| $P_0^v$ | 0.5294121 | | | | |
| $p(loss_i)$ | 0.18170806 | 0.214732 | 0.2543282 | 0.3963506 | 0.2477791 |
| $p(loss_{vp})$ | 0.129694 | 0.148882 | 0.17085 | 0.243532 | 0.167284 |
| | Sub-system 5 | Sub-system 6 | Sub-system 7 | Sub-system 8 | Sub-system 9 |
| $p[state9]$ | 0.00826996 | 0.0127697 | 0.046367 | 0.00826996 | 0.056717 |
| $p[state10]$ | 0.0437441 | 0.0588695 | 0.154069 | 0.0437441 | 0.18139 |
| $p[state11]$ | 0.129694 | 0.156351 | 0.287117 | 0.129694 | 0.319704 |
| $p(loss_i)$ | 0.18170806 | 0.2279902 | 0.487553 | 0.18170806 | 0.557811 |
| $p(loss_{vp})$ | 0.129694 | 0.156351 | 0.287117 | 0.129694 | 0.319704 |

| Test Case 12 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $p[state2]$ | 0.00287919 | | | | |
| $p[state5]$ | 0.0293729 | | | | |
| $p[state8]$ | 0.15982 | | | | |
| $p[state9]$ | 0.00935799 | 0.00856335 | 0.00711422 | 0.00684048 | 0.00787966 |
| $p[state10]$ | 0.106034 | 0.100194 | 0.0894519 | 0.0873924 | 0.0951515 |
| $p[state11]$ | 0.6276 | 0.610219 | 0.577304 | 0.570802 | 0.594953 |
| $P_0^v$ | 0.81967209 | | | | |
| $p(loss_i)$ | 0.74299199 | 0.71897635 | 0.67387012 | 0.66503488 | 0.69798416 |
| $p(loss_{vp})$ | 0.6276 | 0.610219 | 0.577304 | 0.570802 | 0.594953 |
| | Sub-system 5 | Sub-system 6 | Sub-system 7 | Sub-system 8 | Sub-system 9 |
| $p[state9]$ | 0.0100357 | 0.00684048 | 0.00711422 | 0.00787966 | 0.00737866 |
| $p[state10]$ | 0.111022 | 0.0873924 | 0.0894519 | 0.0951515 | 0.0914297 |
| $p[state11]$ | 0.642282 | 0.570802 | 0.577304 | 0.594953 | 0.583483 |
| $p(loss_i)$ | 0.7633397 | 0.66503488 | 0.67387012 | 0.69798416 | 0.68229136 |
| $p(loss_{vp})$ | 0.642282 | 0.570802 | 0.577304 | 0.594953 | 0.583483 |

| Test Case 13 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $p[state2]$ | 0.00911322 | | | | |
| $p[state5]$ | 0.0557786 | | | | |
| $p[state8]$ | 0.181599 | | | | |
| $p[state9]$ | 0.024404 | 0.0198077 | 0.011416 | 0.00994458 | 0.0157862 |
| $p[state10]$ | 0.138626 | 0.119313 | 0.0822165 | 0.0751897 | 0.101983 |
| $p[state11]$ | 0.429185 | 0.393494 | 0.317903 | 0.301936 | 0.359656 |
| $P_0^v$ | 0.67567582 | | | | |
| $p(loss_i)$ | 0.592215 | 0.5326147 | 0.4115355 | 0.38707028 | 0.4774252 |
| $p(loss_{vp})$ | 0.429185 | 0.393494 | 0.317903 | 0.301936 | 0.359659 |
| | Sub-system 5 | Sub-system 6 | Sub-system 7 | Sub-system 8 | Sub-system 9 |
| $p[state9]$ | 0.0281543 | 0.00994458 | 0.011416 | 0.0157862 | 0.0128927 |
| $p[state10]$ | 0.154231 | 0.0751897 | 0.0822165 | 0.101983 | 0.0890496 |
| $p[state11]$ | 0.456991 | 0.301936 | 0.317903 | 0.359656 | 0.332807 |
| $p(loss_i)$ | 0.6393763 | 0.38707028 | 0.4115355 | 0.4774252 | 0.4347493 |
| $p(loss_{vp})$ | 0.456991 | 0.301936 | 0.317903 | 0.359656 | 0.332807 |
| | Sub-system 10 | Sub-system 11 | Sub-system 12 | Sub-system 13 | Sub-system 14 |
| $p[state9]$ | 0.011416 | 0.0120073 | 0.0198077 | 0.0336815 | 0.00965306 |
| $p[state10]$ | 0.0822165 | 0.0849756 | 0.119313 | 0.177312 | 0.073766 |
| $p[state11]$ | 0.317903 | 0.323989 | 0.393493 | 0.497203 | 0.298611 |
| $p(loss_i)$ | 0.4115355 | 0.4209719 | 0.5326137 | 0.7081965 | 0.38203006 |
| $p(loss_{vp})$ | 0.317903 | 0.323989 | 0.393493 | 0.497203 | 0.298611 |

Table B.5: The subsystem results for the test cases in Table B.3.

## B.2.4  Approximation Method Results - Subsystems with Thinned Arrival Rates

| Test Case 1 | | |
| --- | --- | --- |
| | Subsystem 0 | Subsystem 1 |
| $\lambda'_{vp}$ | 0.6666666667 | 0.6647925733 |
| $p[state2]$ | 0.0648661 | |
| $p[state5]$ | 0.0379788 | |
| $p[state8]$ | 0.011991 | |
| $p[state9]$ | 0.003088 | 0.113122 |
| $p[state10]$ | 0.00309695 | 0.0778069 |
| $p[state11]$ | 0.00281114 | 0.0308389 |
| $P_0^v$ | 0.11764704 | |
| $p(loss_i)$ | 0.00899609 | 0.2217678 |
| $p(loss_{vp})$ | 0.00281114 | 0.0308389 |

| Test Case 2 | | |
| --- | --- | --- |
| | Subsystem 0 | Subsystem 1 |
| $\lambda'_{vp}$ | 0.6666666667 | 0.66194756 |
| $p[state2]$ | 0.0468319 | |
| $p[state5]$ | 0.0431208 | |
| $p[state8]$ | 0.0206157 | |
| $p[state9]$ | 0.0149447 | 0.0149796 |
| $p[state10]$ | 0.0118586 | 0.0117814 |
| $p[state11]$ | 0.00707866 | 0.00696881 |
| $P_0^v$ | 0.11764706 | |
| $p(loss_i)$ | 0.03388196 | 0.03372981 |
| $p(loss_{vp})$ | 0.00707866 | 0.00696881 |

| Test Case 3 | | | |
|---|---|---|---|
| | Subsystem 0 | Subsystem 1 | Subsystem 2 |
| $\lambda'_{vp}$ | 0.5 | 0.49846081 | 0.4965914823 |
| $p[state2]$ | 0.0355065 | | |
| $p[state5]$ | 0.0274267 | | |
| $p[state8]$ | 0.0109114 | | |
| $p[state9]$ | 0.0123274 | 0.0164724 | 0.0270393 |
| $p[state10]$ | 0.0070354 | 0.00917697 | 0.0145653 |
| $p[state11]$ | 0.00307838 | 0.0037502 | 0.00535503 |
| $P_0^v$ | 0.07692298 | | |
| $p(loss_i)$ | 0.02244118 | 0.02939957 | 0.04695963 |
| $p(loss_{vp})$ | 0.00307838 | 0.0037502 | 0.00535503 |

| Test Case 4 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $\lambda'_{vp}$ | 0.5 | 0.49719782 | 0.4967032325 | 0.4966071865 | 0.444721171 |
| $p[state1]$ | 0.0716426 | | | | |
| $p[state3]$ | 0.109821 | | | | |
| $p[state5]$ | 0.0848404 | | | | |
| $p[state7]$ | 0.0440654 | | | | |
| $p[state9]$ | 0.0173598 | | | | |
| $p[state10]$ | 0.00649288 | 0.000752166 | 0.0000843687 | 0.193394 | 0.0000090577 |
| $p[state11]$ | 0.00560436 | 0.00099475 | 0.000193367 | 0.104481 | 0.0000315444 |
| $P_0^v$ | 0.33333356 | | | | |
| $p(loss_i)$ | 0.01209724 | 0.001746916 | 0.0002777357 | 0.297875 | 0.0000406021 |
| $p(loss_{vp})$ | 0.00560436 | 0.00099475 | 0.000193367 | 0.104481 | 0.0000315444 |

| Test Case 5 | | |
|---|---|---|
| | Subsystem 0 | Subsystem 1 |
| $\lambda'_{vp}$ | 0.5 | 0.3840095 |
| $p[state1]$ | 0.0158814 | |
| $p[state3]$ | 0.0854714 | |
| $p[state4]$ | 0.451698 | 0.171135 |
| $p[state5]$ | 0.231981 | 0.0786689 |
| $P_0^v$ | 0.3333338 | |
| $p(loss_i)$ | 0.683679 | 0.2498039 |
| $p(loss_{vp})$ | 0.231981 | 0.0786689 |

| Test Case 6 | | | | | |
|---|---|---|---|---|---|
| | Subsystem 0 | Subsystem 1 | Subsystem 2 | Subsystem 3 | Subsystem 4 |
| $\lambda'_{vp}$ | 3.0 | 2.269404 | 1.981296354 | 1.84943593 | 1.467385004 |
| $p[state2]$ | 0.0246787 | | | | |
| $p[state5]$ | 0.0892361 | | | | |
| $p[state8]$ | 0.171965 | | | | |
| $p[state9]$ | 0.0335396 | 0.0232137 | 0.0101452 | 0.101334 | 0.0182344 |
| $p[state10]$ | 0.119279 | 0.069585 | 0.0328703 | 0.195195 | 0.0363453 |
| $p[state11]$ | 0.243532 | 0.126953 | 0.0665526 | 0.206577 | 0.0483241 |
| $P_0^v$ | 0.5294118 | | | | |
| $p(loss_i)$ | 0.3963506 | 0.2197517 | 0.1095681 | 0.503106 | 0.1029038 |
| $p(loss_{vp})$ | 0.243532 | 0.126953 | 0.0665526 | 0.206577 | 0.0483241 |

| Test Case 7 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $\lambda'_{vp}$ | 2.0 | 1.553434 | 1.286383161 | 1.10805315 | 0.9800630387 |
| $p[state2]$ | 0.00909703 | | | | |
| $p[state5]$ | 0.0462568 | | | | |
| $p[state8]$ | 0.121363 | | | | |
| $p[state9]$ | 0.0928268 | 0.121909 | 0.14633 | 0.166843 | 0.184216 |
| $p[state10]$ | 0.194118 | 0.195681 | 0.19307 | 0.188688 | 0.183635 |
| $p[state11]$ | 0.223283 | 0.171910 | 0.138629 | 0.115509 | 0.0986191 |
| $P_0^v$ | 0.39999983 | | | | |
| $p(loss_i)$ | 0.5102278 | 0.489500 | 0.478029 | 0.471040 | 0.4664701 |
| $p(loss_{vp})$ | 0.223283 | 0.17191 | 0.138629 | 0.115509 | 0.0986191 |

| Test Case 8 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $\lambda'_{vp}$ | 10.0 | 4.03242 | 2.904156949 | 2.350554935 | 2.001962937 |
| $p[state2]$ | 0.00470939 | | | | |
| $p[state5]$ | 0.0394083 | | | | |
| $p[state8]$ | 0.178797 | | | | |
| $p[state9]$ | 0.00795959 | 0.0172968 | 0.0223655 | 0.0277143 | 0.024505 |
| $p[state10]$ | 0.0957427 | 0.0930737 | 0.0853959 | 0.0827073 | 0.0641935 |
| $p[state11]$ | 0.596758 | 0.279798 | 0.190624 | 0.148302 | 0.104337 |
| $P_0^v$ | 0.81967269 | | | | |
| $p(loss_i)$ | 0.70046029 | 0.3901685 | 0.2983854 | 0.2587236 | 0.1930355 |
| $p(loss_{vp})$ | 0.596758 | 0.279798 | 0.190624 | 0.148302 | 0.104337 |

| Test Case 9 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $\lambda'_{vp}$ | 2.0 | 1.695768 | 1.375469644 | 1.28583744 | 1.246902668 |
| $p[state2]$ | 0.0286281 | | | | |
| $p[state5]$ | 0.0854074 | | | | |
| $p[state8]$ | 0.133848 | | | | |
| $p[state9]$ | 0.0492607 | 0.111299 | 0.0389518 | 0.0129201 | 0.0177556 |
| $p[state10]$ | 0.112594 | 0.195776 | 0.0632496 | 0.0236734 | 0.0294222 |
| $p[state11]$ | 0.152116 | 0.188881 | 0.0651648 | 0.0302797 | 0.0336102 |
| $P_0^v$ | 0.3999995 | | | | |
| $p(loss_i)$ | 0.3139707 | 0.495956 | 0.1673662 | 0.0668725 | 0.080788 |
| $p(loss_{vp})$ | 0.152116 | 0.188881 | 0.0651648 | 0.0302797 | 0.0336102 |

| | Sub-system 5 | Sub-system 6 |
|---|---|---|
| $\lambda'_{vp}$ | 1.20499402 | 1.134094823 |
| $p[state9]$ | 0.0507586 | 0.126904 |
| $p[state10]$ | 0.0691443 | 0.149308 |
| $p[state11]$ | 0.0588378 | 0.0983241 |
| $p(loss_i)$ | 0.1787407 | 0.3745361 |
| $p(loss_{vp})$ | 0.0588378 | 0.0983241 |

| Test Case 10 | | | | |
|---|---|---|---|---|
| | Subsystem 0 | Subsystem 1 | Subsystem 2 | Subsystem 3 |
| $\lambda'_{vp}$ | 10.0 | 3.78613 | 2.736679128 | 2.294721869 |
| $p[state3]$ | 0.00923877 | | | |
| $p[state7]$ | 0.101439 | - | | |
| $p[state8]$ | 0.00302275 | 0.0211372 | 0.0286248 | 0.0852898 |
| $p[state9]$ | 0.0315837 | 0.0849413 | 0.084428 | 0.196675 |
| $p[state10]$ | 0.168679 | 0.179301 | 0.133497 | 0.229618 |
| $p[state11]$ | 0.621387 | 0.277183 | 0.161494 | 0.187957 |
| $P_0^v$ | 0.73206477 | | | |
| $p(loss_i)$ | 0.82467245 | 0.5625625 | 0.4080438 | 0.6995398 |
| $p(loss_{vp})$ | 0.621387 | 0.277183 | 0.161494 | 0.187957 |

| Test Case 11 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $\lambda'_{vp}$ | 3.0 | 2.610918 | 2.310735536 | 2.055186672 | 1.731803049 |
| $p[state2]$ | 0.0865361 | | | | |
| $p[state5]$ | 0.154033 | | | | |
| $p[state8]$ | 0.159149 | | | | |
| $p[state9]$ | 0.00826996 | 0.0108905 | 0.0161478 | 0.0480468 | 0.0152438 |
| $p[state10]$ | 0.0437441 | 0.0461804 | 0.0541605 | 0.113174 | 0.0381683 |
| $p[state11]$ | 0.129694 | 0.114972 | 0.110592 | 0.15735 | 0.0619186 |
| $P_0^v$ | 0.5294121 | | | | |
| $p(loss_i)$ | 0.18170806 | 0.1720429 | 0.1809003 | 0.3185708 | 0.1153307 |
| $p(loss_{vp})$ | 0.129694 | 0.114972 | 0.110592 | 0.15735 | 0.0619186 |
| | Sub-system 5 | Sub-system 6 | Sub-system 7 | Sub-system 8 | Sub-system 9 |
| $\lambda'_{vp}$ | 1.624572229 | 1.577109377 | 1.508287005 | 1.29639078 | 1.276530203 |
| $p[state9]$ | 0.00408577 | 0.0111061 | 0.0982813 | 0.00299408 | 0.147365 |
| $p[state10]$ | 0.0137517 | 0.0269899 | 0.156357 | 0.0082224 | 0.192893 |
| $p[state11]$ | 0.0292156 | 0.0436383 | 0.140488 | 0.0153199 | 0.137368 |
| $p(loss_i)$ | 0.04705307 | 0.0817343 | 0.3951263 | 0.02653638 | 0.477626 |
| $p(loss_{vp})$ | 0.0292156 | 0.0436383 | 0.140488 | 0.0153199 | 0.137368 |

| Test Case 12 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $\lambda'_{vp}$ | 10.0 | 3.724 | 2.594607624 | 2.299904306 | 2.134244729 |
| $p[state2]$ | 0.00287919 | | | | |
| $p[state5]$ | 0.0293729 | | | | |
| $p[state8]$ | 0.15982 | | | | |
| $p[state9]$ | 0.00935799 | 0.0270231 | 0.0108642 | 0.00639749 | 0.0273196 |
| $p[state10]$ | 0.106034 | 0.120678 | 0.0458176 | 0.0280841 | 0.0744993 |
| $p[state11]$ | 0.6276 | 0.303274 | 0.113583 | 0.0720289 | 0.124076 |
| $P_0^v$ | 0.81967209 | | | | |
| $p(loss_i)$ | 0.74299199 | 0.4509751 | 0.1702648 | 0.10651049 | 0.2258949 |
| $p(loss_{vp})$ | 0.6276 | 0.303274 | 0.113583 | 0.0720289 | 0.124076 |
| | Sub-system 5 | Sub-system 6 | Sub-system 7 | Sub-system 8 | Sub-system 9 |
| $lambda'_{vp}$ | 1.86943618 | 1.479038084 | 1.445852907 | 1.40037129 | 1.31169824 |
| $p[state9]$ | 0.100137 | 0.00358625 | 0.00832726 | 0.0349515 | 0.0157285 |
| $p[state10]$ | 0.195077 | 0.0111367 | 0.0196022 | 0.0588117 | 0.0283393 |
| $p[state11]$ | 0.208832 | 0.022437 | 0.0314566 | 0.0633211 | 0.0351292 |
| $p(loss_i)$ | 0.504046 | 0.03715995 | 0.05938606 | 0.1570843 | 0.079197 |
| $p(loss_{vp})$ | 0.208832 | 0.022437 | 0.0314566 | 0.0633211 | 0.0351292 |

| Test Case 13 | | | | | |
|---|---|---|---|---|---|
| | Sub-system 0 | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 |
| $\lambda'_{vp}$ | 5.0 | 2.854075 | 2.195457237 | 2.017712141 | 1.912599629 |
| $p[state2]$ | 0.00911322 | | | | |
| $p[state5]$ | 0.0557786 | | | | |
| $p[state8]$ | 0.181599 | | | | |
| $p[state9]$ | 0.024404 | 0.0352128 | 0.0101034 | 0.00546187 | 0.0290996 |
| $p[state10]$ | 0.138626 | 0.118713 | 0.036659 | 0.0217903 | 0.0701682 |
| $p[state11]$ | 0.429185 | 0.230764 | 0.0809604 | 0.0520949 | 0.104855 |
| $P_0^v$ | 0.67567582 | | | | |
| $p(loss_i)$ | 0.592215 | 0.3846898 | 0.1277228 | 0.07934707 | 0.2041228 |
| $p(loss_{vp})$ | 0.429185 | 0.230764 | 0.0809604 | 0.0520949 | 0.104855 |
| | Sub-system 5 | Sub-system 6 | Sub-system 7 | Sub-system 8 | Sub-system 9 |
| $\lambda'_{vp}$ | 1.712053995 | 1.385414637 | 1.359654376 | 1.322746422 | 1.298151407 |
| $p[state9]$ | 0.110174 | 0.00327647 | 0.0081415 | 0.0361607 | 0.0157576 |
| $p[state10]$ | 0.195745 | 0.00958807 | 0.0178334 | 0.0569213 | 0.0280318 |
| $p[state11]$ | 0.190788 | 0.0185939 | 0.0271451 | 0.0575462 | 0.0343771 |
| $p(loss_i)$ | 0.496707 | 0.03145844 | 0.05312 | 0.1506282 | 0.0781665 |
| $p(loss_{vp})$ | 0.190788 | 0.0185939 | 0.0271451 | 0.0575462 | 0.0343771 |
| | Sub-system 10 | Sub-system 11 | Sub-system 12 | Sub-system 13 | Sub-system 14 |
| $\lambda'_{vp}$ | 1.253524726 | 1.225579147 | 1.195257461 | 1.104399248 | 0.9480825784 |
| $p[state9]$ | 0.00793676 | 0.0107879 | 0.0764276 | 0.217907 | 0.001447 |
| $p[state10]$ | 0.0157539 | 0.019364 | 0.0988265 | 0.242694 | 0.0031328 |
| $p[state11]$ | 0.0222936 | 0.0247407 | 0.0760156 | 0.14154 | 0.0051388 |
| $p(loss_i)$ | 0.04598426 | 0.0548926 | 0.2512697 | 0.602141 | 0.0097186 |
| $p(loss_{vp})$ | 0.0222936 | 0.0247407 | 0.0760156 | 0.14154 | 0.0051388 |

Table B.6: The subsystem results for the test cases in Table B.3 using the thinned arrival rates.

## B.2.5 Approximation Results

| test case | simulation | approximation 1 | approximation 2 | approximation 3 |
|---|---|---|---|---|
| 1 | 0.155723 | 0.1279670882 | 0.1526495922 | 0.1279909988 |
| 2 | 0.026349 | 0.02048031 | 0.0249475267 | 0.020476752 |
| 3 | 0.029519 | 0.0210958195 | 0.0278296082 | 0.0212831812 |
| 4 | 0.197246 | 0.1579370545 | 0.1922870149 | 0.1589117163 |
| 5 | 0.574883 | 0.5450921875 | 0.5737497899 | 0.5512751574 |
| 6 | 0.357552 | 0.2995222374 | 0.3870244105 | 0.3218891222 |
| 7 | 0.449853 | 0.4145792 | 0.4841418182 | 0.420423868 |
| 8 | 0.566914 | 0.6055293015 | 0.6333843356 | 0.4550272344 |
| 9 | 0.301966 | 0.3040607569 | 0.3476019332 | 0.2494779279 |
| 10 | 0.659053 | 0.6563151373 | 0.7227124858 | 0.5845200899 |
| 11 | 0.336325 | 0.2571874587 | 0.3779988864 | 0.2279330061 |
| 12 | 0.534212 | 0.6108164743 | 0.6708655137 | 0.4455632433 |
| 13 | 0.403717 | 0.4118978431 | 0.5409671368 | 0.3507906972 |

| test case | approximation 4 | approximation 5 | approximation 6 | approximation 7 |
|---|---|---|---|---|
| 1 | 0.127949905 | 0.1526216976 | 0.1544947868 | 0.1563032872 |
| 2 | 0.0204386471 | 0.024899656 | 0.0260799903 | 0.0272033034 |
| 3 | 0.0217680581 | 0.0285921381 | 0.0293792685 | 0.0301368321 |
| 4 | 0.1588358415 | 0.192491977 | 0.1939700191 | 0.1977803942 |
| 5 | 0.5492621511 | 0.5751546321 | 0.589236926 | 0.5811663016 |
| 6 | 0.256402651 | 0.333557863 | 0.3167456913 | 0.4255573085 |
| 7 | 0.4002407967 | 0.4621984766 | 0.4592925085 | 0.4898935225 |
| 8 | 0.3831633678 | 0.3793267061 | 0.258175063 | 0.6516920424 |
| 9 | 0.2285957302 | 0.3040318705 | 0.3012242426 | 0.3472722732 |
| 10 | 0.5329598304 | 0.6162632913 | 0.5222511128 | 0.7433802703 |
| 11 | 0.1931079123 | 0.3044244303 | 0.2981373623 | 0.3936333358 |
| 12 | 0.3468376771 | 0.4047688726 | 0.3707220893 | 0.6134887228 |
| 13 | 0.2695833847 | 0.3865485485 | 0.3816964597 | 0.4789522017 |

| test case | approximation 8 | approximation 9 | approximation 10 | approximation 11 |
|---|---|---|---|---|
| 1 | 0.1526172491 | 0.1525792755 | 0.1548778002 | 0.1548655503 |
| 2 | 0.0248996459 | 0.0248785667 | 0.0252003629 | 0.0251642715 |
| 3 | 0.0285915948 | 0.0285765052 | 0.0285074953 | 0.0292290279 |
| 4 | 0.1924974604 | 0.1922133155 | 0.1973727157 | 0.1974190718 |
| 5 | 0.5753453091 | 0.5715599547 | 0.5805106822 | 0.5807683361 |
| 6 | 0.3327602679 | 0.3201297697 | 0.4539021471 | 0.3880966904 |
| 7 | 0.4584185614 | 0.4537914498 | 0.4991822786 | 0.4763140148 |
| 8 | 0.440647721 | 0.3815268022 | 0.7719900595 | 0.5689518336 |
| 9 | 0.3042940545 | 0.2962388156 | 0.3621465456 | 0.3283240978 |
| 10 | 0.633712907 | 0.5726647175 | 0.5609871263 | 0.6990231371 |
| 11 | 0.3052299532 | 0.2898102071 | 0.365595362 | 0.3565219836 |
| 12 | 0.4423040785 | 0.3758648926 | 0.7654747956 | 0.5239709891 |
| 13 | 0.3891507666 | 0.3598292264 | 0.5972008452 | 0.4285545932 |

**Table B.7**: Loss probabilities calculated by the approximation methods.

## B.2.6  The Effect of VP Arrival Rates on the Approximations

### B.2.6.1  The Effect of VP Arrival Rates on $P_i^r$

| **Test Case 1** | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.007205 +/- 0.000008 | 0.00899609 | 0.00899609 |
| $P_1^r$ | 0.221696 +/- 0.000039 | 0.2218431 | 0.2217678 |
| $P_{vp}$ | 0.031932 +/- 0.000026 | 0.0269227568 | 0.0269139777 |

| **Test Case 2** | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.033247 +/- 0.000015 | 0.03388196 | 0.03388196 |
| $P_1^r$ | 0.033248 +/- 0.000014 | 0.03388196 | 0.03372981 |
| $P_{vp}$ | 0.012435 +/- 0.000015 | 0.0078371686 | 0.0078198751 |

| **Test Case 3** | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.021927 +/- 0.000013 | 0.02244118 | 0.02244118 |
| $P_1^r$ | 0.028937 +/- 0.000015 | 0.02668944 | 0.02939957 |
| $P_2^r$ | 0.046635 +/- 0.000020 | 0.04704279 | 0.04695963 |
| $P_{vp}$ | 0.010448 +/- 0.000017 | 0.0071762526 | 0.0073643891 |

| **Test Case 4** | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.010245 +/- 0.000013 | 0.01209724 | 0.01209724 |
| $P_1^r$ | 0.001170 +/- 0.000004 | 0.00176265 | 0.001746916 |
| $P_2^r$ | 0.000112 +/- 0.000001 | 0.0002829299 | 0.0002777357 |
| $P_3^r$ | 0.297873 +/- 0.000052 | 0.298021 | 0.297875 |
| $P_4^r$ | 0.000010 +/- 0.000000 | 0.000064985 | 0.00004060131 |
| $P_{vp}$ | 0.108033 +/- 0.000079 | 0.1026588097 | 0.1026003715 |

| **Test Case 5** | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.681756 +/- 0.000046 | 0.683679 | 0.683679 |
| $P_1^r$ | 0.238057 +/- 0.000069 | 0.262477 | 0.2498039 |
| $P_{vp}$ | 0.255408 +/- 0.000100 | 0.2555690202 | 0.2542327624 |

| Test Case 6 | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.288909 +/- 0.000081 | 0.3963506 | 0.3963506 |
| $P_1^r$ | 0.142535 +/- 0.000069 | 0.2942393 | 0.2197517 |
| $P_2^r$ | 0.049234 +/- 0.000044 | 0.2182163 | 0.1095581 |
| $P_3^r$ | 0.516964 +/- 0.000068 | 0.5576657 | 0.503106 |
| $P_4^r$ | 0.095786 +/- 0.000059 | 0.2586775 | 0.1029045 |
| $P_{vp}$ | 0.397190 +/- 0.000069 | 0.4715915092 | 0.430438229 |

| Test Case 7 | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.456582 +/- 0.000102 | 0.5102278 | 0.5102278 |
| $P_1^r$ | 0.456648 +/- 0.000101 | 0.5102278 | 0.489500 |
| $P_2^r$ | 0.456621 +/- 0.000103 | 0.5102278 | 0.478029 |
| $P_3^r$ | 0.456526 +/- 0.000102 | 0.5102278 | 0.471040 |
| $P_4^r$ | 0.456767 +/- 0.000105 | 0.5102278 | 0.4664701 |
| $P_{vp}$ | 0.382046 +/- 0.000117 | 0.3887270649 | 0.3852674365 |

| Test Case 8 | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.393947 +/- 0.000106 | 0.70046029 | 0.70046029 |
| $P_1^r$ | 0.349093 +/- 0.000106 | 0.6937711 | 0.3901685 |
| $P_2^r$ | 0.371595 +/- 0.000105 | 0.69706827 | 0.2983854 |
| $P_3^r$ | 0.393898 +/- 0.000105 | 0.70046029 | 0.2587236 |
| $P_4^r$ | 0.349023 +/- 0.000103 | 0.6937711 | 0.1930355 |
| $P_{vp}$ | 0.685323 +/- 0.000046 | 0.7439447624 | 0.7568324341 |

| Test Case 9 | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.235428 +/- 0.000090 | 0.3139707 | 0.3139707 |
| $P_1^r$ | 0.472179 +/- 0.000082 | 0.5102278 | 0.495956 |
| $P_2^r$ | 0.131320 +/- 0.000073 | 0.224456 | 0.1673662 |
| $P_3^r$ | 0.038002 +/- 0.000044 | 0.1301533 | 0.0668732 |
| $P_4^r$ | 0.053859 +/- 0.000051 | 0.1486036 | 0.0668732 |
| $P_5^r$ | 0.157623 +/- 0.000076 | 0.2477595 | 0.1787404 |
| $P_6^r$ | 0.370169 +/- 0.000090 | 0.4252666 | 0.3745361 |
| $P_{vp}$ | 0.334825 +/- 0.000099 | 0.3666261069 | 0.3501509254 |

| Test Case 10 | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.664808 +/- 0.000088 | 0.82467245 | 0.82467245 |
| $P_1^r$ | 0.471443 +/- 0.000122 | 0.78571964 | 0.5625625 |
| $P_2^r$ | 0.355418 +/- 0.000131 | 0.75390688 | 0.4080438 |
| $P_3^r$ | 0.717811 +/- 0.000074 | 0.83927612 | 0.6995398 |
| $P_{vp}$ | 0.705378 +/- 0.000052 | 0.7309769378 | 0.7220787551 |

| Test Case 11 | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.010377 +/- 0.000022 | 0.18170806 | 0.18170806 |
| $P_1^r$ | 0.033295 +/- 0.000041 | 0.214732 | 0.1720429 |
| $P_2^r$ | 0.072630 +/- 0.000059 | 0.2543282 | 0.1809003 |
| $P_3^r$ | 0.264732 +/- 0.000092 | 0.3963506 | 0.3185708 |
| $P_4^r$ | 0.065452 +/- 0.000059 | 0.2477791 | 0.1153315 |
| $P_5^r$ | 0.010360 +/- 0.000022 | 0.18170816 | 0.04705349 |
| $P_6^r$ | 0.045262 +/- 0.000048 | 0.2279902 | 0.0919749 |
| $P_7^r$ | 0.398113 +/- 0.000088 | 0.487553 | 0.3944443 |
| $P_8^r$ | 0.010371 +/- 0.000023 | 0.18170806 | 0.02611425 |
| $P_9^r$ | 0.497091 +/- 0.000080 | 0.5576657 | 0.477318 |
| $P_{vp}$ | 0.436047 +/- 0.000081 | 0.5159151121 | 0.4821716451 |

| Test Case 12 | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.485500 +/- 0.000106 | 0.74299199 | 0.74299199 |
| $P_1^r$ | 0.358032 +/- 0.000118 | 0.71897635 | 0.4509751 |
| $P_2^r$ | 0.077799 +/- 0.000082 | 0.67387012 | 0.1702648 |
| $P_3^r$ | 0.032575 +/- 0.000055 | 0.66503488 | 0.10651049 |
| $P_4^r$ | 0.228659 +/- 0.000115 | 0.69798416 | 0.1742193 |
| $P_5^r$ | 0.574578 +/- 0.000097 | 0.7633397 | 0.5075489 |
| $P_6^r$ | 0.032626 +/- 0.000056 | 0.66503488 | 0.03992935 |
| $P_7^r$ | 0.077810 +/- 0.000084 | 0.67387012 | 0.06233683 |
| $P_8^r$ | 0.228624 +/- 0.000115 | 0.69798416 | 0.1600288 |
| $P_9^r$ | 0.128130 +/- 0.000100 | 0.68229136 | 0.0814024 |
| $P_{vp}$ | 0.745521 +/- 0.000049 | 0.8194697093 | 0.7454520189 |

| Test Case 13 | | | |
|---|---|---|---|
| | simulation | calculated with $\lambda_{vp}$ | calculated with $\lambda'_{vp_i}$ |
| $P_0^r$ | 0.378473 +/- 0.000115 | 0.592215 | 0.592215 |
| $P_1^r$ | 0.243930 +/- 0.000125 | 0.5326147 | 0.3846898 |
| $P_2^r$ | 0.024268 +/- 0.000048 | 0.4115355 | 0.1277228 |
| $P_3^r$ | 0.006201 +/- 0.000023 | 0.38707028 | 0.07934707 |
| $P_4^r$ | 0.125512 +/- 0.000100 | 0.4774252 | 0.2041228 |
| $P_5^r$ | 0.479565 +/- 0.000110 | 0.6393763 | 0.5060818 |
| $P_6^r$ | 0.006231 +/- 0.000024 | 0.38707028 | 0.02923268 |
| $P_7^r$ | 0.024277 +/- 0.000048 | 0.4115355 | 0.05059671 |
| $P_8^r$ | 0.125571 +/- 0.000099 | 0.4774252 | 0.02618034 |
| $P_9^r$ | 0.052285 +/- 0.000071 | 0.4347493 | 0.0760662 |
| $P_{10}^r$ | 0.024290 +/- 0.000048 | 0.4115355 | 0.0443931 |
| $P_{11}^r$ | 0.034473 +/- 0.000057 | 0.4209719 | 0.0533573 |
| $P_{12}^r$ | 0.243867 +/- 0.000122 | 0.5326147 | 0.2498185 |
| $P_{13}^r$ | 0.612963 +/- 0.000093 | 0.7081965 | 0.601757 |
| $P_{14}^r$ | 0.004017 +/- 0.000020 | 0.38203006 | 0.00938642 |
| $P_{vp}$ | 0.640513 +/- 0.000080 | 0.6756490375 | 0.6638116626 |

Table B.8: $P_i^r$ and $P_{vp}$ values found by simulation of the original system, solving the Markov chain for each of the subsystems using $\lambda_{vp_i}$, and solving the Markov chain for each of the subsystems using $\lambda_{vp_i}'$ as the arrival rate for VP calls.

## B.2.6.2 Alternative Calculation Methods for $P_i^r$

**Calculation of the Value of $\sum \lambda_i P(loss_i)'$**

| test case | simulation | totally thinned | final $\lambda'_{vp}$ | middle $\lambda'_{vp}$ | average $\lambda'_{vp}$ |
|---|---|---|---|---|---|
| 1 | 0.445800 | 0.4465342967 | 0.44652209 | 0.4466848967 | 0.4466035833 |
| 2 | 0.044331 | 0.0450745133 | 0.04497308 | 0.0451759467 | 0.0450742933 |
| 3 | 0.071054 | 0.0719334324 | 0.0718691457 | 0.0719541523 | 0.0719490844 |
| 4 | 1.503094 | 1.505759539 | 1.493439073 | 1.505704335 | 1.503376735 |
| 5 | 3.673247 | 3.695954889 | 3.681894889 | 3.710036111 | 3.695531 |
| 6 | 2.910004 | 3.192714343 | 2.69358549 | 3.058161049 | 3.13845364 |
| 7 | 9.132772 | 9.6610676 | 9.329402 | 9.56058 | 9.64342 |
| 8 | 2.279523 | 2.290800932 | 1.282299708 | 1.829844812 | 2.568147887 |
| 9 | 3.950023 | 4.33585175 | 4.060155613 | 4.177133787 | 4.288616647 |
| 10 | 7.665683 | 8.4655722 | 7.524757133 | 8.226684367 | 8.624338133 |
| 11 | 3.870690 | 4.163129173 | 3.577091053 | 4.12424613 | 4.266712225 |
| 12 | 5.232941 | 5.835976496 | 4.08417248 | 4.865395695 | 5.57926465 |
| 13 | 8.121225 | 9.326295813 | 7.91975479 | 8.370133524 | 8.942029613 |

Table B.9: Re-calculation of the value of $\sum \lambda_i P(loss_i)'$ using different methods of calculating $\lambda_{vp}$.

