DYNAMIC SUBFRAME ALLOCATION IN A RESERVATION MULTIPLE ACCESS CHANNEL

by

PETER D. ROORDA

B.A.Sc. The University of Waterloo, 1991

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER'S OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

(Department of Electrical Engineering)

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

July 1993

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

(Signature) _____

Department of Electrical Engineering

The University of British Columbia
Vancouver, Canada

Date July 21, 1993

DE-6 (2/88)

Abstract

Multiaccess protocols allow multiple geographically diverse users to communicate with a central station or each other over a single broadcast channel. Reservation protocols are a class of multiaccess protocols appropriate for satellite and radio networks. This thesis investigates dynamic time slot access control schemes for reservation multiaccess protocols similar to those investigated by Roberts [1] and others. Previous work has considered this model only for the case of a static frame subdivision, the majority of the work relying on approximate methods for analysis. In this thesis, schemes are considered where the subframe allocation is controlled on a frame by frame basis based on the current state of the finite user population.

The controlled system is modelled as a Markov decision process with the subframe allocation as the decision variable. Optimality equations are provided and using infinite horizon dynamic programming techniques, the control policy that maximizes the average throughput on the channel based on complete state information is found. In addition to the optimal policy, we propose two heuristic dynamic allocation schemes that while being suboptimal, have implementation advantages over the optimal. Performance results indicate that the implementation of dynamic subframe allocation into the model significantly improves the performance of the network over previously considered fixed allocation schemes, both in terms of lowering the average message transmission delay and increasing the channel capacity. Proposed are methods of estimating the unknown state using a variation of Rivest's [2] pseudo-Bayesian scheme to allow approximation of the dynamic control schemes which depend on full state information. Simulation results indicate that the implementation of state estimation does not significantly erode the performance of the dynamic policies. Extending the model to incorporate multipacket messages is discussed and performance results indicate that in that case also, protocol performance may be improved using dynamic allocation.

Contents

List of Tables

List of Figures

Acknowledgment

I would like to thank my supervisor, Dr. Victor Leung, for his valuable guidance and suggestions in both my research and in the preparation of this document.

I also wish to thank the National Science and Engineering Research Council of Canada for the funding that made this work possible.

Chapter 1 INTRODUCTION

1.1 Motivation and Objectives

The research presented in this thesis considers the idea of dynamically allocating channel resources to the reservation and data subchannels based on system state feedback in a reservation multiple access protocol. While a great deal of research has been done on reservation protocols, the work has primarily focussed on systems where the subchannel allocation is fixed or where dynamic allocation is based on heuristic algorithms. No one has investigated the idea of fully using the available system state information to optimally partition the channel dynamically.

We consider a simple reservation model originally proposed by Roberts [1], altering it to allow a frame-by-frame allocation partition control. Without such control, useful data slots potentially lie idle, causing decreased bandwidth efficiency. By incorporating this type of control, the system may adaptively allocate channel resources according to the state of the system to increase the system throughput. It is obviously desirable to optimize with respect to system throughput the dynamic allocation strategy. This desire motivates us to develop a fairly simple reservation model that allows exact mathematical analysis and optimization. While the theoretical appeal of such an optimal result is indeed significant, it is also important to bear in mind the implementation complexity of any proposed system. For this reason we consider not only optimal allocation control, but also heuristic suboptimal strategies which achieve near optimal performance while holding advantages for real system implementation.

In this thesis, it is necessary to make simplifications in the reservation model to allow exact numerical analysis of the system performance. One of these assumptions is that the system state is known by all users. Since this is not necessarily the case in a real network, it is necessary to develop a channel state estimation technique to allow implementation of

proposed state-dependant control schemes. Finally, it is of interest to extend the simple single-packet message model to one where messages are of variable length. This extension of the model is appealing in that it incorporates a more general family of multiaccess networks.

Specifically our objectives are as follows:

1. to determine, for a fairly simple reservation model, the optimal subframe allocation policy based on system state feedback which maximizes the system throughput,
2. to compare the performance of the optimal policy performance to previously considered fixed allocation schemes,
3. to investigate and propose heuristic policies that, while suboptimal, show performance close to optimal and have implementation advantages over the optimal,
4. to develop a simple and effective state estimator that is necessary for state dependant allocation schemes,
5. to investigate the extension of these results to multipacket message models.

1.2 Background

1.2.1 Multiaccess protocols

Multiaccess protocols for packet communication systems enable many users at diverse locations to efficiently share a common broadcast channel for communicating with a central station or with each other by packet transmissions. The designer of such protocols strives to achieve a channel access discipline that achieves communication that is both reliable and bandwidth efficient. While the fixed assignment techniques of time division multiplexing and frequency division multiplexing are well suited protocols for systems where user traffic is continuous, these schemes become extremely inefficient when user traffic is bursty and low rate as is often the case for data traffic. In this case, we desire more flexible protocols where idle users do not use bandwidth needlessly and users with data to send may transmit without undue delay. Many such protocols have been implemented and many more proposed

in the communications literature. The wide variety in protocols reflects the differences in the physical characteristics of the communications channel being shared, the nature of the traffic and the characteristics of the user population.

One common characteristic in all multiaccess protocols is that in each there is an amount of channel bandwidth used to coordinate station access in the flexible assignment scheme. Thus there is a basic split in the channel, between control overhead and useful data transfer. In simple contention based random access systems the control overhead is represented by the idle and collision times between useful data packets. For token rings and busses, the token passing time is the control overhead. For reservation multiple access protocols, the control overhead is the reservation subchannel used for reservation request transfer. In all of these cases, the efficiency of the protocol is dictated by how small the control overhead can be kept relative to the amount of useful data exchange. For the most part, the great body of work in the literature on multiaccess protocols is composed of the invention and analysis of heuristic protocols that control channel access with a minimum of overhead. Finding *the* optimal policy has been elusive and research into optimal protocols has lagged behind the more successful heuristic proposals. In this thesis, we consider the optimization of performance for a specific protocol within a class of multiaccess protocols referred to as reservation protocols, where perhaps the channel split between control overhead and data transmission is made most explicitly.

1.2.2 Reservation multiaccess protocols

Within a reservation multiaccess scheme, a terminal with data to send transmits a reservation request on the shared channel to request future channel time to send its data. This request is received (by the scheduling station in a centrally controlled system or by all users in a distributed-control system) and used to allocate channel time to the requesting user based on the protocol's assignment discipline. Thus, the broadcast channel is essentially split into two subchannels – a reservation subchannel for reservation request traffic and a

data subchannel for transmitting data messages on reserved channel time. The channel split may be achieved by either time-division [1], [3], [4], [5], [6] or frequency division [7], [8]. Requests for reservations may be sent explicitly as small request packets [1], [6], [9] or be implied by the successful transmission of an initial data packet [3], [4], [8]. Finally, in the reservation subchannel, protocols may use fixed assignment [9] or random access [1], [4], [8] as a discipline for request packet transmission.

In this thesis, a protocol is considered that is very similar to that first proposed by Roberts [1]. The channel is time-divided into the reservation subchannel and data subchannel. Reservation requests are sent explicitly in a S-ALOHA fashion via minislots in the reservation subchannel. Successful reservation requests join a global queue and those stations in the queue have channel time allocated to them in a first come first served basis in the data subchannel. It is well known that such a protocol is very efficient for systems where the number of users is large, the round trip signal propagation time is long compared to the packet length and the users' message lengths are long in comparison with the reservation request packets. As such, reservation protocols of this type are most appropriate for VSAT (very small aperture terminal) satellite networks, mobile satellite networks or packet radio networks.

Previous analyses of protocols using minislots for reservation have considered systems where the partition between the reservation and data subframes is fixed [1], [5], [10]. This brings about an obvious shortcoming in the protocol. By having a fixed length data subframe, slots may be wasted when there are no stations waiting to send. This inefficient partitioning of the channel prompts us to investigate how better to allocate frame slots, to allow the subframe resource allocation to be more sensitive to the current state of the system. Thus, in this thesis, we consider the system where the subframe allocation is dynamically set according to feedback information on a frame-by-frame basis. It is shown in this thesis that the use of dynamic allocation in the protocol can produce significantly higher throughputs and lower

average delays than the previously considered fixed schemes for packet transmissions in medium and high traffic situations.

While use of such dynamic allocation in reservation models is not entirely new, ours is the first attempt to optimize this allocation for the model under consideration on the basis of full state feedback. Rubin [6] proposed a minislot reservation protocol that automatically adjusts allocation based on a portion of the state information and some traffic estimates, but he abandons the fixed frame size and makes no attempt to optimize the allocation. Protocols that use first packet transmission for reserving future slots proposed in [3], [4], [8] implicitly use partial state information to dynamically allocate channel time but their approach is heuristic in nature, again with no attempt at optimization. The success of the heuristic dynamic allocation schemes in these similar protocols further motivates us to not only extend the concept of dynamic allocation to the minislot reservation model, but to optimize that allocation using Markov decision process techniques.

1.2.3 Markov decision process theory

Szpankowski [10] uses a detailed Markovian analysis of Roberts [1] model to determine system performance with a fixed subframe allocation. By reformulating the model from a simple Markov process with fixed allocation to a Markov decision process with dynamic subframe allocation, the Markovian formulation becomes a tool for design rather than for simple analysis. Markov decision process theory includes a well developed group of techniques for deriving optimal decision policies in Markovian systems. Recent advances in the field address infinite horizon models with undiscounted rewards, a class that includes most applications in the field of communications.

By formulating the reservation model as a Markov decision process, we are able to analyze different dynamic policies explicitly, find the optimal allocation strategy using dynamic programming techniques and, by Markovian analysis, determine the optimal performance

under such a strategy. While novel to this application, this technique has been applied successfully to other communication problems in areas of random access control [11],[12], queueing control [13], [14], and flow control [15], [16].

1.2.4 State estimation

While we assume that full state information is available when deriving the optimal allocation policy, it must be recognized that the system state is only partially observable by the stations. Specifically there is a need to estimate the number of stations contending on the reservation subframe based on the history of observations on that frame. This problem is encountered for controlling the S-ALOHA random access broadcast channel. Segall [17] considers the general problem of recursive state estimation for a partially observed Markov process. His approach, though optimal, proves to be unwieldy and fairly impractical. Other researchers [2], [18], [19] propose simpler techniques to get a good estimate, their performance summarized and compared in [20]. Rivest's [2] pseudo-Bayesian scheme emerges as an especially simple and effective scheme for estimation and lends itself quite handily to be adapted to our model. Thus in this thesis, a useful state estimation technique is developed that bears close resemblance to that proposed by Rivest [2].

1.3 Structure of the Thesis

The thesis is organized as follows. Chapter 2 provides a detailed description of the reservation model and formulates this model as a Markov decision process. The model bears close resemblance to that proposed by Roberts [1] but with the added capability of allowing dynamic allocation. The assumptions made in the model are discussed in detail. From this model, the states, actions, rewards and transition probabilities that uniquely characterize the Markov decision process are defined. In the Markov formulation, the specific formulas necessary for determining the throughput and average delay in the system when a control policy is specified are derived.

Chapter 3 presents the optimality equations and the dynamic programming algorithm to determine the optimal allocation scheme for a given set of system parameters based on the Markov decision process formulation of chapter 2. Furthermore, two simpler heuristic schemes are proposed which are much more attractive for implementation while giving performance close to optimal. Exact numerical results for all the policies are calculated using the formulas presented in chapter 2 and are presented for comparison among the dynamic schemes and with previously considered fixed schemes. Results indicate that the use of dynamic allocation policies improves performance dramatically over the fixed allocation policies that have been previously considered. System capacity under different allocation schemes is examined and numerical results indicate consistently higher capacities for the dynamic policies than fixed allocation. The advantages and disadvantages of the different policies in terms of performance and implementation issues are discussed.

Chapter 4 addresses the problem of state estimation, deriving the expressions for the optimal true Bayesian estimator and an approximate pseudo-Bayesian estimation algorithm appropriate for our model. The latter is much more attractive for real-time implementation and simulation results presented in the chapter indicate that the use of the pseudo-Bayesian estimator does not significantly undermine the performance of the dynamic schemes.

In chapter 5, we consider variations of the reservation model that handle multipacket messages. The extension of the results for the single packet message model to the multipacket model are discussed and some simulation results are presented that indicate that dynamic allocation schemes improve performance in this case also.

Finally, chapter 6 concludes the thesis, highlighting the major points and proposing further avenues of research.

Chapter 2 NETWORK MODEL AND MARKOV DECISION PROCESS FORMULATION

In this chapter the network model that is used for analysis is described in depth and the model is formulated as a Markov decision process. The model chosen is fairly archetypical of reservation models with minislot reservations. The chief difference is that we allow dynamic subchannel allocation. Since most reservation protocols split the channel into a subchannel for securing the reservation and a subchannel for servicing reservation, the idea of dynamic allocation of channel resources would appear to be extendable to other reservation models and it is likely that improvements similar to those that are revealed in the reservation model that we consider would be possible in other reservation protocols when dynamic allocation is implemented. We restrict our attention in this work to one simple model. For exact analysis and to enable true optimization we are forced to make some simplifying assumptions about the model to allow the Markovian formulation. The assumptions are discussed in detail in this chapter.

2.1 The Basic Reservation Model

We consider a reservation model with a finite number M users, each of these users having a message source and a single message buffer. These stations have no means for intercommunication outside of the single broadcast channel on which all stations transmit and receive. Channel time is divided into frames of length F slots, each slot being equivalent to the length of one packet. We shall assume that all stations are synchronized such that packet transmissions start at the beginning of a slot and occupy exactly one slot. Frames are subdivided into a reservation subframe and a data subframe. The reservation subframe is composed of L slots, each of which is further subdivided into V minislots, each minislot

being the length of a reservation request packet. Thus, there are $F - L$ data slots in the data subframe. (See Figure 1.)

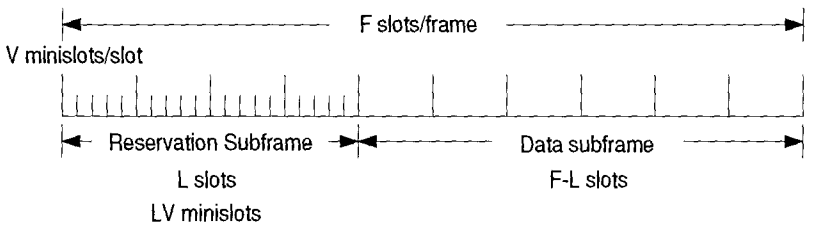

Figure 1. The frame structure of the reservation protocol

The *LV* reservation slots in the reservation subframe are accessed in slotted ALOHA mode. Once a station successfully sends a reservation request packet in a reservation minislot, the message joins a global queue and gets slots reserved in upcoming data subframes. (This global queue has been referred to in the literature as the "queue in the sky" [10] since it does not exist in a single location, but rather is maintained by distributed control of all stations. All stations monitor the number of packets in the queue based on observations on the channel, and track their own place in the queue.) We initially focus our attention on a situation where all messages are composed of a single packet. (In chapter 5 we consider the multipacket message model.) Thus each station is in one of three modes. When it has no message, the station is in idle mode. When a new message arrives the station goes into contention mode. When the station has successfully sent a reservation request on the reservation subchannel, it enters reservation mode. Finally, when the station reaches the head of the global queue and the packet is sent in the reserved data slots, the station goes back into idle mode to wait until a new message arrives.

In Roberts' protocol [1], it is assumed that the channel time allocated to the reservation subframe, L, is fixed for all frames. If we allow L to become a control variable, set according

to the state of the system, it is proposed that gains can be made in the performance of the protocol.

The following are the key assumptions made in the model:

1. There are a M users with identical access procedures and traffic characteristics.
2. Each slot may be divided into V minislots, each of which can hold 1 reservation request packet.
3. Each idle user generates a message in a frame with probability p, referred to as the arrival probability. Until the message is completely sent, the station is considered to be blocked. (ie. All newly arriving packets to that station are lost.)
4. When a new message arrives, the station does not send a reservation request until the following frame. Similarly, when a reservation request is successfully made, the message may join the global queue only at the beginning of the next frame. Finally when a station completes a message transmission, it remains blocked until the beginning of the next frame. (ie. All station states are updated at the start of each frame.)
5. When a station is in contention mode, it will send one reservation request in one of the LV minislots (chosen at random) of the current frame with transmission probability p_s. The station repeats this until the message is sent successfully at which point the station enters the global queue.
6. The channel is error free except for collisions, and all stations are perfectly synchronized.

2.2 Delay Modelling

An important characteristic of any broadcast network model is how the round trip delay is modelled. In local area networks (LAN's) the delay is typically very short with respect to the message size. For packet radio networks the delay is much greater. For satellite networks, this round trip propagation delay is in the order of a quarter second which typically spans the length of a large number of data packets. This range in delay characteristics makes it

difficult to create a general network model for consideration. Reservation protocols demand a fairly high level of complexity for individual stations to coordinate access and maintain synchronization. For this reason, the use of such protocols seem to be best suited for packet radio and satellite networks. We discuss in this section the issues involved in delay modelling for these applications.