### B.2.6.3 Effect of $P_i^r$ on the Approximation Solution

Re-calculation of Approximation 11 Using Different Methods for Calculating $\sum \lambda_i P(loss_i)'$

| test case | simulation | approximation 11 thinned | approximation 11 final | approximation 11 middle |
|---|---|---|---|---|
| 1 | 0.155723 | 0.1548655503 | 0.1548614801 | 0.1549157659 |
| 2 | 0.026349 | 0.0251642715 | 0.0251134952 | 0.0252150479 |
| 3 | 0.029519 | 0.0292290279 | 0.0292041741 | 0.0292370384 |
| 4 | 0.197246 | 0.1974190718 | 0.1958557246 | 0.1974120668 |
| 5 | 0.574883 | 0.5807683361 | 0.5786252804 | 0.5829146266 |
| 6 | 0.357552 | 0.3880966904 | 0.3414869845 | 0.3755318198 |
| 7 | 0.449853 | 0.4763140148 | 0.4608736456 | 0.4716359124 |
| 8 | 0.566914 | 0.5689518336 | 0.4877088178 | 0.5318180509 |
| 9 | 0.301966 | 0.3283321767 | 0.3096786151 | 0.3175933418 |
| 10 | 0.659053 | 0.6990231371 | 0.649732613 | 0.6865074930 |
| 11 | 0.336325 | 0.3565219836 | 0.3159890903 | 0.3538326667 |
| 12 | 0.534212 | 0.5239709891 | 0.4306477151 | 0.4722655602 |
| 13 | 0.403717 | 0.4285545932 | 0.373138683 | 0.3908830268 |

**Table B.10:** Re-calculation of Approximation 11 using different methods for calculating $\sum \lambda_i P(loss_i)'$.

# B.3 Optimal VP Capacity Assignment in a Simple Network

## B.3.1 Results

### B.3.1.1 Loss Probability and Setup Probability Results

| | Test Case 1 | | | |
|---|---|---|---|---|
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.074193 | 1.000000 | 0.073744 +/- 0.000025 | 1.000000 +/- 0.000000 |
| 1 | 0.094485 | 0.852754 | 0.096195 +/- 0.000027 | 0.852477 +/- 0.000017 |
| 2 | 0.154866 | 0.767991 | 0.155723 +/- 0.000031 | 0.767768 +/- 0.000024 |
| 3 | 0.275693 | 0.701010 | 0.275582 +/- 0.000036 | 0.701025 +/- 0.000030 |
| 4 | 0.473346 | 0.579833 | 0.473146 +/- 0.000036 | 0.580019 +/- 0.000039 |
| | Test Case 2 | | | |
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.012231 | 1.000000 | 0.011500 +/- 0.000010 | 1.000000 +/- 0.000000 |
| 1 | 0.013027 | 0.797360 | 0.014156 +/- 0.000010 | 0.797132 +/- 0.000019 |
| 2 | 0.025164 | 0.698290 | 0.026349 +/- 0.000013 | 0.697947 +/- 0.000025 |
| 3 | 0.083192 | 0.645683 | 0.083354 +/- 0.000020 | 0.645643 +/- 0.000027 |
| 4 | 0.268254 | 0.546395 | 0.268039 +/- 0.000029 | 0.546566 +/- 0.000032 |
| 5 | 0.666917 | 0.000000 | 0.666862 +/- 0.000029 | 0.000000 +/- 0.000000 |
| | Test Case 3 | | | |
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.009001 | 1.000000 | 0.008496 +/- 0.000009 | 1.000000 +/- 0.000000 |
| 1 | 0.011396 | 0.869728 | 0.011940 +/- 0.000009 | 0.869668 +/- 0.000016 |
| 2 | 0.029233 | 0.816308 | 0.029519 +/- 0.000013 | 0.816266 +/- 0.000022 |
| 3 | 0.105994 | 0.786650 | 0.105837 +/- 0.000022 | 0.786686 +/- 0.000026 |
| 4 | 0.334042 | 0.710377 | 0.333931 +/- 0.000031 | 0.710470 +/- 0.000033 |
| 5 | 0.806849 | 0.000000 | 0.806847 +/- 0.000025 | 0.000000 +/- 0.000000 |

| Test Case 4 | | | | |
|---|---|---|---|---|
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.160781 | 1.000000 | 0.160542 +/- 0.000039 | 1.000000 +/- 0.000000 |
| 1 | 0.197395 | 0.947381 | 0.197246 +/- 0.000037 | 0.947400 +/- 0.000011 |
| 2 | 0.261966 | 0.920769 | 0.261595 +/- 0.000041 | 0.920802 +/- 0.000018 |
| 3 | 0.354252 | 0.903141 | 0.353974 +/- 0.000041 | 0.903171 +/- 0.000022 |
| 4 | 0.480813 | 0.878178 | 0.480714 +/- 0.000037 | 0.878205 +/- 0.000028 |
| 5 | 0.662393 | 0.812390 | 0.662463 +/- 0.000030 | 0.812317 +/- 0.000040 |
| 6 | 0.936652 | 0.000000 | 0.936641 +/- 0.000016 | 0.000000 +/- 0.000000 |

| Test Case 5 | | | | |
|---|---|---|---|---|
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.487718 | 1.000000 | 0.486386 +/- 0.000040 | 1.000000 +/- 0.000000 |
| 1 | 0.580768 | 0.879732 | 0.574883 +/- 0.000037 | 0.881373 +/- 0.000023 |
| 2 | 0.727478 | 0.743828 | 0.724613 +/- 0.000030 | 0.746476 +/- 0.000044 |
| 3 | 0.925327 | 0.000000 | 0.925303 +/- 0.000017 | 0.000000 +/- 0.000000 |

| Test Case 6 | | | | |
|---|---|---|---|---|
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.330390 | 1.000000 | 0.308393 +/- 0.000041 | 1.000000 +/- 0.000000 |
| 1 | 0.348530 | 0.899637 | 0.325494 +/- 0.000043 | 0.903057 +/- 0.000017 |
| 2 | 0.388115 | 0.798859 | 0.357552 +/- 0.000042 | 0.808445 +/- 0.000023 |
| 3 | 0.461863 | 0.682231 | 0.421578 +/- 0.000039 | 0.704366 +/- 0.000029 |
| 4 | 0.589593 | 0.494086 | 0.549869 +/- 0.000034 | 0.538736 +/- 0.000036 |
| 5 | 0.767248 | 0.000000 | 0.767270 +/- 0.000022 | 0.000000 +/- 0.000000 |

| Test Case 7 | | | | |
|---|---|---|---|---|
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.321110 | 1.000000 | 0.301298 +/- 0.000042 | 1.000000 +/- 0.000000 |
| 1 | 0.383887 | 0.950816 | 0.357382 +/- 0.000042 | 0.952837 +/- 0.000012 |
| 2 | 0.476314 | 0.895843 | 0.449853 +/- 0.000040 | 0.900866 +/- 0.000018 |
| 3 | 0.598189 | 0.821383 | 0.578960 +/- 0.000037 | 0.829564 +/- 0.000028 |
| 4 | 0.746779 | 0.675181 | 0.735916 +/- 0.000027 | 0.688567 +/- 0.000047 |
| 5 | 0.912427 | 0.000000 | 0.912422 +/- 0.000016 | 0.000000 +/- 0.000000 |

| Test Case 8 | | | | |
|---|---|---|---|---|
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.561063 | 1.000000 | 0.548290 +/- 0.000041 | 1.000000 +/- 0.000000 |
| 1 | 0.559986 | 0.871741 | 0.555834 +/- 0.000042 | 0.872891 +/- 0.000028 |
| 2 | 0.568952 | 0.740293 | 0.566914 +/- 0.000044 | 0.741502 +/- 0.000038 |
| 3 | 0.597651 | 0.586596 | 0.584959 +/- 0.000043 | 0.599219 +/- 0.000045 |
| 4 | 0.660813 | 0.353309 | 0.620026 +/- 0.000040 | 0.422657 +/- 0.000046 |
| 5 | 0.729304 | 0.000000 | 0.729277 +/- 0.000033 | 0.000000 +/- 0.000000 |

| Test Case 9 | | | | |
| --- | --- | --- | --- | --- |
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.230054 | 1.000000 | 0.218409 +/- 0.000038 | 1.000000 +/- 0.000000 |
| 1 | 0.263686 | 0.940823 | 0.245705 +/- 0.000039 | 0.942229 +/- 0.000012 |
| 2 | 0.328347 | 0.883226 | 0.301966 +/- 0.000040 | 0.887631 +/- 0.000018 |
| 3 | 0.441585 | 0.815193 | 0.414139 +/- 0.000037 | 0.823860 +/- 0.000024 |
| 4 | 0.623227 | 0.686099 | 0.605130 +/- 0.000030 | 0.700499 +/- 0.000036 |
| 5 | 0.874078 | 0.000000 | 0.874078 +/- 0.000019 | 0.000000 +/- 0.000000 |

| Test Case 10 | | | | |
| --- | --- | --- | --- | --- |
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.610499 | 1.000000 | 0.591046 +/- 0.000040 | 1.000000 +/- 0.000000 |
| 1 | 0.627446 | 0.890739 | 0.605513 +/- 0.000039 | 0.896801 +/- 0.000027 |
| 2 | 0.655823 | 0.765401 | 0.626346 +/- 0.000040 | 0.783867 +/- 0.000037 |
| 3 | 0.699023 | 0.601395 | 0.659053 +/- 0.000038 | 0.648167 +/- 0.000045 |
| 4 | 0.761078 | 0.337817 | 0.715377 +/- 0.000033 | 0.444057 +/- 0.000050 |
| 5 | 0.804755 | 0.000000 | 0.804751 +/- 0.000027 | 0.000000 +/- 0.000000 |

| Test Case 11 | | | | |
| --- | --- | --- | --- | --- |
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.280632 | 1.000000 | 0.277364 +/- 0.000040 | 1.000000 +/- 0.000000 |
| 1 | 0.304710 | 0.929955 | 0.298124 +/- 0.000040 | 0.930616 +/- 0.000015 |
| 2 | 0.356522 | 0.857535 | 0.336325 +/- 0.000040 | 0.861875 +/- 0.000021 |
| 3 | 0.446651 | 0.769815 | 0.410724 +/- 0.000038 | 0.783842 +/- 0.000024 |
| 4 | 0.600826 | 0.612563 | 0.559918 +/- 0.000032 | 0.648592 +/- 0.000033 |
| 5 | 0.826634 | 0.000000 | 0.826651 +/- 0.000019 | 0.000000 +/- 0.000000 |

| Test Case 12 | | | | |
| --- | --- | --- | --- | --- |
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.472026 | 1.000000 | 0.502186 +/- 0.000041 | 1.000000 +/- 0.000000 |
| 1 | 0.490088 | 0.924934 | 0.514918 +/- 0.000042 | 0.921092 +/- 0.000022 |
| 2 | 0.523971 | 0.840498 | 0.534212 +/- 0.000041 | 0.836951 +/- 0.000030 |
| 3 | 0.584999 | 0.728158 | 0.567793 +/- 0.000041 | 0.738904 +/- 0.000038 |
| 4 | 0.694688 | 0.512717 | 0.648602 +/- 0.000033 | 0.576525 +/- 0.000042 |
| 5 | 0.816401 | 0.000000 | 0.816382 +/- 0.000025 | 0.000000 +/- 0.000000 |

| Test Case 13 | | | | |
|---|---|---|---|---|
| | Theoretical | | Simulation | |
| $V$ | $P(loss)$ | $P(setup)$ | $P(loss)$ | $P(setup)$ |
| 0 | 0.327366 | 1.000000 | 0.337322 +/- 0.000042 | 1.000000 +/- 0.000000 |
| 1 | 0.365946 | 0.953145 | 0.361580 +/- 0.000042 | 0.953462 +/- 0.000013 |
| 2 | 0.427918 | 0.898945 | 0.403717 +/- 0.000040 | 0.903033 +/- 0.000019 |
| 3 | 0.525820 | 0.823191 | 0.485248 +/- 0.000037 | 0.837146 +/- 0.000025 |
| 4 | 0.681583 | 0.663186 | 0.635318 +/- 0.000029 | 0.705914 +/- 0.000036 |
| 5 | 0.872525 | 0.000000 | 0.872513 +/- 0.000017 | 0.000000 +/- 0.000000 |

Table B.11: Theoretical and simulated values of $P(loss)$ and $P(setup)$ for the 13 test cases from Table B.3.

## B.3.1.2 Theoretical and Simulated $F$ Values

| Test Case 1 – $\alpha = 0.1$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.907419 | 0.907374 |
| 1 | 0.776927 | 0.776849 |
| 2 | 0.706678 | 0.706564 |
| 3 | 0.658478 | 0.658481 |
| 4 | 0.569184 | 0.569332 |
| 5 | 0.077795 | 0.077789 |

| Test Case 1 – $\alpha = 0.2$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.814839 | 0.814749 |
| 1 | 0.701100 | 0.701221 |
| 2 | 0.645366 | 0.645359 |
| 3 | 0.615947 | 0.615936 |
| 4 | 0.558536 | 0.558644 |
| 5 | 0.155590 | 0.155577 |

| Test Case 1 – $\alpha = 0.3$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.722258 | 0.722123 |
| 1 | 0.625273 | 0.625592 |
| 2 | 0.584054 | 0.584155 |
| 3 | 0.573415 | 0.573392 |
| 4 | 0.547887 | 0.547957 |
| 5 | 0.233386 | 0.233366 |

| Test Case 1 – $\alpha = 0.4$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.629677 | 0.629498 |
| 1 | 0.549446 | 0.549964 |
| 2 | 0.522741 | 0.522950 |
| 3 | 0.530883 | 0.530848 |
| 4 | 0.537238 | 0.537270 |
| 5 | 0.311181 | 0.311154 |

| Test Case 1 – $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.537096 | 0.536872 |
| 1 | 0.473620 | 0.474336 |
| 2 | 0.461429 | 0.461746 |
| 3 | 0.488352 | 0.488304 |
| 4 | 0.526590 | 0.526582 |
| 5 | 0.388976 | 0.388943 |

| Test Case 1 – $\alpha = 0.6$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.444516 | 0.444246 |
| 1 | 0.397793 | 0.398708 |
| 2 | 0.400116 | 0.400541 |
| 3 | 0.445820 | 0.445759 |
| 4 | 0.515941 | 0.515895 |
| 5 | 0.466771 | 0.466732 |

| Test Case 1 – $\alpha = 0.7$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.351935 | 0.351621 |
| 1 | 0.321966 | 0.323080 |
| 2 | 0.338804 | 0.339337 |
| 3 | 0.403288 | 0.403215 |
| 4 | 0.505292 | 0.505208 |
| 5 | 0.544566 | 0.544520 |

| Test Case 1 – $\alpha = 0.8$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.259354 | 0.258995 |
| 1 | 0.246139 | 0.247451 |
| 2 | 0.277491 | 0.278132 |
| 3 | 0.360756 | 0.360671 |
| 4 | 0.494643 | 0.494521 |
| 5 | 0.622362 | 0.622309 |

| Test Case 1 – $\alpha = 0.9$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.166774 | 0.166370 |
| 1 | 0.170312 | 0.171823 |
| 2 | 0.216178 | 0.216927 |
| 3 | 0.318225 | 0.318126 |
| 4 | 0.483995 | 0.483833 |
| 5 | 0.700157 | 0.700097 |

| Test Case 2 − $\alpha = 0.1$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.901223 | 0.901150 |
| 1 | 0.718927 | 0.718834 |
| 2 | 0.630977 | 0.630787 |
| 3 | 0.589434 | 0.589414 |
| 4 | 0.518581 | 0.518713 |
| 5 | 0.066692 | 0.066686 |

| Test Case 2 − $\alpha = 0.2$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.802446 | 0.802300 |
| 1 | 0.640493 | 0.640537 |
| 2 | 0.563665 | 0.563627 |
| 3 | 0.533185 | 0.533185 |
| 4 | 0.490767 | 0.490861 |
| 5 | 0.133383 | 0.133372 |

| Test Case 2 − $\alpha = 0.3$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.703669 | 0.703450 |
| 1 | 0.562060 | 0.562239 |
| 2 | 0.496352 | 0.496468 |
| 3 | 0.476936 | 0.476956 |
| 4 | 0.462953 | 0.463008 |
| 5 | 0.200075 | 0.200059 |

| Test Case 2 − $\alpha = 0.4$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.604892 | 0.604600 |
| 1 | 0.483627 | 0.483942 |
| 2 | 0.429040 | 0.429308 |
| 3 | 0.420687 | 0.420727 |
| 4 | 0.435139 | 0.435155 |
| 5 | 0.266767 | 0.266745 |

| Test Case 2 − $\alpha = 0.5$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.506116 | 0.505750 |
| 1 | 0.405193 | 0.405644 |
| 2 | 0.361727 | 0.362148 |
| 3 | 0.364438 | 0.364499 |
| 4 | 0.407324 | 0.407303 |
| 5 | 0.333458 | 0.333431 |

| Test Case 2 − $\alpha = 0.6$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.407339 | 0.406900 |
| 1 | 0.326760 | 0.327346 |
| 2 | 0.294414 | 0.294988 |
| 3 | 0.308188 | 0.308270 |
| 4 | 0.379510 | 0.379450 |
| 5 | 0.400150 | 0.400117 |

| Test Case 2 − $\alpha = 0.7$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.308562 | 0.308050 |
| 1 | 0.248327 | 0.249049 |
| 2 | 0.227102 | 0.227828 |
| 3 | 0.251939 | 0.252041 |
| 4 | 0.351696 | 0.351597 |
| 5 | 0.466842 | 0.466803 |

| Test Case 2 − $\alpha = 0.8$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.209785 | 0.209200 |
| 1 | 0.169894 | 0.170751 |
| 2 | 0.159789 | 0.160669 |
| 3 | 0.195690 | 0.195812 |
| 4 | 0.323882 | 0.323744 |
| 5 | 0.533534 | 0.533490 |

| Test Case 2 − $\alpha = 0.9$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.111008 | 0.110350 |
| 1 | 0.091460 | 0.092454 |
| 2 | 0.092477 | 0.093509 |
| 3 | 0.139441 | 0.139583 |
| 4 | 0.296068 | 0.295892 |
| 5 | 0.600225 | 0.600176 |

| Test Case 3 – $\alpha = 0.1$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.900900 | 0.900850 |
| 1 | 0.783895 | 0.783895 |
| 2 | 0.737600 | 0.737591 |
| 3 | 0.718584 | 0.718601 |
| 4 | 0.672744 | 0.672816 |
| 5 | 0.080685 | 0.080685 |

| Test Case 3 – $\alpha = 0.2$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.801800 | 0.801699 |
| 1 | 0.698062 | 0.698122 |
| 2 | 0.658893 | 0.658917 |
| 3 | 0.650519 | 0.650516 |
| 4 | 0.635110 | 0.635162 |
| 5 | 0.161370 | 0.161369 |

| Test Case 3 – $\alpha = 0.3$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.702700 | 0.702549 |
| 1 | 0.612228 | 0.612350 |
| 2 | 0.580186 | 0.580242 |
| 3 | 0.582453 | 0.582431 |
| 4 | 0.597477 | 0.597508 |
| 5 | 0.242055 | 0.242054 |

| Test Case 3 – $\alpha = 0.4$ | | |
|---|---|---|
| V | .Theoretical $F$ | Simulation $F$ |
| 0 | 0.603600 | 0.603398 |
| 1 | 0.526395 | 0.526577 |
| 2 | 0.501478 | 0.501567 |
| 3 | 0.514388 | 0.514346 |
| 4 | 0.559843 | 0.559854 |
| 5 | 0.322740 | 0.322739 |

| Test Case 3 – $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.504501 | 0.504248 |
| 1 | 0.440562 | 0.440804 |
| 2 | 0.422770 | 0.422893 |
| 3 | 0.446322 | 0.446261 |
| 4 | 0.522209 | 0.522201 |
| 5 | 0.403425 | 0.403423 |

| Test Case 3 – $\alpha = 0.6$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.405401 | 0.405098 |
| 1 | 0.354729 | 0.355031 |
| 2 | 0.344063 | 0.344218 |
| 3 | 0.378256 | 0.378177 |
| 4 | 0.484576 | 0.484547 |
| 5 | 0.484109 | 0.484108 |

| Test Case 3 – $\alpha = 0.7$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.306301 | 0.305947 |
| 1 | 0.268896 | 0.269258 |
| 2 | 0.265356 | 0.265543 |
| 3 | 0.310191 | 0.310092 |
| 4 | 0.446943 | 0.446893 |
| 5 | 0.564794 | 0.564793 |

| Test Case 3 – $\alpha = 0.8$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.207201 | 0.206797 |
| 1 | 0.183062 | 0.183486 |
| 2 | 0.186648 | 0.186868 |
| 3 | 0.242125 | 0.242007 |
| 4 | 0.409309 | 0.409239 |
| 5 | 0.645479 | 0.645478 |

| Test Case 3 – $\alpha = 0.9$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.108101 | 0.107646 |
| 1 | 0.097229 | 0.097713 |
| 2 | 0.107940 | 0.108194 |
| 3 | 0.174060 | 0.173922 |
| 4 | 0.371675 | 0.371585 |
| 5 | 0.726164 | 0.726162 |

| Test Case 4 − $\alpha = 0.1$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.916078 | 0.916054 |
| 1 | 0.872382 | 0.872385 |
| 2 | 0.854889 | 0.854881 |
| 3 | 0.848252 | 0.848251 |
| 4 | 0.838442 | 0.838456 |
| 5 | 0.797390 | 0.797332 |
| 6 | 0.093665 | 0.093664 |

| Test Case 4 − $\alpha = 0.2$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.832156 | 0.832108 |
| 1 | 0.797384 | 0.797369 |
| 2 | 0.789008 | 0.788961 |
| 3 | 0.793363 | 0.793332 |
| 4 | 0.798705 | 0.798707 |
| 5 | 0.782391 | 0.782346 |
| 6 | 0.187330 | 0.187328 |

| Test Case 4 − $\alpha = 0.3$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.748234 | 0.748163 |
| 1 | 0.722385 | 0.722354 |
| 2 | 0.723128 | 0.723040 |
| 3 | 0.738474 | 0.738412 |
| 4 | 0.758969 | 0.758958 |
| 5 | 0.767391 | 0.767361 |
| 6 | 0.280996 | 0.280992 |

| Test Case 4 − $\alpha = 0.4$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.664312 | 0.664217 |
| 1 | 0.647387 | 0.647338 |
| 2 | 0.657248 | 0.657119 |
| 3 | 0.683585 | 0.683492 |
| 4 | 0.719232 | 0.719209 |
| 5 | 0.752391 | 0.752375 |
| 6 | 0.374661 | 0.374656 |

| Test Case 4 − $\alpha = 0.6$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.496469 | 0.496325 |
| 1 | 0.497389 | 0.497308 |
| 2 | 0.525487 | 0.525278 |
| 3 | 0.573808 | 0.573653 |
| 4 | 0.639759 | 0.639710 |
| 5 | 0.722392 | 0.722405 |
| 6 | 0.561991 | 0.561985 |

| Test Case 4 − $\alpha = 0.7$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.412547 | 0.412379 |
| 1 | 0.422391 | 0.422292 |
| 2 | 0.459607 | 0.459357 |
| 3 | 0.518919 | 0.518733 |
| 4 | 0.600023 | 0.599961 |
| 5 | 0.707392 | 0.707419 |
| 6 | 0.468326 | 0.655649 |

| Test Case 4 − $\alpha = 0.8$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.328625 | 0.328434 |
| 1 | 0.347392 | 0.347277 |
| 2 | 0.393727 | 0.393436 |
| 3 | 0.464030 | 0.463813 |
| 4 | 0.560286 | 0.560212 |
| 5 | 0.692392 | 0.692434 |
| 6 | 0.749322 | 0.749313 |

| Test Case 4 − $\alpha = 0.9$ | | |
| --- | --- | --- |
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.244703 | 0.244488 |
| 1 | 0.272394 | 0.272261 |
| 2 | 0.327846 | 0.327516 |
| 3 | 0.409141 | 0.408894 |
| 4 | 0.520549 | 0.520463 |
| 5 | 0.677393 | 0.677448 |
| 6 | 0.842987 | 0.842977 |

| Test Case 4 – $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.580391 | 0.580271 |
| 1 | 0.572388 | 0.572323 |
| 2 | 0.591367 | 0.591198 |
| 3 | 0.628696 | 0.679459 |
| 4 | 0.679496 | 0.679459 |
| 5 | 0.737391 | 0.737390 |
| 6 | 0.468326 | 0.468320 |

| Test Case 5 – $\alpha = 0.1$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.948772 | 0.948639 |
| 1 | 0.849836 | 0.850724 |
| 2 | 0.742193 | 0.744290 |
| 3 | 0.092533 | 0.092530 |

| Test Case 5 – $\alpha = 0.2$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.897544 | 0.897277 |
| 1 | 0.819939 | 0.820075 |
| 2 | 0.740558 | 0.742103 |
| 3 | 0.185065 | 0.185061 |

| Test Case 5 – $\alpha = 0.3$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.846315 | 0.845916 |
| 1 | 0.790043 | 0.789426 |
| 2 | 0.738923 | 0.739917 |
| 3 | 0.277598 | 0.277591 |

| Test Case 5 – $\alpha = 0.4$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.795087 | 0.794554 |
| 1 | 0.760146 | 0.758777 |
| 2 | 0.737288 | 0.737731 |
| 3 | 0.370131 | 0.370121 |

| Test Case 5 – $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.743859 | 0.743193 |
| 1 | 0.730250 | 0.728128 |
| 2 | 0.735653 | 0.735545 |
| 3 | 0.462664 | 0.462651 |

| Test Case 5 – $\alpha = 0.6$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.692631 | 0.691832 |
| 1 | 0.700354 | 0.697479 |
| 2 | 0.734018 | 0.733358 |
| 3 | 0.555196 | 0.555182 |

| Test Case 5 – $\alpha = 0.7$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.641403 | 0.640470 |
| 1 | 0.670457 | 0.666830 |
| 2 | 0.732383 | 0.731172 |
| 3 | 0.647729 | 0.647712 |

| Test Case 5 – $\alpha = 0.8$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.590174 | 0.589109 |
| 1 | 0.640561 | 0.636181 |
| 2 | 0.730748 | 0.728986 |
| 3 | 0.740262 | 0.740242 |

| Test Case 5 – $\alpha = 0.9$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.538946 | 0.537747 |
| 1 | 0.610664 | 0.605532 |
| 2 | 0.729113 | 0.726799 |
| 3 | 0.832794 | 0.832773 |

| Test Case 6 – $\alpha = 0.1$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.933039 | 0.930839 |
| 1 | 0.844526 | 0.845301 |
| 2 | 0.757785 | 0.763356 |
| 3 | 0.660194 | 0.676087 |
| 4 | 0.503637 | 0.539849 |
| 5 | 0.076725 | 0.076727 |

| Test Case 6 – $\alpha = 0.2$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.866078 | 0.861679 |
| 1 | 0.789416 | 0.787544 |
| 2 | 0.716710 | 0.718266 |
| 3 | 0.638157 | 0.647808 |
| 4 | 0.513187 | 0.540963 |
| 5 | 0.153450 | 0.153454 |

| Test Case 6 – $\alpha = 0.3$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.799117 | 0.792518 |
| 1 | 0.734305 | 0.729788 |
| 2 | 0.675636 | 0.673177 |
| 3 | 0.616121 | 0.619530 |
| 4 | 0.522738 | 0.542076 |
| 5 | 0.230174 | 0.230181 |

| Test Case 6 – $\alpha = 0.4$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.732156 | 0.723357 |
| 1 | 0.679194 | 0.672032 |
| 2 | 0.634561 | 0.628088 |
| 3 | 0.594084 | 0.591251 |
| 4 | 0.532289 | 0.543189 |
| 5 | 0.306899 | 0.306908 |

| Test Case 6 – $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.665195 | 0.654196 |
| 1 | 0.624084 | 0.614275 |
| 2 | 0.593487 | 0.582998 |
| 3 | 0.572047 | 0.562972 |
| 4 | 0.541840 | 0.544303 |
| 5 | 0.383624 | 0.383635 |

| Test Case 6 – $\alpha = 0.6$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.598234 | 0.585036 |
| 1 | 0.568973 | 0.556519 |
| 2 | 0.552413 | 0.537909 |
| 3 | 0.550010 | 0.534693 |
| 4 | 0.551390 | 0.545416 |
| 5 | 0.460349 | 0.460362 |

| Test Case 6 – $\alpha = 0.7$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.531273 | 0.515875 |
| 1 | 0.513862 | 0.498763 |
| 2 | 0.511338 | 0.492820 |
| 3 | 0.527973 | 0.506414 |
| 4 | 0.560941 | 0.546529 |
| 5 | 0.537074 | 0.537089 |

| Test Case 6 – $\alpha = 0.8$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.464312 | 0.446714 |
| 1 | 0.458751 | 0.441007 |
| 2 | 0.470264 | 0.447731 |
| 3 | 0.505937 | 0.478136 |
| 4 | 0.570492 | 0.547642 |
| 5 | 0.613798 | 0.613816 |

| Test Case 6 – $\alpha = 0.9$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.397351 | 0.377554 |
| 1 | 0.403641 | 0.383250 |
| 2 | 0.429189 | 0.402641 |
| 3 | 0.483900 | 0.449857 |
| 4 | 0.580042 | 0.548756 |
| 5 | 0.690523 | 0.690543 |

| Test Case 7 - $\alpha = 0.1$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.932111 | 0.930130 |
| 1 | 0.894123 | 0.893292 |
| 2 | 0.853890 | 0.855765 |
| 3 | 0.799064 | 0.804504 |
| 4 | 0.682341 | 0.693302 |
| 5 | 0.091243 | 0.091242 |

| Test Case 7 - $\alpha = 0.2$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.864222 | 0.860260 |
| 1 | 0.837430 | 0.833746 |
| 2 | 0.811937 | 0.810663 |
| 3 | 0.776744 | 0.779443 |
| 4 | 0.689501 | 0.698037 |
| 5 | 0.182485 | 0.182484 |

| Test Case 7 - $\alpha = 0.3$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.796333 | 0.790389 |
| 1 | 0.780737 | 0.774201 |
| 2 | 0.769984 | 0.765562 |
| 3 | 0.754425 | 0.754383 |
| 4 | 0.696660 | 0.702772 |
| 5 | 0.273728 | 0.273727 |

| Test Case 7 - $\alpha = 0.4$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.728444 | 0.720519 |
| 1 | 0.724044 | 0.714655 |
| 2 | 0.728031 | 0.720461 |
| 3 | 0.732105 | 0.729322 |
| 4 | 0.703820 | 0.707507 |
| 5 | 0.364971 | 0.364969 |

| Test Case 7 - $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.660555 | 0.650649 |
| 1 | 0.667351 | 0.655110 |
| 2 | 0.686079 | 0.675360 |
| 3 | 0.709786 | 0.704262 |
| 4 | 0.710980 | 0.712241 |
| 5 | 0.456213 | 0.456211 |

| Test Case 7 - $\alpha = 0.6$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.592666 | 0.580779 |
| 1 | 0.610659 | 0.595564 |
| 2 | 0.644126 | 0.630258 |
| 3 | 0.687467 | 0.679202 |
| 4 | 0.718140 | 0.716976 |
| 5 | 0.547456 | 0.547453 |

| Test Case 7 - $\alpha = 0.7$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.524777 | 0.510909 |
| 1 | 0.553966 | 0.536019 |
| 2 | 0.602173 | 0.585157 |
| 3 | 0.665147 | 0.654141 |
| 4 | 0.725300 | 0.721711 |
| 5 | 0.638699 | 0.638695 |

| Test Case 7 - $\alpha = 0.8$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.456888 | 0.441038 |
| 1 | 0.497273 | 0.476473 |
| 2 | 0.560220 | 0.540056 |
| 3 | 0.642828 | 0.629081 |
| 4 | 0.732459 | 0.726446 |
| 5 | 0.729942 | 0.729938 |

| Test Case 7 - $\alpha = 0.9$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.388999 | 0.371168 |
| 1 | 0.440580 | 0.416927 |
| 2 | 0.518267 | 0.494954 |
| 3 | 0.620508 | 0.604020 |
| 4 | 0.739619 | 0.731181 |
| 5 | 0.821184 | 0.821180 |

| Test Case 8 – $\alpha = 0.1$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.956106 | 0.954829 |
| 1 | 0.840565 | 0.841185 |
| 2 | 0.723159 | 0.724043 |
| 3 | 0.587701 | 0.597793 |
| 4 | 0.384059 | 0.442394 |
| 5 | 0.072930 | 0.072928 |

| Test Case 8 – $\alpha = 0.2$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.912213 | 0.909658 |
| 1 | 0.809390 | 0.809480 |
| 2 | 0.706025 | 0.706584 |
| 3 | 0.589913 | 0.596367 |
| 4 | 0.414810 | 0.462131 |
| 5 | 0.145861 | 0.145855 |

| Test Case 8 – $\alpha = 0.3$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.868319 | 0.864487 |
| 1 | 0.778215 | 0.777774 |
| 2 | 0.688891 | 0.689126 |
| 3 | 0.589913 | 0.594941 |
| 4 | 0.476311 | 0.481868 |
| 5 | 0.218791 | 0.218783 |

| Test Case 8 – $\alpha = 0.4$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.824425 | 0.819316 |
| 1 | 0.747039 | 0.746068 |
| 2 | 0.671757 | 0.671667 |
| 3 | 0.591018 | 0.593515 |
| 4 | 0.476311 | 0.501605 |
| 5 | 0.291722 | 0.291711 |

| Test Case 8 – $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.780531 | 0.774145 |
| 1 | 0.715863 | 0.714363 |
| 2 | 0.654622 | 0.654208 |
| 3 | 0.592124 | 0.592089 |
| 4 | 0.507061 | 0.521342 |
| 5 | 0.364652 | 0.364638 |

| Test Case 8 – $\alpha = 0.6$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.736638 | 0.728974 |
| 1 | 0.684688 | 0.682657 |
| 2 | 0.637488 | 0.636749 |
| 3 | 0.593229 | 0.590663 |
| 4 | 0.537811 | 0.541078 |
| 5 | 0.437582 | 0.437566 |

| Test Case 8 – $\alpha = 0.7$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.692744 | 0.683803 |
| 1 | 0.653513 | 0.650951 |
| 2 | 0.620354 | 0.619290 |
| 3 | 0.594334 | 0.589237 |
| 4 | 0.568562 | 0.560815 |
| 5 | 0.510513 | 0.510494 |

| Test Case 8 – $\alpha = 0.8$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.648850 | 0.638632 |
| 1 | 0.622337 | 0.619245 |
| 2 | 0.603220 | 0.601832 |
| 3 | 0.595440 | 0.587811 |
| 4 | 0.599312 | 0.580552 |
| 5 | 0.583443 | 0.583422 |

| Test Case 8 – $\alpha = 0.9$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.604957 | 0.593461 |
| 1 | 0.591162 | 0.587540 |
| 2 | 0.586086 | 0.584373 |
| 3 | 0.595440 | 0.586385 |
| 4 | 0.630063 | 0.600289 |
| 5 | 0.656374 | 0.656349 |

| Test Case 9 – $\alpha = 0.1$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.923005 | 0.921841 |
| 1 | 0.873109 | 0.872577 |
| 2 | 0.827738 | 0.829064 |
| 3 | 0.777832 | 0.782888 |
| 4 | 0.679812 | 0.690962 |
| 5 | 0.087408 | 0.087408 |

| Test Case 9 – $\alpha = 0.2$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.846011 | 0.843682 |
| 1 | 0.805396 | 0.802924 |
| 2 | 0.772250 | 0.770498 |
| 3 | 0.740471 | 0.741916 |
| 4 | 0.673525 | 0.681425 |
| 5 | 0.174816 | 0.174816 |

| Test Case 9 – $\alpha = 0.3$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.769016 | 0.765523 |
| 1 | 0.737682 | 0.733272 |
| 2 | 0.716762 | 0.711931 |
| 3 | 0.703111 | 0.700944 |
| 4 | 0.667237 | 0.671888 |
| 5 | 0.262223 | 0.262223 |

| Test Case 9 – $\alpha = 0.4$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.692022 | 0.687364 |
| 1 | 0.669968 | 0.663619 |
| 2 | 0.661274 | 0.653365 |
| 3 | 0.665750 | 0.659972 |
| 4 | 0.660950 | 0.662351 |
| 5 | 0.349631 | 0.349631 |

| Test Case 9 – $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.615027 | 0.609205 |
| 1 | 0.602254 | 0.593967 |
| 2 | 0.605787 | 0.594799 |
| 3 | 0.628389 | 0.619000 |
| 4 | 0.654663 | 0.652814 |
| 5 | 0.437039 | 0.437039 |

| Test Case 9 – $\alpha = 0.6$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.538032 | 0.531045 |
| 1 | 0.534541 | 0.524315 |
| 2 | 0.550299 | 0.536232 |
| 3 | 0.591028 | 0.578027 |
| 4 | 0.648376 | 0.643278 |
| 5 | 0.524447 | 0.524447 |

| Test Case 9 – $\alpha = 0.7$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.461038 | 0.452886 |
| 1 | 0.466827 | 0.454662 |
| 2 | 0.494811 | 0.477666 |
| 3 | 0.553667 | 0.537055 |
| 4 | 0.642089 | 0.633741 |
| 5 | 0.611855 | 0.611855 |

| Test Case 9 – $\alpha = 0.8$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.384043 | 0.374727 |
| 1 | 0.399113 | 0.385010 |
| 2 | 0.439323 | 0.419099 |
| 2 | 0.516307 | 0.496083 |
| 4 | 0.635801 | 0.624204 |
| 5 | 0.699262 | 0.699262 |

| Test Case 9 – $\alpha = 0.9$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.307049 | 0.296568 |
| 1 | 0.331400 | 0.315357 |
| 2 | 0.383835 | 0.360532 |
| 3 | 0.478946 | 0.455111 |
| 4 | 0.629514 | 0.614667 |
| 5 | 0.786670 | 0.786670 |

| Test Case 10 – $\alpha = 0.1$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.961050 | 0.959105 |
| 1 | 0.864410 | 0.867672 |
| 2 | 0.754443 | 0.768115 |
| 3 | 0.611158 | 0.649256 |
| 4 | 0.380143 | 0.471189 |
| 5 | 0.080476 | 0.080475 |

| Test Case 10 – $\alpha = 0.2$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.922100 | 0.918209 |
| 1 | 0.838080 | 0.838543 |
| 2 | 0.743485 | 0.752363 |
| 3 | 0.620921 | 0.650344 |
| 4 | 0.422469 | 0.498321 |
| 5 | 0.160951 | 0.160950 |

| Test Case 10 – $\alpha = 0.3$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.883150 | 0.877314 |
| 1 | 0.811751 | 0.809415 |
| 2 | 0.732528 | 0.736611 |
| 3 | 0.630683 | 0.651433 |
| 4 | 0.464795 | 0.525453 |
| 5 | 0.241426 | 0.241425 |

| Test Case 10 – $\alpha = 0.4$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.844200 | 0.836418 |
| 1 | 0.785422 | 0.780286 |
| 2 | 0.721570 | 0.720859 |
| 3 | 0.640446 | 0.652521 |
| 4 | 0.507121 | 0.552585 |
| 5 | 0.321902 | 0.321900 |

| Test Case 10 – $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.805249 | 0.795523 |
| 1 | 0.759092 | 0.751157 |
| 2 | 0.710612 | 0.705106 |
| 3 | 0.650209 | 0.653610 |
| 4 | 0.549447 | 0.579717 |
| 5 | 0.402377 | 0.402375 |

| Test Case 10 – $\alpha = 0.6$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.766299 | 0.754628 |
| 1 | 0.732763 | 0.722028 |
| 2 | 0.699654 | 0.689354 |
| 3 | 0.659972 | 0.654699 |
| 4 | 0.591774 | 0.606849 |
| 5 | 0.482853 | 0.482851 |

| Test Case 10 – $\alpha = 0.7$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.727349 | 0.713732 |
| 1 | 0.706434 | 0.692899 |
| 2 | 0.688696 | 0.673602 |
| 3 | 0.669735 | 0.655787 |
| 4 | 0.634100 | 0.633981 |
| 5 | 0.563329 | 0.563326 |

| Test Case 10 – $\alpha = 0.8$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.688399 | 0.672837 |
| 1 | 0.680105 | 0.663771 |
| 2 | 0.677739 | 0.657850 |
| 3 | 0.679497 | 0.656876 |
| 4 | 0.676426 | 0.661113 |
| 5 | 0.643804 | 0.643801 |

| Test Case 10 – $\alpha = 0.9$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.649449 | 0.631941 |
| 1 | 0.653775 | 0.634642 |
| 2 | 0.666781 | 0.642098 |
| 3 | 0.689260 | 0.657964 |
| 4 | 0.718752 | 0.688245 |
| 5 | 0.724279 | 0.724276 |

| Test Case 11 – $\alpha = 0.1$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.928063 | 0.927736 |
| 1 | 0.867430 | 0.867367 |
| 2 | 0.807434 | 0.809320 |
| 3 | 0.737499 | 0.746530 |
| 4 | 0.611389 | 0.639725 |
| 5 | 0.082663 | 0.082665 |

| Test Case 11 – $\alpha = 0.2$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.856126 | 0.855473 |
| 1 | 0.804906 | 0.804118 |
| 2 | 0.757332 | 0.756765 |
| 3 | 0.705182 | 0.709218 |
| 4 | 0.610216 | 0.630857 |
| 5 | 0.165327 | 0.165330 |

| Test Case 11 – $\alpha = 0.3$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.784190 | 0.783209 |
| 1 | 0.742382 | 0.740868 |
| 2 | 0.707231 | 0.704210 |
| 3 | 0.672866 | 0.671907 |
| 4 | 0.609042 | 0.621990 |
| 5 | 0.247990 | 0.247995 |

| Test Case 11 – $\alpha = 0.4$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.712253 | 0.710946 |
| 1 | 0.679857 | 0.677619 |
| 2 | 0.657130 | 0.651655 |
| 3 | 0.640549 | 0.634595 |
| 4 | 0.607868 | 0.613122 |
| 5 | 0.330654 | 0.330660 |

| Test Case 11 – $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.640316 | 0.638682 |
| 1 | 0.617332 | 0.614370 |
| 2 | 0.607028 | 0.599100 |
| 3 | 0.608233 | 0.597283 |
| 4 | 0.606694 | 0.604255 |
| 5 | 0.413317 | 0.413326 |

| Test Case 11 – $\alpha = 0.6$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.568379 | 0.566418 |
| 1 | 0.554808 | 0.551121 |
| 2 | 0.556927 | 0.546545 |
| 3 | 0.575917 | 0.559971 |
| 4 | 0.605521 | 0.595388 |
| 5 | 0.495980 | 0.495991 |

| Test Case 11 – $\alpha = 0.7$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.496442 | 0.494155 |
| 1 | 0.492283 | 0.487872 |
| 2 | 0.506826 | 0.493990 |
| 3 | 0.543600 | 0.522659 |
| 4 | 0.604347 | 0.586520 |
| 5 | 0.578644 | 0.578656 |

| Test Case 11 – $\alpha = 0.8$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.424506 | 0.421891 |
| 1 | 0.429759 | 0.424622 |
| 2 | 0.456725 | 0.441435 |
| 3 | 0.511284 | 0.485348 |
| 4 | 0.603173 | 0.577653 |
| 5 | 0.661307 | 0.661321 |

| Test Case 11 – $\alpha = 0.9$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.352569 | 0.349628 |
| 1 | 0.367234 | 0.361373 |
| 2 | 0.406623 | 0.388880 |
| 3 | 0.478967 | 0.448036 |
| 4 | 0.602000 | 0.568785 |
| 5 | 0.743971 | 0.743986 |

| Test Case 12 – $\alpha = 0.1$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.947203 | 0.950219 |
| 1 | 0.881449 | 0.880475 |
| 2 | 0.808845 | 0.806677 |
| 3 | 0.713842 | 0.721793 |
| 4 | 0.530914 | 0.583733 |
| 5 | 0.081640 | 0.081638 |

| Test Case 12 – $\alpha = 0.2$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.894405 | 0.900437 |
| 1 | 0.837965 | 0.839857 |
| 2 | 0.777193 | 0.776403 |
| 3 | 0.699526 | 0.704682 |
| 4 | 0.549111 | 0.590940 |
| 5 | 0.163280 | 0.163276 |

| Test Case 12 – $\alpha = 0.3$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.841608 | 0.850656 |
| 1 | 0.794480 | 0.799240 |
| 2 | 0.745540 | 0.746129 |
| 3 | 0.685210 | 0.687571 |
| 4 | 0.567308 | 0.598148 |
| 5 | 0.244920 | 0.244915 |

| Test Case 12 – $\alpha = 0.4$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.788810 | 0.800874 |
| 1 | 0.750996 | 0.758622 |
| 2 | 0.713887 | 0.715855 |
| 3 | 0.670894 | 0.670460 |
| 4 | 0.585505 | 0.605356 |
| 5 | 0.326560 | 0.326553 |

| Test Case 12 – $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.736013 | 0.751093 |
| 1 | 0.707511 | 0.718005 |
| 2 | 0.682234 | 0.685582 |
| 3 | 0.656578 | 0.653348 |
| 4 | 0.603703 | 0.612564 |
| 5 | 0.408201 | 0.408191 |

| Test Case 12 – $\alpha = 0.6$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.683216 | 0.701312 |
| 1 | 0.664026 | 0.677388 |
| 2 | 0.650582 | 0.655308 |
| 3 | 0.642263 | 0.636237 |
| 4 | 0.621900 | 0.619771 |
| 5 | 0.489841 | 0.489829 |

| Test Case 12 – $\alpha = 0.7$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.630418 | 0.651530 |
| 1 | 0.620542 | 0.636770 |
| 2 | 0.618929 | 0.625034 |
| 3 | 0.627947 | 0.619126 |
| 4 | 0.640097 | 0.626979 |
| 5 | 0.571481 | 0.571467 |

| Test Case 12 – $\alpha = 0.8$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.577621 | 0.601749 |
| 1 | 0.577057 | 0.596153 |
| 2 | 0.587276 | 0.594760 |
| 3 | 0.613631 | 0.602015 |
| 4 | 0.658294 | 0.634187 |
| 5 | 0.653121 | 0.653106 |

| Test Case 12 – $\alpha = 0.9$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.524823 | 0.551967 |
| 1 | 0.533573 | 0.555535 |
| 2 | 0.555624 | 0.564486 |
| 3 | 0.599315 | 0.584904 |
| 4 | 0.676491 | 0.641394 |
| 5 | 0.734761 | 0.734744 |

| Test Case 13 − $\alpha = 0.1$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.932737 | 0.933732 |
| 1 | 0.894425 | 0.894274 |
| 2 | 0.851842 | 0.853101 |
| 3 | 0.793454 | 0.801956 |
| 4 | 0.665026 | 0.698854 |
| 5 | 0.087253 | 0.087251 |

| Test Case 13 − $\alpha = 0.2$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.865473 | 0.867464 |
| 1 | 0.835705 | 0.835086 |
| 2 | 0.804740 | 0.803170 |
| 3 | 0.763717 | 0.766766 |
| 4 | 0.666865 | 0.691795 |
| 5 | 0.174505 | 0.174503 |

| Test Case 13 − $\alpha = 0.3$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.798210 | 0.801197 |
| 1 | 0.776985 | 0.775897 |
| 2 | 0.757637 | 0.753238 |
| 3 | 0.733980 | 0.731577 |
| 4 | 0.668705 | 0.684735 |
| 5 | 0.261757 | 0.261754 |

| Test Case 13 − $\alpha = 0.4$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.730946 | 0.734929 |
| 1 | 0.718265 | 0.716709 |
| 2 | 0.710534 | 0.703307 |
| 3 | 0.704243 | 0.696387 |
| 4 | 0.670545 | 0.677676 |
| 5 | 0.349010 | 0.349005 |

| Test Case 13 − $\alpha = 0.5$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.663683 | 0.668661 |
| 1 | 0.659546 | 0.657521 |
| 2 | 0.663431 | 0.653375 |
| 3 | 0.674505 | 0.661197 |

| Test Case 13 − $\alpha = 0.6$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.596420 | 0.602393 |
| 1 | 0.600826 | 0.598333 |
| 2 | 0.616329 | 0.603443 |
| 3 | 0.644768 | 0.626007 |
| 4 | 0.674224 | 0.663556 |
| 5 | 0.523515 | 0.523508 |

| Test Case 13 − $\alpha = 0.7$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.529156 | 0.536125 |
| 1 | 0.542106 | 0.539145 |
| 2 | 0.569226 | 0.553512 |
| 3 | 0.615031 | 0.590817 |
| 4 | 0.676064 | 0.656497 |
| 5 | 0.610767 | 0.610759 |

| Test Case 13 − $\alpha = 0.8$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.461893 | 0.469858 |
| 1 | 0.483386 | 0.479956 |
| 2 | 0.522123 | 0.503580 |
| 3 | 0.585294 | 0.555628 |
| 4 | 0.677904 | 0.649437 |
| 5 | 0.698020 | 0.698010 |

| Test Case 13 − $\alpha = 0.9$ | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 0 | 0.394629 | 0.403590 |
| 1 | 0.424666 | 0.420768 |
| 2 | 0.475021 | 0.453649 |
| 3 | 0.555557 | 0.520438 |
| 4 | 0.679743 | 0.642378 |
| 5 | 0.785273 | 0.785262 |

| Test Case 13 − $\alpha = 0.5$ (cont.) | | |
|---|---|---|
| V | Theoretical $F$ | Simulation $F$ |
| 4 | 0.6727385 | 0.670616 |
| 5 | 0.436262 | 0.436256 |

**Table B.12**: Optimal capacity distribution obtained by theoretical calculation and simulation results.

## B.4 Optimal VPDBA Solution for a Simple Network

### B.4.1 Test Cases

| Test Case 1 | Test Case 2 | Test Case 3 | Test Case 4 |
|---|---|---|---|
| 5 nodes | 5 nodes | 5 nodes | 5 nodes |
| 5 servers/node | 5 servers/node | 5 servers/node | 5 servers/node |
| $\lambda_{01} = 0.1$ | $\lambda_{01} = 0.25$ | $\lambda_{01} = 0.2$ | $\lambda_{01} = 0.2$ |
| $\lambda_{02} = 0.1$ | $\lambda_{02} = 0.25$ | $\lambda_{02} = 1.1$ | $\lambda_{02} = 1.1$ |
| $\lambda_{03} = 10.0$ | $\lambda_{03} = 0.25$ | $\lambda_{03} = 0.5$ | $\lambda_{03} = 0.2$ |
| $\lambda_{04} = 0.1$ | $\lambda_{04} = 0.3$ | $\lambda_{04} = 0.7$ | $\lambda_{04} = 0.1$ |
| $\lambda_{12} = 0.1$ | $\lambda_{12} = 0.3$ | $\lambda_{12} = 0.3$ | $\lambda_{12} = 0.5$ |
| $\lambda_{13} = 0.1$ | $\lambda_{13} = 0.4$ | $\lambda_{13} = 0.3$ | $\lambda_{13} = 0.2$ |
| $\lambda_{14} = 0.1$ | $\lambda_{14} = 0.25$ | $\lambda_{14} = 0.2$ | $\lambda_{14} = 0.2$ |
| $\lambda_{23} = 0.1$ | $\lambda_{23} = 0.25$ | $\lambda_{23} = 0.2$ | $\lambda_{23} = 0.2$ |
| $\lambda_{24} = 0.1$ | $\lambda_{24} = 0.3$ | $\lambda_{24} = 0.2$ | $\lambda_{24} = 0.2$ |
| $\lambda_{34} = 0.1$ | $\lambda_{34} = 0.5$ | $\lambda_{34} = 0.3$ | $\lambda_{34} = 0.2$ |
| $\mu = 1.0$ | $\mu = 1.0$ | $\mu = 1.0$ | $\mu = 1.0$ |

| Test Case 5 | Test Case 6 | Test Case 7 | Test Case 8 |
|---|---|---|---|
| 5 nodes | 5 nodes | 5 nodes | 5 nodes |
| 5 servers/node | 5 servers/node | 5 servers/node | 5 servers/node |
| $\lambda_{01} = 0.2$ | $\lambda_{01} = 0.2$ | $\lambda_{01} = 0.1$ | $\lambda_{01} = 0.6$ |
| $\lambda_{02} = 0.2$ | $\lambda_{02} = 0.7$ | $\lambda_{02} = 0.1$ | $\lambda_{02} = 0.2$ |
| $\lambda_{03} = 0.2$ | $\lambda_{03} = 0.2$ | $\lambda_{03} = 0.4$ | $\lambda_{03} = 0.2$ |
| $\lambda_{04} = 0.3$ | $\lambda_{04} = 0.2$ | $\lambda_{04} = 0.1$ | $\lambda_{04} = 0.2$ |
| $\lambda_{12} = 0.2$ | $\lambda_{12} = 0.2$ | $\lambda_{12} = 0.1$ | $\lambda_{12} = 0.2$ |
| $\lambda_{13} = 0.2$ | $\lambda_{13} = 0.2$ | $\lambda_{13} = 0.1$ | $\lambda_{13} = 0.2$ |
| $\lambda_{14} = 0.5$ | $\lambda_{14} = 0.2$ | $\lambda_{14} = 0.1$ | $\lambda_{14} = 0.2$ |
| $\lambda_{23} = 0.2$ | $\lambda_{23} = 0.2$ | $\lambda_{23} = 0.9$ | $\lambda_{23} = 0.2$ |
| $\lambda_{24} = 0.2$ | $\lambda_{24} = 0.7$ | $\lambda_{24} = 0.1$ | $\lambda_{24} = 0.2$ |
| $\lambda_{34} = 0.2$ | $\lambda_{34} = 0.2$ | $\lambda_{34} = 0.1$ | $\lambda_{34} = 0.2$ |
| $\mu = 1.0$ | $\mu = 1.0$ | $\mu = 1.0$ | $\mu = 1.0$ |

| Test Case 9 | | | | |
|---|---|---|---|---|
| 10 nodes | | | | |
| 5 servers per node | | | | |
| $\lambda_{01} = 0.05$ | $\lambda_{12} = 0.05$ | $\lambda_{24} = 0.05$ | $\lambda_{37} = 0.05$ | $\lambda_{57} = 0.05$ |
| $\lambda_{02} = 0.05$ | $\lambda_{13} = 0.05$ | $\lambda_{25} = 0.05$ | $\lambda_{38} = 0.05$ | $\lambda_{58} = 0.05$ |
| $\lambda_{03} = 0.05$ | $\lambda_{14} = 0.05$ | $\lambda_{26} = 0.05$ | $\lambda_{39} = 0.05$ | $\lambda_{59} = 0.05$ |
| $\lambda_{04} = 1.05$ | $\lambda_{15} = 0.05$ | $\lambda_{27} = 0.05$ | $\lambda_{45} = 0.05$ | $\lambda_{67} = 0.2$ |
| $\lambda_{05} = 0.05$ | $\lambda_{16} = 0.05$ | $\lambda_{28} = 0.05$ | $\lambda_{46} = 0.7$ | $\lambda_{68} = 0.2$ |
| $\lambda_{06} = 0.05$ | $\lambda_{17} = 0.05$ | $\lambda_{29} = 0.05$ | $\lambda_{47} = 0.05$ | $\lambda_{69} = 0.2$ |
| $\lambda_{07} = 0.05$ | $\lambda_{18} = 0.05$ | $\lambda_{34} = 0.05$ | $\lambda_{48} = 0.05$ | $\lambda_{78} = 0.2$ |
| $\lambda_{08} = 0.05$ | $\lambda_{19} = 0.05$ | $\lambda_{35} = 0.05$ | $\lambda_{49} = 0.05$ | $\lambda_{79} = 0.2$ |
| $\lambda_{09} = 0.05$ | $\lambda_{23} = 0.05$ | $\lambda_{36} = 0.05$ | $\lambda_{56} = 0.05$ | $\lambda_{89} = 0.3$ |

| Test Case 10 | | | | |
|---|---|---|---|---|
| 10 nodes | | | | |
| 5 servers per node | | | | |
| $\lambda_{01} = 0.05$ | $\lambda_{12} = 0.05$ | $\lambda_{24} = 0.05$ | $\lambda_{37} = 1.3$ | $\lambda_{57} = 0.06$ |
| $\lambda_{02} = 0.05$ | $\lambda_{13} = 0.05$ | $\lambda_{25} = 0.05$ | $\lambda_{38} = 0.02$ | $\lambda_{58} = 0.06$ |
| $\lambda_{03} = 1.1$ | $\lambda_{14} = 0.05$ | $\lambda_{26} = 0.05$ | $\lambda_{39} = 0.02$ | $\lambda_{59} = 0.06$ |
| $\lambda_{04} = 0.05$ | $\lambda_{15} = 0.05$ | $\lambda_{27} = 0.05$ | $\lambda_{45} = 0.02$ | $\lambda_{67} = 0.02$ |
| $\lambda_{05} = 0.05$ | $\lambda_{16} = 0.05$ | $\lambda_{28} = 0.05$ | $\lambda_{46} = 0.02$ | $\lambda_{68} = 0.02$ |
| $\lambda_{06} = 0.05$ | $\lambda_{17} = 0.05$ | $\lambda_{29} = 0.05$ | $\lambda_{47} = 0.02$ | $\lambda_{69} = 0.02$ |
| $\lambda_{07} = 0.05$ | $\lambda_{18} = 0.05$ | $\lambda_{34} = 0.02$ | $\lambda_{48} = 0.02$ | $\lambda_{78} = 0.5$ |
| $\lambda_{08} = 0.05$ | $\lambda_{19} = 0.05$ | $\lambda_{35} = 0.02$ | $\lambda_{49} = 0.02$ | $\lambda_{79} = 0.5$ |
| $\lambda_{09} = 0.05$ | $\lambda_{23} = 0.05$ | $\lambda_{36} = 0.02$ | $\lambda_{56} = 0.06$ | $\lambda_{89} = 0.6$ |

**Table B.13**: The test cases for the experiment described in Section 3.5.1.

# Appendix C

# Single Node Network Base Case

## C.1 Test Cases

| test case 1 | test case 2 |
|---|---|
| $i = 2, j = 1$ | $i = 2, j = 1$ |
| $\lambda_{vp_1} = 3.0$ | $\lambda_{vp_1} = 4.0$ |
| $\lambda_{vp_2} = 5.0$ | $\lambda_{vp_2} = 2.0$ |
| $\lambda = 1.5$ | $\lambda = 5.0$ |
| $\mu = 1.0$ | $\mu = 1.0$ |
| $K = 5$ | $K = 5$ |
| $V_1 = 2$ | $V_1 = 1$ |
| $V_2 = 2$ | $V_2 = 1$ |

211

| test case 3 | test case 4 |
|---|---|
| $i = 4, j = 1$ | $i = 4, j = 1$ |
| $\lambda_{vp_1} = 2.0$ | $\lambda_{vp_1} = 0.5$ |
| $\lambda_{vp_2} = 1.5$ | $\lambda_{vp_2} = 3.5$ |
| $\lambda_{vp_3} = 3.0$ | $\lambda_{vp_3} = 0.5$ |
| $\lambda_{vp_4} = 0.5$ | $\lambda_{vp_4} = 3.0$ |
| $\lambda = 1.25$ | $\lambda = 4.0$ |
| $\mu = 1.0$ | $\mu = 1.0$ |
| $K = 8$ | $K = 8$ |
| $V_1 = 2$ | $V_1 = 1$ |
| $V_2 = 2$ | $V_2 = 2$ |
| $V_3 = 2$ | $V_3 = 1$ |
| $V_4 = 1$ | $V_4 = 2$ |
| test case 5 | test case 6 |
| $i = 3, j = 1$ | $i = 3, j = 1$ |
| $\lambda_{vp_1} = 5.0$ | $\lambda_{vp_1} = 1.5$ |
| $\lambda_{vp_2} = 1.25$ | $\lambda_{vp_2} = 1.2$ |
| $\lambda_{vp_3} = 1.5$ | $\lambda_{vp_3} = 1.3$ |
| $\lambda = 1.0$ | $\lambda = 4.0$ |
| $\mu = 1.0$ | $\mu = 1.0$ |
| $K = 6$ | $K = 6$ |
| $V_1 = 2$ | $V_1 = 1$ |
| $V_2 = 1$ | $V_2 = 1$ |
| $V_3 = 1$ | $V_3 = 1$ |
| test case 7 | test case 8 |
| $i = 2, j = 1$ | $i = 3, j = 1$ |
| $\lambda_{vp_1} = 1.5$ | $\lambda_{vp_1} = 5.0$ |
| $\lambda_{vp_2} = 2.5$ | $\lambda_{vp_2} = 3.5$ |
| $\lambda = 3.5$ | $\lambda_{vp_3} = 3.0$ |
| $\mu = 1.0$ | $\lambda = 0.5$ |
| $K = 5$ | $\mu = 1.0$ |
| $V_1 = 1$ | $K = 6$ |
| $V_2 = 1$ | $V_1 = 1$ |
| | $V_2 = 1$ |
| | $V_3 = 1$ |

| test case 9 | test case 10 |
|---|---|
| $i = 10, j = 1$ | $i = 10, j = 1$ |
| $\lambda_{vp_1} = 4.0$ | $\lambda_{vp_1} = 5.0$ |
| $\lambda_{vp_2} = 4.0$ | $\lambda_{vp_2} = 5.0$ |
| $\lambda_{vp_3} = 3.0$ | $\lambda_{vp_3} = 5.0$ |
| $\lambda_{vp_4} = 3.0$ | $\lambda_{vp_4} = 2.0$ |
| $\lambda_{vp_5} = 2.0$ | $\lambda_{vp_5} = 2.0$ |
| $\lambda_{vp_6} = 2.0$ | $\lambda_{vp_6} = 2.0$ |
| $\lambda_{vp_7} = 2.0$ | $\lambda_{vp_7} = 2.0$ |
| $\lambda_{vp_8} = 2.0$ | $\lambda_{vp_8} = 1.0$ |
| $\lambda_{vp_9} = 2.0$ | $\lambda_{vp_9} = 1.0$ |
| $\lambda_{vp_{10}} = 2.0$ | $\lambda_{vp_{10}} = 1.0$ |
| $\lambda = 2.0$ | $\lambda = 3.0$ |
| $\mu = 1.0$ | $\mu = 1.0$ |
| $K = 13$ | $K = 22$ |
| $V_1 = 1$ | $V_1 = 2$ |
| $V_2 = 1$ | $V_2 = 2$ |
| $V_3 = 1$ | $V_3 = 2$ |
| $V_4 = 1$ | $V_4 = 2$ |
| $V_5 = 1$ | $V_5 = 2$ |
| $V_6 = 1$ | $V_6 = 2$ |
| $V_7 = 1$ | $V_7 = 2$ |
| $V_8 = 1$ | $V_8 = 2$ |
| $V_9 = 1$ | $V_9 = 2$ |
| $V_{10} = 1$ | $V_{10} = 2$ |

**Table C.1:** The test cases used in the single node, multiple VP model.

# Appendix D

# General Line Networks

## D.1 Using the Performance Measure Approximation Method

### D.1.1 Validity of Performance Measure Approximation Method

#### D.1.1.1 Test Cases

```
Group 1
nodes: 7
vps : 5
svrs : 7
lambdas: 0.25, 0.5, 0.75, 4.0, 5.0, 0.1, 0.1, 0.1, 0.15, 0.2, 0.2, 0.1, 0.2, 0.2, 0.1, 0.1,
         0.1, 0.1, 0.1, 0.1, 0.2
tc 1) VPs: 01 - 1, 02 - 1, 03 - 1, 04 - 2, 05 - 2
tc 2) VPs: 01 - 1, 12 - 1, 23 - 1, 34 - 1, 45 - 1
tc 3) VPs: 35 - 1, 36 - 1, 45 - 1, 46 - 1, 56 - 1
tc 4) VPs: 03 - 1, 04 - 1, 05 - 1, 46 - 1, 56 - 1
tc 5) VPs: 04 - 2, 05 - 2, 15 - 1, 16 - 1, 24 - 1
```

214

```
Group 2
nodes: 10
vps : 4
svrs : 10
lambdas: 0.2, 0.5, 0.7, 0.1, 0.1, 0.6, 0.2, 0.2, 0.1, 0.3, 0.5, 0.5, 0.1, 0.1, 0.1, 0.2,
         0.1, 0.2, 0.9, 0.1, 0.1, 0.1, 0.2, 0.3, 0.1, 0.1, 1.5, 0.2, 0.1, 0.1, 0.1, 0.7,
         0.2, 0.2, 0.2, 0.4, 0.5, 0.1, 0.1, 0.1, 0.1, 0.1, 1.2, 0.3, 0.5
tc 6) VPs: 24 - 1, 36 - 2, 46 - 1, 78 - 2
tc 7) VPs: 03 - 1, 06 - 1, 24 - 1, 78 - 1
tc 8) VPs: 03 - 1, 06 - 1, 24 - 1, 46 - 1
tc 9) VPs: 01 - 2, 02 - 2, 03 - 2, 04 - 2
tc 10) VPs: 03 - 2, 12 - 1, 23 - 2, 36 - 3
```

```
Group 3
nodes: 5
vps : 2
svrs : 4
lambdas: 0.1, 1.2, 0.7, 0.1, 0.1, 0.1, 1.2, 1.2, 0.1, 0.6
tc 11) VPs: 02 - 1, 03 - 1
tc 12) VPs: 02 - 1, 14 - 1
tc 13) VPs: 14 - 1, 23 - 2
tc 14) VPs: 02 - 2, 23 - 2
tc 15) VPs: 23 - 1, 34 - 1
```

```
Group 4
nodes: 6
vps : 3
svrs : 3
lambdas: 0.1, 0.1, 0.1, 0.1, 0.5, 0.7, 0.1, 0.1, 0.1, 0.5, 0.3, 0.1, 0.1, 0.1, 0.1
tc 16) VPs: 05 - 1, 12 - 1, 23 - 1
tc 17) VPs: 01 - 1, 02 - 1, 03 - 1
tc 18) VPs: 34 - 1, 35 - 1, 45 - 1
tc 19) VPs: 01 - 1, 02 - 1, 23 - 1
tc 20) VPs: 12 - 1, 23 - 1, 24 - 1
```

```
Group 5
nodes: 8
vps : 4
svrs : 6
lambdas: 2.0, 0.2, 0.1, 0.5, 0.1, 0.2, 0.3, 0.1, 1.5, 1.2, 0.1, 0.1, 0.2, 0.1, 0.1, 0.1,
         0.1, 0.9, 0.2, 0.5, 0.1, 0.1, 0.1, 0.1, 0.3, 0.1, 0.1, 0.8
tc 21) VPs: 01 - 2, 13 - 2, 14 - 1, 23 - 1
tc 22) VPs: 01 - 2, 14 - 1, 23 - 1, 67 - 1
tc 23) VPs: 01 - 1, 04 - 1, 13 - 1, 14 - 1
tc 24) VPs: 04 - 1, 13 - 1, 27 - 1, 67 - 1
tc 25) VPs: 04 - 1, 14 - 1, 35 - 1, 67 - 1
```

```
Group 6
nodes: 5
vps : 3
svrs : 5
lambdas: 0.1, 0.1, 0.1, 0.1, 2.0, 3.0, 0.5, 0.3, 0.1, 0.7
tc 26) VPs: 01 - 1, 02 - 1, 03 - 1
tc 27) VPs: 12 - 1, 13 - 2, 14 - 1
tc 28) VPs: 12 - 1, 13 - 2, 34 - 1
tc 29) VPs: 12 - 1, 13 - 3, 34 - 1
tc 30) VPs: 12 - 1, 13 - 1, 23 - 1
```

```
Group 7
nodes: 7
vps : 3
svrs : 5
lambdas: 0.2, 0.2, 0.2, 0.2, 0.2, 0.2, 1.2, 0.2, 0.2, 0.2, 0.7, 0.6, 0.3, 0.2, 0.2, 0.1,
         0.2, 0.1, 0.8, 0.1, 0.1
tc 31) VPs: 12 - 1, 16 - 1, 45 - 1
tc 32) VPs: 12 - 1, 23 - 1, 45 - 1
tc 33) VPs: 16 - 1, 23 - 1, 24 - 1
tc 34) VPs: 34 - 1, 35 - 3, 36 - 1
tc 35) VPs: 23 - 1, 24 - 1, 45 - 1
```

Group 8
nodes: 8
vps : 6
svrs : 10
lambdas: 2.0, 0.2, 1.5, 0.1, 0.1, 0.1, 0.9, 0.8, 1.5, 1.2, 0.1, 0.1, 0.1, 0.1, 0.1, 1.2,
    1.4, 0.1, 0.1, 0.1, 0.7, 0.1, 0.1, 0.5, 0.1, 1.2, 0.5, 2.1
tc 36) VPs: 01 - 1, 03 - 1, 07 - 1, 12 - 1, 13 - 1, 14 - 1
tc 37) VPs: 01 - 1, 12 - 1, 25 - 1, 26 - 1, 37 - 1, 46 - 1
tc 38) VPs: 01 - 1, 03 - 1, 13 - 1, 14 - 1, 25 - 1, 26 - 1
tc 39) VPs: 25 - 1, 26 - 1, 36 - 1, 46 - 1, 56 - 1, 67 - 1
tc 40) VPs: 03 - 1, 12 - 1, 13 - 1, 14 - 1, 56 - 1, 67 - 1

Group 9
nodes: 10
vps : 7
svrs : 8
lambdas: 0.2, 0.9, 1.5, 1.2, 0.1, 0.1, 0.7, 0.3, 0.2, 0.6, 0.3, 0.2, 0.2, 0.1, 1.5, 1.1,
    0.1, 1.7, 0.2, 0.1, 0.1, 0.2, 0.1, 0.3, 0.1, 0.7, 0.6, 0.5, 0.1, 0.1, 0.2, 0.2
    0.5, 0.9, 0.1, 0.1, 0.1, 1.5, 0.1, 1.2, 0.1, 0.1, 0.2, 0.8, 0.1
tc 41) VPs: 03 - 1, 04 - 1, 17 - 1, 23 - 1, 58 - 1, 67 - 1, 79 - 1
tc 42) VPs: 12 - 1, 17 - 1, 23 - 1, 48 - 1, 56 - 1, 67 - 1, 89 - 1
tc 43) VPs: 02 - 1, 03 - 1, 04 - 1, 17 - 1, 18 - 1, 23 - 1, 34 - 1
tc 44) VPs: 17 - 1, 23 - 1, 35 - 1, 48 - 1, 58 - 1, 67 - 1, 79 - 1
tc 45) VPs: 03 - 1, 04 - 1, 07 - 1, 17 - 1, 18 - 1, 23 - 1, 35 - 1

Group 10
nodes: 10
vps : 3
svrs : 5
lambdas: 0.1, 0.1, 1.5, 0.1, 0.1, 0.2, 0.5, 0.7, 0.1, 1.2, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1,
    0.1, 0.1, 0.9, 0.1, 0.1, 0.1, 0.1, 0.9, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1
    0.9, 1.5, 0.1, 1.5, 0.1, 0.1, 0.1, 0.1, 0.1, 1.2, 0.1, 0.1
tc 46) VPs: 03 - 2, 12 - 2, 24 - 1
tc 47) VPs: 03 - 1, 12 - 1, 24 - 1
tc 48) VPs: 03 - 1, 48 - 1, 78 - 1
tc 49) VPs: 12 - 2, 56 - 2, 78 - 2
tc 50) VPs: 03 - 2, 23 - 1, 29 - 1

Table D.1: Test Cases for the Approximation Method validity experiment.