Archival literature on the subject of reservation protocols [1], [10], [21] have largely avoided true delay modelling for analysis of the network model. True delay modelling involves the addition of "wait" states to the straightforward "contention", "reservation" and "idle" states introduced earlier. This addition of states quickly increases the dimensionality of the problem and thus computational barriers become insurmountable very quickly. In later chapters it becomes apparent that such an increase in dimensionality will render exact numerical analysis inachievable. Some researchers have attempted the high road, doing analysis with these added "wait" states, but have been forced to rely on approximate methods to avoid numerical overload [5]. [5] produces a Markovian model with more than $2G/F + 5$ distinct state variables where G is the round trip delay measured in slots. Most analyses have, on the other hand, either assumed that the propagation delay is shorter than the data subframe, thus does not interfere with the state structure [10] or else discounted the presence of delay completely [1], [21]. This simplification keeps the number of state variables low, allowing exact Markovian analysis.

In the model as we have defined it with assumption 4, we assume that the delay is small enough that all stations have the information from the reservation subframe before the beginning of the next frame. There is no real need for stations to monitor the data subframe since by assumption 6 we may rely on successful transmissions and correct distributed maintenance of the global queue. Thus, for a system where control is distributed among the stations or orchestrated by satellite on-board processing, assumption 4 demands that the

longest round trip delay in the system G, must be less than the length of the data subframe (ie. $G < F - L$). (For a network controlled centrally by a hub earth station, the feedback delay would be twice the round trip propagation delay.) Considering that the allocation L may change dynamically from 0 to F, this effectively means that for our model to be strictly accurate, G=0. While this may indicate that our modelling of delay is not completely true, past work indicates that the simplification is not a serious one. Kleinrock and Lam [21] show that adjusting the retransmission probability of a S-ALOHA system such that the average backoff includes the true delay, the results are close to those produced by the computationally expensive true modelling approach. Furthermore, Lim and Meerov's [22] investigations show that in a contention channel, the introduction of delay into the system improves stability and throughput of the system. Thus, our assumption of small or no delay may act as a worst case scenario. These results give credence to the notion that valid performance results may be determined without strictly accurate delay modelling.

2.3 Markov Decision Process Formulation

In the literature, analysis of reservation multiaccess protocols has taken a number of different forms. In many of the cases, the reservation and data subchannels are treated independently with the only stipulation being that their average flows are balanced. While this approach may be valid for a fixed allocation, in the more general model where the subframe allocation may change from frame to frame, it is difficult to avoid handling the two subsystems together. We thus handle the system as a whole, formulating and analyzing it as a two state Markov decision process. From this formulation, we can determine the stationary state probabilities and thus the system performance measures, throughput and average message delay.

A Markov decision process [23] differs from a Markov process in two respects. Firstly, while in a Markov process the probability of being in the current state is a function only

of the previous state, in a Markov decision process the current state of the system is a function of not only the previous state, but also of the "decision" or "action" at that previous state. Secondly, in a Markov decision process there is associated with each state a "reward". The Markov decision process is a useful formulation in that using this formulation we may find the decision policy which maximizes the expected reward of the process. For our reservation model, this translates to finding the allocation policy which maximizes the expected throughput for the network. Furthermore, since the decision policy is a function of the system state, specifying the policy transforms the Markov decision process to a simple Markov process, whose throughput is the expected reward. We now proceed with such a formulation, defining the states, actions, rewards and transition probabilities that uniquely define the Markov decision process.

2.3.1 System state

We let the system state be defined by the two dimensional vector, $S_k = \left(N_1^k, N_2^k\right) \in \mathbf{S}$, where ${N_1}^k$ and ${N_2}^k$ define the number of stations in contention mode and in the global queue respectively at the beginning of frame k. The state space $\mathbf{S}$, is defined by

$$\begin{aligned} N_1^k &\in \{0, ..., M\} \quad , \\ N_2^k &\in \left\{0, ..., M - N_1^k\right\} \quad . \end{aligned} \tag{1}$$

The magnitude of the state space is found to be

$$|\mathbf{S}| = \frac{(M+2)(M+1)}{2} \tag{2}$$

2.3.2 Actions

At the start of frame k, the stations decide on the size of the allocation for the reservation subchannel L_k, based on the current state information. Thus the allocation policy will be defined by

$$L_k = L_k(S_k) \quad . \tag{3}$$

Since we are interested only in stationary policies, we may drop the subscript to give a time invariant policy

$$L(S_k) \in \mathbf{A} \equiv \{0, ..., F\} \quad . \tag{4}$$

Thus the policy L, is simply a mapping from the state space onto the set of allowable actions as defined by $\mathbf{A}$.

2.3.3 Rewards

We would like to find the allocation policy to maximize system throughput. Thus we chose the reward r to be the expected number of successful packets sent in the frame given the state and action variables. Since this expectation is the lesser of the length of the global queue and the data slots in the frame,

$$r_L(S_k) = \min\left(N_2^k, F - L(S_k)\right) \quad . \tag{5}$$

Thus $\bar{r}_L$ is a vector indicating the reward for each state for the action policy, L.

2.3.4 Dynamic equations

Before deriving the transition probabilities, let us define the dynamic equations of the system that dictate the system state transitions. For ease of notation, let

$$\begin{aligned} n_1 \equiv N_1^k \quad , \qquad n_2 \equiv N_2^k \quad , \\ m_1 \equiv N_1^{k+1} \quad , \quad m_2 \equiv N_2^{k+1} \quad , \quad a \equiv L(S_k) \quad . \end{aligned} \tag{6}$$

Further, we let X^k be the number of new messages generated in frame k, Y^k be the number of successful reservation requests sent in frame k, and W^k be the number of message transmissions that are successfully completed in frame k. The role of these variables is seen in Figure 2 which illustrates how stations move through the queueing system.

We may now write the set of dynamic equations as follows:

$$\begin{aligned} m_1 &= n_1 + X^k - Y^k \quad , \\ m_2 &= n_2 + Y^k - W^k \quad . \end{aligned} \tag{7}$$

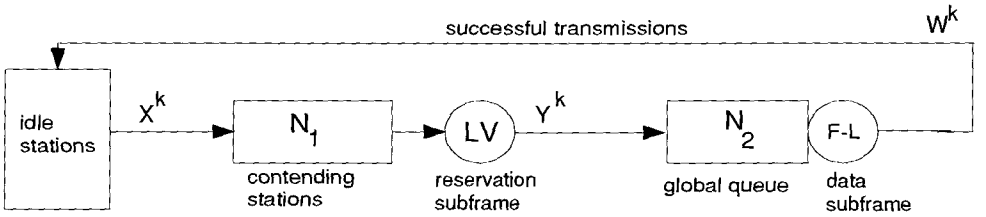

Figure 2. Queuing control representation of the reservation model

2.3.5 State transition probabilities

The derivation of the transition probabilities is somewhat more complex but is simplified by assumptions 3–5. In a Markov decision process, the current state is a function of the previous state and action. Thus a state transition matrix, P^L has elements,

$$P^L_{i,j} = Pr\{S_{k+1} = j \mid S_k = i, L(i)\} \ . \tag{8}$$

The probability distribution of Y^k given that n requests are sent derived in [10] is

$$\begin{aligned} & Pr\Big\{Y^k = i \mid n, a\Big\} \\ & = f(i,n,a) \\ & = \frac{(-1)^i (aV)! n!}{(aV)^n i!} \times \sum_{j=i}^{\min\{aV,n\}} \frac{(-1)^j (aV-j)^{n-j}}{(j-i)!(aV-j)!(n-j)!} \ . \end{aligned} \tag{9}$$

Since the probability that n requests are sent is binomially distributed, we may write the probability of exactly Y^k success given that n_1 stations are in contention mode to be

$$Pr\Big\{Y^k = i \mid n_1, n_2, a\Big\} = \sum_{n=0}^{n_1} \binom{n_1}{n} p_s^n (1-p_s)^{n_1-n} f(i,n,a) \ . \tag{10}$$

By assumption 3, X^k is also distributed binomially,

$$Pr\Big\{X^k = x \mid n_1, n_2, a\Big\} = \binom{M-n_1-n_2}{x} p^x (1-p)^{M-n_1-n_2-x}. \tag{11}$$

Finally,

$$W^k = \min(n_2, F - a) \ . \tag{12}$$

Using these expressions, we may now state the state transition probabilities as

$$\begin{aligned} & p(m_1, m_2 \mid n_1, n_2, a) \\ & = Pr\Big\{ Y^k = m_2 - n_2 + \min(n_2, F - a) \mid n_1, n_2, a \Big\} \times \\ & Pr\Big\{ X^k = m_1 - n_1 + m_2 - n_2 + \min(n_2, F - a) \mid n_1, n_2, a \Big\}. \end{aligned} \tag{13}$$

Since the probability of being in the next state (m_1, m_2) is a function only of the current state (n_1, n_2) and action, a, we can say immediately that the transition probabilities do in fact satisfy the Markovian property, and thus the process is a Markov decision process. Furthermore, the transition probabilities are independent of the frame index k and thus are stationary. We note here again that when a decision policy is specified, the action is a simple function of the current state and these transition probabilities define a unique Markov process.

2.4 Solving for the System Throughput and Delay

2.4.1 Deriving the throughput of the Markov process

Average channel throughput is commonly used as a measure of the effectiveness of a multiaccess protocol. For our purposes, we will consider the throughput as the long run average number of successful packet transmissions per frame, given the initial state S,

$$\eta^L(S) = \lim_{T \to \infty} \frac{1}{T} E\left[\sum_{k=1}^{T} r_L(S_k) \mid S_0 = S \right] \ . \tag{14}$$

All non-degenerate policies result in Markov processes with a single ergodic class (see Appendix A), in which case the throughput is independent of the initial state,

$$\eta^L(S) = \eta^L \ . \tag{15}$$

To find this throughput we must first determine the limiting distribution of the stationary Markov chain.

The most common method to find the limiting distribution is to find the equivalent stationary distribution of the system. A distribution

$$\Pi^* = \{\pi_1^*, \pi_2^*, ...\} \tag{16}$$

is stationary if

$$\begin{aligned} \pi_i^* &\geq 0 \ , \\ \sum_j \pi_j^* &= 1 \ , \\ \text{and} \quad \pi_i^* &= \sum_j \pi_j^* \, P\{S_{k+1} = i \mid S_k = j\} \ . \end{aligned} \tag{17}$$

If the Markov process contains a single recurrent class [24], the stationary distribution is unique and is equivalent to the limiting distribution,

$$\pi_i^* = \lim_{k \to \infty} P\{S_k = i\} \ . \tag{18}$$

Determining the stationary distribution is simply a matter of solving the linear system given by eqns. (17).

We may now find the throughput by finding the expected throughput for each state (the state dependant reward) and summing over the limiting distribution, or more simply the vector product,

$$\eta^L = \Pi^* \, \bar{r}_L \ . \tag{19}$$

2.4.2 Expected packet delay

The expected packet delay for the system is the sum of the reservation request packet delay D_1, and the data packet delay D_2, both in units of frame durations. If we determine the stationary state expectations as follows,

$$\begin{aligned} E[N_1] &= \sum_{n_1=0}^{M} \left\{ n_1 \sum_{n_2=0}^{M-n_1} \pi^*_{(n_1,n_2)} \right\} \ , \\ \text{and} \quad E[N_2] &= \sum_{n_2=0}^{M} \left\{ n_2 \sum_{n_1=0}^{M-n_2} \pi^*_{(n_1,n_2)} \right\} \ , \end{aligned} \tag{20}$$

we may use Little's formula to determine the delays to be

$$D_1 = \frac{F\ E[N_1]}{\eta^L} \quad \text{and} \quad D_2 = \frac{F\ E[N_2]}{\eta^L} \quad . \tag{21}$$

Thus we may now write the overall expected message delay to be

$$D = F\left(\frac{E[N_1] + E[N_2]}{\eta^L}\right) \quad . \tag{22}$$

According to model assumption 4 there is no need to calculate residual delays. By loosening the basic simplifying assumptions of our model such residual delays would be introduced.

2.4.3 Summary of steps to find throughput and delay

The following summarizes the steps taken to determine the throughput and delay for a given stationary control policy $L(N_1, N_2)$.

1. Calculate the transition matrix with eqn. (13).
2. Use the transition matrix to determine the limiting distribution by solving the system of eqns. (17).
3. Find the long run average throughput by incorporating the limiting distribution and state dependant throughputs using eqn. (19).
4. Solve for the expected packet delay using eqns. (20) and (22).

Chapter 3 SUBFRAME ALLOCATION SCHEMES

In the previous chapter we developed the specific tools to determine the throughput and delay performance for the network operating under a decision policy. In this chapter we examine specific policies and compare the performance of the model under these policies. Previous research has focussed chiefly upon fixed allocation schemes. We determine the optimal dynamic policy and two heuristic dynamic schemes which improve on the performance of the fixed schemes. Throughout this section, we continue to assume the ideal situation where the state information is completely known at each decision point. We will loosen this assumption in the following chapter and incorporate state estimation into the system.

3.1 Fixed Allocation

Previous research into the reservation model ([1], [10], [5]) considers only the case where the size of the reservation subframe is fixed with no regard for the system state (ie. $L(N_1, N_2) = L_f$). Szpankowski [10] proposes for his very similar model a simple method for choosing the best fixed allocation. Here we use the same method, applying it to our model. If we assume that throughput on the contention channel is $1/e$ reservation requests per slot, we may write the flow balance equation to be

$$\frac{V L_f}{e} = F - L_f \ . \tag{23}$$

This gives us an optimal fixed allocation of

$$L_f = \frac{F}{1 + V/e} \ . \tag{24}$$

We use the lower and upper integer values of this result as the two best fixed allocations. By choosing the lower of these values, we create a situation where there is a bottleneck in the reservation subchannel. The higher of these values creates a bottleneck in the data subchannel.

For the case where F=5, V=3 and M=20, L_f is 2.377, and the two best fixed allocations schemes are at L_f=2 and L_f=3. For these allocations, we have set the transmission probability p_s to 1.0 and 0.6 respectively. In Figures 3 and 4, we plot for these two fixed allocation schemes the throughput against the arrival probability and the average delay against the throughput as calculated using the procedure summarized in section 2.4.3. In the case of L_f=2 we can see that the bottleneck in the reservation subframe has caused decreased throughput when traffic is high, a phenomenon that is well explored in [10]. The transmission probability p_s for that case was reduced from 1.0 to 0.6 to allow good performance on a fairly large range of traffic levels while keeping the average delay reasonably low. We have chosen a model with small frame and population size here to ease the numerical computations. In following sections we resort to simulation methods to investigate larger population models and compare dynamic and fixed subframe allocation schemes.

3.2 Optimal Allocation

3.2.1 Optimality equations

We wish to optimize the allocation policy with respect to the system throughput. In the model that we have defined, this is equivalent to minimizing the average packet delay. In this section we define the optimality equations used to determine the optimal policy. Finding the optimal allocation policy corresponds to finding a state-dependant policy L^* such that

$$\eta^{L^*}(S) = \max_{L \in \Psi} \eta^L(S) = \eta^*(S) \quad , \tag{25}$$

where Ψ is the set of all stationary, non-randomized policies that are dependant only on the current state. For a finite state and action Markov decision process with average reward criteria and infinite horizon, it has been show [25] that there exists a policy that is optimal within this set. A stationary policy L is said to be unichain if its corresponding Markov chain contains only one recurrent class and possibly a number of transient states. In this case

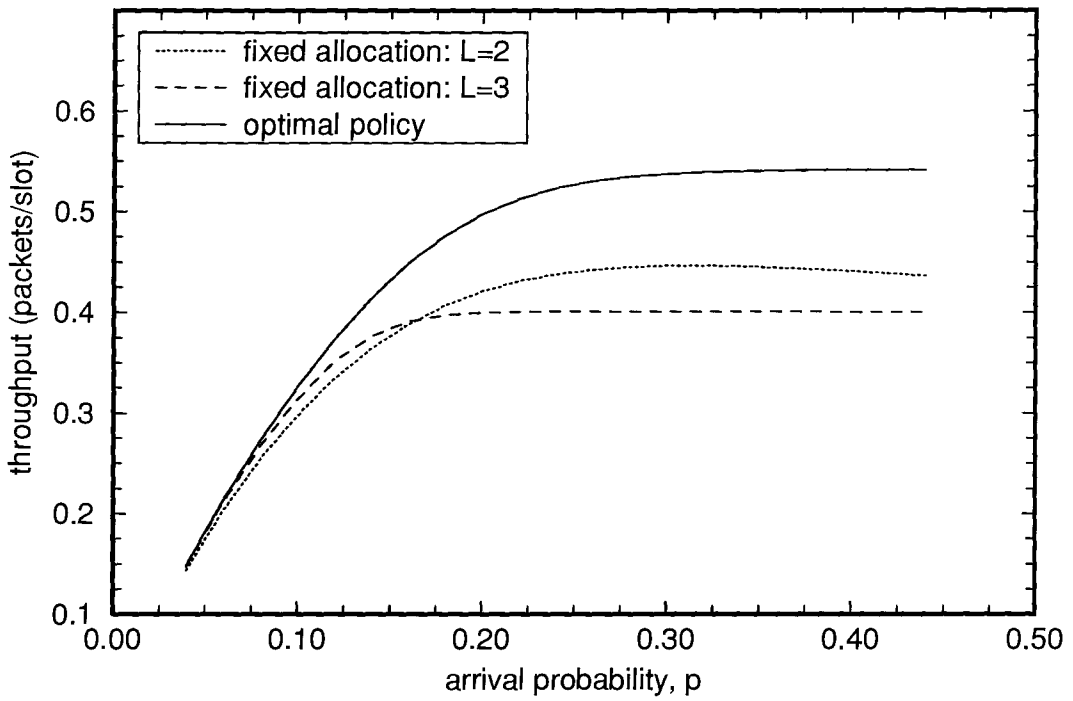

Figure 3. Throughput vs. arrival probability for the parameter set (M=20, F=5, V=3) : fixed allocation policies and optimal policy