## D.1.1.2 Resulting Performance Measures

| | Simulation | Approx | | Simulation | Approx |
|---|---|---|---|---|---|
| TC | loss | loss | TC | loss | loss |
| 1 | 0.549379 +/- 0.000502 | 0.554407 | 26 | 0.460865 +/- 0.000550 | 0.468077 |
| 2 | 0.495045 +/- 0.000647 | 0.594643 | 27 | 0.387924 +/- 0.000559 | 0.402469 |
| 3 | 0.503720 +/- 0.000611 | 0.625038 | 28 | 0.328512 +/- 0.000598 | 0.337274 |
| 4 | 0.490419 +/- 0.000617 | 0.450801 | 29 | 0.357098 +/- 0.000506 | 0.372691 |
| 5 | 0.629567 +/- 0.000457 | 0.530322 | 30 | 0.316844 +/- 0.000566 | 0.318484 |
| 6 | 0.073567 +/- 0.000383 | 0.089747 | 31 | 0.194324 +/- 0.000490 | 0.217626 |
| 7 | 0.069823 +/- 0.000394 | 0.086957 | 32 | 0.161011 +/- 0.000459 | 0.192002 |
| 8 | 0.076551 +/- 0.000403 | 0.093485 | 33 | 0.225118 +/- 0.000483 | 0.245247 |
| 9 | 0.114152 +/- 0.000450 | 0.129921 | 34 | 0.452609 +/ 0.000421 | 0.488913 |
| 10 | 0.100664 +/- 0.000418 | 0.126659 | 35 | 0.182791 +/- 0.000487 | 0.218268 |
| 11 | 0.295017 +/- 0.000518 | 0.260248 | 36 | 0.127131 +/- 0.000484 | 0.100877 |
| 12 | 0.277005 +/- 0.000531 | 0.245954 | 37 | 0.117944 +/- 0.000488 | 0.139692 |
| 13 | 0.295373 +/- 0.000458 | 0.299583 | 38 | 0.126389 +/- 0.000504 | 0.095151 |
| 14 | 0.274853 +/- 0.000498 | 0.280703 | 39 | 0.110950 +/- 0.000472 | 0.128465 |
| 15 | 0.243788 +/- 0.000512 | 0.291492 | 40 | 0.111459 +/- 0.000514 | 0.091604 |
| 16 | 0.303058 +/- 0.000442 | 0.291398 | 41 | 0.345124 +/- 0.000548 | 0.328658 |
| 17 | 0.432089 +/- 0.000424 | 0.441115 | 42 | 0.347917 +/- 0.000582 | 0.384114 |
| 18 | 0.273881 +/- 0.000439 | 0.33613 | 43 | 0.379874 +/- 0.000562 | 0.358472 |
| 19 | 0.277101 +/- 0.000468 | 0.286079 | 44 | 0.342274 +/- 0.000581 | 0.373529 |
| 20 | 0.281213 +/- 0.000427 | 0.303632 | 45 | 0.420412 +/- 0.000553 | 0.394339 |
| 21 | 0.248396 +/- 0.000493 | 0.216875 | 46 | 0.395872 +/- 0.000563 | 0.39224 |
| 22 | 0.217696 +/- 0.000517 | 0.22939 | 47 | 0.383244 +/- 0.000532 | 0.385417 |
| 23 | 0.232128 +/- 0.000532 | 0.216523 | 48 | 0.394456 +/- 0.000541 | 0.40211 |
| 24 | 0.235079 +/- 0.000514 | 0.23342 | 49 | 0.398088 +/- 0.000541 | 0.442564 |
| 25 | 0.234598 +/- 0.000501 | 0.249636 | 50 | 0.453218 +/- 0.000512 | 0.468268 |

Table D.2: Resulting P(loss) values calculated via simulation and our approximation method.

| TC | Simulation setup | Approx setup | TC | Simulation setup | Approx setup |
|----|------------------|--------------|----|------------------|--------------|
| 1  | 0.275599 +/- 0.000475 | 0.280555 | 26 | 0.927605 +/- 0.000290 | 0.919485 |
| 2  | 0.911990 +/- 0.000309 | 0.832144 | 27 | 0.437290 +/- 0.000472 | 0.442896 |
| 3  | 0.915457 +/- 0.000316 | 0.84706  | 28 | 0.469870 +/- 0.000465 | 0.460163 |
| 4  | 0.640014 +/- 0.000490 | 0.660909 | 29 | 0.324430 +/- 0.000450 | 0.310393 |
| 5  | 0.218178 +/- 0.000520 | 0.414874 | 30 | 0.655028 +/- 0.000421 | 0.64746  |
| 6  | 0.772498 +/- 0.000305 | 0.763891 | 31 | 0.728424 +/- 0.000311 | 0.712814 |
| 7  | 0.855233 +/- 0.000241 | 0.848767 | 32 | 0.746230 +/- 0.000326 | 0.716627 |
| 8  | 0.865142 +/- 0.000217 | 0.859216 | 33 | 0.795273 +/- 0.000316 | 0.790254 |
| 9  | 0.884757 +/- 0.000251 | 0.875826 | 34 | 0.891150 +/ 0.000363 | 0.880129 |
| 10 | 0.806275 +/- 0.000292 | 0.790836 | 35 | 0.799126 +/- 0.000299 | 0.775373 |
| 11 | 0.748769 +/- 0.000317 | 0.758968 | 36 | 0.776752 +/- 0.000282 | 0.781466 |
| 12 | 0.720482 +/- 0.000345 | 0.741089 | 37 | 0.823243 +/- 0.000269 | 0.81513  |
| 13 | 0.618813 +/- 0.000397 | 0.612063 | 38 | 0.762918 +/- 0.000318 | 0.774272 |
| 14 | 0.538885 +/- 0.000406 | 0.49334  | 39 | 0.796559 +/- 0.000288 | 0.781698 |
| 15 | 0.774631 +/- 0.000294 | 0.74037  | 40 | 0.775405 +/- 0.000317 | 0.738031 |
| 16 | 0.501293 +/- 0.000433 | 0.488949 | 41 | 0.701423 +/- 0.000388 | 0.639387 |
| 17 | 0.845331 +/- 0.000394 | 0.842008 | 42 | 0.788092 +/- 0.000347 | 0.652571 |
| 18 | 0.878750 +/- 0.000306 | 0.859503 | 43 | 0.725109 +/- 0.000375 | 0.737725 |
| 19 | 0.770539 +/- 0.000363 | 0.740103 | 44 | 0.722342 +/- 0.000353 | 0.636102 |
| 20 | 0.562476 +/- 0.000402 | 0.516401 | 45 | 0.683397 +/- 0.000409 | 0.70844  |
| 21 | 0.629237 +/- 0.000397 | 0.620786 | 46 | 0.722550 +/- 0.000391 | 0.698776 |
| 22 | 0.717075 +/- 0.000342 | 0.690592 | 47 | 0.817454 +/- 0.000319 | 0.799109 |
| 23 | 0.728978 +/- 0.000346 | 0.726632 | 48 | 0.799709 +/- 0.000343 | 0.787197 |
| 24 | 0.764955 +/- 0.000320 | 0.762816 | 49 | 0.672234 +/- 0.000420 | 0.578402 |
| 25 | 0.789983 +/- 0.000315 | 0.777636 | 50 | 0.796922 +/- 0.000335 | 0.794365 |

**Table D.3**: Resulting P(setup) values calculated via simulation and our approximation method.

| | Simulation | Approx | | Simulation | Approx |
|---|---|---|---|---|---|
| TC | nosetup | nosetup | TC | nosetup | nosetup |
| 1 | 0.326433 +/- 0.000439 | 0.32058 | 26 | 0.039031 +/- 0.000161 | 0.042828 |
| 2 | 0.044437 +/- 0.000156 | 0.068042 | 27 | 0.344419 +/- 0.000408 | 0.332886 |
| 3 | 0.041956 +/- 0.000163 | 0.057347 | 28 | 0.355977 +/- 0.000451 | 0.357764 |
| 4 | 0.183441 +/- 0.000328 | 0.186228 | 29 | 0.434322 +/- 0.000420 | 0.432596 |
| 5 | 0.289615 +/- 0.000416 | 0.274821 | 30 | 0.235670 +/- 0.000351 | 0.240261 |
| 6 | 0.210766 +/- 0.000296 | 0.214919 | 31 | 0.218803 +/- 0.000284 | 0.224687 |
| 7 | 0.134658 +/- 0.000227 | 0.138083 | 32 | 0.212910 +/- 0.000296 | 0.228965 |
| 8 | 0.124535 +/- 0.000208 | 0.127623 | 33 | 0.158638 +/- 0.000259 | 0.158306 |
| 9 | 0.102088 +/- 0.000228 | 0.108042 | 34 | 0.059604 +/- 0.000206 | 0.061265 |
| 10 | 0.174223 +/- 0.000269 | 0.182671 | 35 | 0.164154 +/- 0.000254 | 0.175598 |
| 11 | 0.177112 +/- 0.000249 | 0.178304 | 36 | 0.194866 +/- 0.000266 | 0.196489 |
| 12 | 0.202091 +/- 0.000296 | 0.195231 | 37 | 0.155908 +/- 0.000248 | 0.159045 |
| 13 | 0.268592 +/- 0.000315 | 0.271718 | 38 | 0.207117 +/- 0.000301 | 0.20425 |
| 14 | 0.334375 +/- 0.000363 | 0.364439 | 39 | 0.180868 +/- 0.000267 | 0.190258 |
| 15 | 0.170426 +/- 0.000246 | 0.18395 | 40 | 0.199559 +/- 0.000292 | 0.237971 |
| 16 | 0.347567 +/- 0.000353 | 0.362132 | 41 | 0.195525 +/- 0.000273 | 0.242095 |
| 17 | 0.087838 +/- 0.000234 | 0.088299 | 42 | 0.138179 +/- 0.000243 | 0.213977 |
| 18 | 0.088043 +/- 0.000231 | 0.093272 | 43 | 0.170463 +/- 0.000259 | 0.168257 |
| 19 | 0.165877 +/- 0.000282 | 0.185546 | 44 | 0.182617 +/- 0.000252 | 0.227972 |
| 20 | 0.314484 +/- 0.000327 | 0.336762 | 45 | 0.183492 +/- 0.000258 | 0.176587 |
| 21 | 0.278663 +/- 0.000327 | 0.296972 | 46 | 0.167610 +/- 0.000258 | 0.183072 |
| 22 | 0.221330 +/- 0.000284 | 0.238433 | 47 | 0.112583 +/- 0.000207 | 0.123464 |
| 23 | 0.208108 +/- 0.000288 | 0.214178 | 48 | 0.121282 +/- 0.000219 | 0.127233 |
| 24 | 0.179790 +/- 0.000267 | 0.18182 | 49 | 0.197280 +/- 0.000279 | 0.235014 |
| 25 | 0.160745 +/- 0.000250 | 0.166854 | 50 | 0.111037 +/- 0.000199 | 0.109343 |

**Table D.4**: Resulting P(nosetup) values calculated via simulation and our approximation method.

## D.1.2 VPDBA Solution Experiment

### D.1.2.1 Test Cases

| test case 1 | test case 2 | test case 3 |
|---|---|---|
| $N = 5$ | $N = 5$ | $N = 5$ |
| $\lambda_{vp01} = 0.1$ | $\lambda_{vp01} = 0.7$ | $\lambda_{vp01} = 0.4$ |
| $\lambda_{vp02} = 0.5$ | $\lambda_{vp02} = 0.5$ | $\lambda_{vp02} = 0.1$ |
| $\lambda_{vp03} = 1.9$ | $\lambda_{vp03} = 0.5$ | $\lambda_{vp03} = 0.2$ |
| $\lambda_{vp04} = 1.2$ | $\lambda_{vp04} = 0.2$ | $\lambda_{vp04} = 0.1$ |
| $\lambda_{vp12} = 0.1$ | $\lambda_{vp12} = 0.1$ | $\lambda_{vp12} = 0.2$ |
| $\lambda_{vp13} = 0.1$ | $\lambda_{vp13} = 1.2$ | $\lambda_{vp13} = 0.4$ |
| $\lambda_{vp14} = 0.3$ | $\lambda_{vp14} = 0.2$ | $\lambda_{vp14} = 0.2$ |
| $\lambda_{vp23} = 0.1$ | $\lambda_{vp23} = 0.5$ | $\lambda_{vp23} = 0.4$ |
| $\lambda_{vp24} = 0.1$ | $\lambda_{vp24} = 0.7$ | $\lambda_{vp24} = 0.4$ |
| $\lambda_{vp34} = 0.2$ | $\lambda_{vp34} = 0.1$ | $\lambda_{vp34} = 0.6$ |
| $\mu = 1.0$ | $\mu = 1.0$ | $\mu = 1.0$ |
| $K = 5$ | $K = 4$ | $K = 4$ |

**Table D.5**: The test cases used in the line network model experiment.

# Appendix E

# Algorithm for VPDBA

## E.1   Previous Algorithms for VPDBA

### E.1.1   Gain/Loss Ratio Method (Arvidsson)

1. Assign high, initially acceptable call loss levels $\gamma(s, o, d)$ for all traffic classes $s$, and source destination pairs $(o, d)$

2. Find the shortest path available to each traffic $(s, o, d)$

3. Compute the GAIN achieved for each traffic $(s, o, d)$ if 1 unit of capacity is added to the shortest path for traffic $(s, o, d)$

4. Compute the LOSS achieved for each traffic $(t, i, j)$ if 1 unit of capacity is added to the shortest path for traffic $(s, o, d)$

5. Find the traffic $(s_{max}, o_{max}, d_{max})$ that would yield the highest gain/loss ratio

6. If the highest gain/loss ratio is NOT equal to 0

   (a) Assign 1 unit of capacity to $(s_{max}, o_{max}, d_{max})$

   (b) go to step 3

7. Otherwise

   (a) if low (acceptable) loss levels have been achieved $\forall(s, o, d)$ or if all capacity has been assigned to VPs

      - STOP

   (b) otherwise

      - reduce the acceptable loss levels $\forall(s, o, d)$

      - return to step 3

## E.1.2 Overall Blocking Rate Method (Cheng and Lin)

1. Assume that all traffic is carried on 1-hop VP paths which are initially assigned all capacity on each link

2. While (improvement is possible)

   (a) PHASE I - find the shortest path for each traffic i,j (hold VP assignments fixed and alter the call routing assignments to reduce the overall call blocking rate)

   (b) PHASE II (GREEDY HEURISTIC for capacity adjustment)

      i. calculate the current blocking rate

      ii. for each VP, compute the change in overall blocking rate if 1 unit of capacity were added to this VP and subtracted for every 1-hop VP on physical path q

      iii. if there exists an assignment that reduces the blocking rate

         A. select the capacity change that results in the maximum blocking rate reduction and adjust the path capacities accordingly

iv. otherwise STOP with the current VPs as the solution

### E.1.3 Blocking Drift Method (Aneroussis and Lazar)

1. all VPs have an initial capacity of 0

2. compute blocking probabilities for all SD pairs for the given arrival rates

3. STEP 1

   (a) Find the SD pairs for which blocking constraints are not satisfied

   (b) If none are found, proceed to Step 2

   (c) Otherwise

       i. Consider every VP whose capacity can be increased as a Bandwidth Increase Candidate (BIC). The capacity of a VP is increased by removing the necessary capacity from all the links on the paths of the VP.

       ii. if the BIC set is empty

          • EXIT – there is no spare capacity available in the network

       iii. otherwise

          A. Compute the blocking drift $D_b$ for the current vector of VP capacities = $\sum_{w,k} max(0, P_w^k - \beta_w^k$ which represents "how far" we are from satisfying the blocking constraints

          B. For each BIC, create an alternative solution by assigning 1 unit of capacity to the BIC while holding all other VPs at their current capacity

          C. For each alternative solution, calculate the blocking drift

      D. If the new blocking drift is greater than the current blocking drift for all alternative solutions there is no way of satisfying the blocking constraints for the given network load - STOP

      E. Otherwise, select the alternative with the lowest blocking drift and make it the current solution

4. STEP 2

    (a) Check for signaling violations which are not considered in our problem formulation so we ignore this step

5. If there are any remaining blocking or signaling constraint violations, return to STEP 1

6. STEP 3

    (a) Every VP whose capacity can be increased by one unit is considered a BIC

    (b) Compute the throughput for each SD pair: $\gamma_w^k = (1 - P_w^k)\lambda_w^k$

    (c) Compute the network revenue $J = \sum_w \sum_k \gamma_w^k F_w^k$ where $F_w^k$ denotes the revenue obtained by accepting one call of SD pair $w$ and traffic class $k$

    (d) For each BIC, create an alternative solution and compute the resulting network revenue

    (e) Eliminate alternatives for which blocking or signaling capacity constraints are violated

    (f) If no alternatives remain, STOP

(g) Otherwise, select the alternative with the highest revenue increase and repeat

step 3

## E.2 Comparing Our Algorithm to Previous Algorithms

### E.2.1 Test Cases

| test case 1 | test case 2 | test case 3 | test case 4 | test case 5 |
|---|---|---|---|---|
| $N = 5$ | $N = 5$ | $N = 5$ | $N = 5$ | $N = 5$ |
| $\lambda_{01} = 0.1$ | $\lambda_{01} = 0.7$ | $\lambda_{01} = 0.4$ | $\lambda_{01} = 0.1$ | $\lambda_{01} = 0.1$ |
| $\lambda_{02} = 0.5$ | $\lambda_{02} = 0.5$ | $\lambda_{02} = 0.1$ | $\lambda_{02} = 0.3$ | $\lambda_{02} = 1.2$ |
| $\lambda_{03} = 1.9$ | $\lambda_{03} = 0.5$ | $\lambda_{03} = 0.2$ | $\lambda_{03} = 0.1$ | $\lambda_{03} = 0.7$ |
| $\lambda_{04} = 1.2$ | $\lambda_{04} = 0.2$ | $\lambda_{04} = 0.1$ | $\lambda_{04} = 0.2$ | $\lambda_{04} = 0.1$ |
| $\lambda_{12} = 0.1$ | $\lambda_{12} = 0.1$ | $\lambda_{12} = 0.2$ | $\lambda_{12} = 0.4$ | $\lambda_{12} = 0.1$ |
| $\lambda_{13} = 0.1$ | $\lambda_{13} = 1.2$ | $\lambda_{13} = 0.4$ | $\lambda_{13} = 0.2$ | $\lambda_{13} = 0.1$ |
| $\lambda_{14} = 0.3$ | $\lambda_{14} = 0.2$ | $\lambda_{14} = 0.2$ | $\lambda_{14} = 0.1$ | $\lambda_{14} = 1.2$ |
| $\lambda_{23} = 0.1$ | $\lambda_{23} = 0.5$ | $\lambda_{23} = 0.4$ | $\lambda_{23} = 2.0$ | $\lambda_{23} = 1.2$ |
| $\lambda_{24} = 0.1$ | $\lambda_{24} = 0.7$ | $\lambda_{24} = 0.4$ | $\lambda_{24} = 0.1$ | $\lambda_{24} = 0.1$ |
| $\lambda_{34} = 0.2$ | $\lambda_{34} = 0.1$ | $\lambda_{34} = 0.6$ | $\lambda_{34} = 1.25$ | $\lambda_{34} = 0.6$ |
| $K = 5$ | $K = 4$ | $K = 4$ | $K = 5$ | $K = 4$ |
| test case 6 | test case 7 | test case 8 | test case 9 | test case 10 |
| $N = 6$ | $N = 5$ | $N = 5$ | $N = 4$ | $N = 5$ |
| $\lambda_{01} = 0.1$ | $\lambda_{01} = 0.1$ | $\lambda_{01} = 0.2$ | $\lambda_{01} = 0.1$ | $\lambda_{01} = 0.4$ |
| $\lambda_{02} = 0.1$ | $\lambda_{02} = 0.1$ | $\lambda_{02} = 0.2$ | $\lambda_{02} = 0.1$ | $\lambda_{02} = 0.1$ |
| $\lambda_{03} = 0.1$ | $\lambda_{03} = 0.1$ | $\lambda_{03} = 0.1$ | $\lambda_{03} = 0.1$ | $\lambda_{03} = 0.1$ |
| $\lambda_{04} = 0.1$ | $\lambda_{04} = 0.1$ | $\lambda_{04} = 0.2$ | $\lambda_{12} = 0.05$ | $\lambda_{04} = 0.2$ |
| $\lambda_{05} = 0.5$ | $\lambda_{12} = 2.0$ | $\lambda_{12} = 0.1$ | $\lambda_{13} = 0.05$ | $\lambda_{12} = 0.1$ |
| $\lambda_{12} = 0.7$ | $\lambda_{13} = 3.0$ | $\lambda_{13} = 0.1$ | $\lambda_{23} = 0.15$ | $\lambda_{13} = 2.0$ |
| $\lambda_{13} = 0.1$ | $\lambda_{14} = 0.5$ | $\lambda_{14} = 0.1$ | $K = 4$ | $\lambda_{14} = 0.2$ |
| $\lambda_{14} = 0.1$ | $\lambda_{23} = 0.3$ | $\lambda_{23} = 0.2$ | | $\lambda_{23} = 1.75$ |
| $\lambda_{15} = 0.1$ | $\lambda_{24} = 0.1$ | $\lambda_{24} = 0.9$ | | $\lambda_{24} = 0.1$ |
| $\lambda_{23} = 0.1$ | $\lambda_{34} = 0.7$ | $\lambda_{34} = 0.3$ | | $\lambda_{34} = 0.1$ |
| $\lambda_{24} = 0.5$ | $K = 5$ | $K = 4$ | | $K = 4$ |
| $\lambda_{25} = 0.3$ | | | | |
| $\lambda_{34} = 0.1$ | | | | |
| $\lambda_{35} = 0.1$ | | | | |
| $\lambda_{45} = 0.1$ | | | | |
| $K = 3$ | | | | |

**Table E.1:** Test cases used to compare my algorithm to the previous algorithms.
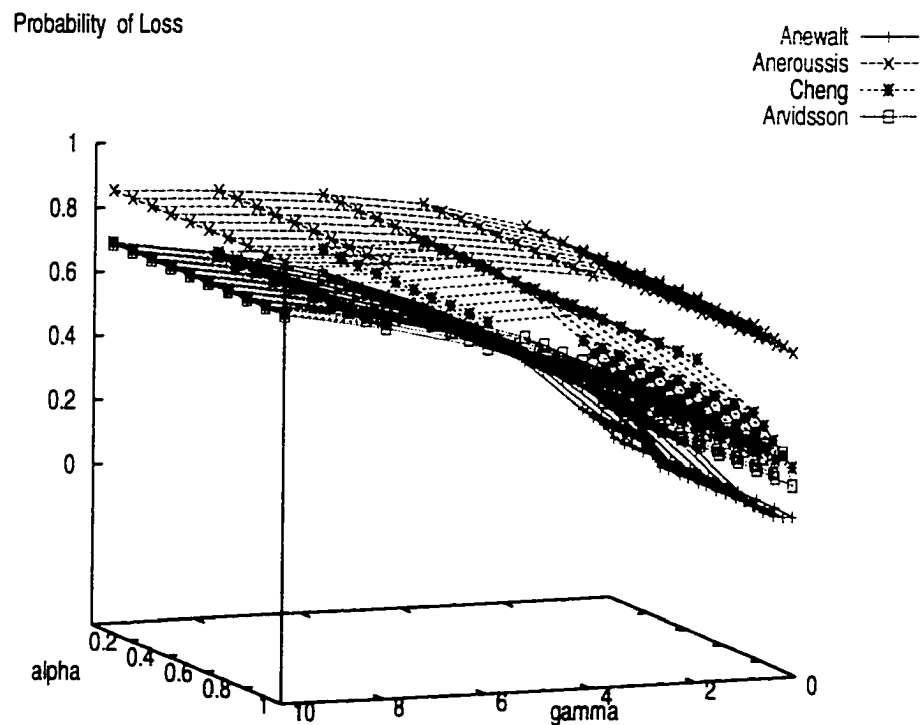
## E.2.2 Results
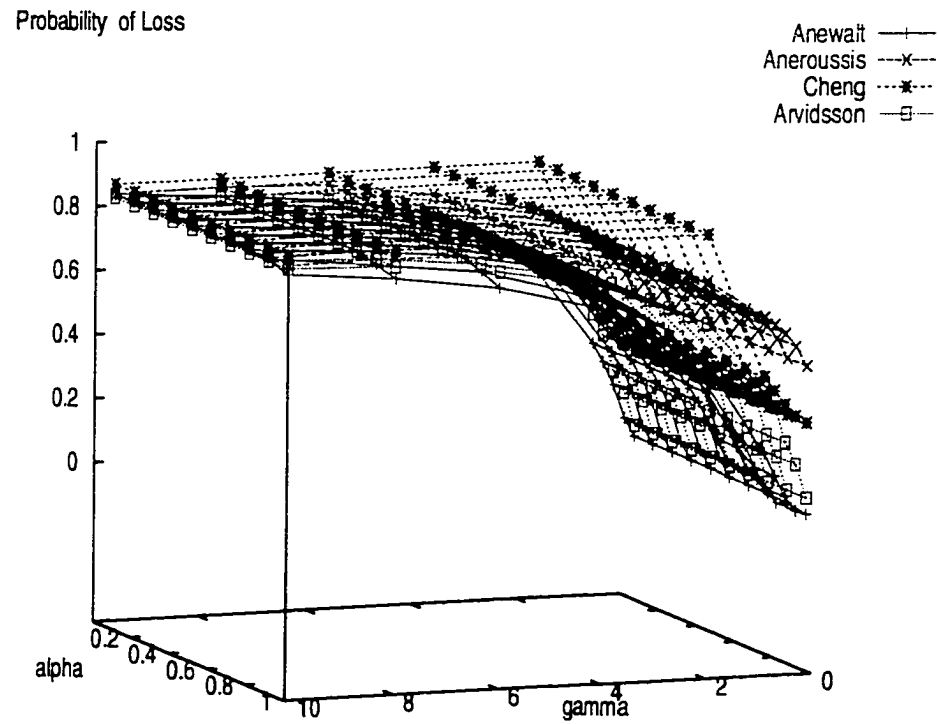


**Figure E.1**: Comparing the Loss Probabilities calculated for our network and the 3 other networks Test Case 1.
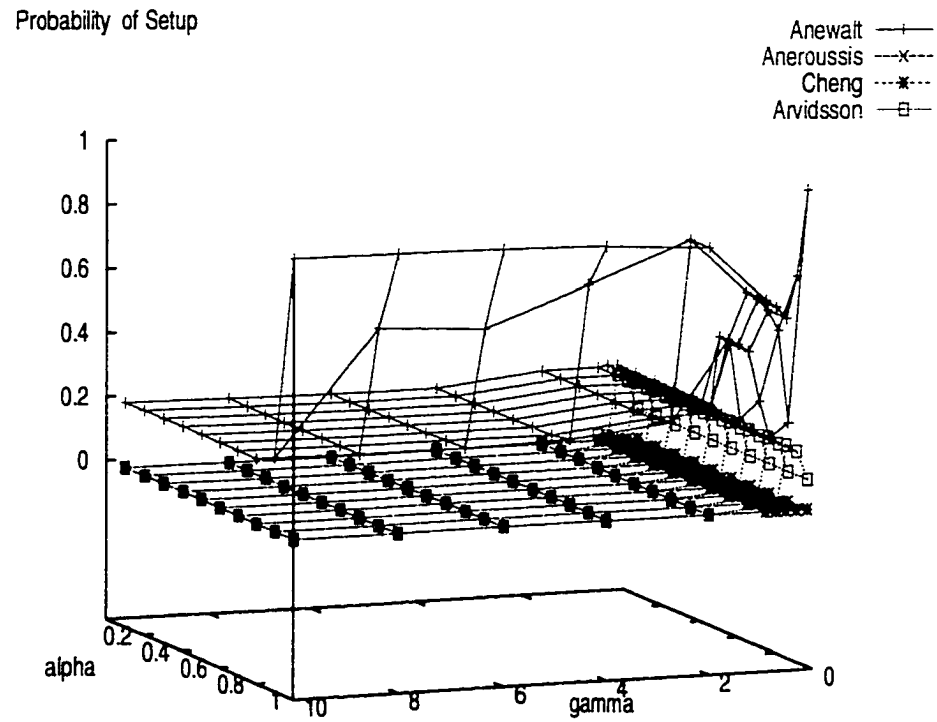
**Figure E.2**: Comparing the Setup Probabilities calculated for our network and the 3 other networks Test Case 1.
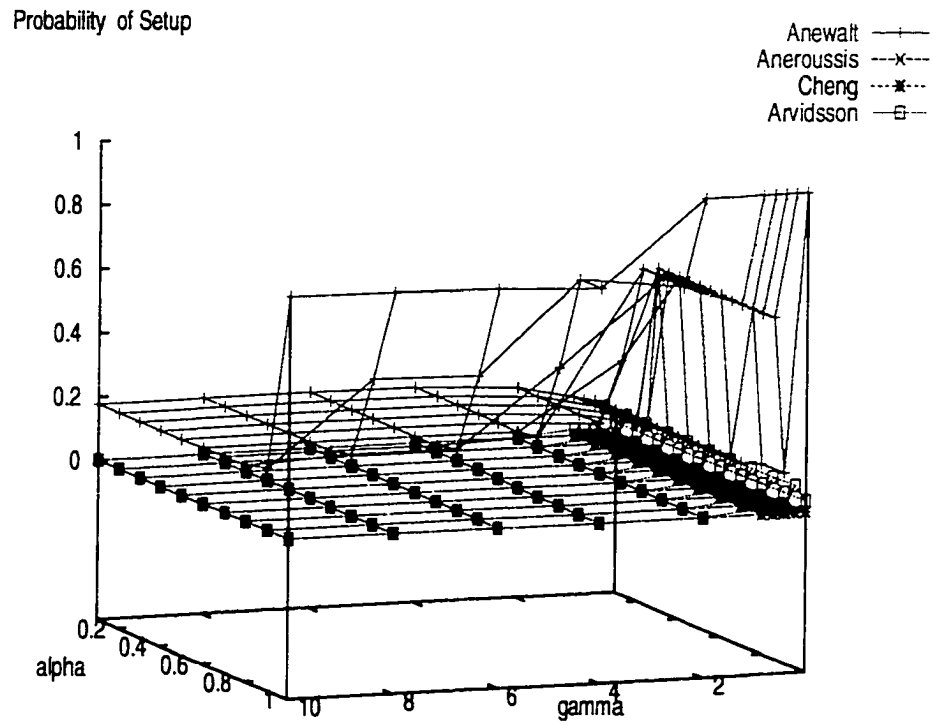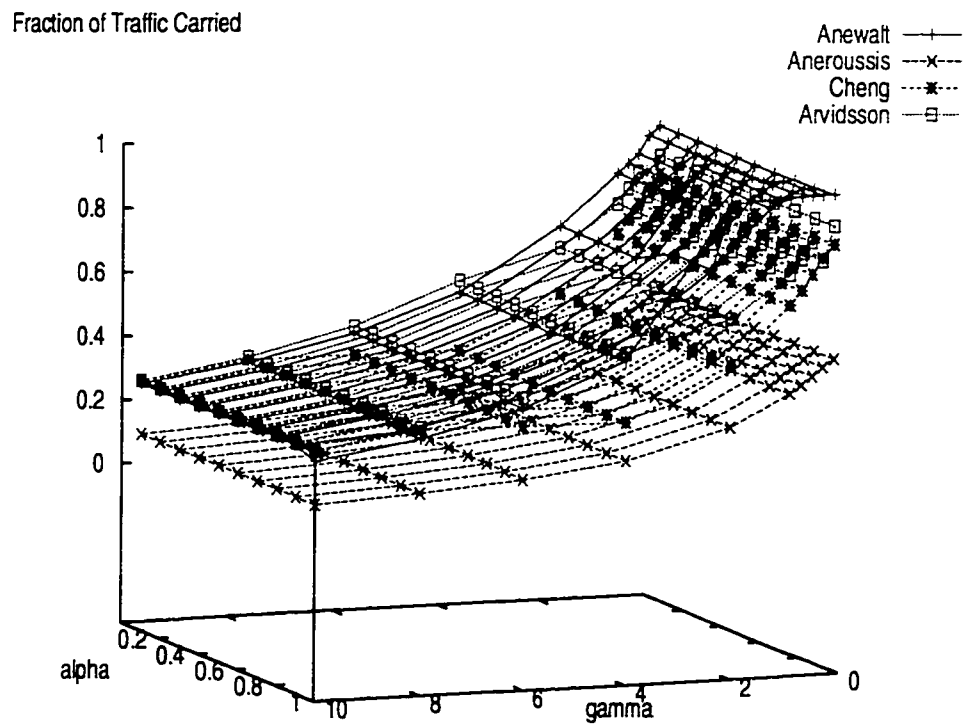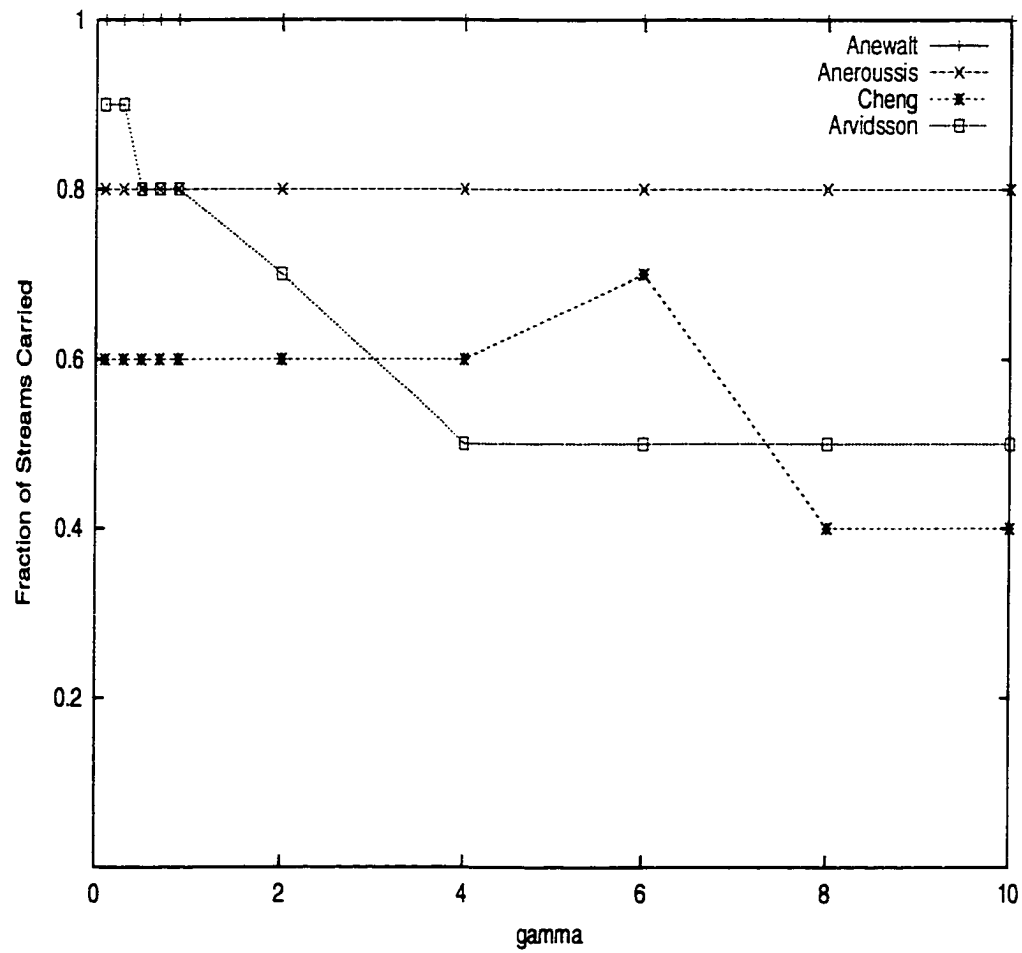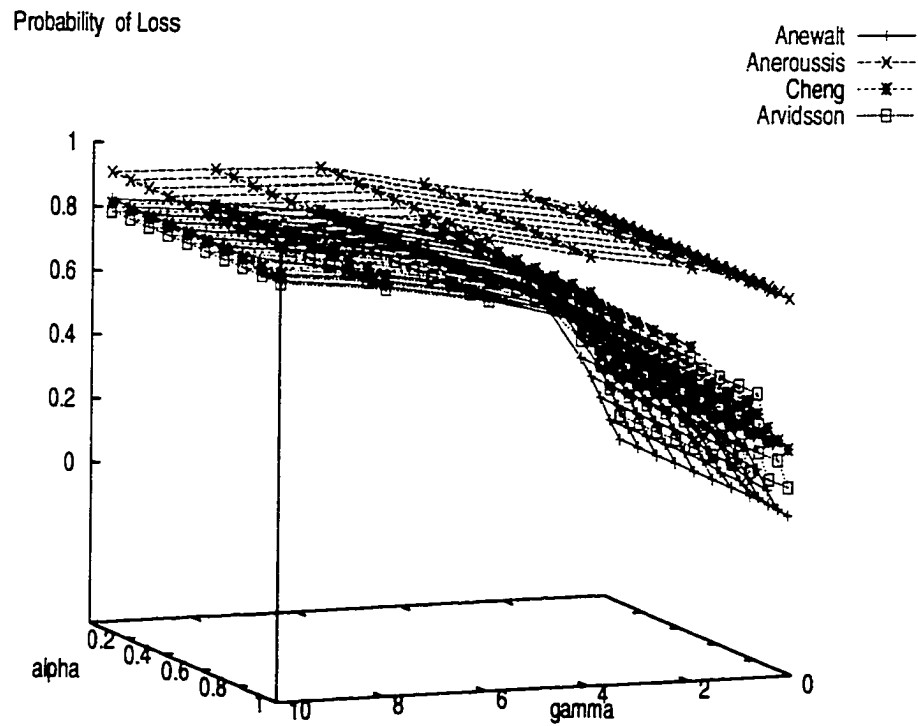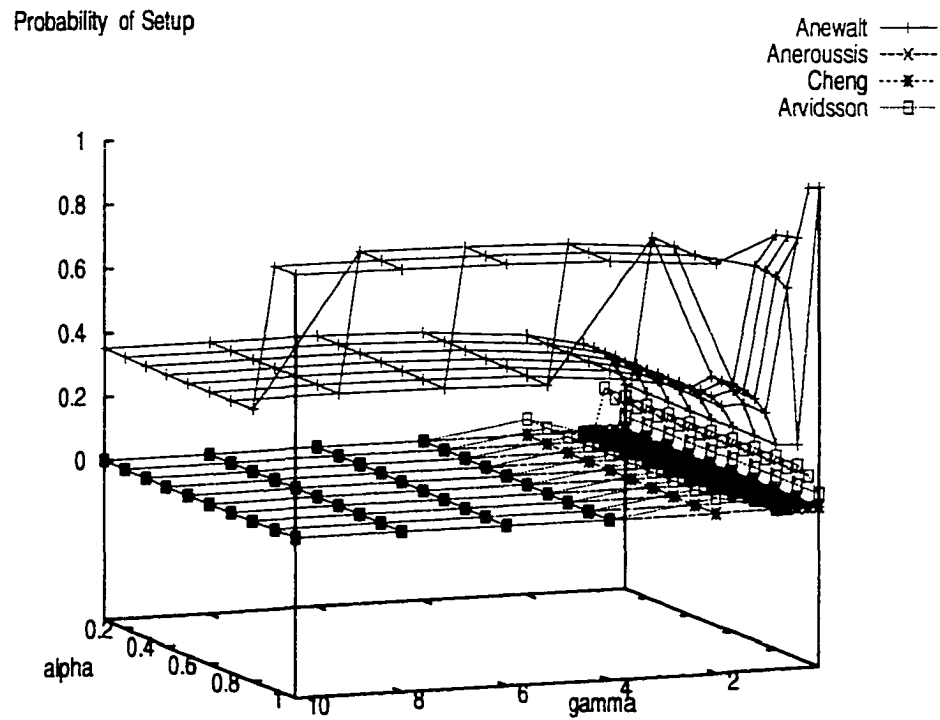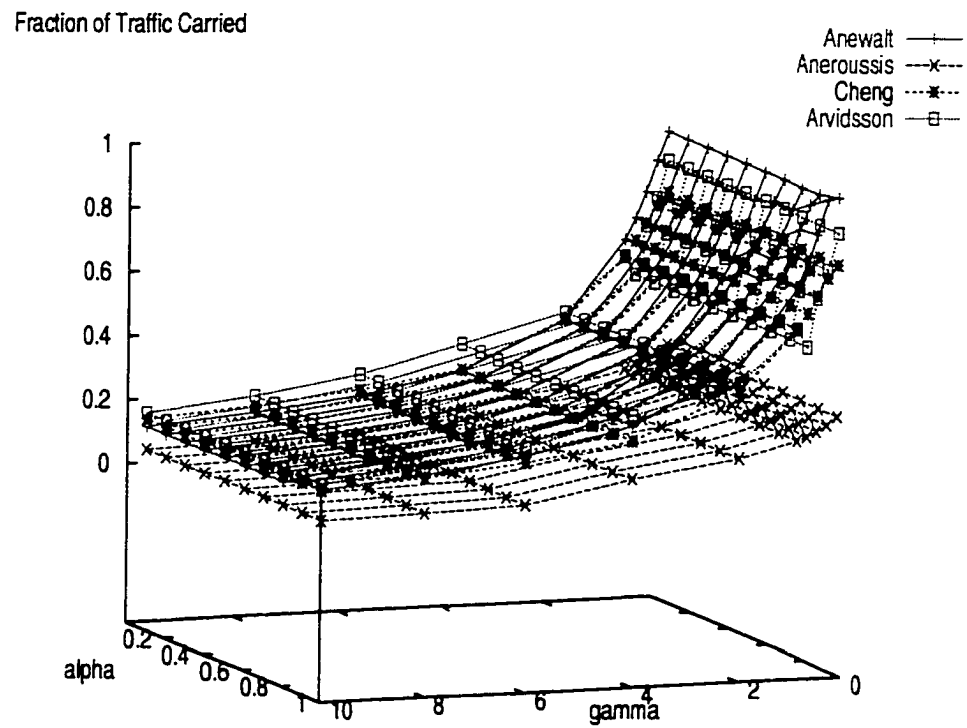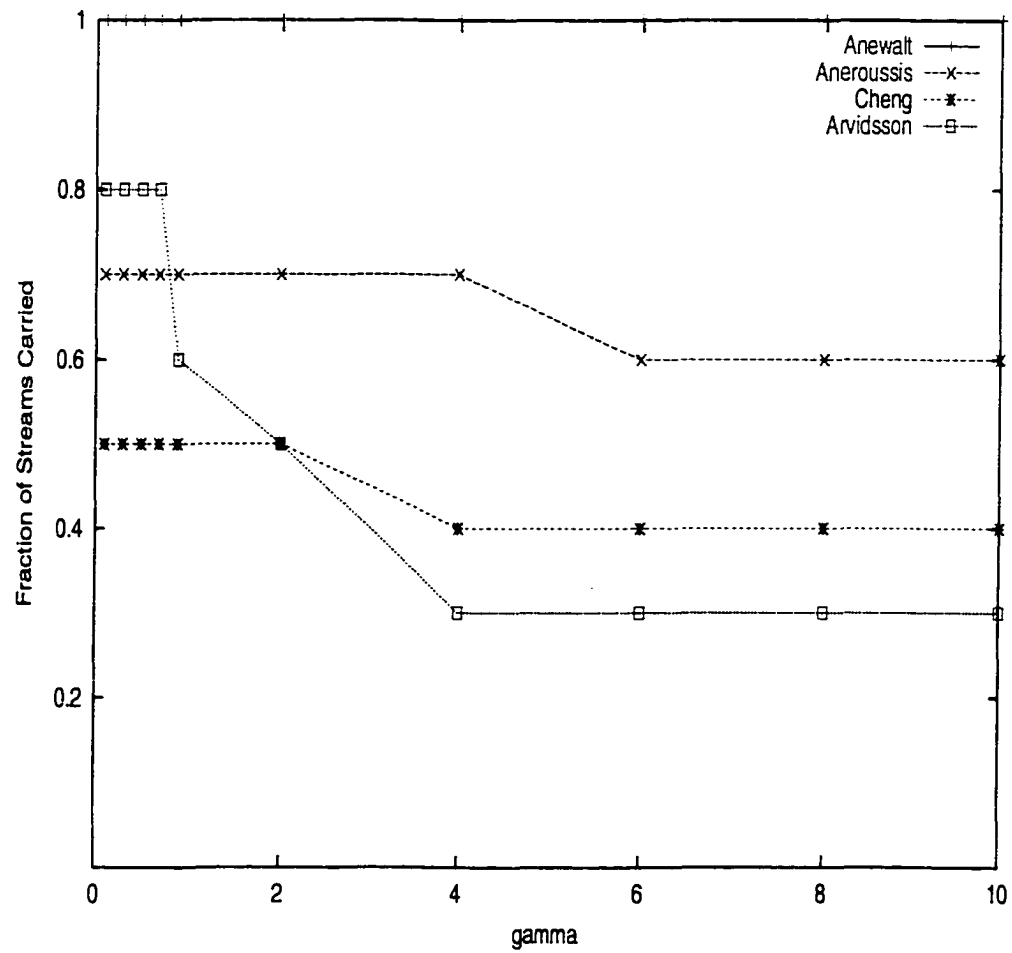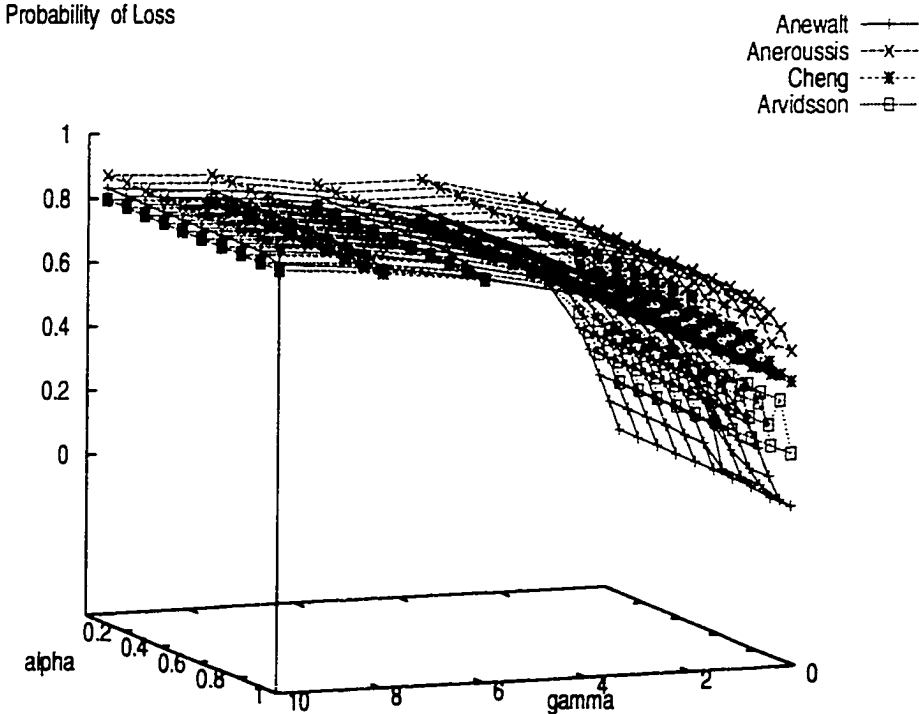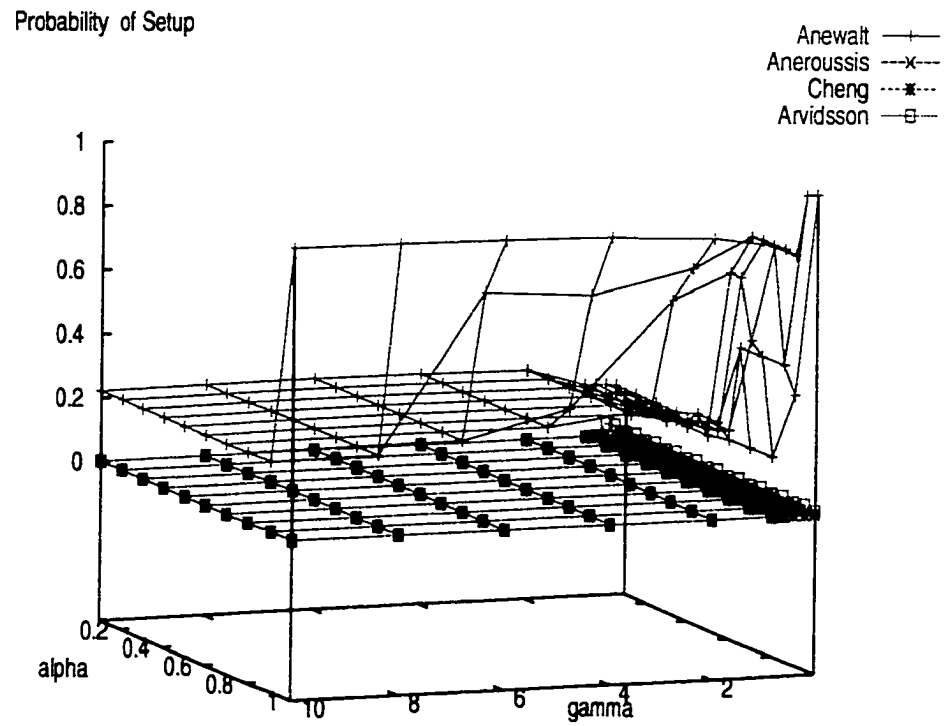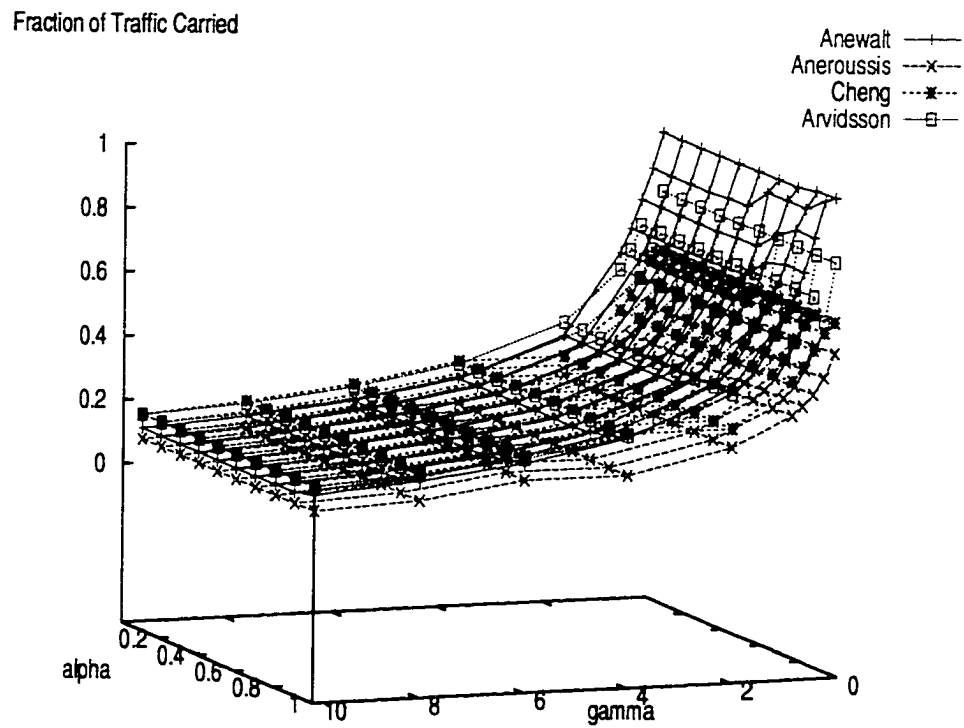
Fraction of Traffic Carried

Anewalt ——+——
Aneroussis ---×---
Cheng ---✱---
Arvidsson --□--
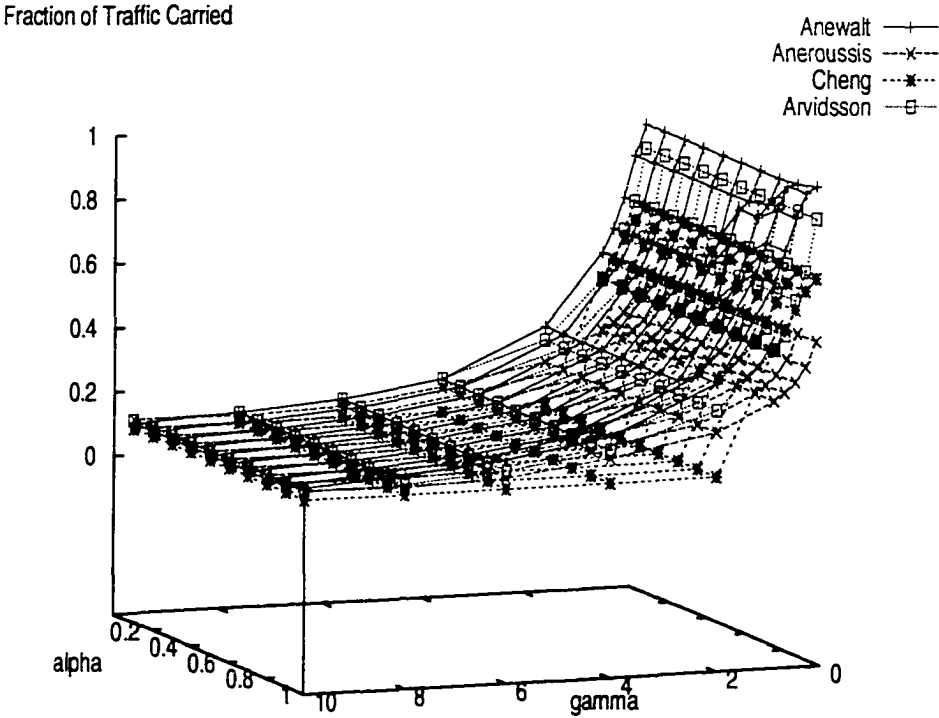


**Figure E.3**: Comparing the Fraction of Traffic Carried calculated for our network and the 3 other networks Test Case 1.
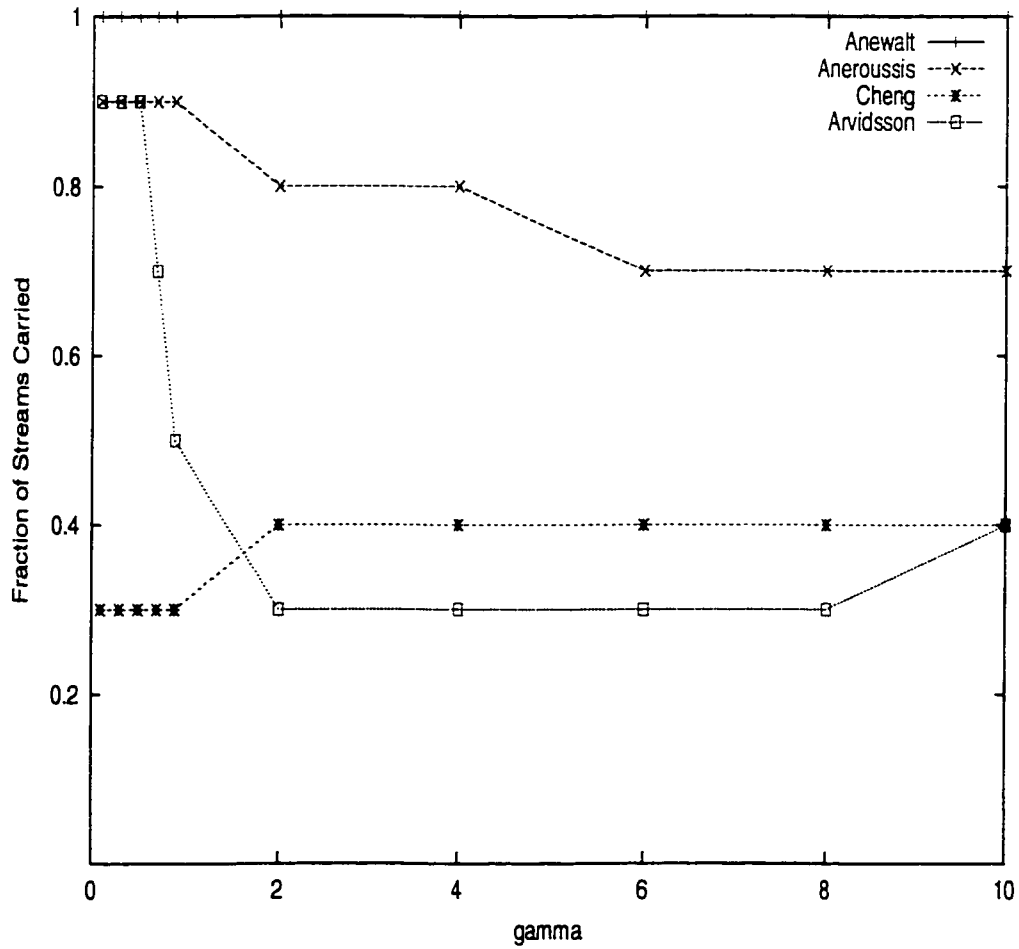
**Figure E.4**: Comparing the Fraction of Streams Carried calculated for our network and the 3 other networks Test Case 1.
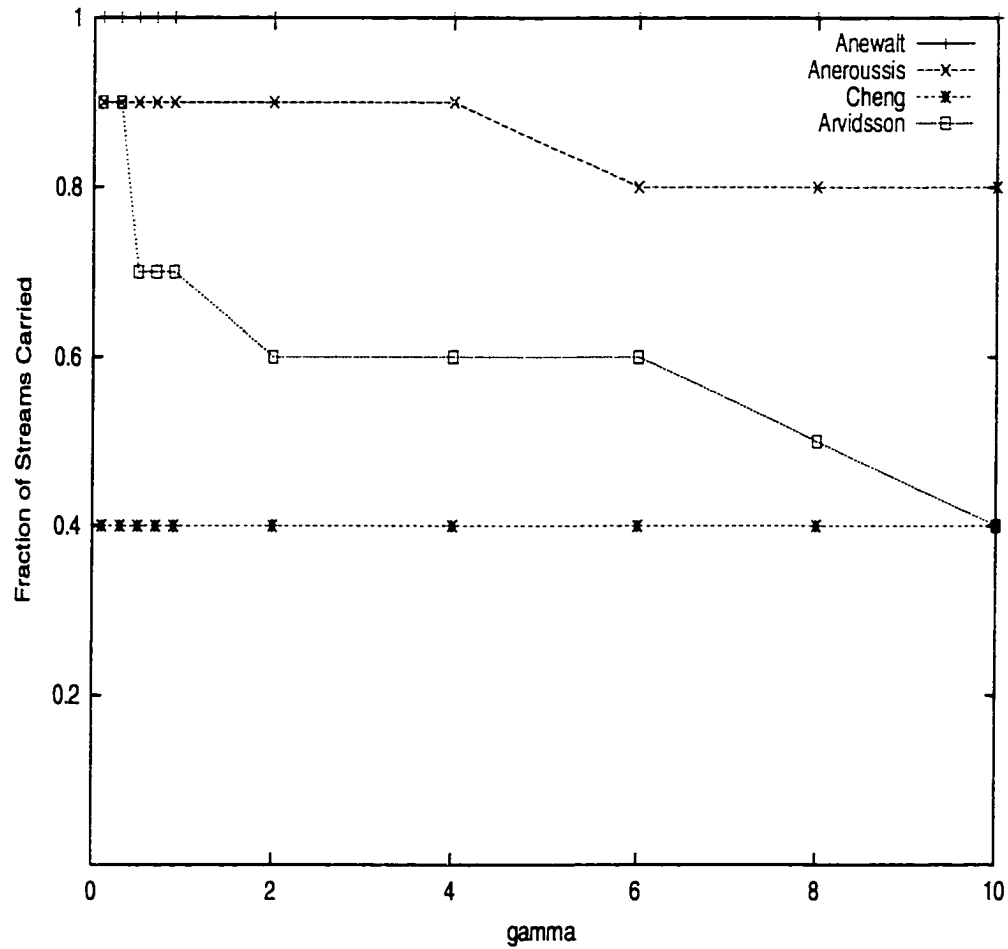
Probability of Loss

Anewalt ⸺+⸺
Aneroussis ⸺x⸺
Cheng ⸺*⸺
Arvidsson ⸺◻⸺
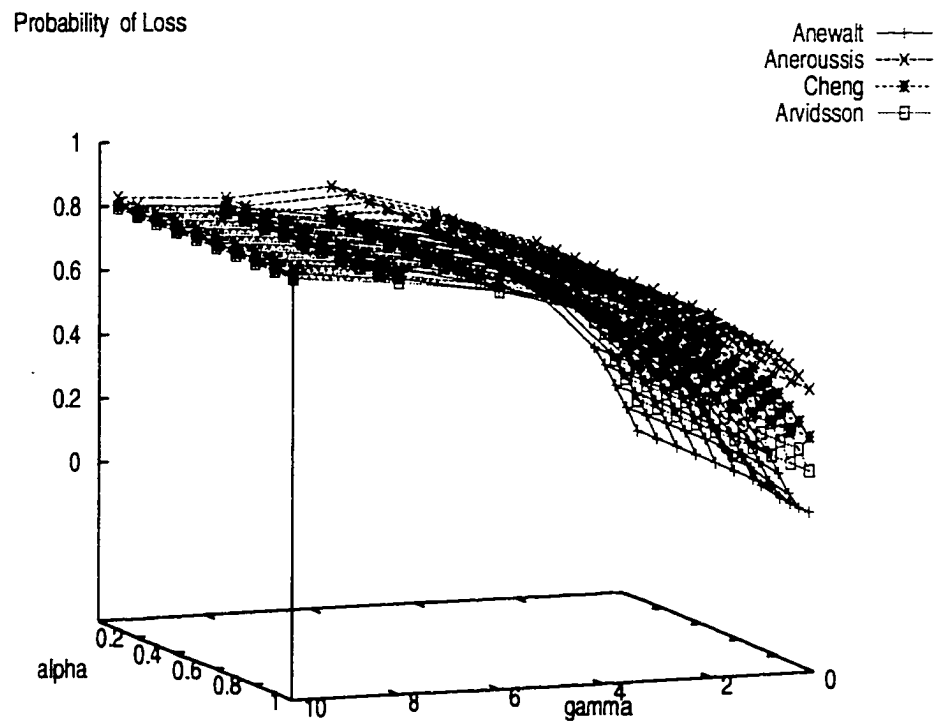


**Figure E.5**: Comparing the Loss Probabilities calculated for our network and the 3 other networks Test Case 2.
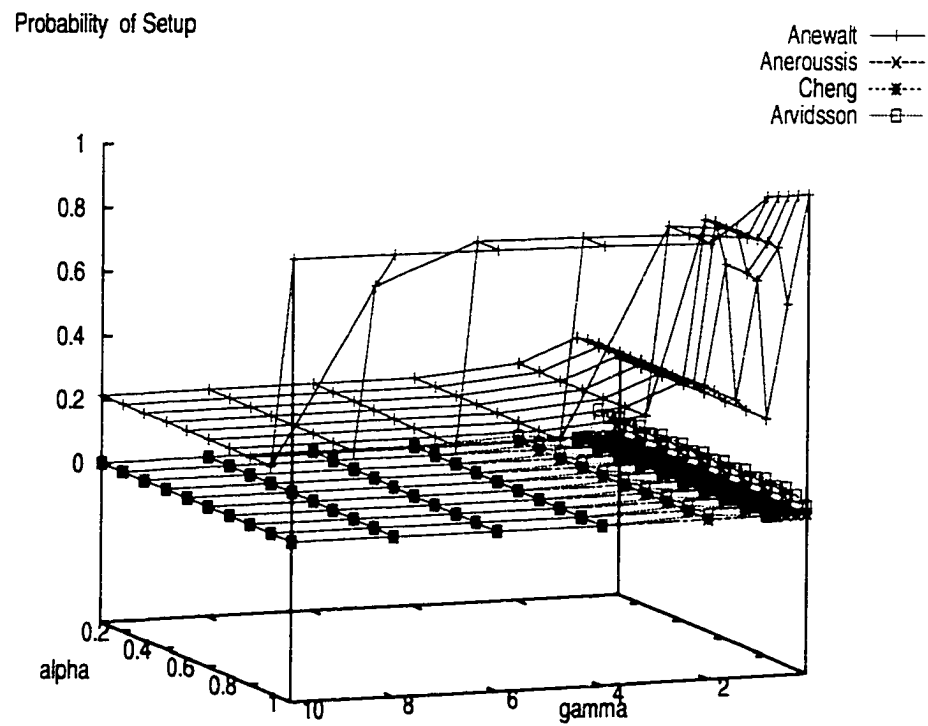
**Figure E.6**: Comparing the Setup Probabilities calculated for our network and the 3 other networks Test Case 2.
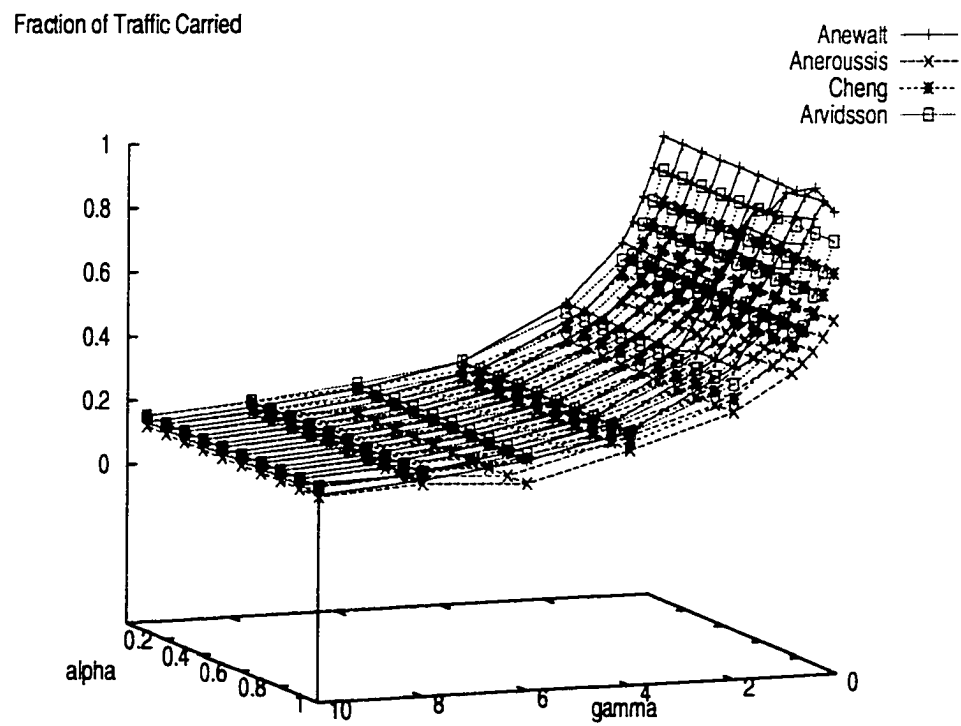
**Figure E.7**: Comparing the Fraction of Traffic Carried calculated for our network and the 3 other networks Test Case 2.
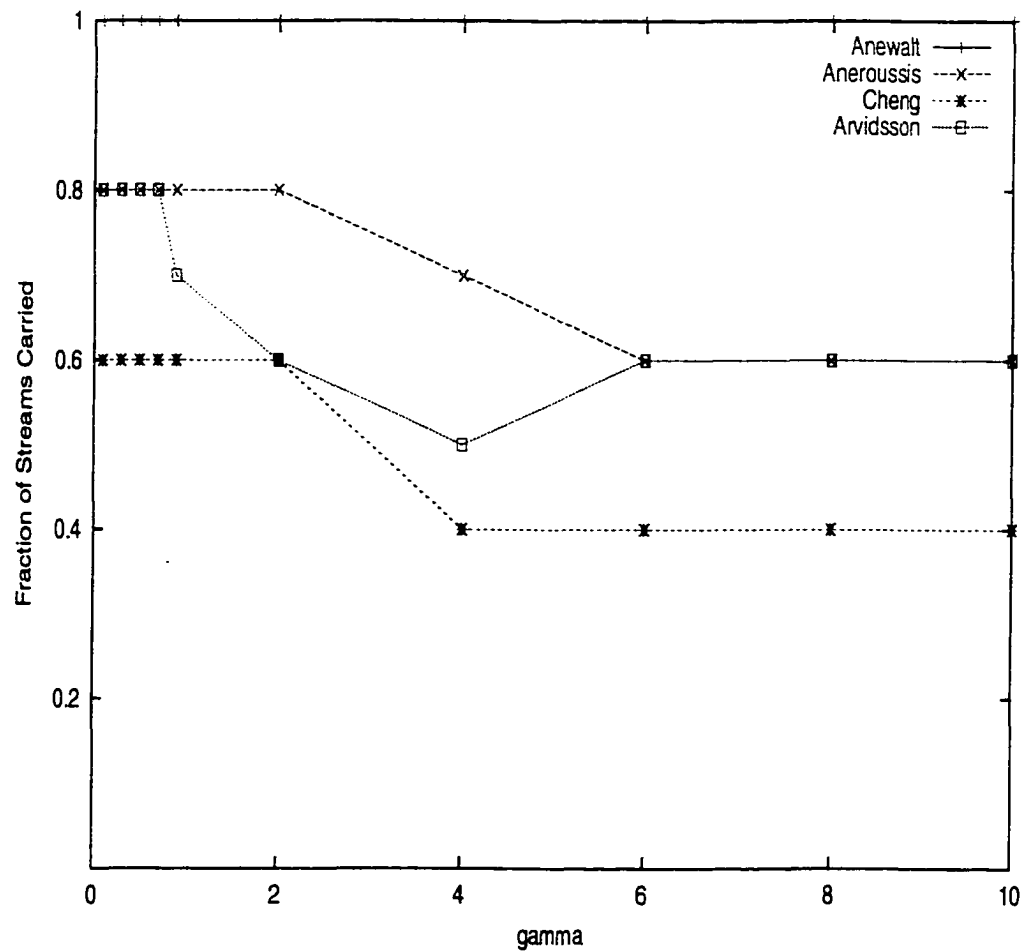
**Figure E.8**: Comparing the Fraction of Streams Carried calculated for our network and the 3 other networks Test Case 2.
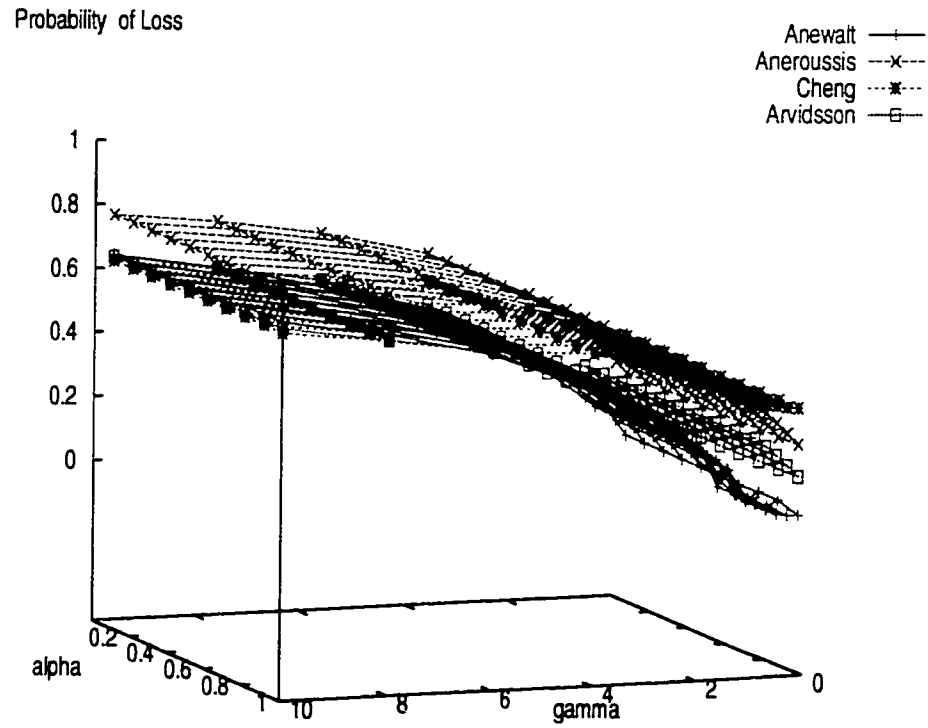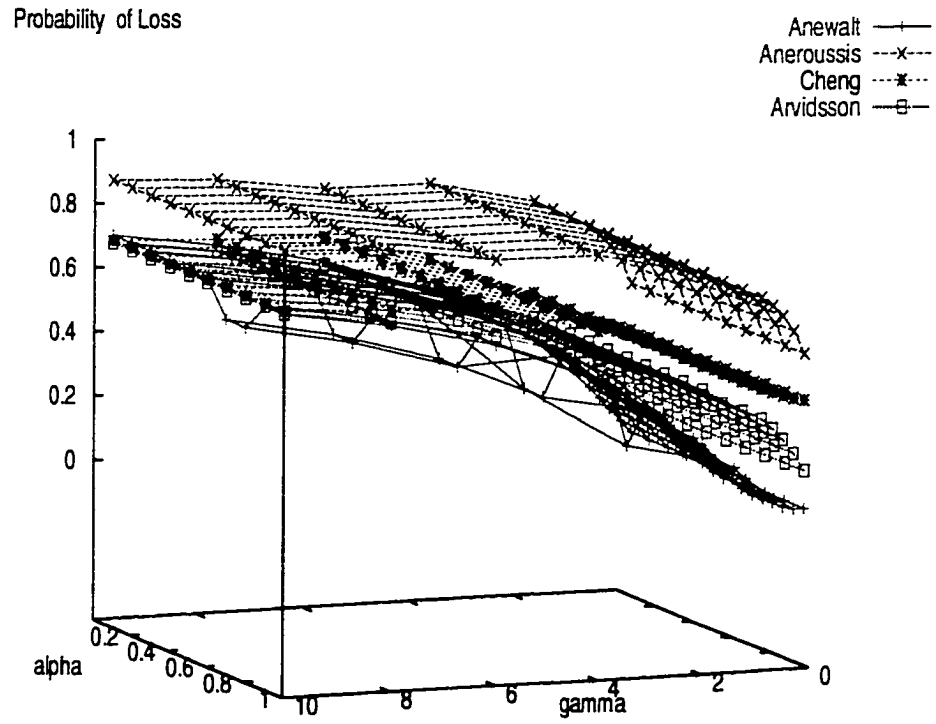
**Figure E.9**: Comparing the Loss Probabilities calculated for our network and the 3 other networks Test Case 3.
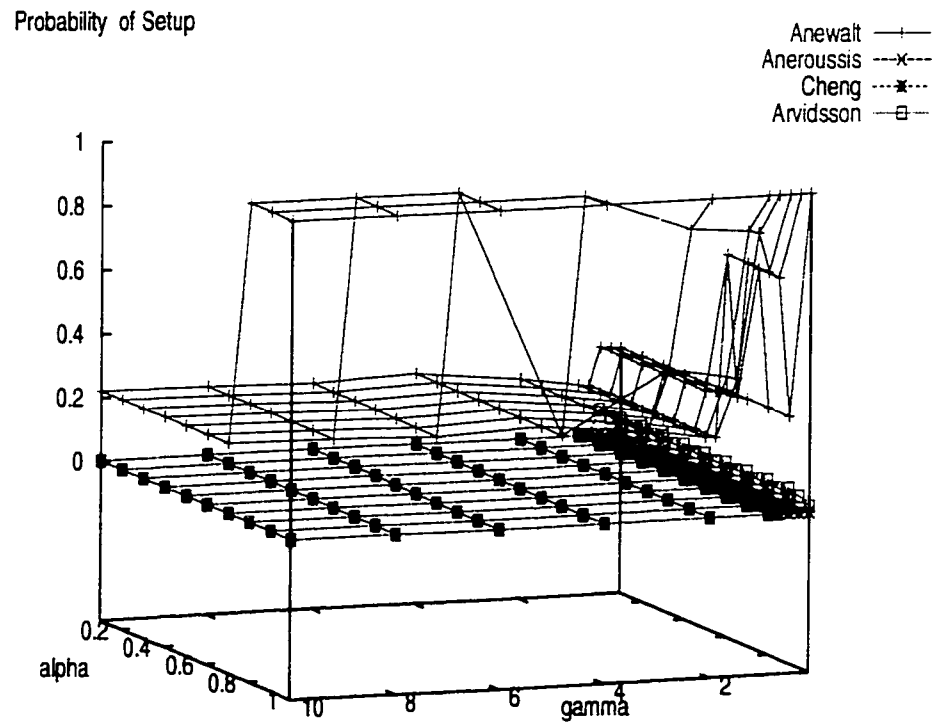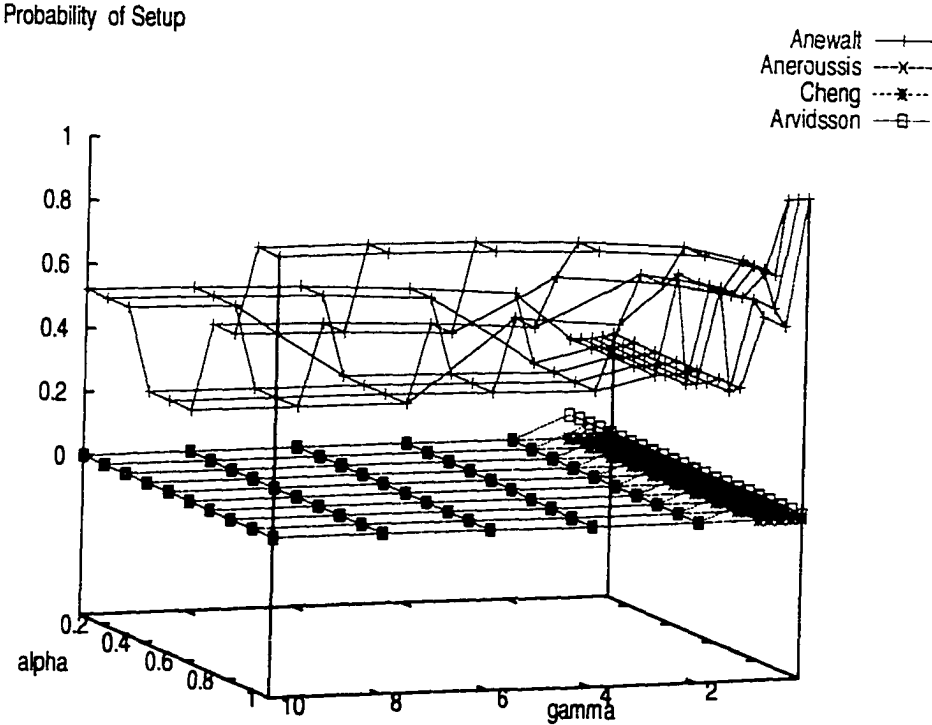
**Figure E.10**: Comparing the Setup Probabilities calculated for our network and the 3 other networks Test Case 3.
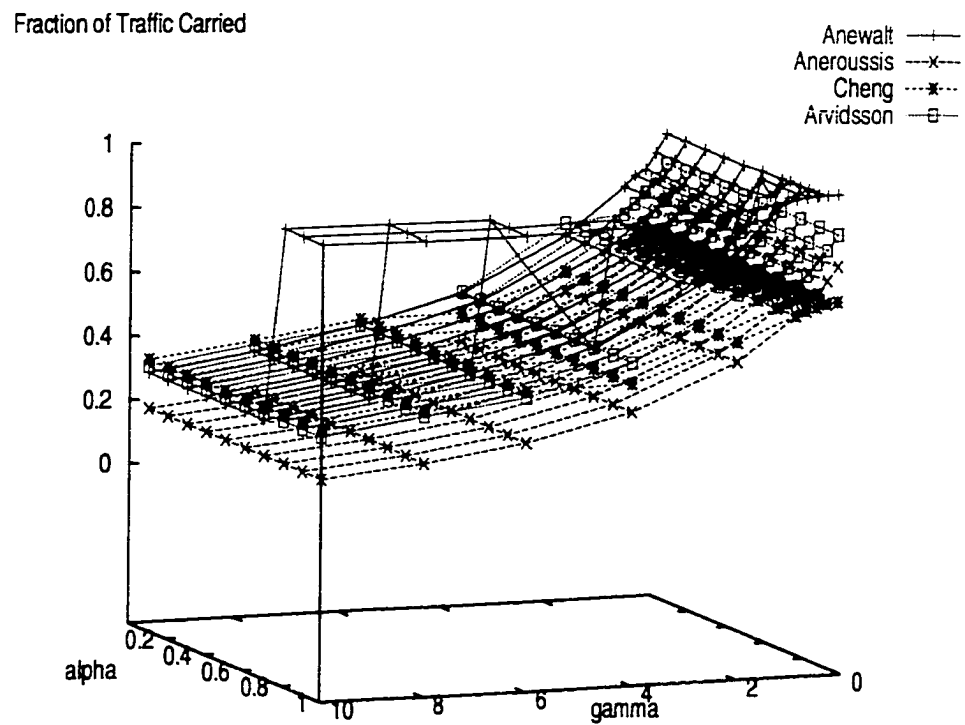
**Figure E.11**: Comparing the Fraction of Traffic Carried calculated for our network and the 3 other networks Test Case 3.
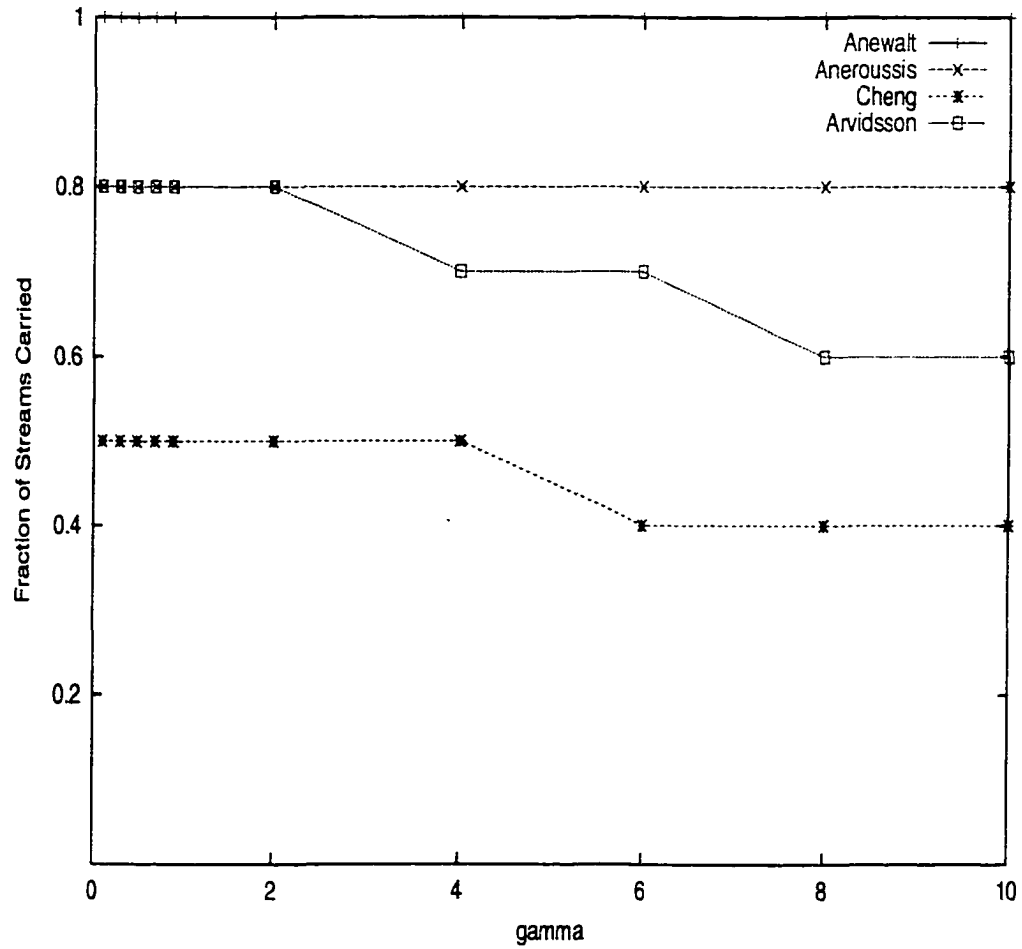
**Figure E.12**: Comparing the Fraction of Streams Carried calculated for our network and the 3 other networks Test Case 3.