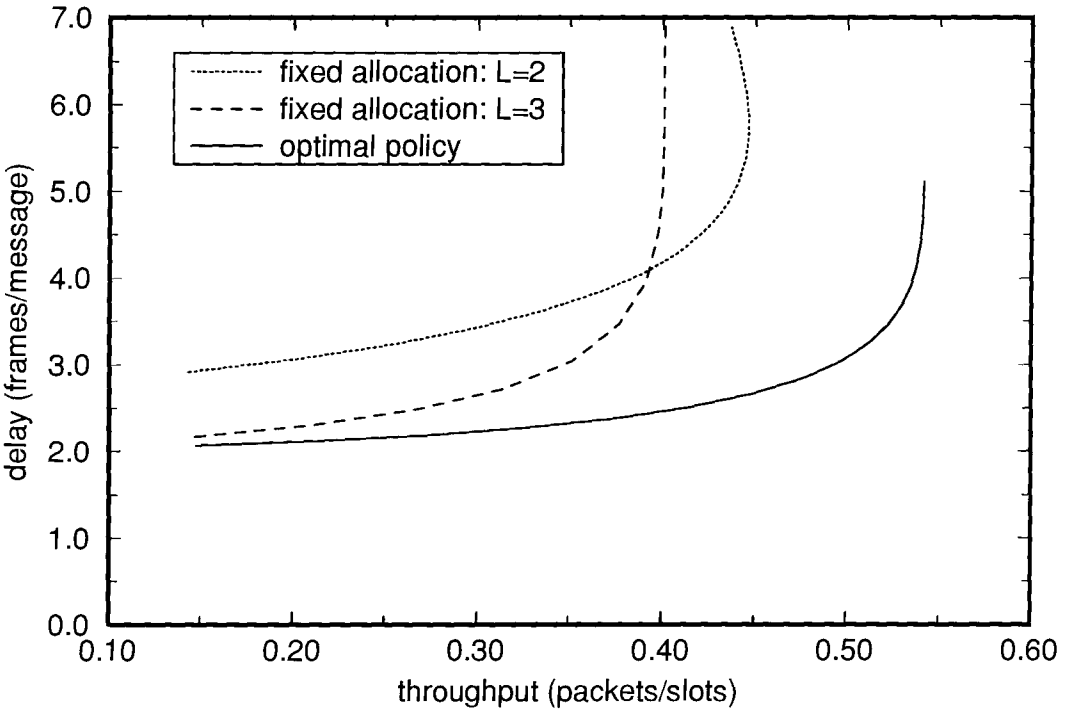

Figure 4. Average delay vs. throughput for the parameter set (M=20, F=5, V=3) : fixed allocation policies and optimal policy

$\eta^L(S)$ is independent of the state and is thus a constant [23]. When all stationary policies are unichain, η^* satisfies the set of optimality equations,

$$\bar{v} = \max_L \left\{ \bar{r}_L - \eta^* + P^L \bar{v} \right\} \quad . \tag{26}$$

Recall that P^L is the policy dependant transition probability matrix whose elements are defined by eqn. (13). These optimality equations are derived and fully explained in [23] and are the equations that uniquely characterize the optimal policy, L^*, and its corresponding optimal expected average reward, η^*. The decision rule L which achieves the maximum average reward when a $\bar{v}$ that solves the equation is inserted is the optimal policy [23]. $\bar{v}$ is referred to in the literature as the bias or the relative value function and is defined by eqn. (26) up to an additive constant. It is called the relative value function because its elements, $v(S)$; $S \in \mathbf{S}$, indicate the relative value in terms of total reward of being in one state versus another. That is, $v(i) - v(j)$, is the difference in the total reward when a system starts in state i versus a system that starts in state j. Since there are $|\mathbf{S}| + 1$ unknowns in eqn. (26) (η^* and $v(i)$, $i = 1, \ldots, |\mathbf{S}|$) for $|\mathbf{S}|$ equations, $\bar{v}$ is solvable only up to an additive constant, thus the values of the relative value function only have relevance with respect to one another.

Looking now at the optimality equations we see that the expression on the right is the sum of the immediate reward relative to the average reward for that frame, $\bar{r}_L - \eta^*$, and the expected relative value for the next frame,

$$P^L \bar{v} \, , \tag{27}$$

when the policy L is used. This gives us an intuitive sense of what the optimality equations represent. The details of the rigorous derivation of (26) are quite complex and the reader is referred to such texts as [23], [25] and [26] for complete descriptions.

For this optimality equation to hold for our model it must be shown that all policies are unichain. This is demonstrated in Appendix A.

3.2.2 Solving for the optimal policy with dynamic programming

Using these optimality equations we would like to find both the optimal allocation policy L^*, and the system throughput, η^*. There exists a number of methods to solve these equations. Both the policy iteration algorithm [27] and the modified policy iteration algorithm [23] for average reward criterion are presented here. The former is the simpler of the two but the latter is preferable for its reduced computational load and its attractive convergence characteristics. We provide here the specific algorithms and discuss the issues of applying these algorithms to solving the network control problem.

3.2.2.1 The policy iteration algorithm

Policy iteration is an efficient procedure for finding average reward optimal policies in Markov decision processes (MDP's). It is guaranteed to converge for MDP's with a finite number of states and actions when all policies are unichain. These conditions hold for our network control application. The algorithm was first proposed by [27] and consists of four basic steps. First an initial policy is chosen arbitrarily as the starting point. Second, the policy is evaluated by determining the throughput for that policy. Thirdly an improved policy is found. Finally, a stopping criteria is used to determine whether to repeat the second and third step to continue to improve the policy. It is guaranteed that at each improvement step, an equal or better policy is found [23] so the last step stops the iterations if the policy has not changed. The algorithm is:

1. (Initialization) Set n=0 and choose an initial decision rule $L_n \in \Psi$.
2. (Policy evaluation) Obtain the average reward of the policy, η_n and relative value vector, $\bar{v}_n$ by solving

$$0 = \bar{r}_{L_n} - \eta_n + \left(P^{L_n} - I\right)\bar{v}_n \quad . \tag{28}$$

Note that $\bar{v}_n$, is uniquely solvable up to an additive constant.

3. (Policy improvement) Choose L_{n+1} to satisfy

$$\bar{r}_{L_{n+1}} + P^{L_{n+1}}\bar{v}_n = \max_{L\in\Psi}\left\{\bar{r}_L + P^L\bar{v}_n\right\} \tag{29}$$

setting $L_{n+1}(S) = L_n(S)$ for each S if possible. Here we have used the relative value function, updated in step 2, to find an improved policy.

4. (Loop) If $L_{n+1} = L_n$, stop and set $L^* = L_n$. Otherwise increment n by 1 and return to step 2.

This algorithm produces a series of decision rules and their respective throughputs and is guaranteed to converge to solution in a finite number of steps [23]. The algorithm has been shown to be equivalent to Newton's method for finding the minimum of a function [24]. An initial policy is arbitrarily chosen and by policy improvement the policy converges toward the optimal.

The relative value function $\bar{v}_n$ is determined by solving eqn. (28) and is unique up to an additive constant. For convenience, we have set $v_n(S_0) = 0$, where S_0, is equivalent to the empty system state ($(N_1, N_2) = (0,0)$). This setting is commonly made and does not effect the result of the algorithm.

Note that the solution of eqn. (28) involves solving a system of $|\mathbf{S}|$ linear equations where $|\mathbf{S}|$ is defined by eqn. (2). Since $|\mathbf{S}|$ is of the order M^2, the complexity of solving the system of equations grows with M^6. Since in finding the optimal decision rule many iterations are potentially needed, this evaluation step becomes very costly. The modified policy iteration was developed to reduce this computational load by avoiding the explicit solution of the $|\mathbf{S}| \times |\mathbf{S}|$ system while still guaranteeing convergence when all policies are unichain.

3.2.2.2 The modified policy iteration algorithm

The modified policy iteration algorithm breaks down the policy evaluation step of regular policy iteration to reduce the computational expense of solving the $|\mathbf{S}| \times |\mathbf{S}|$ system. Similarly

to the policy iteration algorithm, there is a policy improvement step and an evaluation step, but this time, the evaluation is done iteratively over m steps rather than exactly. We avoid solving explicitly the $|\mathbf{S}| \times |\mathbf{S}|$ system. Note that when m is infinite, the algorithm is equivalent to policy iteration since the exact policy evaluation is achieved. The algorithm is:

1. (Initialization) Select $\bar{v}_0$, specify ϵ>0 and set *n=0*.
2. (Policy Improvement) Choose L_{n+1} to satisfy

$$\bar{r}_{L_{n+1}} + P^{L_{n+1}}\bar{v}_n = \max_{L \in \Psi} \left\{ \bar{r}_L + P^L \bar{v}_n \right\} \tag{30}$$

   setting $L_{n+1}(S) = L_n(S)$ for each S if possible.
3. (Partial Policy Evaluation)
   a. Set *k=0* and

$$\bar{u}_n^0 = \bar{r}_{L_{n+1}} + P^{L_{n+1}}\bar{v}_n \tag{31}$$

   b. If $sp(\bar{u}_n^0 - \bar{v}_n) < \epsilon$ go to 4. Otherwise go to c.
   c. If *k=m*, go to e. Otherwise compute $\bar{u}_n^{k+1} = \bar{r}_{L_{n+1}} + P^{L_{n+1}}\bar{u}_n^k$.
   d. Increment k by 1 and return to c.
   e. Set $v_{n+1} = u_n^m$, increment n, and go to step 2.
4. (Final Policy Improvement) Choose L_ϵ to satisfy

$$\bar{r}_{L_\epsilon} + P^{L_\epsilon}\bar{v}_n = \max_{L \in \Psi} \left\{ \bar{r}_L + P^L \bar{v}_n \right\} \tag{32}$$

This algorithm is guaranteed to converge within a finite number of iterations to an ϵ-optimal decision rule (ie. $\eta_{L_\epsilon} > \eta^* - \epsilon$), where ϵ may be chosen to be arbitrarily small [23]. In this algorithm, step 3 uses an iterative method to approximate the true relative value

function, $\bar{v}_{n+1}$ with the iterative result, $\bar{u}_n^m$ produced by m iterations. The span semi-norm used here is defined by

$$sp(\bar{x}) = \max_i (x_i) - \min_i (x_i) \quad . \tag{33}$$

and is used for the stopping criterion in step 3b. The algorithm stops when the norm of approximate relative value for policy, L_{n+1} differs from that of policy L_n by less than ϵ. In our application, $m = 6$ appeared to work well for all the systems considered. For full details of the modified policy iteration algorithm, and the rationale behind its development, the reader is referred to [23].

Applying the modified policy iteration algorithm reduced the computational effort of solving for the optimal policy by up to 60–70% compared to policy iteration when all but the most trivial systems were analyzed. Despite this, the complexity still grows dramatically with M since the solution of the system remains to be found, though only partially at each step. Since there do not exist methods computationally more efficient than the modified policy iteration algorithm for solving Markov decision processes with large state spaces [23], such complexity must be endured or the search for the optimal abandoned.

3.2.3 Optimal performance

Here we present some performance results for the optimal policies in a small network model and compare them with performance results for fixed allocation policies. First, some mention should be made of how the retransmission probability p_s is chosen for our models. The literature on slotted ALOHA channels has given close consideration to the choice of this parameter and it has been shown in that case that when p_s is fixed, congestion among colliding packets renders the system unstable when the user population in infinite and bistable when the population is finite [21]. For a finite population model, the bistable character is revealed in the way the delay-throughput performance becomes two-valued at high throughput levels and throughput degrades as the traffic intensity continues to increase. [10] shows that the

bistable effect is also found in the reservation model when the subframe size is fixed (which is expected since the reservation subsystem alone is effectively a slotted ALOHA channel). The approximate methods used to expose this bistability, while seemingly effective when the subframe allocation is static, are not easily extendable to the dynamic allocation model that we are considering. Thus we leave detailed stability analysis beyond the scope of our work. We choose network models where the network parameters are such that we may set p_s=1.0 for the dynamic policies without serious performance degradation due to stability problems. While it may be more appropriate to choose p_s adaptively as a decreasing function of N_1 as a means to stabilize the reservation subchannel we will maintain a fixed retransmission probability for our analysis and leave consideration of an adaptive transmission probability to further research. For the fixed allocation schemes, we are at times forced to reduce p_s to avoid too serious a degradation in throughput due to congestion. The reader is referred to [10] for a full treatment of the bistable character of the reservation model when a fixed allocation policy is used.

The modified policy iteration was implemented with ϵ chosen small enough that further reduction did not affect the optimal policy. Table 1 shows a sampling of optimal policies for the parameter set (M=20, V=3, F=5, p_s=1.0). We may first note that the optimal policies change with the traffic level (as indicated by the arrival probability, p). Furthermore, the optimal policies are a function of both state variables, thus both state variables contain information useful for allocation control. For the case when complete state information is available, the average throughput and delays were calculated for the optimal policies. Figures 3 and 4 presented previously in section 3.1 plot the throughput against the message arrival probability and the delay versus throughput respectively. Since these curves are based on complete state information, the optimal results represent an upper bound on performance for the model.

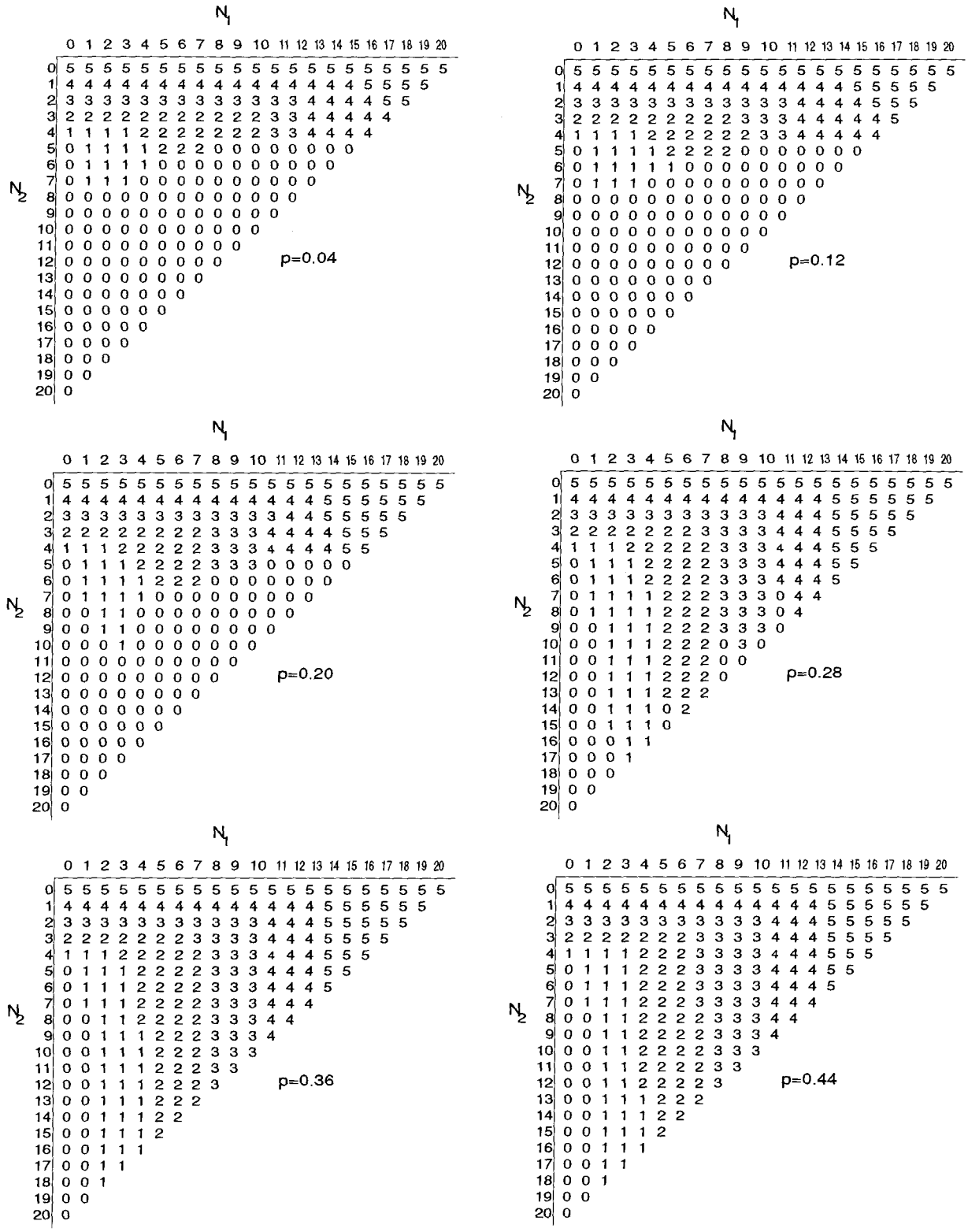

Table 1. Optimal policies for a range of arrival probabilities for the parameters (M=20, F=5, V=3)

We can see that at medium and high traffic levels, the improvement of the optimal over the fixed schemes is indeed dramatic, showing higher capacity in general, lower delays for the same throughput levels and higher throughputs for the same traffic levels. The best fixed policy capacity occurs for the allocation L=2 at capacity of near 0.45 packets per slot. By comparison, optimal allocation gives a capacity of approximately 0.54 packets per slot, an improvement of 20% over the fixed allocation. From Figure 4 we see that while for the fixed policies the average delay increases rapidly when throughput approaches capacity of 0.35–0.45, the delay of the optimal allocation remains fairly flat in that region, near the minimum of 2 frames per message, as system throughput extends to levels over 0.5 where the delay then increases quickly as throughput reaches capacity. By examining the specific policies, we see that the optimal policy achieves this result by adjusting the reservation subframe allocation such that the attempt rate is near the optimal of 1 request per minislot, ensuring sufficient reservation minislots to avoid excessive collisions in the reservation subframe.