**Figure E.13**: Comparing the Loss Probabilities calculated for our network and the 3 other networks Test Case 4.

**Figure E.14**: Comparing the Setup Probabilities calculated for our network and the 3 other networks Test Case 4.
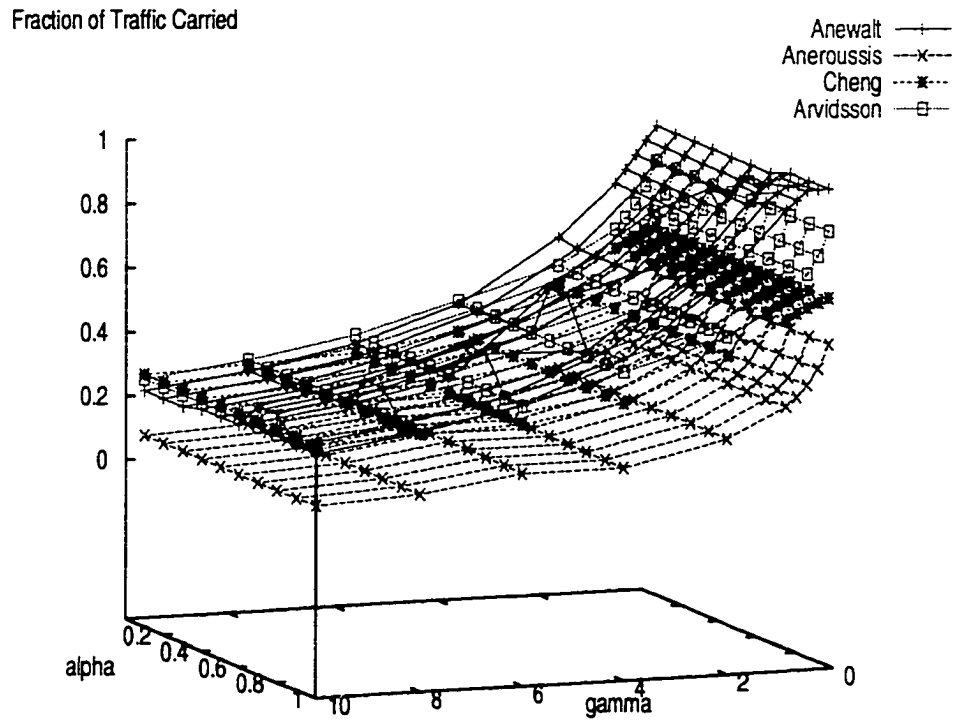
**Figure E.15**: Comparing the Fraction of Traffic Carried calculated for our network and the 3 other networks Test Case 4.
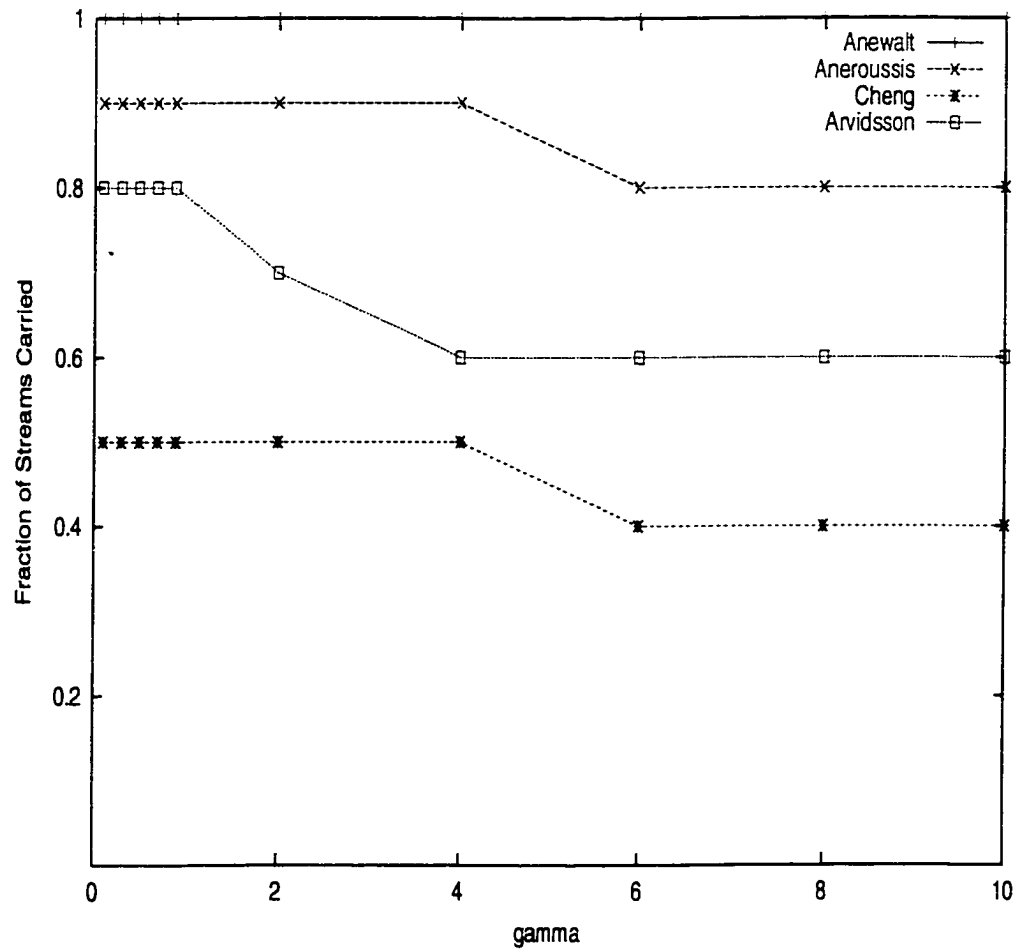
**Figure E.16**: Comparing the Fraction of Streams Carried calculated for our network and the 3 other networks Test Case 4.
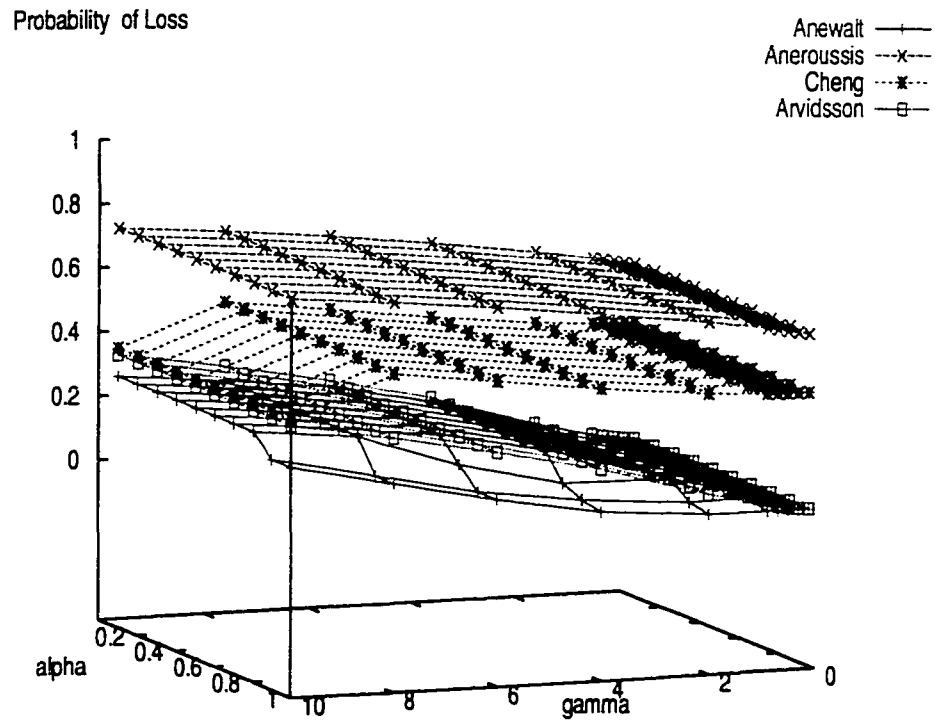
Probability of Loss

Anewalt ——+——
Aneroussis ---×---
Cheng ---✶---
Arvidsson ——◻——



**Figure E.17**: Comparing the Loss Probabilities calculated for our network and the 3 other networks Test Case 5.

Probability of Setup

Anewalt ——+——
Aneroussis ---x---
Cheng ---*---
Arvidsson ---ᵍ---



**Figure E.18**: Comparing the Setup Probabilities calculated for our network and the 3 other networks Test Case 5.

**Figure E.19**: Comparing the Fraction of Traffic Carried calculated for our network and the 3 other networks Test Case 5.

**Figure E.20**: Comparing the Fraction of Streams Carried calculated for our network and the 3 other networks Test Case 5.

**Figure E.21**: Comparing the Loss Probabilities calculated for our network and the 3 other networks Test Case 7.
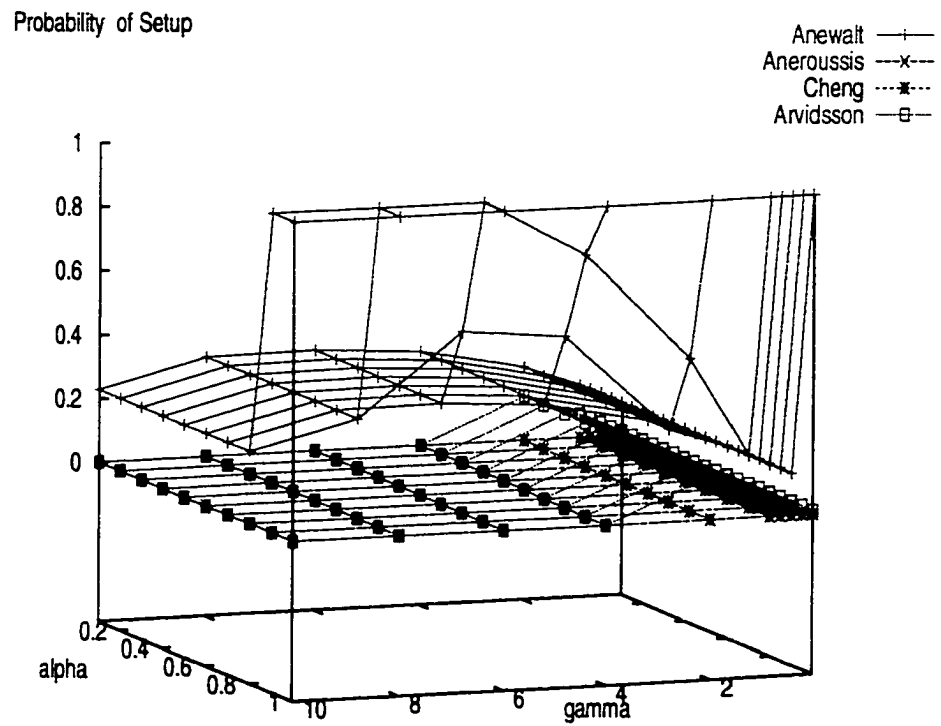
**Figure E.22**: Comparing the Setup Probabilities calculated for our network and the 3 other networks Test Case 7.
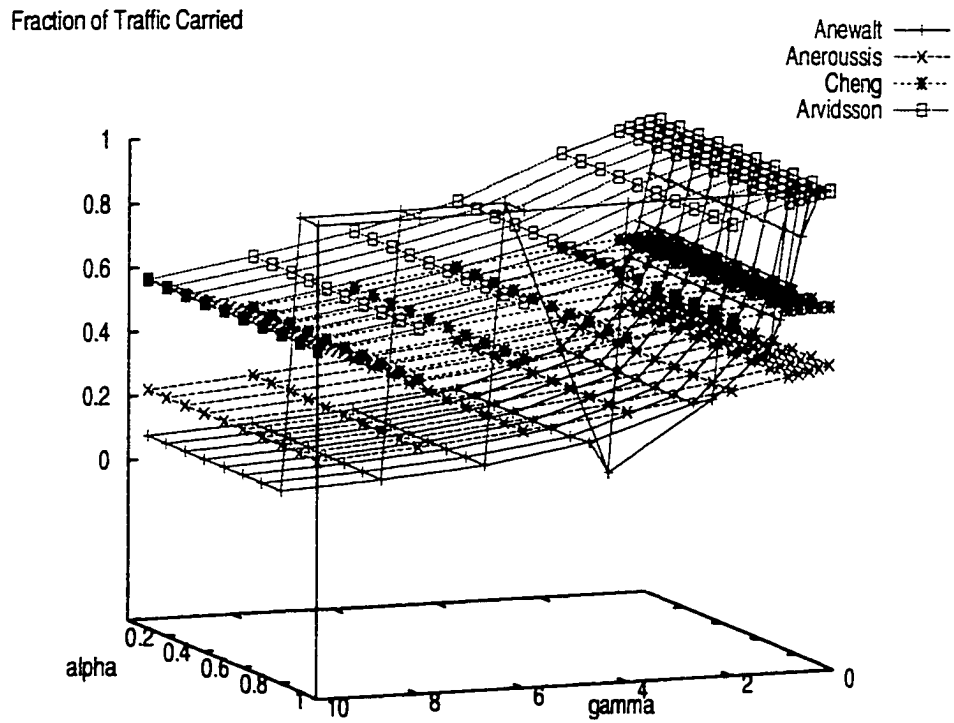
**Figure E.23**: Comparing the Fraction of Traffic Carried calculated for our network and the 3 other networks Test Case 7.
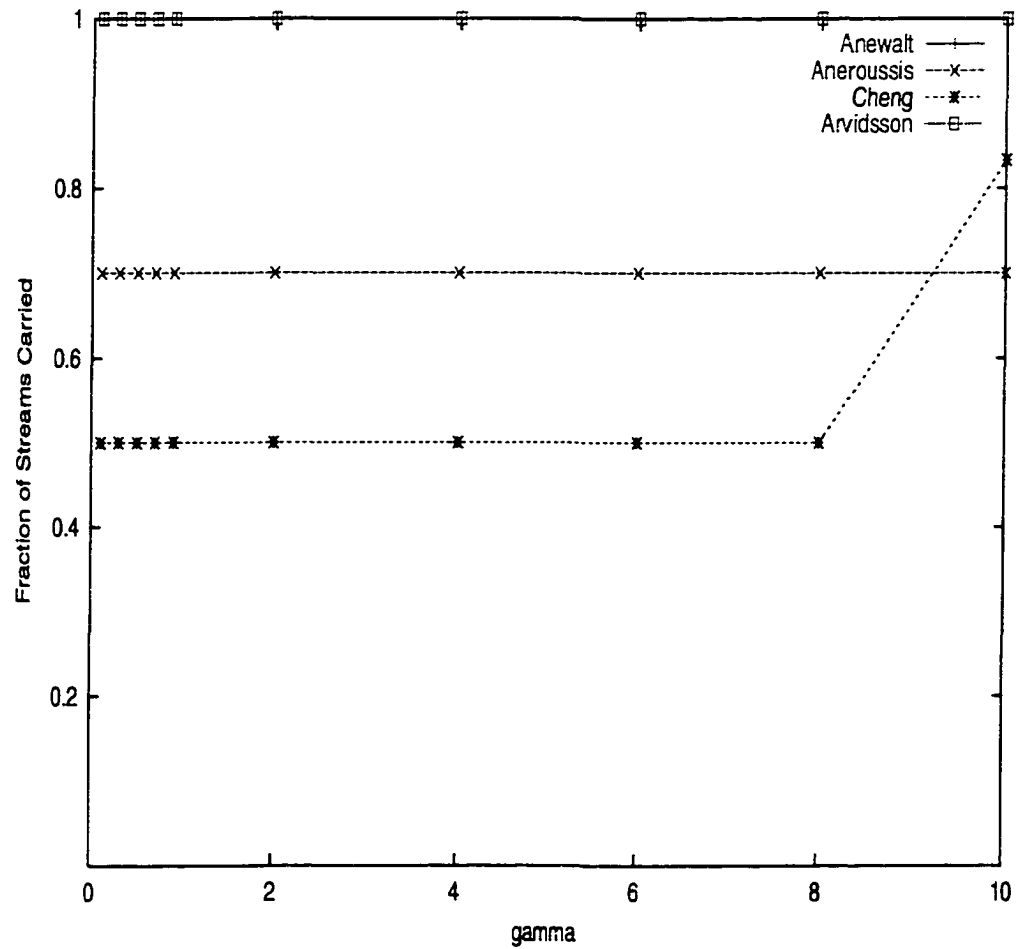
**Figure E.24**: Comparing the Fraction of Streams Carried calculated for our network and the 3 other networks Test Case 7.
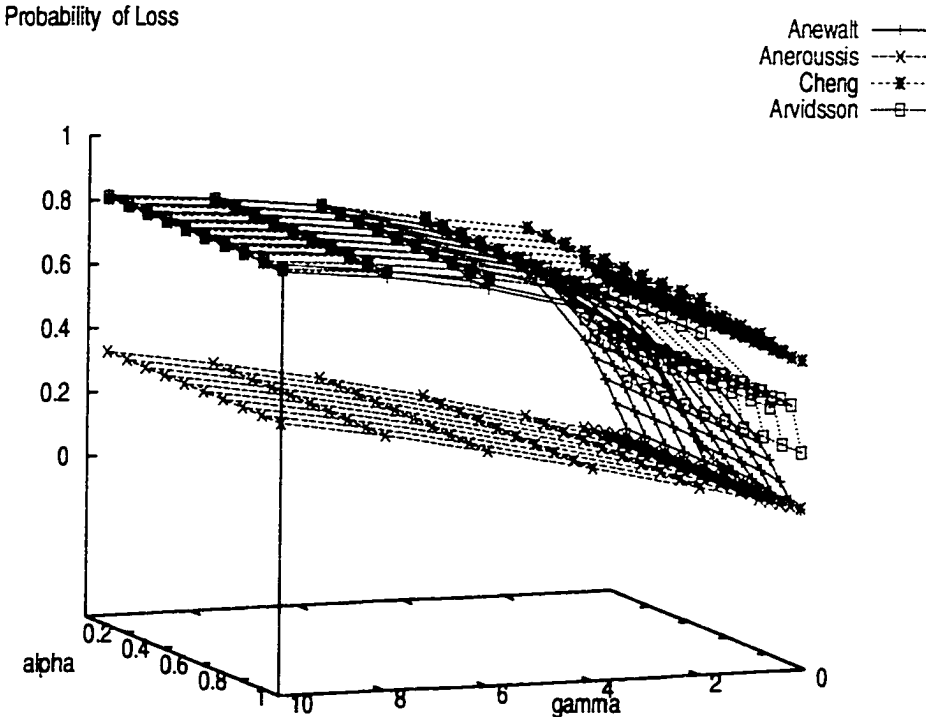
**Figure E.25**: Comparing the Loss Probabilities calculated for our network and the 3 other networks Test Case 8.

**Figure E.26**: Comparing the Setup Probabilities calculated for our network and the 3 other networks Test Case 8.

**Figure E.27**: Comparing the Fraction of Traffic Carried calculated for our network and the 3 other networks Test Case 8.

**Figure E.28**: Comparing the Fraction of Streams Carried calculated for our network and the 3 other networks Test Case 8.

**Figure E.29**: Comparing the Loss Probabilities calculated for our network and the 3 other networks Test Case 9.
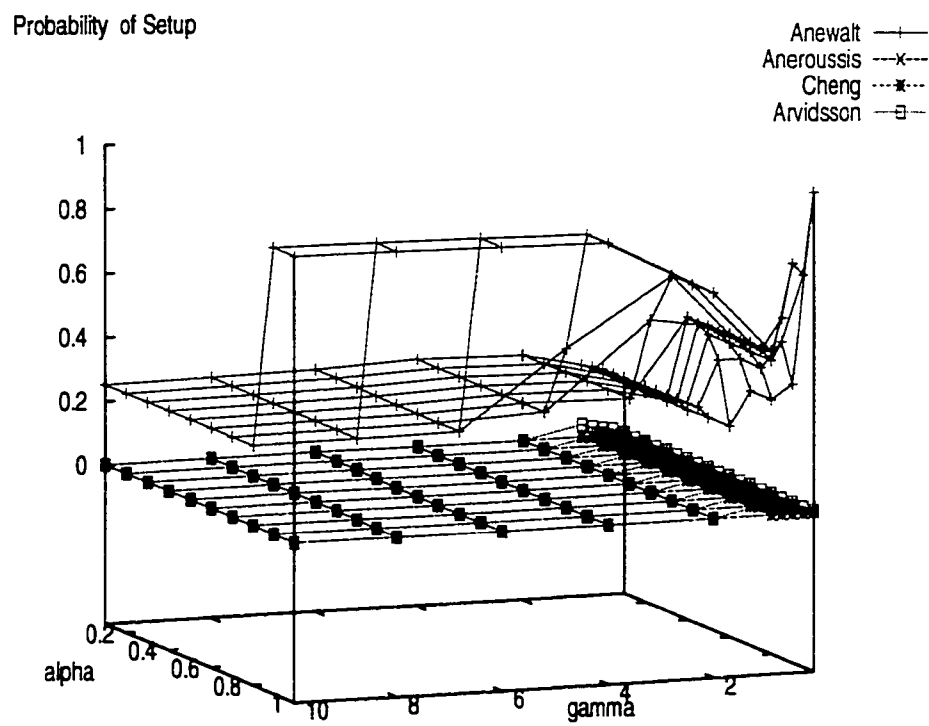
**Figure E.30**: Comparing the Setup Probabilities calculated for our network and the 3 other networks Test Case 9.
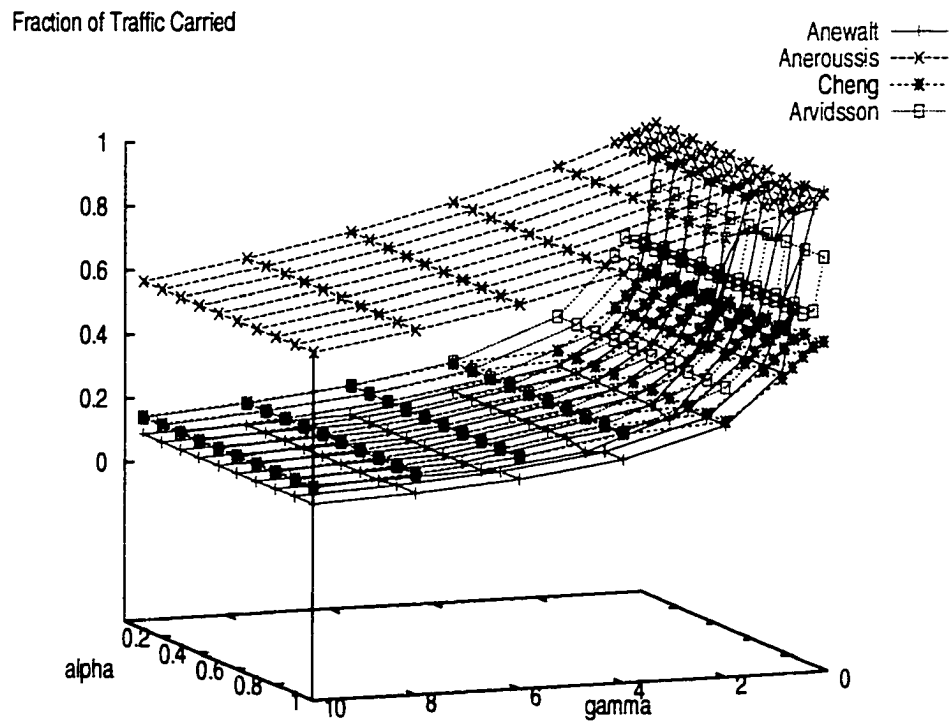
Fraction of Traffic Carried

Anewalt ——+——
Aneroussis ——×——
Cheng ····*····
Arvidsson ——□——

Figure E.31: Comparing the Fraction of Traffic Carried calculated for our network and the 3 other networks Test Case 9.

**Figure E.32**: Comparing the Fraction of Streams Carried calculated for our network and the 3 other networks Test Case 9.

**Figure E.33**: Comparing the Loss Probabilities calculated for our network and the 3 other networks Test Case 10.

Probability of Setup

Anewalt ——+——
Aneroussis ---x---
Cheng ---*---
Arvidsson —a—



**Figure E.34**: Comparing the Setup Probabilities calculated for our network and the 3 other networks Test Case 10.

**Figure E.35**: Comparing the Fraction of Traffic Carried calculated for our network and the 3 other networks Test Case 10.
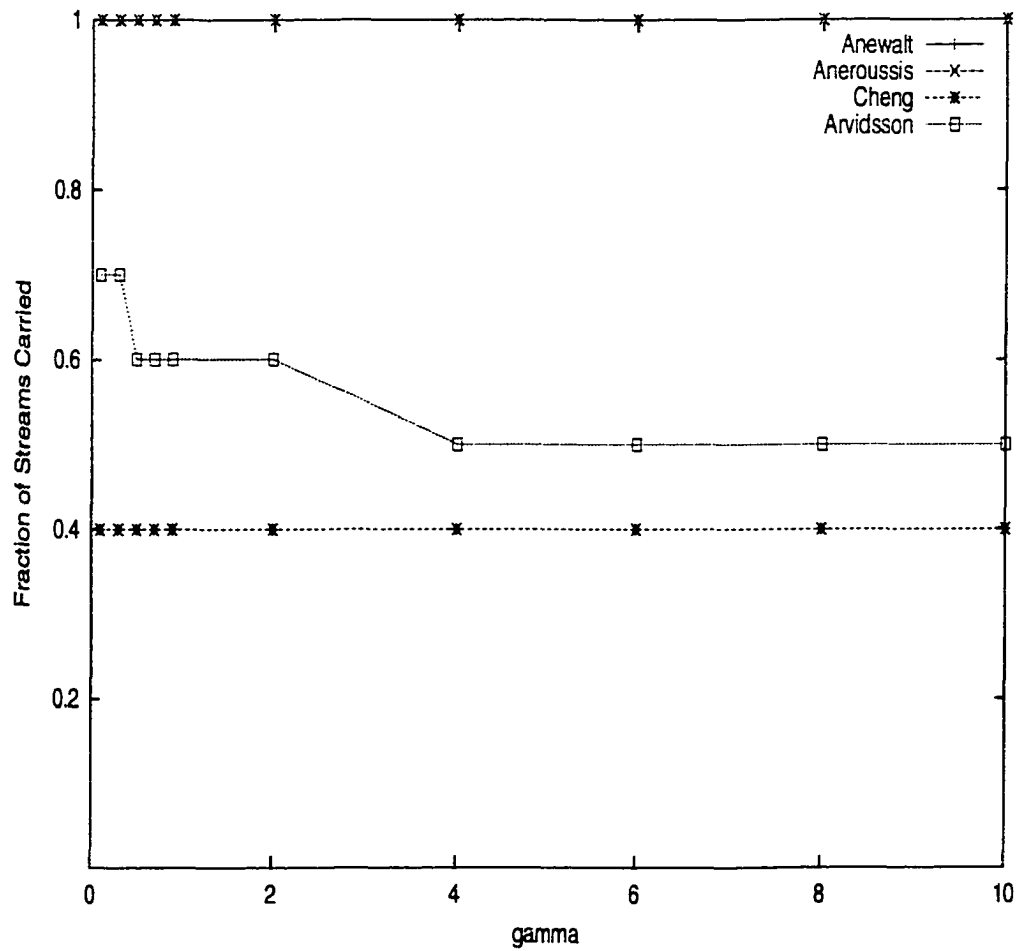
**Figure E.36**: Comparing the Fraction of Streams Carried calculated for our network and the 3 other networks Test Case 10.