While these optimal results look promising, there remain some key problems with achieving these results. First, the optimal policy for the network is a function of the traffic level as indicated by the arrival probability, p. In all real systems, we can expect the traffic levels to fluctuate over time. Adapting the optimal policies to this fluctuation would effectively involve changing policies. It would be preferable for the model to maintain good performance with a single policy for all traffic levels. Secondly, the optimal allocation at each frame is a function of both state variables. While the state of the global queue is known with certainty, the number of contending stations must be estimated by implementing an appropriate state estimator, increasing complexity in the transmitting/receiving stations. Finally, the complexity of the modified policy iteration algorithm prevents us from finding the optimal policies for networks with large population models. Many applications of reservation protocols involve such large populations and with existing computational means we may not hope to use these

analytical Markov techniques to solve for the optimal policy and performance when the dimension of the state space grows into the thousands and tens of thousands. The following sections propose two heuristic dynamic schemes that address these problems while maintaining performance close to the optimal. The state estimation problem is considered in chapter 4.

3.3 "Data Priority" (DP) Allocation

Rather than finding the policy that maximizes the long run average throughput as we have done in the previous section, let us consider a simple policy that sets the allocation to the value that maximizes the successful data packet transmissions in that frame. We shall refer to this policy as the "data priority" (DP) policy since as many slots as are needed to service the global queue (up to the obvious upper limit of F) are allocated to the data subframe, the remaining slots (if any) allocated to the reservation subframe. Thus,

$$L_{DP}(N_1, N_2) = \max(0, F - N_2) \ . \tag{34}$$

This policy has an advantage over the optimal in that it is only dependant on N_2, and thus no state estimation techniques are required for N_1 for implementing this allocation policy.

$L_{DP}(N_1, N_2)$

N_2 \ N_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
1	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	
2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3			
3	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2				
4	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1					
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0							
8	0	0	0	0	0	0	0	0	0	0	0	0	0								
9	0	0	0	0	0	0	0	0	0	0	0	0									
10	0	0	0	0	0	0	0	0	0	0	0										
11	0	0	0	0	0	0	0	0	0	0											
12	0	0	0	0	0	0	0	0	0												
13	0	0	0	0	0	0	0	0													
14	0	0	0	0	0	0	0	0													
15	0	0	0	0	0	0	0														
16	0	0	0	0	0	0															
17	0	0	0	0																	
18	0	0	0																		
19	0	0																			
20	0																				

Table 2. DP policy for the parameters (M=20, F=5, V=3)

Table 2 shows the DP policy values for the small model with parameter set (M=20, F=5, V=3). The performance results for the DP policy implemented in the small population model are plotted in Figures 5 and 6. As expected, we see that the DP policy shows a significant improvement over the fixed schemes while the throughput levels fall short of the optimal for medium and high traffic levels. The capacity of the DP policy is approximately 0.515, a 15% improvement over highest capacity for a fixed allocation scheme. This is less than the 20% improvement achieved by the optimal policy. However, the DP policy performance does show a slight degradation at high traffic levels. The throughput begins to decrease with the traffic level, a sign indicative of an increasing number of collisions in the reservation subchannel. Since the DP policy does not explicitly use information regarding the number of contending stations, this congestion problem is not unexpected.

Since the derivation of the DP policy is simple with respect to the dynamic programming techniques involved in finding the optimal policy, we are no longer restricted to small state space models for finding the policy. For a very large model though, deriving the throughput and average delay are costly if the analytical approach of last chapter is used. Solving the global balance equations is an operation of the order, $o\left(|S|^3\right) = o(M^6)$ and thus for a network of only 100 users, the number of floating point operations already runs into the hundreds of billions. To test the DP policy and compare it to the fixed policies for large models, we resort to simulation methods. Calculation of confidence levels for this simulation model cannot be done using the usual statistical methods since there is a correlation between successive delay and throughput values. We simply choose a high simulation duration. The relative smoothness of our performance curves indicates that the simulation lengths chosen are indeed sufficient for comparison among the different policies. Using the parameter set (M=150, F=12, V=12) we used discrete time simulation over a 20,000 frame duration to determine the throughput and delay for the model when the DP policy and the two best fixed allocation were used. Data were gathered over a range of traffic levels and the results appear

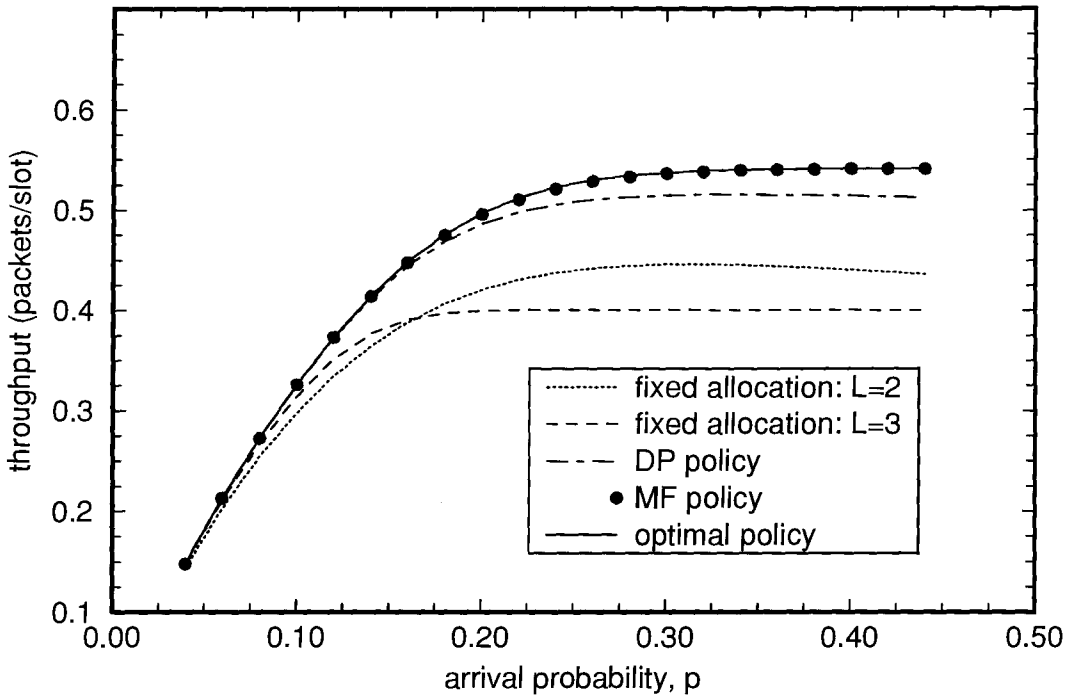

Figure 5. Throughput vs. arrival probability for the parameter set (M=20, F=5, V=3) : fixed allocation policy, optimal policy, MF policy and DP policy

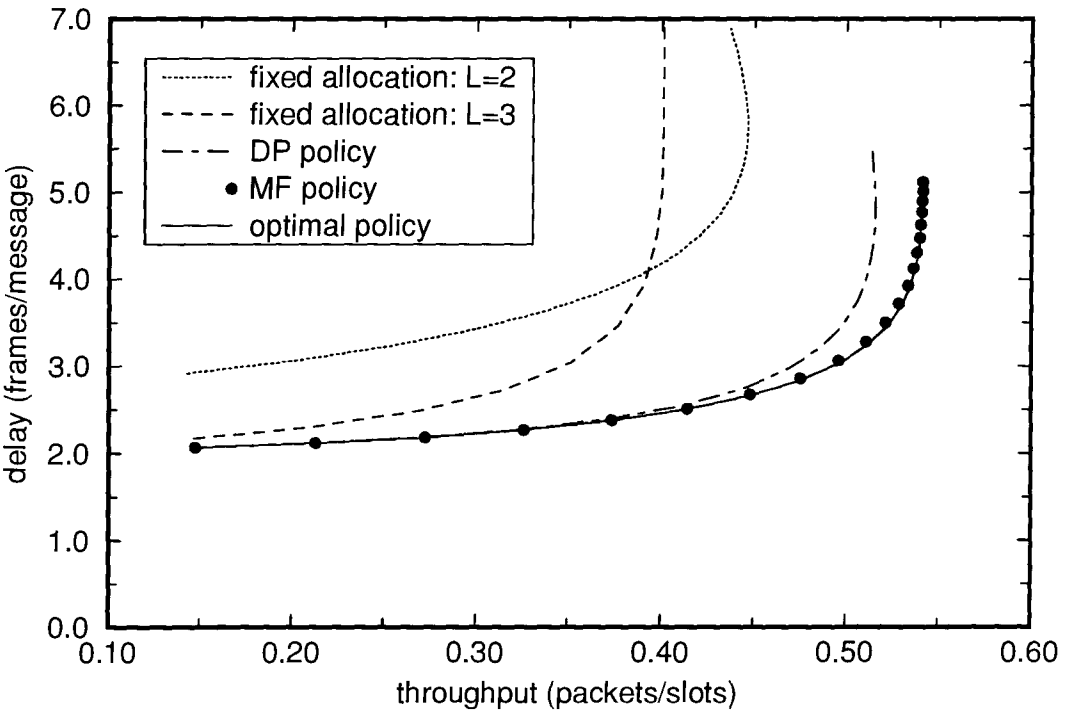

Figure 6. Average delay vs. throughput for the parameter set (M=20, F=5, V=3) : fixed allocation policy, optimal policy, MF policy and DP policy

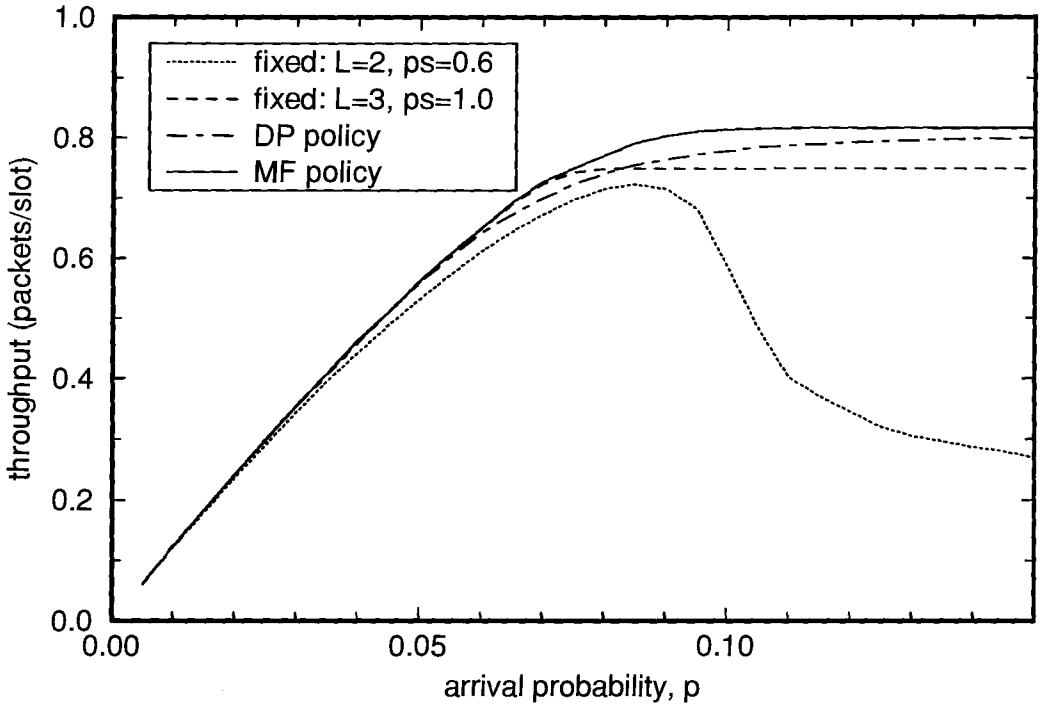

Figure 7. Throughput vs. arrival probability for the parameter set (M=150, F=12, V=12) : fixed allocation policy, MF policy and DP policy

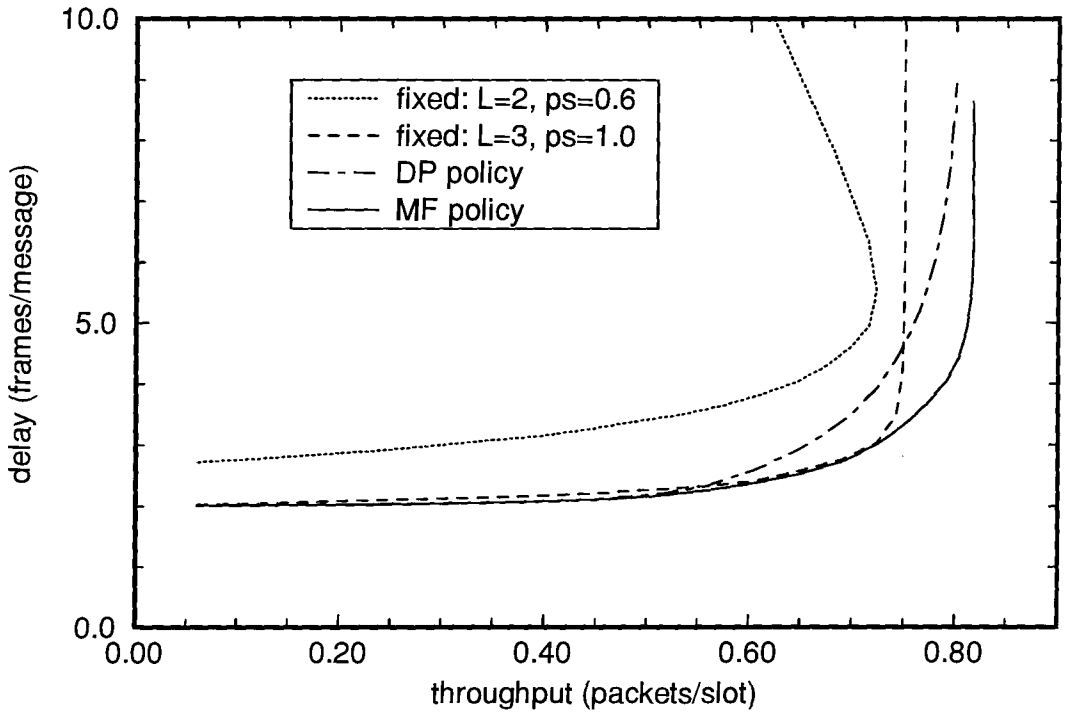

Figure 8. Average delay vs. throughput for the parameter set (M=150, F=12, V=12) : fixed allocation policy, MF policy and DP policy

in Figures 7 and 8 where throughput is plotted against the arrival probability and the delay is plotted versus the throughput respectively. It is apparent that in this larger model, the DP policy, while showing a capacity increase of 6.7% over the fixed policies, actually shows higher delays in a region of medium throughput levels. This indicates that while the DP policy avoids wasted data slots, the fact that the fixed policies balance the allocation between reservations and data does create better performance at some traffic levels. Note in Figure 8 that for the fixed allocation, L=2, there is a serious degradation in throughput as the traffic level increases. This is a result of high congestion in the reservation subsystem. This effect may be reduced by a further reduction of the retransmission probability from p=0.6 at the expense of higher delays. The dynamic polices do not appear to suffer from the destabilizing effect of congestion in this model.

3.4 "Maximum Flow" (MF) Allocation

We now consider another policy for subframe allocation. We see in the previous section that while the DP policy shows higher capacity than the fixed allocation schemes, it is still outdone by the optimal policy and shows congestion effects at high traffic levels. The optimal policy achieves this advantage by ensuring a balance between reservations and data packets while the DP policy simply maximizes the data packet transmissions, W^k.

In light of these observations we now develop a method to improve on the DP policy by better balancing the flow of traffic through the system while drastically simplifying the derivation from the modified policy iteration algorithm used for the optimal policy to a simple one step maximization. We propose a simple maximization of a new criterion, which we will refer to as the "flow". We define the "flow" in frame k as

$$F^k = \alpha Y^k + W^k \quad , \tag{35}$$

where α is a weighting coefficient. Since Y^k is a random variable, we must use the expectation, $E[F^k]$ as our criterion. The policy which maximizes this expectation will be referred to henceforth as the "maximum flow" (MF) policy.

From [5] we know that if n' stations each sends a request packet in one of LV minislots (chosen at random), the expected number of minislots containing a single transmission, Y, is

$$E[Y \mid n'] = n'\left(1 - \frac{1}{LV}\right)^{n'-1} . \tag{36}$$

Thus,

$$\begin{aligned} E[Y \mid N_1] &= \sum_{n'=0}^{N_1} \binom{N_1}{n'} p_s^{n'} (1-p_s)^{N_1-n'} n' \left(1 - \frac{1}{LV}\right)^{n'-1} \\ &= p_s N_1 \left(1 - \frac{p_s}{LV}\right)^{N_1-1} . \end{aligned} \tag{37}$$

Thus, from eqns. (35) and (37), we may write the expected flow as a function of the state, (N_1, N_2) and the reservation subframe allocation L,

$$E[F^k] = \alpha p_s N_1 \left(1 - \frac{p_s}{LV}\right)^{N_1-1} + \min(N_2, F - L) . \tag{38}$$

It is left now only to determine a good choice for α. By considering Figure 2, we see that α dictates the value of a successful reservation request transmission with respect to a data packet transmission. If we choose α=0, we give full priority to the data successes and thus the MF policy would become identical to the DP policy. Alternatively, if we choose a very high α, we give high priority to the reservation requests, allocating a high number of reservation slots for even a small number of contending stations. A desirable choice of α would lie between these extremes in an attempt to balance the flow equitably. It has been well established in the literature that the highest throughput on a S-ALOHA channel with an infinite user population is $1/e$. Thus, in a large population model, we may hope for an average V/e successful reservation requests per slot. This gives us an idea of the relative worth of a

successful request transmission and we set

$$\alpha = \frac{e}{V} \quad . \tag{39}$$

This proved to be a good choice for the weighting coefficient. Although a detailed sensitivity analysis was not done on this choice the close resemblance of the resulting MF policy to the optimal and their similar performance leads us to believe that for this policy derivation this weighting would appear to be the best or near the best.

We can now find the MF policy, L_{MF}, by maximizing the expected flow,

$$E\left[F^k \mid L_{MF}(n_1, n_2)\right] = \max_{L \in A} \left\{ E\left[F^k \mid L\right] \right\} \quad . \tag{40}$$

The calculations involved then in finding the MF policy are very manageable compared to modified policy iteration; there are no expensive iterative steps nor risk of non-convergence.

N_2 \ N_1

N_2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
1	4	4	4	4	4	4	4	4	4	4	4	4	4	4	5	5	5	5	5	5	
2	3	3	3	3	3	3	3	3	3	3	3	4	4	4	5	5	5	5	5		
3	2	2	2	2	2	2	2	2	3	3	3	4	4	4	5	5	5	5			
4	1	1	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5				
5	0	0	1	1	1	2	2	2	3	3	3	4	4	4	5	5					
6	0	0	1	1	1	2	2	2	3	3	3	4	4	4	5						
7	0	0	1	1	1	2	2	2	3	3	3	4	4	4							
8	0	0	1	1	1	2	2	2	3	3	3	4	4								
9	0	0	1	1	1	2	2	2	3	3	3	4									
10	0	0	1	1	1	2	2	2	3	3	3										
11	0	0	1	1	1	2	2	2	3	3											
12	0	0	1	1	1	2	2	2	3												
13	0	0	1	1	1	2	2	2													
14	0	0	1	1	1	2	2														
15	0	0	1	1	1	2															
16	0	0	1	1	1																
17	0	0	1	1																	
18	0	0	1																		
19	0	0																			
20	0																				

$L_{MF}(N_1, N_2)$

Table 3. MF policy for the parameters (M=20, F=5, V=3)

For the small model (parameters : M=20, F=5, V=3) the MF policy was calculated and is shown on table 3. The average throughput and message delay were calculated for this

policy for varying arrival probabilities and the delay-throughput curve and throughput-arrival probability curve are plotted in Figures 5 and 6 shown in the previous section with the other policies under consideration. Let us first note that the MF policy is independent of the traffic level as represented by the arrival probability, p. The policy appears to follow a structure and bears a very close resemblance to the optimal policy for medium and high traffic levels (see table 1). From Figures 5 and 6 we see that the performance of the MF policy virtually matches the optimal. Although it is not apparent from the plots, the MF policy throughput is in the order of 0.01% less than the optimal. Although the specific values of MF policies are quite different than the optimal policy values at low traffic levels, the throughput and delay are virtually the same since the system is generally in low occupancy states (ie. $N_1 + N_2 \ll M$) where the policies produce equivalent allocations. In these low occupancy states, the optimal and MF policies allocate the subframe very similarly to the DP policy. These numerical results for the small model indicate that the MF policy is a very good approximation to the optimal and gives better performance than the DP policy.

Again, a larger model (parameters : M=150, F=12, V=12) was analyzed under the MF policy and the simulation results are shown in Figures 7 and 8 (see previous section). The MF policy shows a higher throughput capacity than the DP policy and consistently lower delays for the same throughput. This is especially apparent in medium and high throughput levels. While the performance gain over the fixed allocation schemes is less dramatic for the large model than the small model, it is still substantial. For medium and low throughput levels, the MF policy has delays only negligibly different than the DP policy or the best fixed allocation policy. Again, it must be noted that these results are based on the unrealistic assumption that complete state information is available to all stations. The following chapter investigates state estimation implementation and shows that estimation has little effect on the performance results.

3.5 Capacity Analysis

We have demonstrated with both analytical and simulation results in previous sections that dynamic allocation policies in the reservation model have advantages over the fixed allocation schemes in terms of delay-throughput performance. In particular, the dynamic policies show dramatically higher capacity, especially in the small model. The performance results shown have been the result of rather cumbersome analytical or simulation algorithms that are time consuming and fairly complex. In this section, we develop a simple approximate method for predicting the capacity gain of using dynamic allocation versus fixed allocation. Results show that the capacity of the fixed schemes are highly sensitive to traffic and channel parameters and exhibits a granular effect that prevents them from achieving ideal capacity. Dynamic schemes, we will show, do not suffer from such an effect and their capacity is robust to changing system parameters.

For our approximate analysis, we consider an infinite user population model. Although this is a divergence from the model that we have used to this point, it allows us to get some general results and is a valid approximation when the population size is large with respect to the number of minislots in the frame. For an infinite population model, it is well known that unless some sort of input control is used, a S-ALOHA channel is unstable [28]. Thus for this capacity analysis, we will assume such control is incorporated. It is well known then that for a stabilized S-ALOHA channel, the throughput approaches $1/e$ as the number of contending stations grows without bound [28]. Using this assumption [28] points out the ideal capacity of a reservation multiaccess channel with S-ALOHA used for transmission of reservation requests. This ideal approximation assumes that at capacity, both the reservation subchannel and the data subchannel are operating at their individual capacities. These individual capacities are $1/e$ and 1 for reservation minislots and data slots respectively. For the transmission of a packet in a system operating at capacity then, the expected number of minislots used for

successful transmission of the reservation request is e. Summing the channel time used for request transmission with the single data slot for the packet transmission, the average number of slots used by the station for its packet transmission is

$$1 + \frac{e}{V} \quad . \tag{41}$$

Thus the ideal capacity (in packets/slot) is the reciprocal,

$$C = \frac{1}{1 + \frac{e}{V}} \quad . \tag{42}$$

This ideal capacity approximation assumes that the subframe allocation is such that the flow is balanced between the reservation and data subframes.

Indeed, the fixed allocation protocol achieves this capacity when the subframe partition, L_f is set to the ideal allocation which balances the flow through the system,

$$L' = \frac{F}{1 + \frac{V}{e}} \quad . \tag{43}$$

But, we are constrained in the system to set L_f to an integer value. As L_f deviates from the ideal, the system exhibits a bottleneck in the reservation subchannel if $L_f < L'$, and a bottleneck in the data subchannel if $L_f > L'$. The capacity of the channel is then

$$C(L_f) = \min\left(\frac{VL_f}{Fe}, \frac{F - L_f}{F}\right) \quad . \tag{44}$$

The maximum capacity for the fixed allocation schemes and the ideal capacity from eqn. (42) are plotted against the number of minislots per slot, V in Figures 9 and 10 for F=12 and F=5 respectively. Notice that for the fixed allocation, the capacity has a granular effect. When L_f may be chosen to satisfy the flow balance of eqn. (43) the fixed allocation capacity matches the ideal but between those points, the capacity drops blow the ideal in a step like function, each step representing a different fixed allocation. The steps result from the necessity in the model of choosing an integer allocation. The deviation of the fixed allocation capacity from

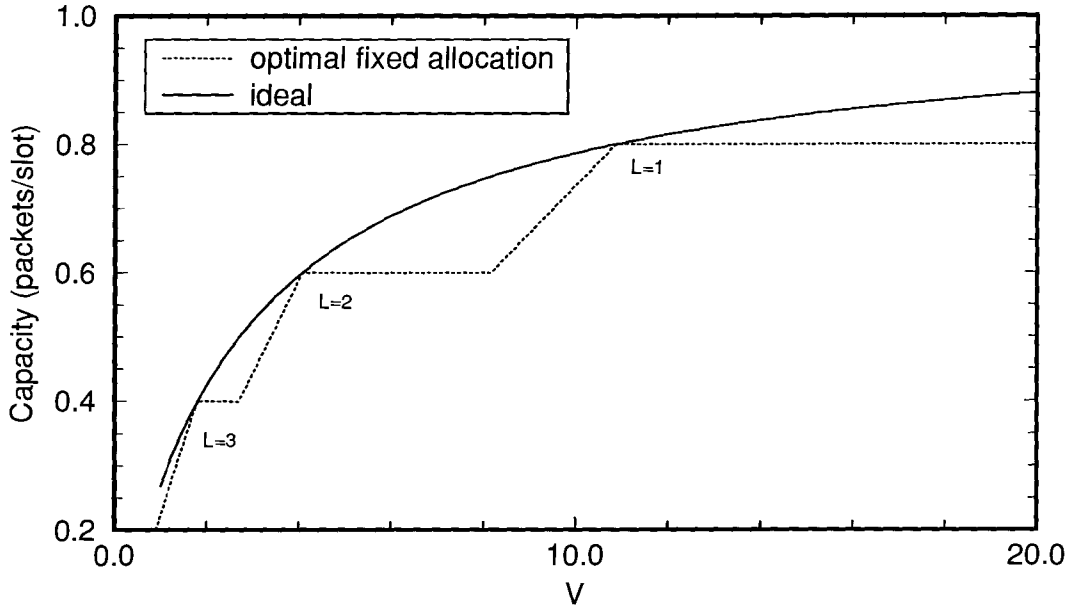

Figure 9. Approximate capacity plotted against the number of minislots per slot, V, for the ideal capacity and the best fixed allocation policies' capacity for the case F=5

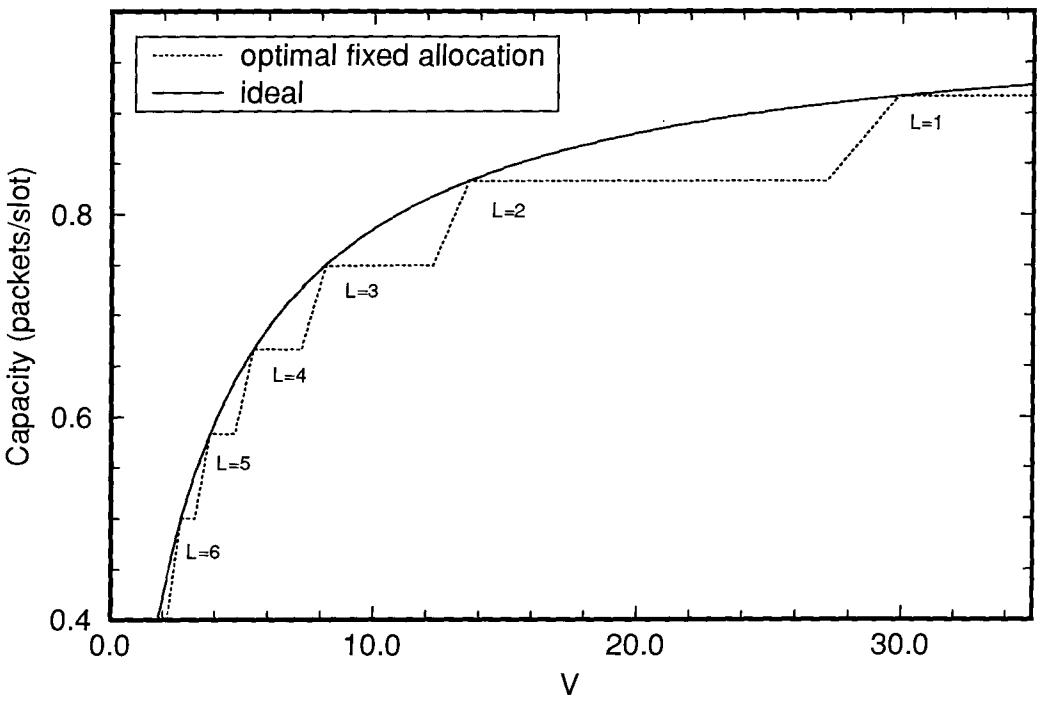

Figure 10. Approximate capacity plotted against the number of minislots per slot, V, for the ideal capacity and the best fixed allocation policies' capacity for the case F=5

the ideal is quite dramatic for certain values of V. For example, we see in Figure 9 that for V=8, the best fixed allocation is L_f=2, producing a capacity of 0.6. This is 19% lower than the ideal capacity of 0.74. Examining the figures, we see that the lower capacities resulting from this granular effect are most pronounced in the smaller frame size model and at lower values of V where the ideal capacity curve is the steepest.

By using a dynamic scheme, we may avoid this granular effect. Let us assume a dynamic allocation scheme, $L(n_1, n_2)$, where no data slots are left idle (ie. $L(n_1, n_2) \geq F - n_2$). For the MF, DP and optimal policies this criterion holds. At capacity, the number of messages in the system, $n_1 + n_2$, grows without bound. In the case of the DP policy, n_2 will decrease from one frame to the next if $n_2 \geq F$. It will increase when $n_2 < F$ by at most the number of minislots in the reservation subframe, $(F - n_2)V$. Therefore n_2 is strictly finite and $n_1 \rightarrow \infty$. Thus for the DP policy, the expected number of messages moving into the global queue by successful transmission of the reservation request on the reservation subchannel is

$$E[Y \mid n_1, n_2] = \frac{VL(n_1, n_2)}{e} \quad . \tag{45}$$

The expected number of complete messages sent over the data subchannel when no data slots are idle is

$$E[W \mid n_1, n_2] = F - L(n_1, n_2) \quad . \tag{46}$$

The steady state flow balance equation at capacity,

$$\sum_{n_1, n_2} P(n_1, n_2) E[Y \mid n_1, n_2] = \sum_{n_1, n_2} P(n_1, n_2) E[W \mid n_1, n_2] \tag{47}$$

reduces to

$$E[L] = \frac{F}{1 + \frac{V}{e}} \quad . \tag{48}$$

Thus the capacity of the DP allocation policy is

$$C = \frac{1}{1 + \frac{e}{V}} \quad , \tag{49}$$

the same as the ideal.

It is not as easy to find a general expression of the capacity of the optimal and MF policies since in these cases n_1 does not necessarily grow without bound at capacity. In general though we may say that for the optimal policy, the capacity will at worst match the DP capacity. Since according to eqns. (40) and (35), the MF policy will only favour allocating slots to the reservation subframe when the expected reservation request throughput is greater than *V/e* per slot, we can expect that for the MF policy,

$$E[Y \mid n_1, n_2] \geq \frac{VL(n_1, n_2)}{e} \tag{50}$$

at capacity. Thus for the optimal and MF policies, the capacity matches or exceed the ideal,

$$C \geq \frac{1}{1 + \frac{e}{V}} \ . \tag{51}$$

Examining the numerical results in the previous section, we would expect that these capacities would be very close to this bound. Certainly for high traffic levels, we would expect n_1 to grow large, the reservation request throughput to approach 1/*e*, and thus the capacity of the MF or optimal policies would approach the ideal value. To make a stronger claim would involve a closer examination of the Markov process underlying the system. This question involves the Markovian analysis of a countable state process, an avenue of investigation that may encounter technical barriers.

This capacity analysis uses a model different than our throughput and delay analysis in that we have assumed an infinite user population and implemented input control. These changes are necessary to provide a simple approximation. In the finite population model with no input control, the approximations are close provided some care is taken with the choice of p_s for all policies to prevent congestion from eroding the throughput on the channel. Generally, the retransmission probability, p_s, should be chosen as high as possible to minimize the request packet transmission delay, while being low enough to prevent too many collisions in the

reservation subframe from decreasing the throughput. In the model examined in this work, it has been found that p_s can be safely set to 1.0 except for the fixed allocation policies. This issue of stability for this model is explored for the fixed allocation policy in [5] and [10]. We leave detailed analysis of this issue for the dynamic policies to further research.