# Appendix F

# The General Network

## F.1 Evaluating Algorithm Performance and Comparison of Our Algorithm to Other Algorithms

### F.1.1 Test Cases

| Test Case 1 | | | | | | |
|---|---|---|---|---|---|---|
| number of nodes: 7 | | | | | | |
| undirected edges: (0,2), (1,2), (2,3), (3,4), (3,5), (3,6) | | | | | | |
| capacity: 6 units in each direction | | | | | | |
| traffic distribution 1: | | | | | | |
| 01: 0.1 | 10: 0.2 | 20: 0.1 | 30: 0.3 | 40: 0.2 | 50: 0.1 | 60: 0.2 |
| 02: 0.1 | 12: 2.0 | 21: 0.5 | 31: 0.3 | 41: 0.2 | 51: 0.1 | 61: 0.2 |
| 03: 0.1 | 13: 4.0 | 23: 0.5 | 32: 0.3 | 42: 0.2 | 52: 0.1 | 62: 0.2 |
| 04: 0.1 | 14: 0.5 | 24: 0.1 | 34: 0.7 | 43: 0.2 | 53: 0.1 | 63: 0.2 |
| 05: 0.1 | 15: 3.2 | 25: 3.0 | 35: 0.7 | 45: 0.2 | 54: 0.1 | 64: 0.2 |
| 06: 0.1 | 16: 2.2 | 26: 0.1 | 36: 0.5 | 46: 0.2 | 56: 0.1 | 65: 0.2 |
| traffic distribution 2: | | | | | | |
| 01: 0.01 | 10: 0.02 | 20: 0.01 | 30: 0.03 | 40: 0.02 | 50: 0.01 | 60: 0.02 |
| 02: 0.01 | 12: 0.20 | 21: 0.05 | 31: 0.03 | 41: 0.02 | 51: 0.01 | 61: 0.02 |
| 03: 0.01 | 13: 0.40 | 23: 0.05 | 32: 0.03 | 42: 0.02 | 52: 0.01 | 62: 0.02 |
| 04: 0.01 | 14: 0.05 | 24: 0.01 | 34: 0.07 | 43: 0.02 | 53: 0.01 | 63: 0.02 |
| 05: 0.01 | 15: 0.32 | 25: 0.30 | 35: 0.07 | 45: 0.02 | 54: 0.01 | 64: 0.02 |
| 06: 0.01 | 16: 0.22 | 26: 0.01 | 36: 0.05 | 46: 0.02 | 56: 0.01 | 65: 0.02 |

| Test Case 2 | | | | | |
|---|---|---|---|---|---|
| number of nodes: 6 | | | | | |
| undirected edges: (0,1), (0,2), (0,3), (2,3), (2,4), (2,5), (3,4) | | | | | |
| capacity: 4 units in each direction | | | | | |
| traffic distribution 1: | | | | | |
| 01: 0.1 | 10: 0.6 | 20: 0.1 | 30: 1.6 | 40: 1.1 | 50: 1.6 |
| 02: 5.2 | 12: 0.7 | 21: 1.1 | 31: 1.7 | 41: 1.2 | 51: 2.7 |
| 03: 0.3 | 13: 0.8 | 23: 1.1 | 32: 1.8 | 42: 1.3 | 52: 1.8 |
| 04: 0.4 | 14: 0.9 | 24: 1.1 | 34: 0.9 | 43: 1.4 | 53: 1.9 |
| 05: 0.5 | 15: 1.0 | 25: 0.5 | 35: 2.0 | 45: 2.5 | 54: 0.9 |
| traffic distribution 2: | | | | | |
| 01: 0.1 | 10: 0.2 | 20: 0.1 | 30: 1.6 | 40: 1.1 | 50: 1.6 |
| 02: 0.2 | 12: 0.1 | 21: 0.1 | 31: 0.7 | 41: 1.2 | 51: 0.7 |
| 03: 2.3 | 13: 0.8 | 23: 0.1 | 32: 2.8 | 42: 1.3 | 52: 1.1 |
| 04: 2.4 | 14: 0.9 | 24: 1.1 | 34: 0.9 | 43: 1.4 | 53: 1.1 |
| 05: 0.5 | 15: 0.1 | 25: 2.5 | 35: 2.0 | 45: 0.5 | 54: 0.9 |

| Test Case 3 | | | | | |
|---|---|---|---|---|---|
| number of nodes: 6 | | | | | |
| undirected edges: (0,1), (0,2), (0,5), (1,3), (2,3), (2,4), (2,5), (3,4), (4,5) | | | | | |
| capacity: 4 units in each direction | | | | | |
| traffic distribution 1: | | | | | |
| 01: 0.25 | 10: 0.1 | 20: 1.3 | 30: 0.2 | 40: 0.1 | 50: 0.1 |
| 02: 0.25 | 12: 0.3 | 21: 0.2 | 31: 2.0 | 41: 0.2 | 51: 0.1 |
| 03: 0.25 | 13: 0.5 | 23: 0.3 | 32: 0.5 | 42: 0.3 | 52: 0.2 |
| 04: 0.2 | 14: 0.7 | 24: 0.4 | 34: 0.2 | 43: 0.4 | 53: 0.3 |
| 05: 0.1 | 15: 0.1 | 25: 0.5 | 35: 0.1 | 45: 0.5 | 54: 0.3 |
| traffic distribution 2: | | | | | |
| 01: 1.25 | 10: 1.1 | 20: 1.3 | 30: 1.2 | 40: 1.1 | 50: 1.1 |
| 02: 1.25 | 12: 1.3 | 21: 1.2 | 31: 2.0 | 41: 1.2 | 51: 1.1 |
| 03: 1.25 | 13: 1.5 | 23: 1.3 | 32: 1.5 | 42: 1.3 | 52: 2.2 |
| 04: 1.2 | 14: 1.7 | 24: 1.4 | 34: 1.2 | 43: 1.4 | 53: 1.3 |
| 05: 1.1 | 15: 1.1 | 25: 1.5 | 35: 2.1 | 45: 1.5 | 54: 1.3 |

**Table F.1**: Test Cases used to evaluate algorithm performance and to compare my algorithms to previous algorithms in general networks

## F.1.2 Results

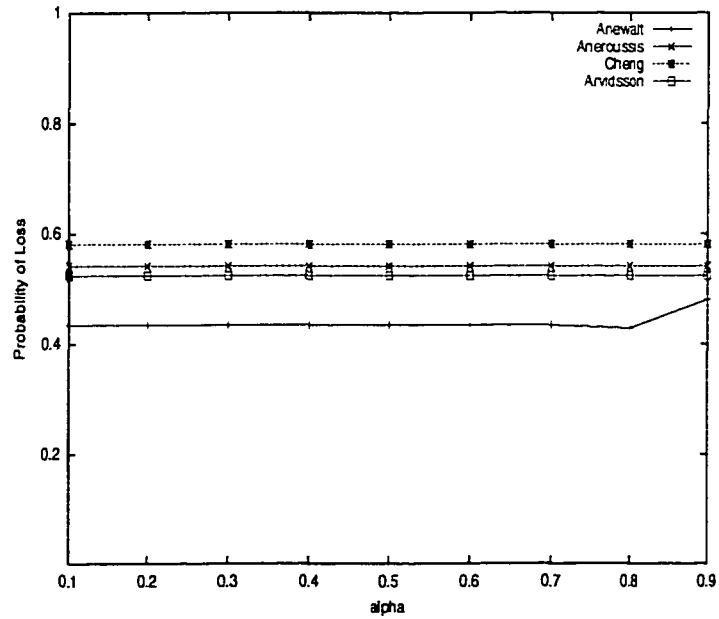### F.1.2.1 Graphs Comparing Performance Measures



**Figure F.1**: Comparing the Probability of Loss for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 1, Traffic Distribution 1.
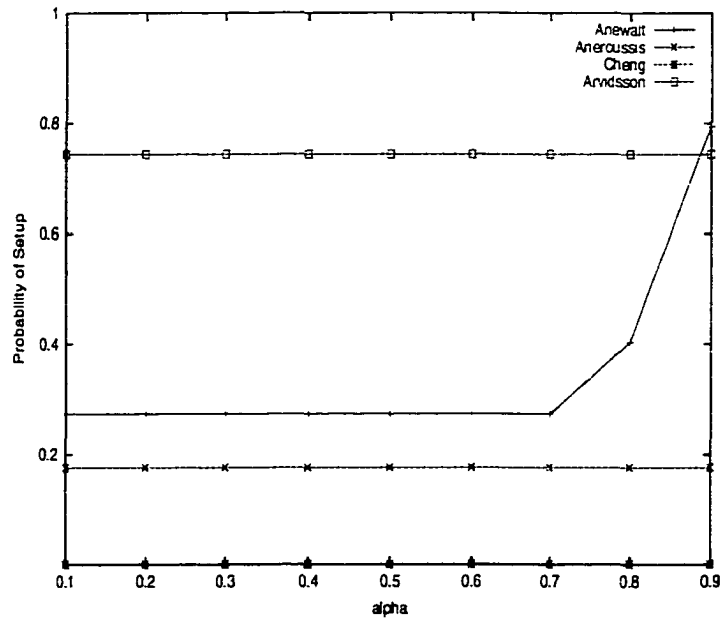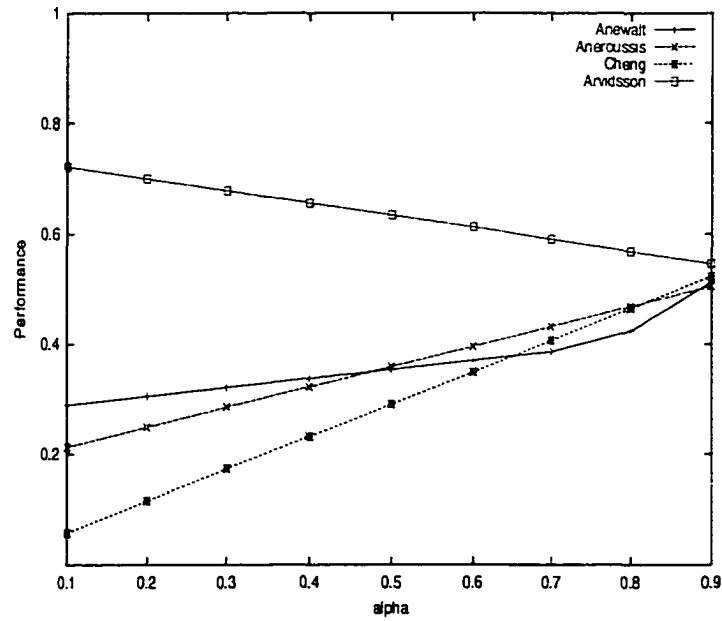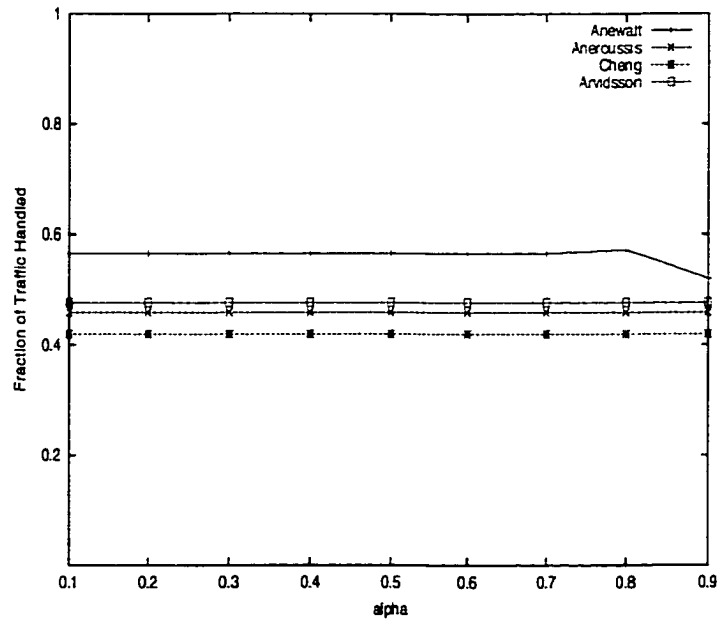
**Figure F.2**: Comparing the Probability of Setup for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 1, Traffic Distribution 1.



**Figure F.3**: Comparing the Overall Performance of the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 1, Traffic Distribution 1.

**Figure F.4**: Comparing the Fraction of Traffic handled by the layout produced from o ur algorithm and the layouts produced from the three other algorithms for Test Case 1, Traffic Distribution 1.
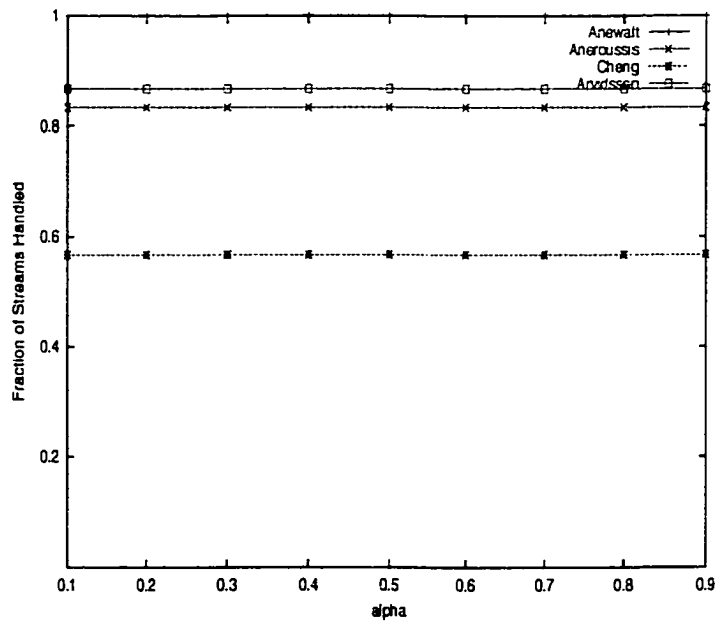


**Figure F.5**: Comparing the Fraction of Streams handled by the layout produced from o ur algorithm and the layouts produced from the three other algorithms for Test C ase 1, Traffic Distribution 1.
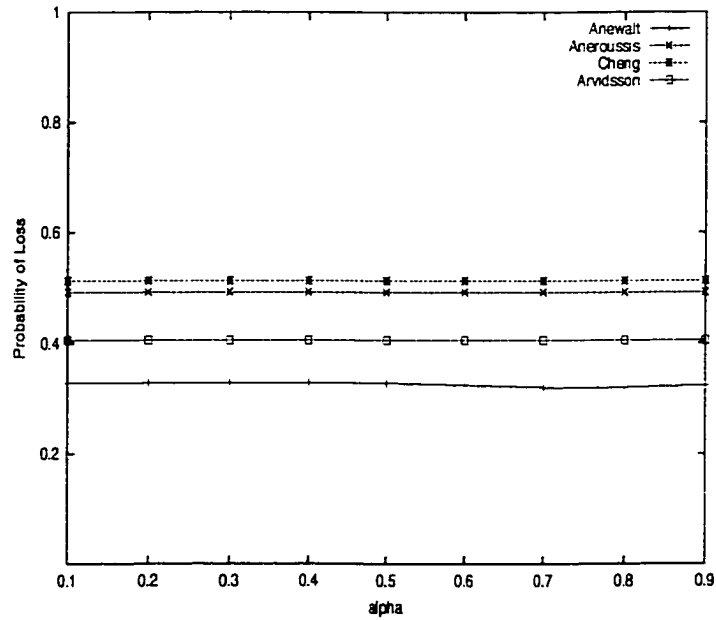
**Figure F.6**: Comparing the Probability of Loss for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 1, Traffic Distribution 2.
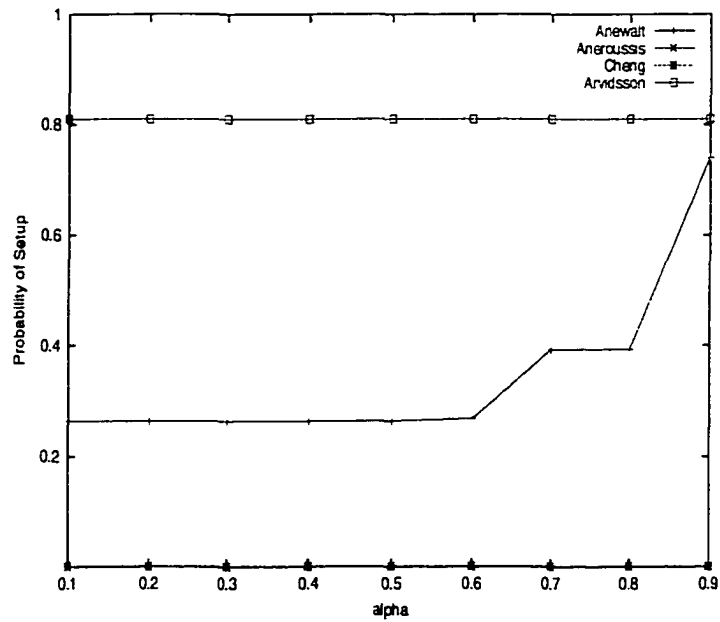
**Figure F.7**: Comparing the Probability of Setup for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 1, Traffic Distribution 2.
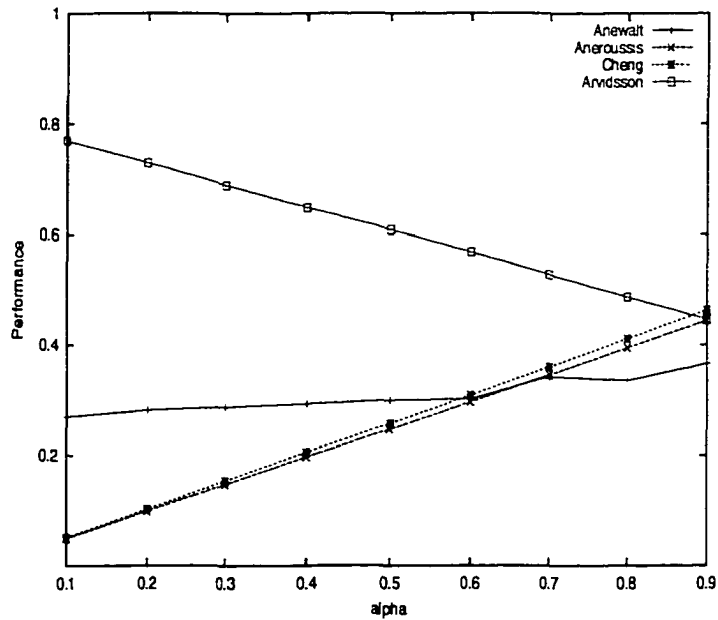


**Figure F.8**: Comparing the Overall Performance of the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 1, Traffic Distribution 2.

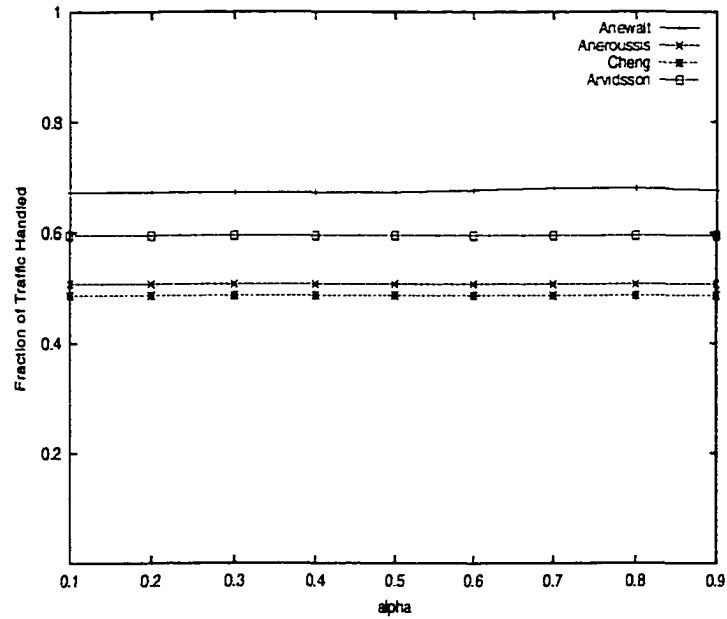**Figure F.9**: Comparing the Fraction of Traffic handled by the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 1, Traffic Distribution 2.



**Figure F.10**: Comparing the Fraction of Streams handled by the layout produced from o ur algorithm and the layouts produced from the three other algorithms for Test C ase 1, Traffic Distribution 2.

**Figure F.11**: Comparing the Probability of Loss for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 2, Traffic Distribution 1.

**Figure F.12**: Comparing the Probability of Setup for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 2, Traffic Distribution 1.



**Figure F.13**: Comparing the Overall Performance of the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 2, Traffic Distribution 1.

**Figure F.14**: Comparing the Fraction of Traffic handled by the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 2, Traffic Distribution 1.
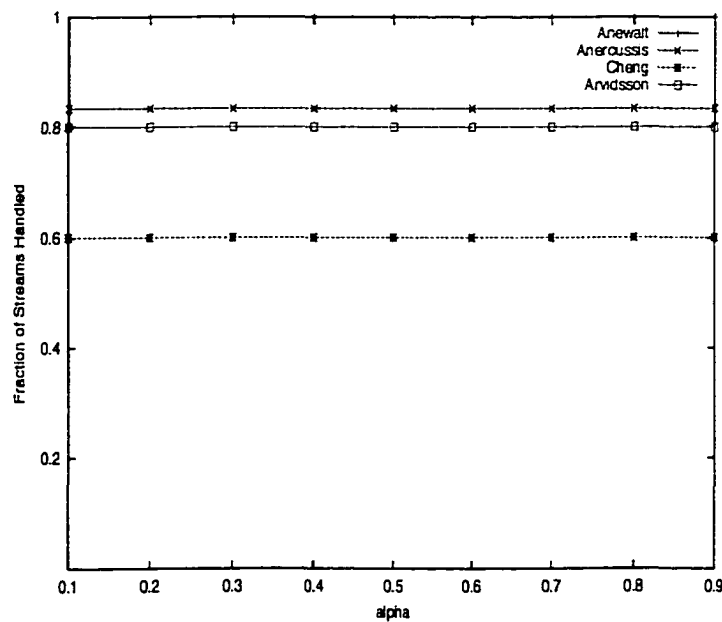


**Figure F.15**: Comparing the Fraction of Streams handled by the layout produced from o ur algorithm and the layouts produced from the three other algorithms for Test Case 2, Traffic Distribution 1.
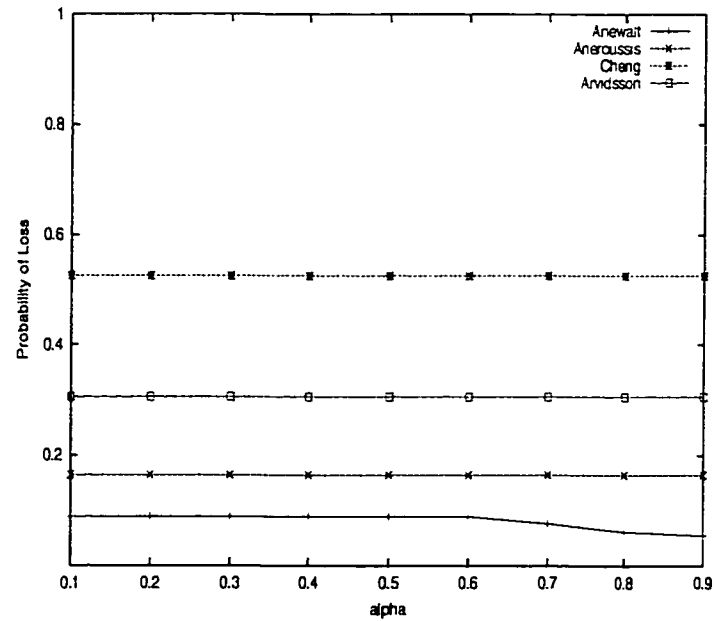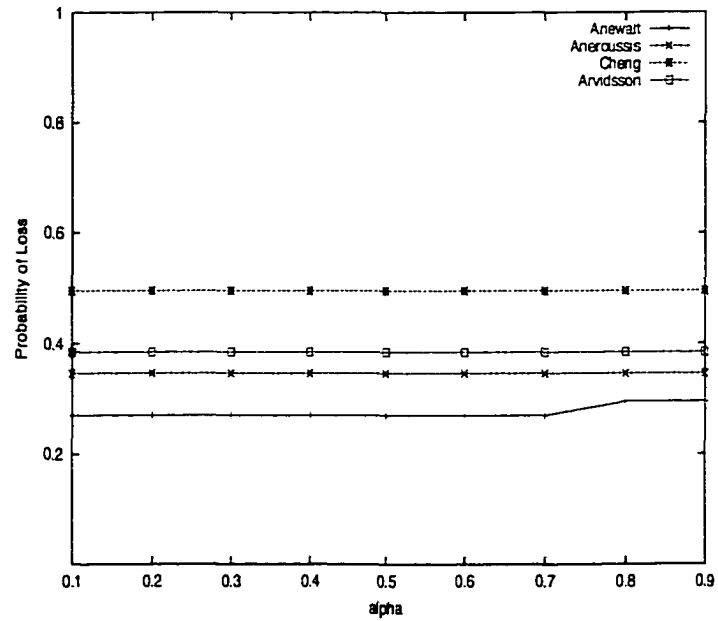
**Figure F.16**: Comparing the Probability of Loss for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 2, Traffic Distribution 2.
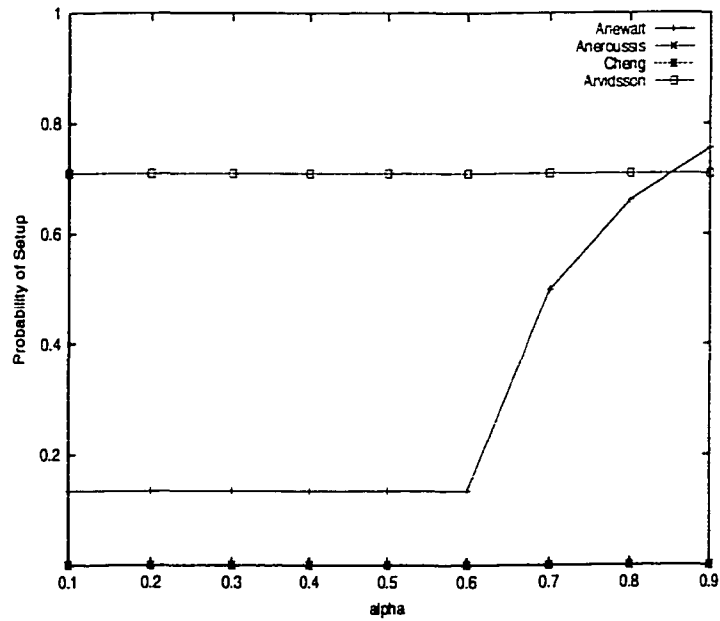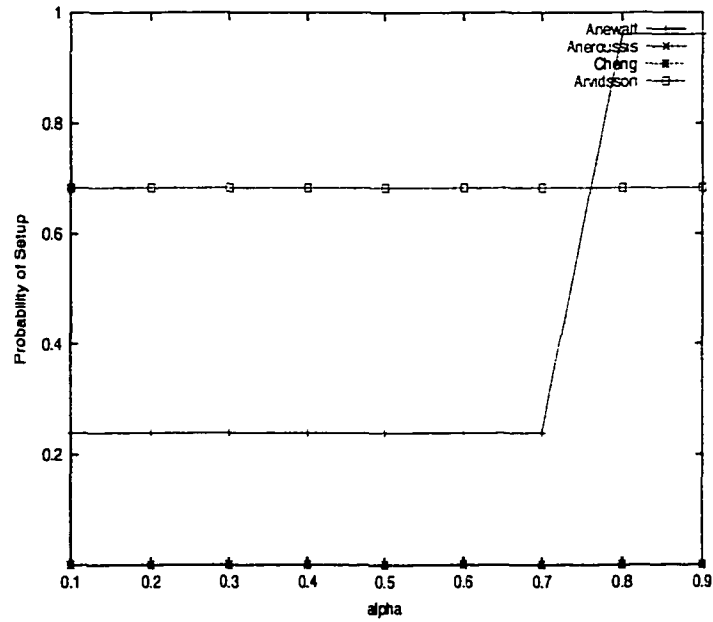
**Figure F.17**: Comparing the Probability of Setup for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 2, Traffic Distribution 2.
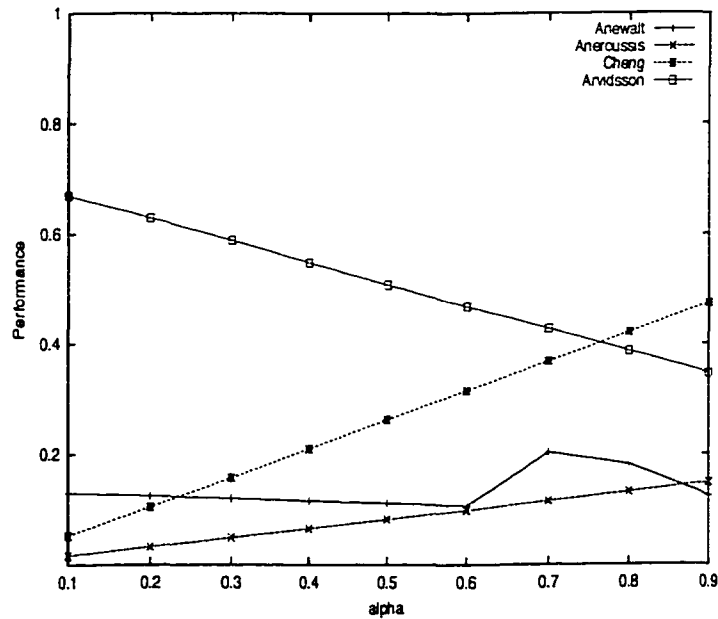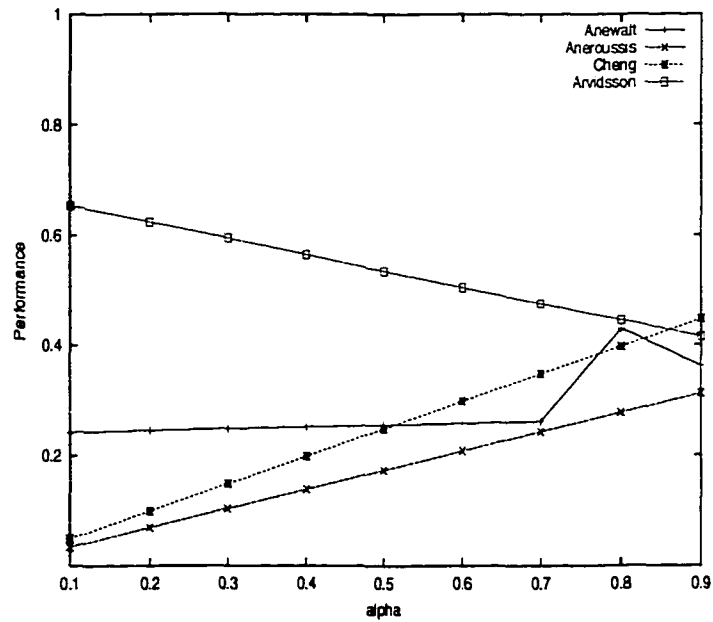


**Figure F.18**: Comparing the Overall Performance of the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 2, Traffic Distribution 2.
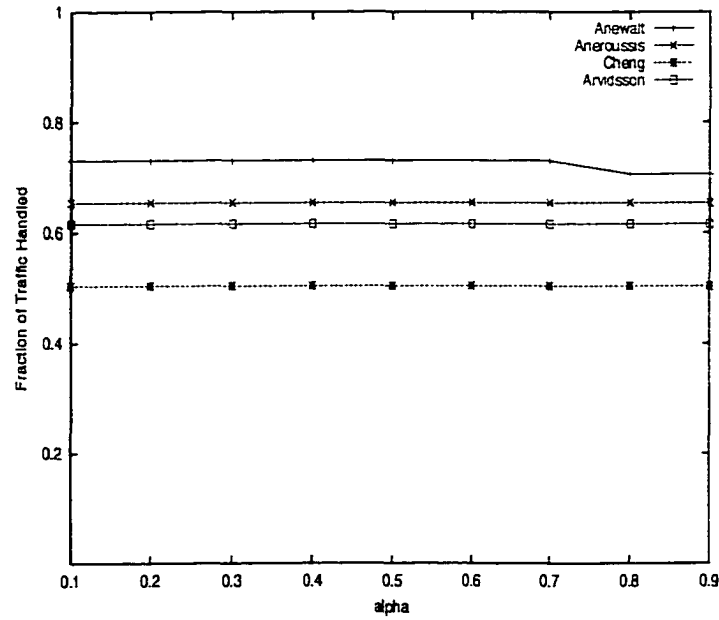
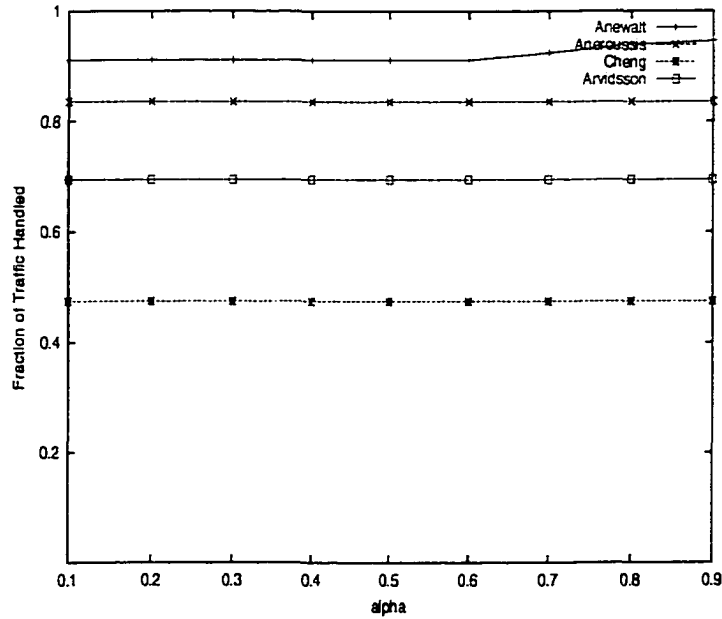**Figure F.19**: Comparing the Fraction of Traffic handled by the layout produced from o ur algorithm and the layouts produced from the three other algorithms for Test Case 2, Traffic Distribution 2.
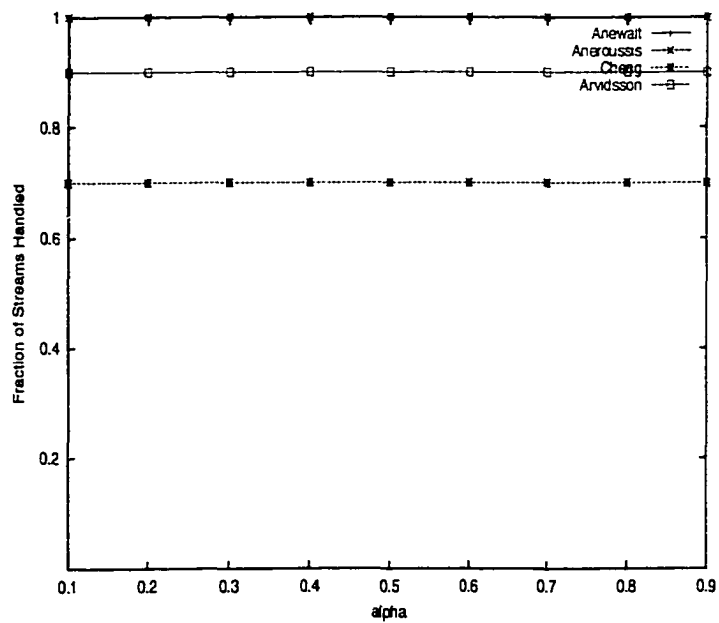


**Figure F.20**: Comparing the Fraction of Streams handled by the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 2, Traffic Distribution 2.
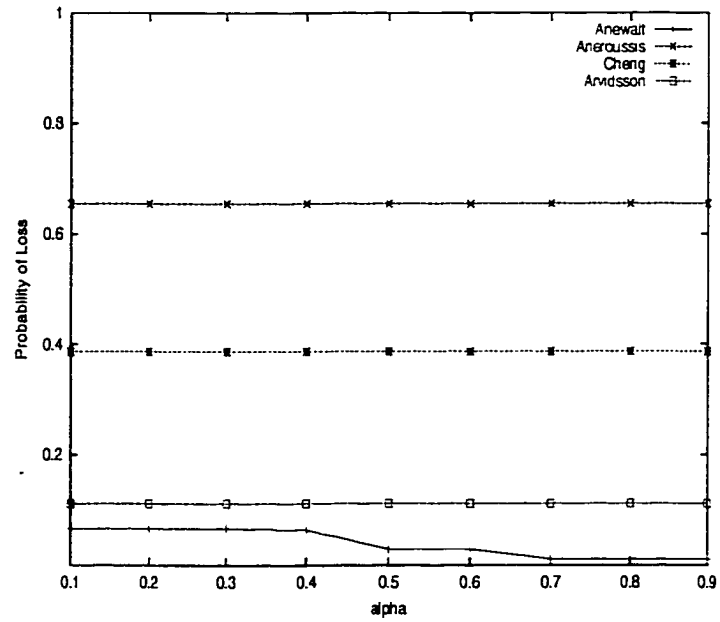
**Figure F.21**: Comparing the Probability of Loss for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 3, Traffic Distribution 2.
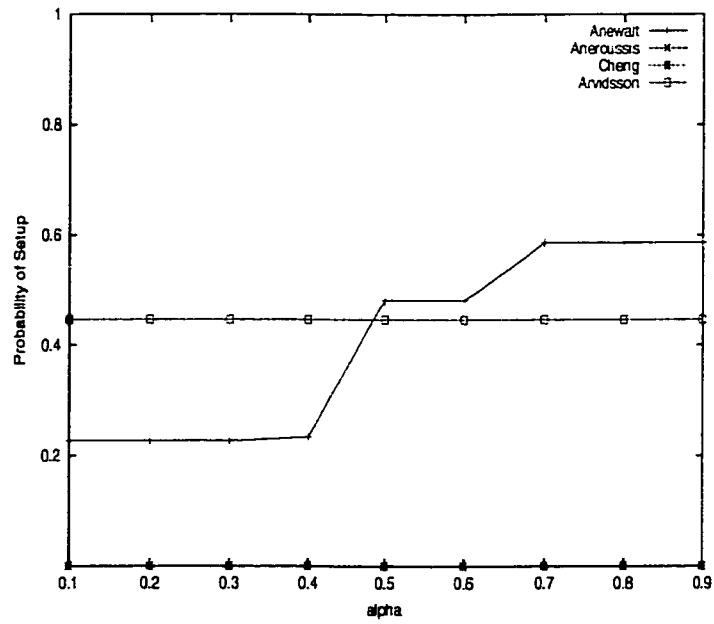
**Figure F.22**: Comparing the Probability of Setup for the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 3, Traffic Distribution 2.
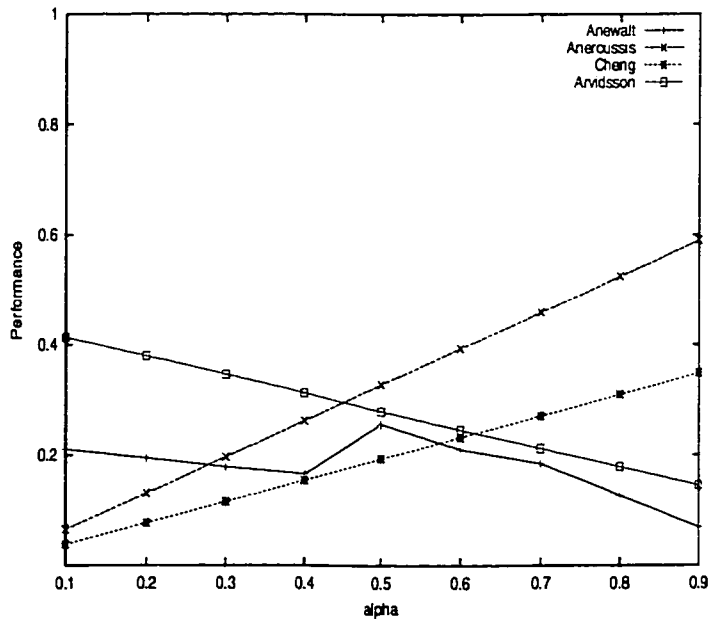