Chapter 4 SYSTEM STATE ESTIMATION

It has been proposed in the opening chapters that by dynamic partitioning of the reservation subframe based on the system state, we may improve the performance of the reservation multiaccess protocol. We do not of course have full state information. While the number of stations in the global queue is readily obtained by monitoring successes in the reservation subframe and message completions in the data subframe, the number of stations in contention mode is not so easily known. This state variable increases as idle stations become active when new messages arrive and decreases with the successful transmissions of reservation requests. Since the stations are geographically diverse and share only the single broadcast channel, the arrival of new messages cannot be monitored with complete accuracy. Rather, if we assume that stations can decipher collisions, idleness and successes in the minislots of the reservation subframe, only partial information is available and a state estimate must be derived based on the history of the reservation subframe observations. We will refer to idle minislots as holes in this discussion. In this chapter, this estimation problem is considered and a specific estimation technique for the system is proposed that is both effective and practical for implementation.

The following are some of the assumptions of our model relevant to the estimation problem.

- All stations monitor the previous reservation subframe, k and can reliably decipher the number of collisions C_k, the number of successes T_k, and the number of idle minislots or holes, H_k. These observations together form the observation vector, $O_k = (H_k, T_k, C_k)$. Noting that $C_k = L_k V - T_k - H_k$, we may drop the unnecessary entry, $O_k = (H_k, T_k)$. This assumption of ternary feedback is made widely in the literature for estimation of system backlog for S-ALOHA stabilization [20].

- All stations monitor the data subframe and can reliably determine the number of message completions, W^k. Thus the number of stations in the global queue is simply determined by

$$N_2^{k+1} = N_2^k + T_k - W^k \ . \tag{52}$$

- All stations maintain an estimate of the common arrival probability, p, for the idle stations in the network. This allows us to define the estimated number of newly arriving packets in frame k to be

$$\hat{\lambda}_k = \left(M - N_2^k - \hat{n}_k\right) p \ , \tag{53}$$

where $\hat{n}_k$ is the estimate in frame k for the number of contending stations, N_1^k. In this chapter, for ease of notation, we let $n_k = N_1^k$.

Based on these assumptions we wish to develop a recursive estimator of the form,

$$\hat{n}_{k+1} = f(\hat{n}_k, O_k) \ . \tag{54}$$

In the literature on S-ALOHA stabilization techniques, there appears a number of papers that address this estimation problem for the single channel model with ternary feedback. (This would correspond to the $L_k V = 1$ version of the model under consideration here.) While Segall [17] and Rivest [2] both present the formulas for optimal recursive estimation in this case, the formulas are impractical and costly to implement. Recognizing this, [2], [19], [20], [29], and [30] all investigate heuristic schemes to produce reliable estimation with much simpler implementation. Cunningham summarizes these efforts among others in [20] and demonstrates with simulations that Rivest's pseudo-Bayesian recursive estimation technique shows very good performance in comparison with the other estimators. Furthermore, the pseudo-Bayesian techniques is both a simple and elegant approximation to the true optimal

Bayesian estimator, and lends itself fairly readily to the extension to the estimation problem at hand.

For these reasons we choose to adapt the pseudo-Bayesian estimator to our model for estimation of n_k. Initially we derive the true Bayesian estimation formulas and reveal their crippling complexity. We then follow an approach that parallels Rivest's [2], developing a simple linear pseudo-Bayesian estimator to approximate the true Bayesian estimator that performs well in simulation under a variety of system parameters.

4.1 True Bayesian Estimation

The true Bayesian estimation procedure is as follows:

1. Assume an a priori distribution, $P\{n_k\}$, of the number of contending stations, n_k.
2. From our observations O_k in frame k, determine the conditional probability, $P\{O_k \mid n_k\}$.
3. Determine the a posteriori distribution by Bayes rule as follows,

$$P\{n_k \mid O_k\} = \frac{P\{O_k \mid n_k\}\ P\{n_k\}}{P(O_k)} \ . \tag{55}$$

.

4. By considering the generation of new packets and the successful transmission of packets, find the distribution,

$$P\{n_{k+1} \mid O_k\} \ , \tag{56}$$

and use its mean as the estimate $\hat{n}_{k+1}$. We may use eqn. (56) as the apriori distribution for the next frame.

Following this procedure strictly, it is necessary to track and update the distribution, $P\{n_k\}$ and eqn. (54) would become

$$P\{n_{k+1}\} = f(P\{n_k\}, O_k) \ . \tag{57}$$

Note that in the finite population model the state space is limited to, $n_k \in [0, 1, \ldots, M]$, thus the apriori distribution is a vector, $\bar{p}_k$ with elements

$$p_{k,i} \equiv P\{n_k = i\}, \quad i = 0, 1, \ldots, M. \tag{58}$$

For all other values of n_k, the probability is necessarily zero. We may assume the system is initially empty, thus

$$\bar{p}_0 = (1, 0, 0, 0, \ldots) \ . \tag{59}$$

The conditional probability of the observation vector in frame k may be written as

$$\begin{aligned} P(O_k \mid n_k) &\equiv P(H_k = h, T_k = t \mid n_k = n) \\ &= \sum_{\forall n'} P(h, t \mid n') \ P(n' \mid n) \ . \end{aligned} \tag{60}$$

Here, n' is the number of requests actually transmitted in the frame. Thus

$$P(n' \mid n) = \binom{n}{n'} p_s^{n'} (1 - p_s)^{n - n'}, \quad 0 \le n' \le n \ . \tag{61}$$

From appendix B,

$$\begin{aligned} P(h, t \mid n') = (-1)^{h+t} \binom{LV}{h} \binom{LV - h}{t} \sum_{i=t}^{LV-h} \sum_{j=h}^{LV} \\ \left[(-1)^{i+j} \binom{LV - h}{j - h} \binom{LV - h - t}{i - t} \left(1 - \frac{j}{LV}\right)^{n'} \left(1 - \frac{i}{LV - h}\right)^{n' - (LV - h)} \right] \ . \end{aligned} \tag{62}$$

If we denote the aposteriori distribution by the vector $\bar{p}'_k$ with elements

$$p'_{k,i} \equiv P\{n_k = i \mid O_k\} \ , \tag{63}$$

it may be updated according to eqn. (55) with

$$p'_{k,n} = \frac{P(O_k \mid n_k = n) \ p_{k,n}}{\sum_i P(O_k \mid n_k = i) \ p_{k,i}} \quad n = 0, 1, \ldots \tag{64}$$

Finally, as in step 4 we must find the distribution of n_{k+1} by considering the generation of new messages and successful transmission of reservation requests. This distribution will act as the apriori distribution for the next frame, $\bar{p}_{k+1}$. Letting X^k be the number of arriving messages in frame k, the new state is updated by

$$n_{k+1} = n_k - T_k + X^k \quad . \tag{65}$$

Since we know that distribution of X^k is binomial conditional on n_k and we know the distribution of the state, we may now calculate the elements of apriori distribution for the next frame with

$$\begin{aligned} p_{k+1,n} &= \sum_i p'_{k,i} P\left\{ X^k = n - i + T_k \mid n_k = i \right\} \\ &= \sum_{i=0}^{\min(M, n+T_k)} p'_{k,i} \binom{M - i - N_2^k}{n - i + T_k} p^{n-i+t} (1-p)^{M - N_2^k - n - T_k} \quad . \end{aligned} \tag{66}$$

The entire process is repeated for the next frame.

This Bayesian estimation procedure recursively updates a distribution rather than simply a single estimate of the state. This distribution makes optimal use of the available information and by recursion, incorporates the history of observation on the channel. This is made possible only by the Markovian nature of the system. The details of the underlying Markov process are discussed in detail in the next section. To derive a single estimate of the state in the next frame is a simple matter of finding the expectation of the apriori distribution,

$$\hat{n}_{k+1} = \sum_i i \, p_{k+1,i} \quad . \tag{67}$$

While the true Bayesian estimation is the most accurate estimator for our model, it is unlikely that the procedure as presented is of much use for practical implementation in a real-time communication system. The calculations involved in finding the aposteriori distribution according to eqns. (60), (62) and (64) are the largest stumbling block, and as M grows large,

it is evident that the complexity of the algorithm is infeasible for real-time implementation. For this reason we investigate an approximation to the true Bayesian estimate that produces a simple linear estimator. Again, its derivation and resulting form is very similar to that proposed by Rivest [2].

4.2 Pseudo-Bayesian Estimation for the Reservation Subchannel

To derive the approximation to the Bayesian estimator we follow essentially the same procedure, making a couple of key simplifying assumptions that alter the resulting estimate update to be a much simplified expression. Although the simplifying assumptions will necessarily undermine the accuracy of our result, the need for a practical estimator forces us to endure these losses. Furthermore, numerical and simulation results in the following section reveals that the pseudo-Bayesian estimate appears to perform near to the optimal at a small fraction of the computational expense.

Our first simplifying assumption is made of the apriori distribution of n_k. We follow Rivest in assuming that the distribution may be reasonably approximated with a Poisson distribution of mean $\hat{n}_k$. Thus, let

$$P_{\hat{n}_k}\{n_k\} = \frac{e^{-\hat{n}_k}\,\hat{n}_k^{n_k}}{n_k!} \quad . \tag{68}$$

This approximation is made not only to save the amount of information that must be carried from frame to frame but we will see that with some further simplification, this greatly reduces the effort in calculating the aposteriori distribution. The Poisson approximation made in the single channel, infinite user model considered by Rivest is demonstrated to be a reasonably good one. The approximation seems less valid as applied to our model where we consider only finite user populations. Obviously, when $\hat{n}_k$ approaches the upper limit of M, the Poisson distribution breaks down, but if we consider that in typical operation M will be large with respect to the backlog, the Poisson approximation may be a reasonable one. We will proceed

despite these doubts emphasizing that such approximations will be necessary to produce a simpler estimator.

The second key assumption is in our handling of the conditional probability distribution. Rather than confront the unwieldy joint distribution defined in eqns. (60) and (62) we will consider each minislot individually. By doing this we assume that the observations on each minislot are independent of one another. These observations are obviously not independent and for the sake of simplicity we forfeit the information in the correlation of these observations.

Let H_i, T_i and C_i denote the occurrence in minislot i in frame k of a hole, success and collision respectively. The conditional probabilities for minislot i are thus

$$\begin{aligned} P\{H_i \mid n_k\} &\equiv H(n_k) = \left(1 - \frac{p_s}{LV}\right)^{n_k}, \\ P\{T_i \mid n_k\} &\equiv T(n_k) = \frac{p_s n_k}{LV}\left(1 - \frac{p_s}{LV}\right)^{n_k - 1}, \\ P\{C_i \mid n_k\} &\equiv C(n_k) = 1 - H(n_k) - T(n_k) \ . \end{aligned} \tag{69}$$

We wish now to find the aposteriori distributions for each of the three observations. Using Bayes rule,

$$P\{n_k \mid H_i\} = \frac{P\{H_i \mid n_k\} \times P_{\hat{n}_k}(n_k)}{P\{H_i\}} \ . \tag{70}$$

Using eqns. (69) and (68) and with some manipulation on the numerator, we get

$$P\{n_k \mid H_i\} = \frac{e^{-x}\, P_{(\hat{n}_k - x)}(n_k)}{P\{H_i\}} \ , \tag{71}$$

where

$$x = \frac{\hat{n}_k p_s}{LV} \ . \tag{72}$$

Similarly, for the observation of a success or a collision, the aposteriori distributions are

$$\begin{aligned} P\{n_k \mid T_i\} &= \frac{x\, e^{-x}\, P_{(\hat{n}_k - x)}(n_k - 1)}{P\{T_i\}}, \\ P\{n_k \mid C_i\} &= \frac{P_{\hat{n}_k}(n_k)\ (1 - H(n_k) - T(n_k))}{P\{C_i\}} \ . \end{aligned} \tag{73}$$

We see then that in the case of the hole, the mean of this new distribution is

$$\hat{n}_k - x \tag{74}$$

and in the case of a success the mean is

$$\hat{n}_k - x + 1 \ . \tag{75}$$

For a collision and for a collision the mean is derived to be

$$\hat{n}_k + \frac{x^2}{e^x - x - 1} \ . \tag{76}$$

It is worthwhile to note here that in the case of a hole and a success, the Poisson distribution of n_k is maintained while for a collision it is not. For simplicity, we proceed despite this fact, acknowledging at this point that the assumption of the a priori distribution is not exact. It is for this reason that Rivest suffixes the estimator with "pseudo-".

Incorporating the a posteriori distribution means together for observations on all minislots, we now see that our pseudo-Bayesian estimate for the number of contending stations in frame k is

$$\tilde{n}_k = \hat{n}_k - H_k x - T_k x + T_k + C_k\left(\frac{x^2}{e^x - x - 1}\right) \ . \tag{77}$$

At this point, it becomes apparent that the two simplifying approximations (that the apriori distribution is Poisson and the minislot observations are independent) allows us to break down the highly complex procedure in section 4.1 to a simple expression. Adjusting this estimate by our prediction of the number of arriving packets, $\hat{\lambda}$ and the successful transmission of packets observed in frame k, the estimate for frame k+1 becomes

$$\hat{n}_{k+1} = \tilde{n}_k + \left(M - N_2^k - \tilde{n}_k\right)p \ . \tag{78}$$

This provides us with a simple formula to estimate the number of contending stations for the next frame based on the current estimate and observations of the current frame.

Let us now consider how our estimate may be incorporated into the subframe allocation procedure. First it is important to note that the state estimate $\hat{n}_k$ is a real number as defined by eqn. (78), but to choose an allocation in the desired policy, the state must be an integer. We overcome this inconsistency by simply rounding the estimate to the nearest integer. With the common integer estimate, all stations determine the subframe allocation using a simple lookup table, matching the state estimate with its respective allocation, $L_k = L\left(\hat{N}_1, N_2\right)$. As long as the feedback to all stations is identical and all stations use the same algorithm to determine the allocation, the distributed control will be consistent among the stations.

For testing purposes, the pseudo-Bayesian estimation algorithm was implemented into a simulation of the reservation model. The technique was applied to the MF policy on a 20,000 frame simulation and throughput and average delay were calculated. Again, although we cannot determine the confidence intervals of the data points using the usual statistical methods, the comparative smoothness of the resulting curves indicates that this simulation length is sufficient for a simple comparison among control schemes. Both the small and large models considered in the last chapter were examined and Figures 11 and 12 show the delay plotted against the throughput for those two models. We can see that with estimation implemented, the performance of the system is virtually unchanged. There is a slight decrease in the capacity for the small model shown in Figure 11 but the difference is less than 2%, and the MF policy with estimation still shows much better performance than the fixed allocations schemes and the DP policy, where state estimation is unnecessary. These results are encouraging and indicate that while we have been forced to make some simplifying approximations in the derivation of a simple estimator, the resulting pseudo-Bayesian estimate provides for good performance for a dynamic subframe allocation policy.

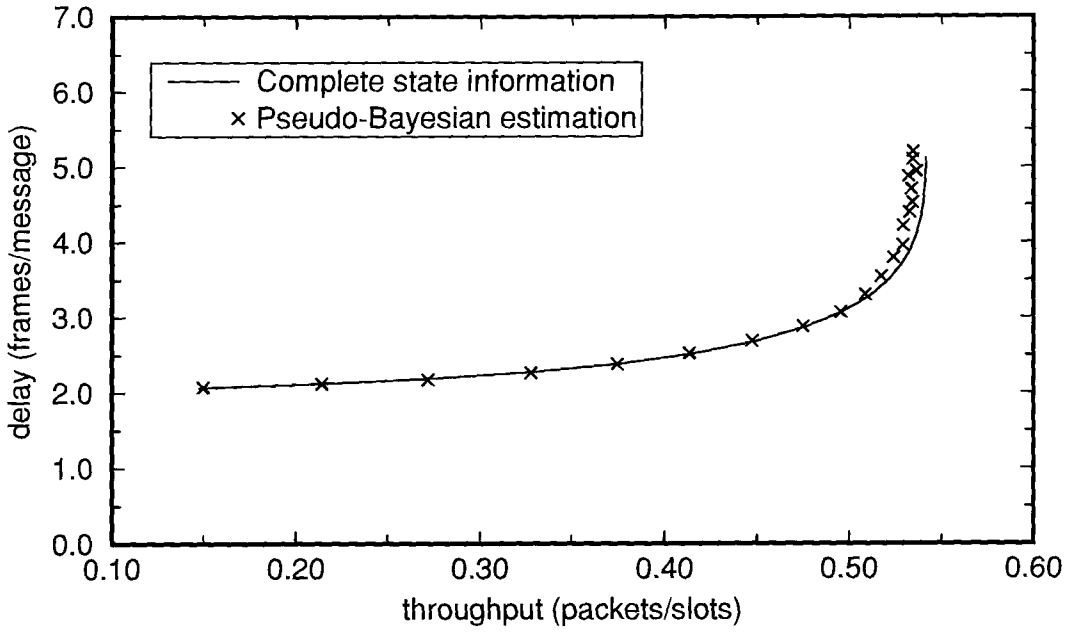

Figure 11. Average delay vs. throughput for the parameter set (M=20, F=5, V=3) : MF policy with and without estimation

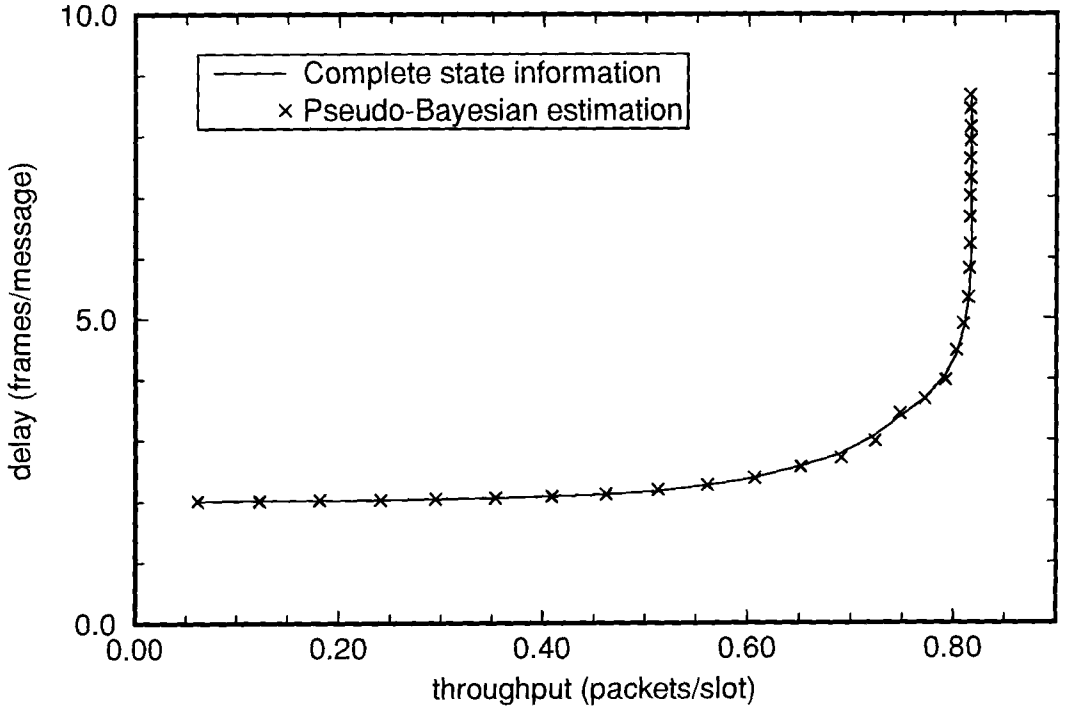

Figure 12. Average delay vs. throughput for the parameter set (M=150, F=12, V=12) : MF policy with and without estimation

The issues of sensitivity of this state estimator to system parameters has not been investigated in this work. Analysis of this type is left to further research.