**Figure F.23**: Comparing the Overall Performance of the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 3, Traffic Distribution 2.
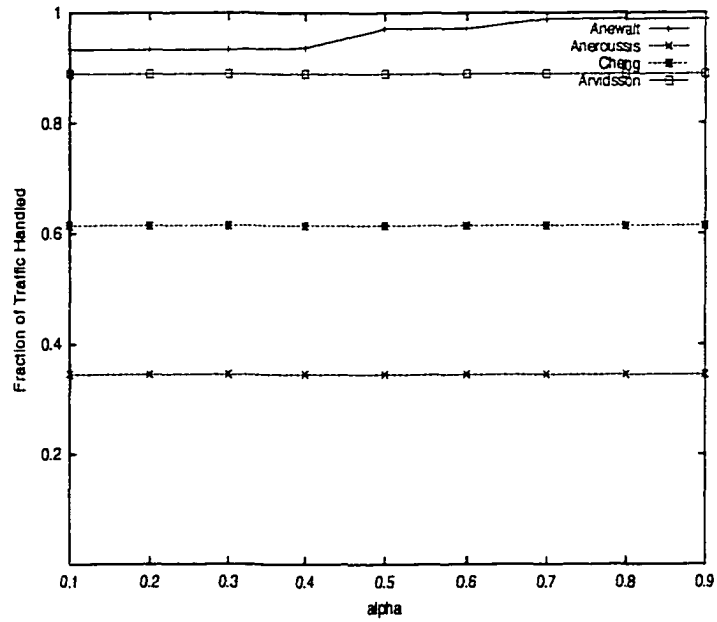
**Figure F.24**: Comparing the Fraction Traffic handled by the layout produced from our algorithm and the layouts produced from the three other algorithms for Test Case 3, Traffic Distribution 2.
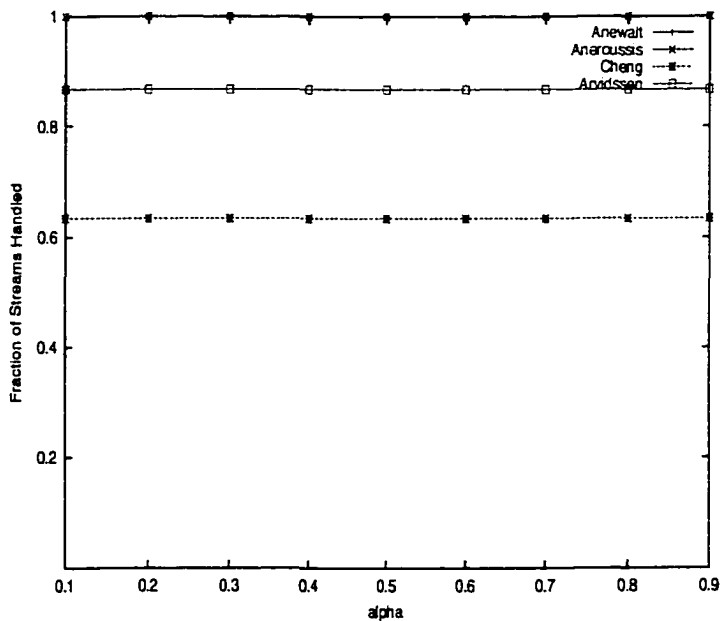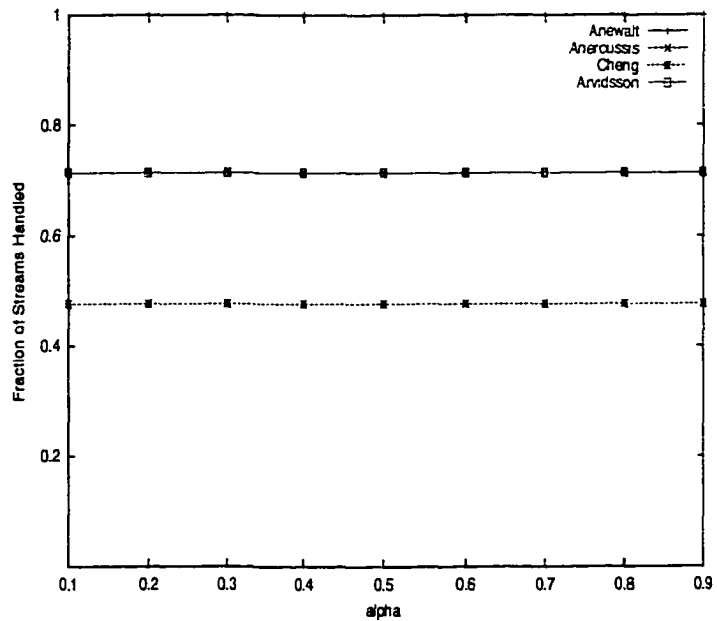


**Figure F.25**: Comparing the Fraction of Streams handled by the layout produced from o ur algorithm and the layouts produced from the three other algorithms for Test Case 3, Traffic Distribution 2.

## F.1.2.2 VPDBA Solutions

**Test Case 1 - Traffic Distribution 1**

| Aneroussis | | Cheng | | Arvidsson | |
|---|---|---|---|---|---|
| src-dest | capacity | src-dest | capacity | src-dest | capacity |
| 01 | 1 | 01 | 5 | 13 | 3 |
| 02 | 1 | 02 | 1 | 15 | 3 |
| 03 | 1 | 10 | 2 | 30 | 1 |
| 04 | 1 | 13 | 2 | 31 | 1 |
| 05 | 1 | 15 | 1 | 32 | 1 |
| 06 | 1 | 16 | 1 | 40 | 1 |
| 10 | 2 | 20 | 1 | 42 | 1 |
| 12 | 4 | 25 | 2 | 62 | 1 |
| 20 | 2 | 30 | 2 | | |
| 21 | 3 | 31 | 1 | | |
| 24 | 1 | 42 | 1 | | |
| 26 | 1 | 45 | 2 | | |
| 34 | 2 | 46 | 3 | | |
| 35 | 2 | 54 | 4 | | |
| 36 | 2 | 56 | 2 | | |
| 40 | 1 | 60 | 1 | | |
| 41 | 1 | 62 | 1 | | |
| 42 | 1 | 63 | 1 | | |
| 43 | 1 | 64 | 2 | | |
| 45 | 1 | 65 | 1 | | |
| 46 | 1 | | | | |
| 50 | 1 | | | | |
| 51 | 1 | | | | |
| 52 | 1 | | | | |
| 53 | 1 | | | | |
| 54 | 1 | | | | |
| 56 | 1 | | | | |
| 63 | 3 | | | | |
| 64 | 1 | | | | |
| 65 | 2 | | | | |

| Anewalt | | | | | |
|---|---|---|---|---|---|
| α: 0.1 - 0.6 | | α: 0.7 | | α: 0.8 - 0.9 | |
| src-dest | capacity | src-dest | capacity | src-dest | capacity |
| 01 | 1 | 12 | 1 | 13 | 1 |
| 02 | 2 | 13 | 2 | 15 | 1 |
| 12 | 1 | 15 | 1 | 25 | 1 |
| 13 | 2 | 16 | 1 | | |
| 15 | 1 | 25 | 1 | | |
| 16 | 1 | 30 | 1 | | |
| 20 | 3 | 31 | 1 | | |
| 21 | 3 | 32 | 1 | | |
| 25 | 1 | 35 | 2 | | |
| 30 | 1 | 36 | 2 | | |
| 31 | 1 | 45 | 1 | | |
| 32 | 1 | 46 | 1 | | |
| 34 | 3 | 56 | 1 | | |
| 35 | 2 | | | | |
| 36 | 2 | | | | |
| 42 | 1 | | | | |
| 43 | 2 | | | | |
| 45 | 1 | | | | |
| 46 | 1 | | | | |
| 53 | 3 | | | | |
| 54 | 1 | | | | |
| 56 | 1 | | | | |
| 62 | 1 | | | | |
| 63 | 3 | | | | |
| 64 | 1 | | | | |

## Test Case 1 - Traffic Distribution 2

| Aneroussis | | Cheng | | Arvidsson | |
|---|---|---|---|---|---|
| src-dest | capacity | src-dest | capacity | src-dest | capacity |
| 01 | 1 | 01 | 5 | 13 | 1 |
| 02 | 1 | 02 | 1 | 14 | 1 |
| 03 | 1 | 10 | 2 | 15 | 1 |
| 04 | 1 | 13 | 2 | 16 | 1 |
| 05 | 1 | 15 | 1 | 23 | 1 |
| 06 | 1 | 16 | 1 | 25 | 1 |
| 10 | 2 | 20 | 1 | 30 | 1 |
| 12 | 4 | 25 | 2 | 31 | 1 |
| 20 | 2 | 30 | 2 | 32 | 1 |
| 21 | 3 | 31 | 1 | 40 | 1 |
| 24 | 1 | 42 | 1 | 42 | 1 |
| 26 | 1 | 45 | 2 | 62 | 1 |
| 35 | 2 | 46 | 3 | | |
| 36 | 2 | 54 | 4 | | |
| 40 | 1 | 56 | 2 | | |
| 41 | 1 | 60 | 1 | | |
| 42 | 1 | 62 | 1 | | |
| 43 | 1 | 63 | 1 | | |
| 45 | 1 | 64 | 2 | | |
| 46 | 1 | 65 | 1 | | |
| 50 | 1 | | | | |
| 51 | 1 | | | | |
| 52 | 1 | | | | |
| 53 | 1 | | | | |
| 54 | 1 | | | | |
| 56 | 1 | | | | |
| 63 | 3 | | | | |
| 64 | 1 | | | | |
| 65 | 2 | | | | |

| Anewalt | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| $\alpha$: 0.1 - 0.3 | | $\alpha$: 0.4 | | $\alpha$: 0.5 - 0.6 | | $\alpha$: 0.7 - 0.9 | |
| src-dest | capacity | src-dest | capacity | src-dest | capacity | src-dest | capacity |
| 01 | 1 | 01 | 1 | 12 | 1 | 13 | 1 |
| 02 | 3 | 12 | 2 | 13 | 1 | 15 | 1 |
| 12 | 2 | 13 | 1 | 15 | 1 | 16 | 1 |
| 13 | 1 | 15 | 1 | 16 | 1 | 25 | 1 |
| 15 | 1 | 16 | 1 | 25 | 2 | | |
| 16 | 1 | 20 | 1 | | | | |
| 20 | 3 | 21 | 2 | | | | |
| 21 | 2 | 25 | 2 | | | | |
| 25 | 2 | 31 | 1 | | | | |
| 31 | 1 | 32 | 1 | | | | |
| 32 | 1 | 34 | 3 | | | | |
| 34 | 3 | 35 | 1 | | | | |
| 35 | 1 | 36 | 2 | | | | |
| 36 | 2 | 40 | 1 | | | | |
| 40 | 1 | 41 | 1 | | | | |
| 41 | 1 | 42 | 1 | | | | |
| 42 | 1 | 45 | 1 | | | | |
| 45 | 1 | 46 | 1 | | | | |
| 46 | 1 | 53 | 2 | | | | |
| 53 | 3 | 54 | 1 | | | | |
| 54 | 1 | 56 | 1 | | | | |
| 56 | 1 | 63 | 3 | | | | |
| 63 | 4 | 64 | 1 | | | | |
| 64 | 1 | | | | | | |

## Test Case 2 - Traffic Distribution 1

| Aneroussis | | Cheng | | Arvidsson | |
|---|---|---|---|---|---|
| src-dest | capacity | src-dest | capacity | src-dest | capacity |
| 01 | 1 | 03 | 1 | 02 | 3 |
| 03 | 3 | 04 | 3 | 15 | 1 |
| 04 | 1 | 13 | 3 | 41 | 1 |
| 05 | 1 | 14 | 1 | 50 | 1 |
| 10 | 1 | 21 | 2 | 51 | 2 |
| 12 | 1 | 23 | 2 | | |
| 13 | 1 | 30 | 3 | | |
| 14 | 1 | 31 | 1 | | |
| 20 | 1 | 32 | 2 | | |
| 21 | 1 | 34 | 4 | | |
| 23 | 3 | 35 | 2 | | |
| 24 | 1 | 42 | 2 | | |
| 25 | 1 | 43 | 4 | | |
| 30 | 3 | 45 | 2 | | |
| 31 | 1 | 50 | 1 | | |
| 32 | 2 | 51 | 1 | | |
| 34 | 4 | 53 | 2 | | |
| 35 | 2 | | | | |
| 40 | 1 | | | | |
| 41 | 1 | | | | |
| 42 | 2 | | | | |
| 43 | 4 | | | | |
| 52 | 2 | | | | |
| 53 | 1 | | | | |
| 54 | 1 | | | | |

| Anewalt | | | | | |
|---|---|---|---|---|---|
| α: 0.1 - 0.7 | | α: 0.8 | | α: 0.9 | |
| src-dest | capacity | src-dest | capacity | src-dest | capacity |
| 02 | 3 | 02 | 3 | 02 | 3 |
| 03 | 1 | 10 | 1 | 45 | 1 |
| 10 | 1 | 13 | 2 | 51 | 1 |
| 13 | 2 | 21 | 1 | 52 | 1 |
| 20 | 1 | 23 | 2 | | |
| 21 | 1 | 24 | 3 | | |
| 23 | 2 | 30 | 2 | | |
| 24 | 3 | 31 | 1 | | |
| 30 | 2 | 32 | 2 | | |
| 31 | 1 | 35 | 1 | | |
| 32 | 2 | 42 | 1 | | |
| 34 | 3 | 45 | 2 | | |
| 35 | 1 | 51 | 1 | | |
| 42 | 1 | 52 | 1 | | |
| 43 | 3 | 53 | 1 | | |
| 45 | 2 | | | | |
| 51 | 1 | | | | |
| 52 | 1 | | | | |
| 53 | 1 | | | | |

## Test Case 2 - Traffic Distribution 2

| Aneroussis | | Cheng | | Arvidsson | |
|---|---|---|---|---|---|
| src-dest | capacity | src-dest | capacity | src-dest | capacity |
| 01 | 1 | 03 | 2 | 04 | 2 |
| 02 | 1 | 04 | 2 | 05 | 1 |
| 03 | 3 | 12 | 1 | 14 | 1 |
| 05 | 1 | 13 | 2 | 40 | 2 |
| 10 | 1 | 14 | 1 | 50 | 1 |
| 12 | 1 | 23 | 2 | 51 | 1 |
| 13 | 1 | 24 | 1 | | |
| 15 | 1 | 30 | 1 | | |
| 20 | 1 | 31 | 3 | | |
| 21 | 1 | 32 | 2 | | |
| 23 | 3 | 34 | 4 | | |
| 24 | 3 | 35 | 2 | | |
| 30 | 3 | 40 | 1 | | |
| 31 | 1 | 41 | 1 | | |
| 32 | 3 | 43 | 4 | | |
| 34 | 4 | 45 | 2 | | |
| 35 | 1 | 50 | 2 | | |
| 40 | 1 | 53 | 2 | | |
| 42 | 2 | | | | |
| 43 | 4 | | | | |
| 45 | 1 | | | | |
| 51 | 1 | | | | |
| 52 | 1 | | | | |
| 53 | 1 | | | | |
| 54 | 1 | | | | |

| Anewalt | | | | | |
|---|---|---|---|---|---|
| $\alpha$: 0.1 - 0.6 | | $\alpha$: 0.7 - 0.8 | | $\alpha$: 0.9 | |
| src-dest | capacity | src-dest | capacity | src-dest | capacity |
| 01 | 1 | 03 | 3 | 04 | 1 |
| 02 | 1 | 04 | 2 | 25 | 2 |
| 03 | 3 | 24 | 1 | 32 | 1 |
| 04 | 2 | 25 | 2 | 35 | 1 |
| 10 | 3 | 30 | 2 | 50 | 1 |
| 23 | 2 | 31 | 1 | | |
| 24 | 1 | 32 | 2 | | |
| 25 | 2 | 35 | 1 | | |
| 30 | 2 | 40 | 1 | | |
| 31 | 1 | 41 | 1 | | |
| 32 | 2 | 42 | 1 | | |
| 34 | 3 | 50 | 1 | | |
| 35 | 1 | 52 | 1 | | |
| 40 | 1 | 53 | 1 | | |
| 41 | 1 | | | | |
| 42 | 1 | | | | |
| 43 | 3 | | | | |
| 50 | 1 | | | | |
| 52 | 1 | | | | |
| 53 | 1 | | | | |

## Test Case 3 - Traffic Distribution 1

| Aneroussis | | Cheng | | Arvidsson | |
| --- | --- | --- | --- | --- | --- |
| src-dest | capacity | src-dest | capacity | src-dest | capacity |
| 01 | 1 | 04 | 2 | 03 | 1 |
| 02 | 2 | 05 | 2 | 04 | 1 |
| 03 | 1 | 12 | 2 | 12 | 2 |
| 04 | 1 | 13 | 2 | 14 | 3 |
| 05 | 3 | 14 | 2 | 15 | 1 |
| 10 | 1 | 15 | 2 | 21 | 2 |
| 12 | 1 | 21 | 1 | 30 | 1 |
| 13 | 1 | 24 | 2 | 35 | 1 |
| 14 | 2 | 25 | 4 | 40 | 1 |
| 15 | 1 | 32 | 4 | 41 | 2 |
| 20 | 2 | 35 | 2 | 51 | 1 |
| 21 | 1 | 40 | 3 | | |
| 23 | 2 | 41 | 4 | | |
| 24 | 3 | 42 | 1 | | |
| 25 | 4 | 45 | 2 | | |
| 30 | 1 | 50 | 1 | | |
| 31 | 2 | 51 | 3 | | |
| 32 | 4 | 53 | 4 | | |
| 34 | 1 | 54 | 4 | | |
| 35 | 1 | | | | |
| 40 | 1 | | | | |
| 41 | 1 | | | | |
| 42 | 3 | | | | |
| 43 | 3 | | | | |
| 45 | 3 | | | | |
| 50 | 3 | | | | |
| 51 | 1 | | | | |
| 52 | 2 | | | | |
| 53 | 2 | | | | |
| 54 | 4 | | | | |

| Anewalt | | | | | | | |
|---|---|---|---|---|---|---|---|
| $\alpha$: 0.1 - 0.6 | | $\alpha$: 0.7 | | $\alpha$: 0.8 | | $\alpha$: 0.9 | |
| src-dest | capacity | src-dest | capacity | src-dest | capacity | src-dest | capacity |
| 01 | 2 | 01 | 1 | 13 | 1 | 20 | 2 |
| 02 | 1 | 02 | 1 | 14 | 1 | 31 | 3 |
| 04 | 1 | 04 | 1 | 20 | 3 | | |
| 05 | 2 | 12 | 1 | 31 | 3 | | |
| 10 | 1 | 13 | 1 | | | | |
| 12 | 1 | 14 | 2 | | | | |
| 13 | 1 | 20 | 3 | | | | |
| 14 | 2 | 24 | 2 | | | | |
| 15 | 1 | 31 | 3 | | | | |
| 20 | 3 | 34 | 1 | | | | |
| 23 | 2 | 51 | 1 | | | | |
| 24 | 2 | | | | | | |
| 25 | 3 | | | | | | |
| 31 | 3 | | | | | | |
| 32 | 3 | | | | | | |
| 34 | 1 | | | | | | |
| 42 | 3 | | | | | | |
| 43 | 3 | | | | | | |
| 45 | 3 | | | | | | |
| 50 | 2 | | | | | | |
| 51 | 1 | | | | | | |
| 52 | 2 | | | | | | |
| 53 | 1 | | | | | | |
| 54 | 3 | | | | | | |

## Test Case 3 - Traffic Distribution 2

| Aneroussis | | Cheng | | Arvidsson | |
|---|---|---|---|---|---|
| Arvidsson src-dest | capacity | src-dest | capacity | src-dest | capacity |
| 01 | 1 | 03 | 1 | 03 | 1 |
| 02 | 1 | 04 | 3 | 04 | 2 |
| 03 | 1 | 05 | 2 | 12 | 1 |
| 04 | 2 | 12 | 1 | 14 | 2 |
| 05 | 3 | 13 | 1 | 15 | 2 |
| 10 | 1 | 14 | 2 | 21 | 1 |
| 12 | 1 | 15 | 2 | 30 | 1 |
| 13 | 2 | 21 | 1 | 35 | 2 |
| 14 | 1 | 24 | 1 | 40 | 2 |
| 15 | 1 | 25 | 4 | 41 | 2 |
| 20 | 1 | 30 | 1 | 51 | 2 |
| 21 | 1 | 32 | 4 | | |
| 23 | 2 | 35 | 2 | | |
| 24 | 2 | 40 | 3 | | |
| 25 | 4 | 41 | 3 | | |
| 30 | 1 | 42 | 1 | | |
| 31 | 1 | 43 | 1 | | |
| 32 | 4 | 45 | 2 | | |
| 34 | 2 | 50 | 2 | | |
| 35 | 1 | 51 | 2 | | |
| 40 | 2 | 53 | 4 | | |
| 41 | 2 | 54 | 4 | | |
| 42 | 2 | | | | |
| 43 | 2 | | | | |
| 45 | 3 | | | | |
| 50 | 3 | | | | |
| 51 | 1 | | | | |
| 52 | 2 | | | | |
| 53 | 2 | | | | |
| 54 | 4 | | | | |

| Anewalt | | | |
|---|---|---|---|
| $\alpha$: 0.1 - 0.7 | | $\alpha$: 0.8 - 0.9 | |
| src-dest | capacity | src-dest | capacity |
| 01 | 1 | 14 | 1 |
| 02 | 1 | 35 | 1 |
| 03 | 1 | | |
| 04 | 1 | | |
| 05 | 3 | | |
| 10 | 1 | | |
| 12 | 1 | | |
| 13 | 1 | | |
| 14 | 1 | | |
| 20 | 1 | | |
| 21 | 1 | | |
| 23 | 2 | | |
| 24 | 2 | | |
| 25 | 3 | | |
| 30 | 1 | | |
| 31 | 2 | | |
| 32 | 3 | | |
| 35 | 2 | | |
| 40 | 1 | | |
| 42 | 2 | | |
| 43 | 3 | | |
| 45 | 1 | | |
| 50 | 3 | | |
| 52 | 2 | | |
| 53 | 1 | | |
| 54 | 3 | | |

**Table F.2**: The resulting VPDBA solutions for each algorithm using the test cases in Table F.1.

# Bibliography

[1] S. AHN, R. TSANG, S. TONG, AND D. DU. Virtual path layout design on atm networks. In *IEEE Infocom*, volume 1, pages 192–199, 1994.

[2] N. ANEROUSIS. Managing virtual circuit and virtual path services on atm networks with quality of service guarantees. Ph.D. Thesis, Columbia University, May 1996.

[3] N. ANEROUSIS AND A. LAZAR. Virtual path control for atm networks with call level quality of service guarantees. In *IEEE Infocom*, pages 312–319, 1996.

[4] N. ANEROUSIS AND A. LAZAR. Virtual path control for atm networks with call level quality of service guarantees. *IEEE Transactions in Networking*, April 1998.

[5] K. ANEWALT. Virtual path survey title. Technical Report, The College of William and Mary, Nov 1998.

[6] T. AOYAMA, I. TOKIZAWA, AND K. SATO. Atm vp-based broadband networks for multimedia services. *IEEE Communications Magazine*, pages 30–39, April 1993.

[7] A. ARVIDSSON. Management of reconfigurable virtual path networks. In *Proceedings of the 14th International Teletraffic Congress (ITC)*, pages 931–940, 1994.

[8] A. ARVIDSSON. Real time management of virtual paths. In *IEEE GLOBECOM '94*, volume 3, pages 1399–1404, 1994.

[9] M. BALL AND A. VAKHUTINSKY. Fault-tolerant virtual path layout in atm networks. *INFORMS Journal on Computing*, 13:76–94, 2001.

[10] L. BECCHETTI AND C. GAIBISSO. Lower bounds for the virtual path layout problem in atm networks. In *SOPFSEM '97: Theory and Practice of Informatics*, 1997.

[11] C. BRUNI, P. D'ANDREA, U. MOCCI, AND C. SCOGLIO. Optimal capacity management of virtual paths in atm networks. In *IEEE GLOBECOM '94*, volume 1, pages 207–211, 1994.

[12] R. BUBENIK, M. GADDIS, AND J. DEHART. Communicating with virtual paths and virtual channels. In *IEEE Infocom '92*, volume 3, pages 1035–1042, 1992.

[13] J. BURGIN. Dynamic capacity management in the bisdn. *International Journal of Digital and Analog Communication Systems*, 3:161–165, 1990.

[14] J. BURGIN AND D. DORMAN. Broadband isdn resource management: The role of virtual paths. *IEEE Communications Magazine*, pages 44–48, Sept. 1991.

[15] J. CHAN AND D. TSANG. Bandwidth allocation of multiple qos classes in atm environment. In *IEEE Infocom '94*, volume 1, pages 360–367, 1994.

[16] K.-T. CHENG AND Y.-S. LIN. On the joint virtual path assignment and virtual circuit routing problem in atm networks. In *GLOBECOM '94*, pages 777–782, 1994.

[17] I. CHLAMTAC, A. FARAGO, AND T. ZHANG. How to establish and utilize virtual paths in atm networks. In *Conference Record of the International Conference on Communications (ICC)*, pages 1268–1372, 1993.

[18] I. CHLAMTAC, A. FARAGO, AND T. ZHANG. Optimizing the system of virtual paths. *IEEE/ACM Transactions on Networking*, 2(6):581–585, 1994.

[19] R. COHEN AND A. SEGALL. Connection management and routing in atm networks. In *IEEE Infocom '94*, volume 1, pages 184–191, 1994.

[20] J. DAIGLE. *Queueing Theory for Telecommunications*. Addison-Welsey Publishing Company, Reading, Massachusetts, 1992.

[21] G. DEVECIANA AND G. KESIDIS. An approach to provisioning for real-time vbr video teleconferencing via end-to-end virtual path connections. In *IEEE International Conference on Communications (ICC)*, volume 2, pages 632–636, 1996.

[22] E. DUTKIEWICZ AND G. ANIDO. Connection admission control in atm networks. In *Proceedings of the ITC Specialists' Seminar on Telecommunication Services for Developing Economies*, volume 1, pages 305–316, 1991.

[23] A. FARAGO, I. CHLAMTAC, AND S. BASAGNI. Virtual path network topology optimization using random graphs. In *IEEE Infocom '99*, volume 2, pages 491–496, 1999.

[24] C. FREI AND B. FALTINGS. Abstraction and constraint satisfaction techniques for planning bandwidth allocation. In *IEEE Infocom '00*, volume 1, pages 235–244, 2000.

[25] V. FROST AND M. MULLEN. Dynamic bandwidth allocation for b-isdn based on end-to-end delay estimates. In *Proceedings of the International Conference on Communications (ICC)*, volume 1, pages 225–231, 1992.

[26] A. GARIGLIO, A. TONIETTI, U. MOCCI, AND C. SCOGLIO. Optimal dimensioning of virtual paths in atm networks. In *Proceedings of the European Conference on Networks and Optical Communications (NOC '96)*, pages 99–105, 1996.

[27] A. GERSHT AND A. SHULMAN. Optimal dynamic virtual path bandwidth allocation and restoration in atm networks. In *IEEE GLOBECOM 1994*, volume 2, pages 770–776, 1994.

[28] O. GERSTEL, I. CIDON, AND S. ZAKS. The layout of virtual paths in atm networks. *IEEE/ACM Transactions on Networking*, 4(6):873–884, 1996.

[29] O. GERSTEL AND A. SEGALL. Dynamic maintenance of the virtual path layout. In *IEEE Infocom '95*, volume 1, pages 330–337, 1995.

[30] S. GUPTA, K. ROSS, AND M. ZARKI. Routing in virtual path based atm networks. In *IEEE GLOBECOM '92*, volume 1, pages 571–575, 1992.

[31] I. HABIB AND T. SAADAWI. Dynamic bandwidth allocation and access control of virtual paths in atm broadband networks. In *High Performance Networking IV*, pages 49–64, 1993.

[32] H. HADAMA, R. KAWAMURA, T.IZAKI, AND I.TOKIZAWA. Direct virtual path configuration in large-scale atm networks. In *IEEE Infocom '94*, volume 1, pages 201–207, 1994.

[33] C. HOU. Routing virtual circuits with timing requirements in virtual path based atm networks. In *IEEE Infocom '96*, volume 1, pages 320–328, 1996.

[34] J. HUI, M. GURSORY, N. MOAYERI, AND R. YATES. A layered broadband switching architecture with physical or virtual path configurations. *IEEE Journal on Selected Areas in Communications*, 9(9):1416–1426, Dec. 1991.

[35] R. HWANG, J. KUROSE, AND D. TOWSLEY. Mdp routing in atm networks using virtual path concept. In *IEEE Infocom '94*, volume 3, pages 1509–1517, 1994.

[36] J. HYMAN, A. LAZAR, AND G. PACIFICI. Modeling vc, vp and vn bandwidth assignment strategies in broadband networks. In *Proceedings of the 4th International Workshop on System Support for Digital Audio and Video*, pages 102–113, Nov 1993.

[37] H. JI AND J. HUI. Statistical resource allocation and pricing in broadband communication networks. In *Broadband Communications: Proceedings of the International IFIP-IEEE Conference on Broadband Communications*, pages 247–258, 1996.

[38] R. KAWAMURA, K.I. SATO, AND I. TOKIZAWA. Self-healing atm networks based on virtual path concept. *IEEE Journal on Selected Areas in Communications*, 12(1):120–127, Jan 1994.

[39] M. KAZUTAKA AND H. KIM. Near–optimal virtual path routing for survivable atm networks. In *IEEE Infocom '94*, volume 1, pages 208–215, 1994.

[40] S. KIM. An optimal establishment of virtual path connections for atm networks. In *IEEE Infocom '95*, volume 1, pages 72–79, 1995.

[41] K. KRISHNAN AND R. CARDWELL. Routing and virtual-path design in atm networks. In *IEEE GLOBECOM '94*, volume 2, pages 765–769, 1994.

[42] A. LAZAR, A. ORDA, AND D. PENDARAKIS. Virtual path bandwidth allocation in multi-user networks. In *IEEE Infocom '95*, volume 1, pages 312–320, 1995.

[43] H. LEVY, T. MENDELSON, AND G. GOREN. Optimal use of virtual paths for connection setup reduction: The single link problem. In *IEEE Infocom '00*, 2000.

[44] Y. LIN, W. SU, AND C. LO. Virtual path management in atm networks. In *IEEE International Conference on Communications*, volume 2, pages 642–646, 1996.

[45] K. LIU, D. PETR, V. FROST, H. ZHU, C. BRAUN, AND W. EDWARDS. Design and analysis of a bandwidth management framework for atm-based broadband isdn. *IEEE Communications Magazine*, pages 138–145, May 1997.

[46] M. LOGOTHETIS AND S. SHIODA. Centralized virtual path bandwidth allocation scheme for atm networks. *IEICE Transactions on Communications*, E75-B(10):1071–1080, Oct. 1992.

[47] K. MASE AND S. SHIODA. Real-time network management for atm networks. *Queueing, Performance and Control in ATM (ITC-13)*, pages 129–136, 1991.

[48] T. NOH, D. VAMAN, AND X. GU. Reconfiguration for service and self-healing in atm networks based on virtual paths. In *Proceedings of the 12th International Conference on Computer Communication (ICCC)*, pages 525–530, 1995.

[49] H. OBARA, M. SASAGAWA, AND I. TOKIZAWA. An atm cross-connect system for broadband transport networks based on virtual path concept. In *IEEE International Conference on Communications (ICC)*, pages 839–843, 1990.

[50] S. OHTA AND K.I. SATO. Dynamic bandwidth control of the virtual path in an asynchronous transfer mode network. *IEEE Transactions on Communications*, 40(7):1239–1247, July 1992.

[51] A. ORDA, G. PACIFICI, AND D. PENDARAKIS. An adaptive virtual path allocation policy for broadband networks. In *IEEE Infocom*, volume 1, pages 329–336, 1996.

[52] A. PITSILLIDES, G. STYLIANOU, C. PATTICHIS, AND A. SEKERCIOGLUAND A. VESILAKOS. Bandwidth allocation for virtual baths (bavp): Investigation of performance of classical constained and genetic algorithm based optimisation techniques. In *IEEE Infocom '00*, pages 1501–1510, 2000.

[53] Z. QIN, T. LO, AND F. WU. The optimal virtual path design in atm networks. In *22nd Conference on Local Computer Networks*, pages 159–167, 1997.

[54] L. REISS AND L. MERAKOS. Shaping of virtual path traffic for atm b-isdn. In *IEEE Infocom '93*, volume 1, pages 168–175, 1993.

[55] S. RHEE AND T. KONSTANTOPOULOS. A decentralized model for virtual path capacity allocation. In *IEEE Infocom '99*, volume 2, pages 491–504, 1999.

[56] K.I. SATO, S. OHTA, AND I. TOKIZAWA. Broad-band atm network architecture based on virtual paths. *IEEE Transactions on Communications*, 38(8):1212–1222, Aug. 1990.

[57] Y. SATO AND K.I. SATO. Virtual path and link capacity design for atm networks. *IEEE Journal on Selected Areas in Communications*, 9(1):104–111, Jan 1991.

[58] Y. SATO, N. YAMANAKA, AND K.I. SATO. Atm network resource management techniques for cbr virtual paths/channels. *IEICE Transactions on Communications*, E79-B(5):684-692, May 1996.

[59] Y. SATO, N. YAMANAKA, K.I. SATO, AND N. TOKURA. Experimental atm transport system and virtual path management techniques. In *IEEE GLOBECOM '91*, volume 3, pages 2110-2116, 1991.

[60] A. SETHI. A model for virtual tree bandwidth allocation in atm networks. In *IEEE Infocom '95*, pages 1222-1229, 1995.

[61] S. SHIODA AND H. UOSE. Virtual path bandwidth control method for atm networks: Successive modification method. *IEICE Transactions*, E74(12):4061-4068, Dec. 1991.

[62] R. SIEBENHAAR. Optimized atm virtual path bandwidth management under fairness constraints. In *IEEE GLOBECOM '94*, volume 2, pages 924-928, 1994.

[63] R. SIEBENHAAR. Multiservice call blocking approximations for virtual path based atm networks with cbr and vbr traffic. In *IEEE Infocom '95*, volume 1, pages 321-329, 1995.

[64] R. SLOSIAR AND D. LATIN. A polynomial-time algorithm for the establishment of primary and alternate paths in atm networks. In *IEEE Infocom '00*, pages 509-518, 2000.

[65] A. TANENBAUM. *Computer Networks*. Prentice-Hall, Inc., 1989.

[66] M. VEERARAGHAVAN, M. KSHIRSAGAR, AND G. CHOUDHURY. Concurrent atm connection setup reducing need for vp provisioning. In *IEEE Infocom '96*, volume 1, pages 303-311, 1996.

[67] R. VEITCH, D. SMITH, AND I. HAWKER. Design and performance issues of protection virtual path networks. In *Broadband Communications '96*, pages 537-548, 1996.

[68] E. WONG, A. CHAN, S. CHAN, AND K. KO. Bandwidth allocation and routing in virtual path based atm networks. In *IEEE International Conference on Communications*, volume 2, pages 647-652, 1996.

[69] J.P. WONG AND R.K. PANKAJ. A diameter based method for virtual path layout in atm networks. In *Broadband Communications '96*, pages 501-512, 1996.

[70] S. ZAKS. Path layout in atm networks. In *SOFSEM '97: Theory and Practice of Informatics*, pages 144-160, 1997.

[71] L. ZHANG. Statistics of cell loss and its application for forward error recovery in atm network. In *Conference Record of the International Conference on Communications*, pages 694-698, June 1992.

# VITA

## Karen Marie Anewalt

Karen Marie Anewalt was born in Pittsburgh, Pennsylvania on June 19, 1972. She graduated from Shaler High School in Pittsburgh, Pennsylvania in 1990. Ms. Anewalt attended Allegheny College in Meadville, Pennsylvania, where she received the Bachelor of Science degree in Mathematics with a minor in Computer Science in 1994. Later that year, she matriculated in the Computer Science program at The College of William and Mary in Williamsburg, Virginia as a Teaching Assistant. In 1996, she received the Master of Science degree in Computer Science from The College of William and Mary, and entered the doctoral program at the same institution. The author holds the rank of Assistant Professor in the Department of Computer Science at Mary Washington College in Fredericksburg, Virginia.