Chapter 5 MULTIPACKET MESSAGE MODELS

Analysis to this point has focussed exclusively on the single packet message model. In addition to this model, it is of interest to investigate multipacket message models. Indeed, as the total message length with respect to the reservation packet length increases, we may expect the efficiency of the reservation protocol to improve accordingly since proportionally there is less overhead for reservations per useful data slot. In this chapter we continue to investigate the use of dynamic subframe allocation but now with models that handle multipacket message lengths.

For our multipacket message model, users generate messages of variable length rather than the single packet length messages considered earlier. Such a message model is more appropriate for traffic composed of e-mail messages, and general data transfers while single packet messages are more appropriate for such short data transfers as credit card checks and other types of short standardized data signalling. In slotted systems, users would break the message into slot sized packets. While the length of the messages may have one of many possible distributions depending on the nature of the traffic, the exponential distribution is generally used. When the message is broken into packets of equal length, the exponentially distributed message lengths lead to geometrically distributed numbers of packets per message. This distribution covers the full range of possible lengths, from single packet messages to very long messages and has convenient statistical properties. When a station sends its reservation packet, it provides reservation for the entire set of data packets in the multipacket message.

For the multipacket models that we will consider, the frame structure remains as in Figure 1, with stations transmitting reservation requests in slotted ALOHA fashion over the reservation subframe. Thus, the recursive estimator discussed in chapter 4 remains valid for the multipacket message models.

Before beginning detailed analysis of multipacket message models, we must first determine the specific service discipline for the model by which the messages in the global queue are allocated slots in the data subframe. In this chapter we consider two distinct service disciplines. The two multipacket message models corresponding to these disciplines we will refer to simply as the M1 model and the M2 model. In both cases, stations send a single reservation request packet for the entire multipacket message. The M1 model is the simpler of the two, wherein stations are restricted to sending only one packet per frame and the message's length is not indicated in the reservation request packet but rather, the message's end is indicated by an end-of-message (EOM) flag. This type of service discipline is seen in such reservation protocols as R-ALOHA [3] and that proposed in [31] where a station is limited to securing only one slot per frame. In a multichannel scenario where one may consider the allocation of a bank of channels rather than slots in a frame as in [8] and [32], this service discipline is appropriate since stations can send their packets on only one channel at a time when they have a single transmitter. Within the M2 model, message length is indicated in the request packet and stations are allowed to transmit multiple packets within a single frame. This is a far more general reservation service discipline for multipacket messages, and is assumed in reservation protocols considered by [4] and [5].

We shall see that the simplicity of the M1 model allows us to carry out a Markov decision process formulation and policy optimization very similar to that of the single packet model. The M2 model exhibits a much more complex state structure and we are forced to abandon attempts at finding an allocation policy based on full state information to optimize throughput performance. Rather we look at what insight we have gained about heuristic policies from the single packet message model and the M1 model to propose a version of the MF policy that shows good improvements over fixed allocation schemes in the M2 model. For both the M1 and M2 models we provide performance curves which back up our claims that dynamic

subframe allocation provides better system performance than fixed subframe allocation.

5.1 The M1 Model

5.1.1 Model and Markov decision process formulation

Stations in the M1 model send a single reservation request packet for their entire message. This reservation request does not indicate the length of the multipacket message and stations are restricted to sending no more than one packet per frame. Similarly to the single packet message model, when the station has successfully sent its reservation request packet, that station joins the global queue, sending its leading untransmitted data packet on the data subframe when it reaches the head of the global queue. When a packet is sent, if it is the last packet in the message, the station goes into idle mode and waits for a new message to arrive. Alternatively, if it is not the last packet, the station rejoins the global queue at the end of the frame to send its next packet in a subsequent frame. Thus stations are served in a round-robin fashion. For all stations to be informed at the beginning of the next frame of the state of the global queue, an EOM flag should indicate that the last packet is being sent. If the global queue is short so that stations send packets in subsequent frames, this EOM flag should be indicated in the next to last packet of the message (or in the reservation request packet for a single packet message) to avoid the station being allocated a wasted data slot after its message is complete.

Let us assume that for the ith message, the number of packets in the message, r_i is geometrically distributed with mean message length, R packets. Thus

$$P(r_i = r) = \frac{1}{R}\left(1 - \frac{1}{R}\right)^{r-1}, \quad r = 1, 2, \ldots \tag{79}$$

Using this very simple multiple message model, the Markov decision process formulation remains very similar to that of the single packet message model. Figure 13 shows how the

stations move through the queuing system. Note that the state of the system at frame k is still completely defined by the two state variables, the number of stations in contention mode, N_1^k, and the number of stations currently in the global queue, N_2^k. The decision variable remains the size of the reservation subframe, $L(S_k)$. Similarly, since the stations are confined to sending only one packet per frame, the reward may be written the same as in the single packet message model,

$$r_L(S_k) = \min\left(N_2^k, F - L(S_k)\right) \quad . \tag{80}$$

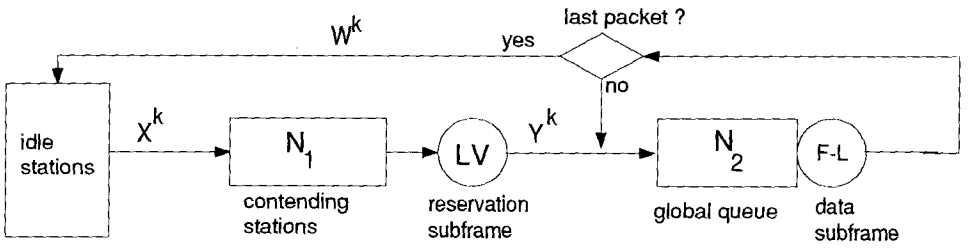

Figure 13. Queueing control representation of the M1 multipacket reservation model

The only difference from the single packet message model appears in the state transition probabilities. Note from Figure 13 that W^k now represents the number of message completions in frame k. The advantage of the geometric distribution becomes evident now as we take advantage of its memoryless property to update the system state. Using the same simplified notation as chapter 2,

$$\begin{gathered} n_1 \equiv N_1^k \quad , \qquad n_2 \equiv N_2^k \quad , \\ m_1 \equiv N_1^{k+1} \quad , \quad m_2 \equiv N_2^{k+1} \quad , \quad a \equiv L(S_k) \quad . \end{gathered} \tag{81}$$

the distribution of W^k conditional on the state and action is

$$Pr\left\{W^k = w \mid n_1, n_2, a\right\} = \sum_{i=w}^{\min(n_2, F-a)} \binom{\min(n_2, F-a)}{i} \left(\frac{1}{R}\right)^i \left(1 - \frac{1}{R}\right)^{\min(n_2, F-a)-i} \quad . \tag{82}$$

The distributions of Y^k and X^k remain the same as in section 2.3.5 eqns. (10) and (11). Combining these distributions we write the transition probabilities of the Markov decision process as

$$\begin{aligned} & p(m_1, m_2 \mid n_1, n_2, a) \\ & = \sum_{w=0}^{\min(n_2, F-a)} [Pr\Big\{W^k = w \mid n_1, n_2, a\Big\} \times \\ & Pr\Big\{Y^k = m_2 - n_2 + w \mid n_1, n_2, a\Big\} \times \\ & Pr\Big\{X^k = m_1 - n_1 + m_2 - n_2 + w \mid n_1, n_2, a\Big\}]. \end{aligned} \tag{83}$$

Again, as in the single packet message model, we see that due to the memoryless property of the message length distribution the state transition probabilities satisfy the Markovian property. Note that the single packet message model is identical to the M1 model with R=1.

5.1.2 Dynamic allocation policies

From the Markov decision process formulation we see that the M1 model has a very similar state structure to that of the single packet model formulated in chapter 2. Thus our approach to finding good policies is also similar. As in the single packet message model, we consider fixed allocation schemes, optimal allocation and two heuristic policies. These policies correspond directly to those examined in chapter 3 for the single packet message model. Some comments are given as to how each of these policies is adapted for the M1 model. Figures 14–17 show performance results of the different policies in the M1 model. These results are discussed in section 5.1.3. It should be noted immediately that the reservation subsystem is unaffected by the extension to the multipacket model and the state estimation technique discussed in chapter 4 remains unchanged for this model.

5.1.2.1 Fixed allocation

Although Szpankowski [10] proposed his method for finding the best fixed allocation for specifically the single packet message model, it is simple to extend it to the M1 model. If the throughput on the reservation subframes assumed to be $1/e$ requests per minislot, then

the average reservation overhead for the message will be e/V slots per message. Setting the proportion of reservation slots to data slots equal to the proportion of the average reservation overhead to average message length we get

$$\frac{L}{F-L} = \frac{e/V}{R} \tag{84}$$

and solve to get

$$L = \frac{F}{1 + RV/e} \quad . \tag{85}$$

We use the integer values of eqn. (85) for the best fixed allocations.

5.1.2.2 Optimal policy

As in section 3.2, we apply the modified policy iteration algorithm with our new transition probability matrix to find the optimal policy for the M1 model which maximizes the system throughput. All nondegenerate policies form unichain Markov processes and thus the optimality equations (eqn. (26)) apply and the algorithm is guaranteed to converge to the optimal policy, $L^*(N_1, N_2)$.

5.1.2.3 DP policy

For the M1 model, the DP policy again maximizes the one-step throughput for the current frame. Due to the constraint that each station may send at most one packet per frame, at most N_2^k packets may be sent in the frame and we specify the DP policy to be

$$L_{DP}(N_1, N_2) = \max\left(0, F - N_2^k\right) \quad . \tag{86}$$

5.1.2.4 MF policy

In section 3.4 a new criteria, referred to as the "flow", was proposed to allow a one step maximization and thereby approximate the optimal policy without the computational

complexity of the dynamic programming techniques. For the M1 policy we again use such a criteria. Since W^k now refers to the number of message completions in the frame, we now introduce the variable Z^k which indicates the number of packet transmissions in frame k. (In the situation where R=1, $Z^k = W^k$, and are thus interchangeable in the single packet message model.) We define the "flow" in frame k as

$$F^k = \alpha Y^k + Z^k \ , \tag{87}$$

and set $\alpha = e/V$. Noting that $E[Y^k]$ is defined by eqn. (37) and that $Z^k = \min(N_2^k, F - L)$, the expectation of F^k may be maximized with respect to L at each state to give the MF policy, $L_{MF}(N_1, N_2)$. It was pointed out in section 3.4 that the MF policy was attractive for the single packet message model in that it was independent of the arrival probability, p. This property is maintained for the M1 version of the MF policy. Furthermore, since $E[F^k]$ is independent of the average message length, R, so too will be the MF policy, and the MF policy derived for the single packet message model will apply unchanged to the M1 model with the same F and V.

5.1.3 Numerical results

These policies were analyzed for a small network model with parameters {M=20, F=5, V=3, R=3} and average delay and throughput were calculated using the methods described in section 2.4. The retransmission probability, p_s is set to unity unless otherwise indicated. The optimal policies for different arrival policies were calculated using the modified policy iteration algorithm and a set of sample policies appear on Table 4. The MF and DP policies are identical to those appearing on tables 2 and 3. Notice that as in the single packet message model, the optimal policies for high traffic levels are very similar to the MF policy, foreshadowing the almost identical performance of the optimal and MF policies.

Figures 14 and 15 plot the throughput versus the arrival probability and the average delay against the throughput respectively for the fixed and dynamic policies. In these curves, we see

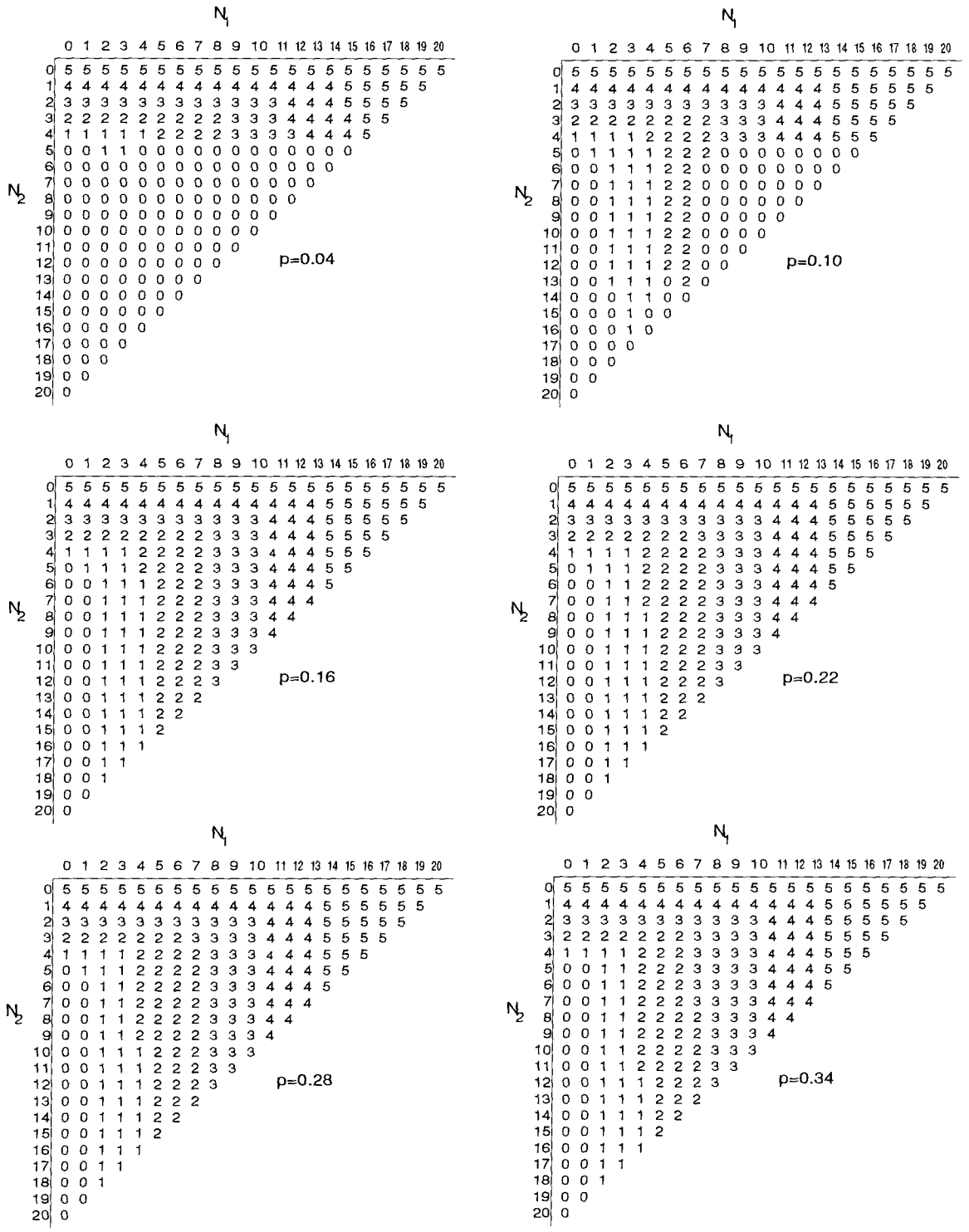

Table 4. Optimal policies for a range of arrival probabilities in the M1 model with parameters (M=20, F=5, V=3, R=3)

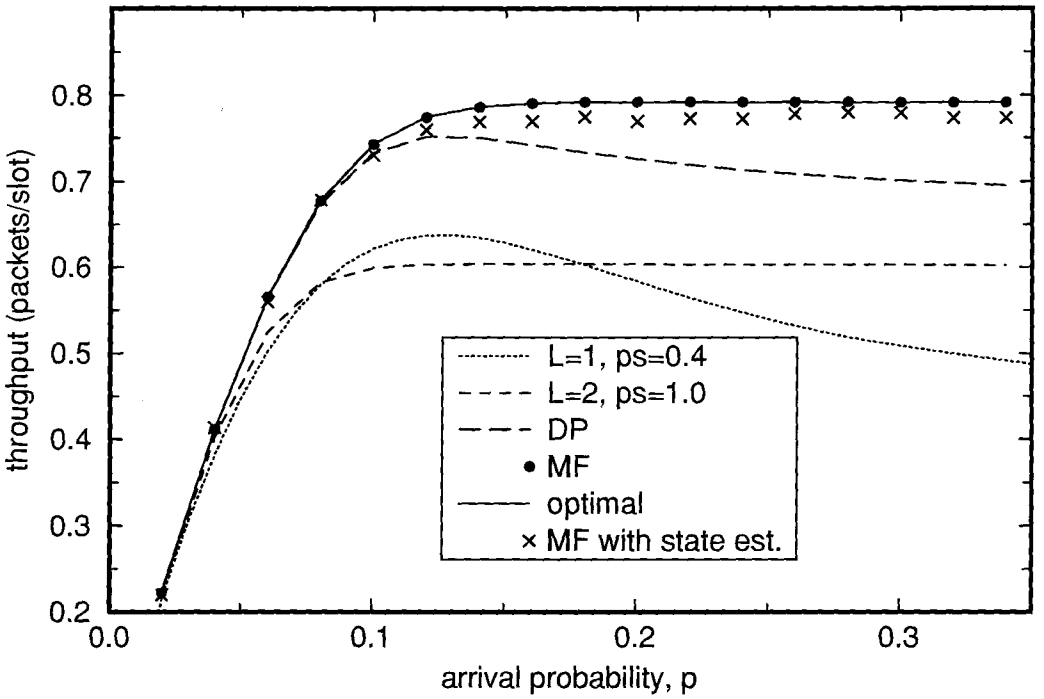

Figure 14. Throughput vs. arrival probability in the M1 model for the parameter set (M=20, F=5, V=3, R=3) : fixed allocation policy, optimal policy, MF policy and DP policy

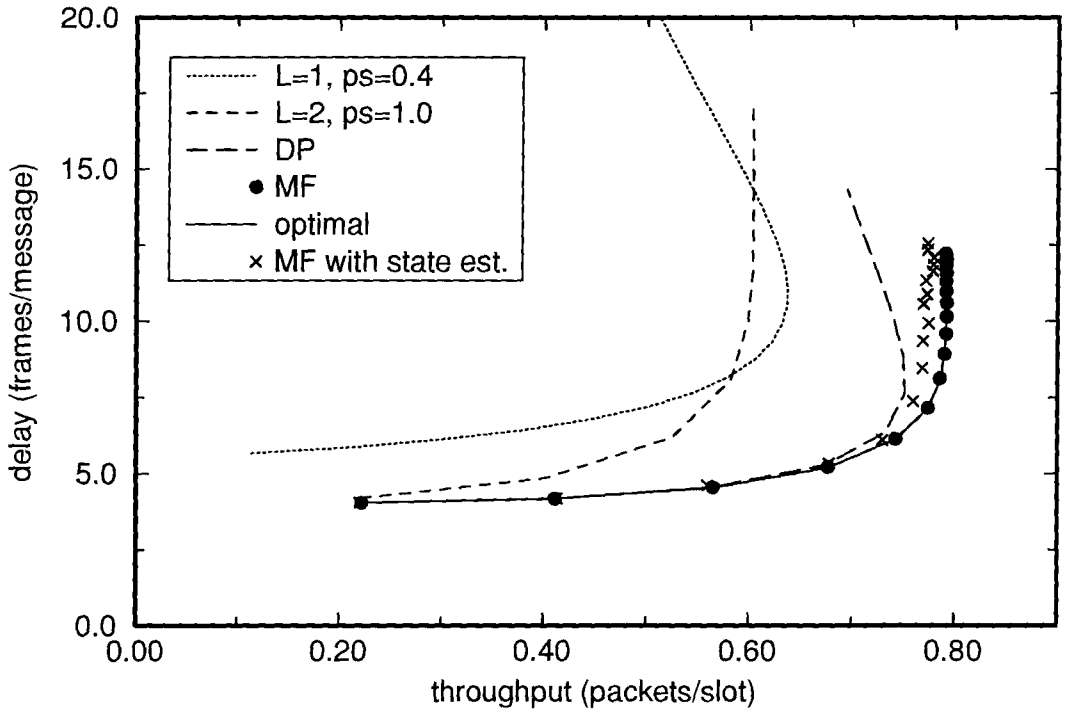

Figure 15. Average delay vs. throughput in the M1 model for the parameter set (M=20, F=5, V=3, R=3) : fixed allocation policy, optimal policy, MF policy and DP policy

that the dynamic policies all show lower average delays for equivalent throughputs and higher throughputs for equivalent arrival probabilities than the fixed policies. For example, for a throughput of 0.58 packets per slot, the fixed allocation schemes operate close to their capacity with a large delay of over 8 frames. The dynamic schemes all show delays of near 4.5 frames when operating at that throughput, a reduction of over 40%. Furthermore, the capacities of the dynamic policies are considerably higher that those of the fixed policies. For this model the highest throughput achieved by a fixed allocation schemes is 0.637 packets per slot when L=1 and p=0.13. In comparison, the highest throughputs achieved by the dynamic policies are 0.751 for the DP policy and 0.792 for the optimal and MF policies, an improvement of 18% and 24% respectively. As in the single packet message model, the MF and the optimal policies under complete state information are indistinguishable in terms of performance and both show slightly lower delays for equivalent throughputs than the DP policy. Included in Figures 14 and 15 are the simulation results of the MF policy implemented with state estimation incorporated. The introduction of state estimation has eroded the capacity of the protocol from the complete state information case, but achieves a capacity about 3% higher than the DP policy and 21% higher than the fixed allocation capacity.

We also examined a larger scale system model with parameters (M=150, F=15, V=8, R=5). In this case, both the modified policy iteration algorithm and the solving of the global balance equations of the Markov processes are extremely complex undertakings and we opt for simulation analysis and abandon the optimal policy. Simulations were conducted for the policies of interest over a range of traffic levels and delay and throughput statistics were gathered. The same simulation length of 20,000 frames was used as in the single packet message model and provides a sufficient accuracy for simple comparison among policies. Figures 16 and 17 show the simulation results plotting throughput against the arrival probability and the delay versus the throughput respectively. State estimation is

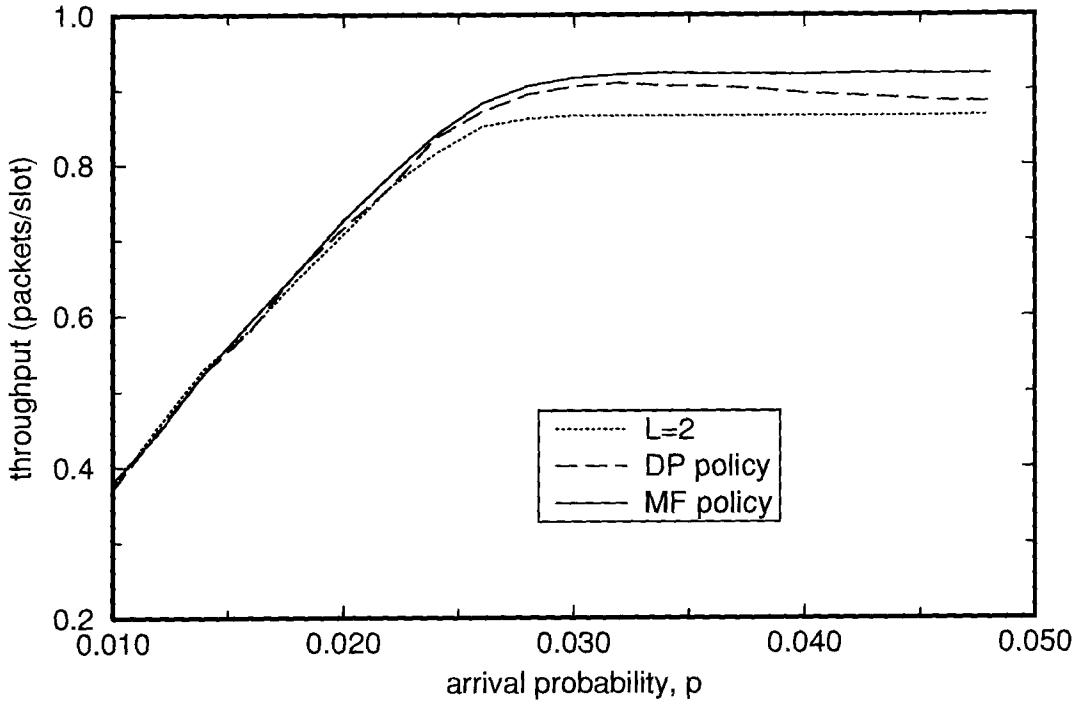

Figure 16. Throughput vs. arrival probability in the M1 model for the parameter set (M=150, F=15, V=8, R=5) : fixed allocation policy, MF policy and DP policy

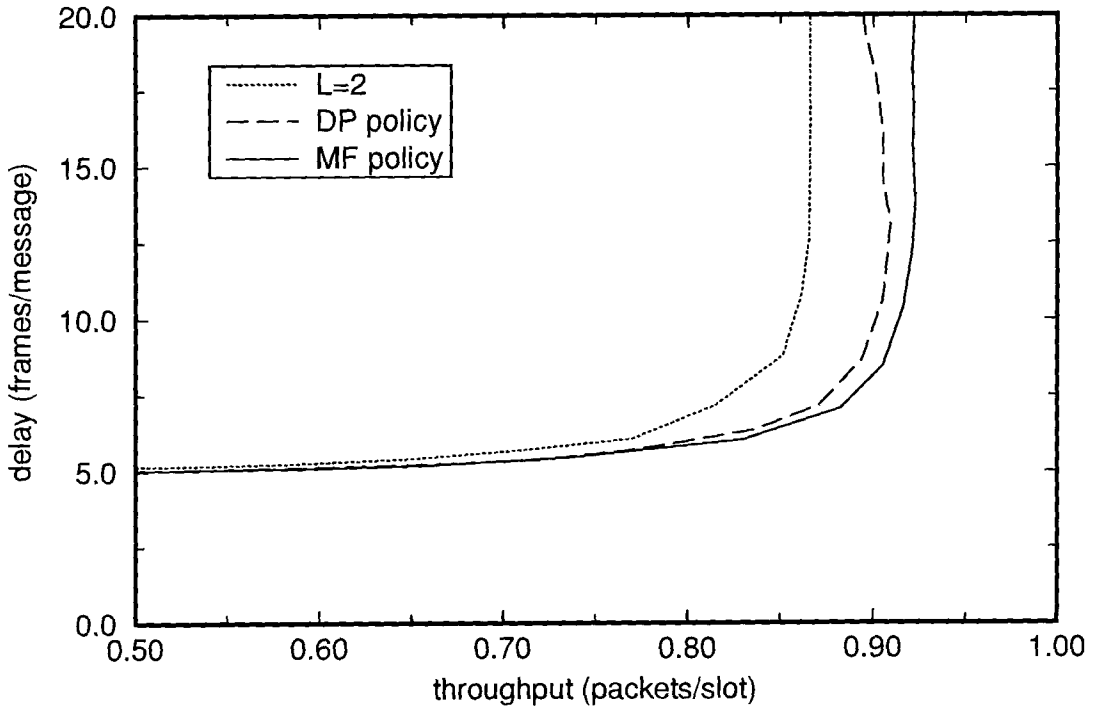

Figure 17. Average delay vs. throughput in the M1 model for the parameter set (M=150, F=15, V=8, R=5) : fixed allocation policy, MF policy and DP policy

implemented for the MF model and the retransmission probabilities are set at unity. We see from these results that as in the small model, the dynamic policies show lower delays in the high throughput regions of the graph and the capacity of the MF policy is the highest. At throughputs up to near 0.8 packets/slot, the delays of all policies are very close. Beyond this, the fixed allocation policy shows a sudden increase in delay as it reaches its capacity at 0.87 packets/slot. In comparison, the DP and MF policies produce capacities of 0.91 and 0.92 respectively, an improvement of approximately 5% over the fixed allocation capacity. This relative improvement achieved by introducing dynamic allocation, while less than that for models with smaller frame size, is still encouraging.

5.1.4 Capacity approximation

The capacity approximation technique described in section 3.5 for the single packet message model may be extended to the M1 model fairly simply. We again look at the case where M is very large and the reservation subsystem throughput is stable with throughput approaching $1/e$ reservation packets per minislot when the number of contending stations grows large. In the M1 case, the average overhead for reservation request packet transmission is e/V and the ideal capacity

$$C = \frac{1}{1 + \frac{e}{RV}} \ , \tag{88}$$

as discussed in [28]. As in section 3.5, this ideal is a capacity approximation that is based on the assumption that perfect flow balance exists between the two subchannels. For a fixed allocation scheme of reservation subframe size L_f, this perfect flow balance is not in general achieved and the capacity is better approximated by

$$C(L_f) = \min\left(\frac{RVL_f}{Fe}, \frac{F - L_f}{F}\right) \tag{89}$$

due to the bottleneck in the reservation subsystem or the data subsystem caused by choosing a whole number of reservation slots. By the same line of argument as in section 3.5 we can

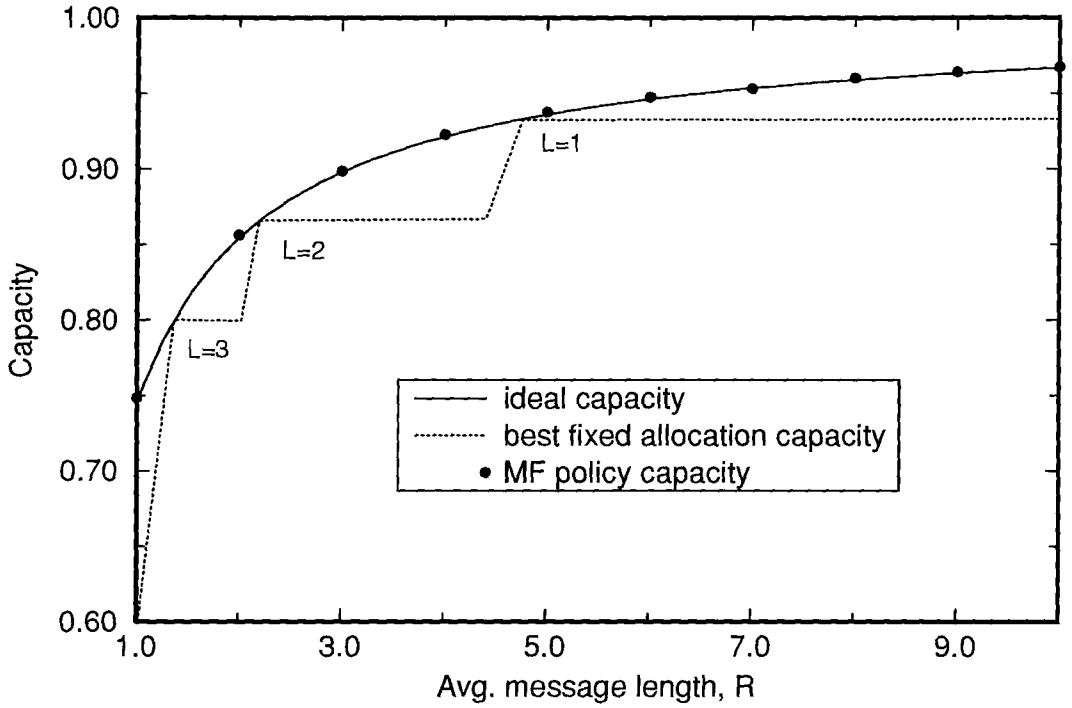

Figure 18. Approximate capacity plotted against the average message length, R, for the ideal capacity, the best fixed allocation policies' capacity and the MF policy capacity for the case F=15

say that the ideal throughput of eqn. (88) is a good approximation for the capacity of the DP policy, MF policy and optimal policies when M is large. We verify this by finding capacities by simulation for the MF policy applied to a model with parameters (M=150, F=15, V=8) and state estimation incorporated for different values of R.

In Figure 18 we plot system capacity against the average message length for the MF simulations, the ideal capacity (eqn. (88)) and the best fixed capacities (eqn. (89)). The granular effect that we observed when capacity was plotted against V is again evident in this figure but now with respect to the average message length, and the difference between the fixed allocation capacity and the ideal capacity is highly sensitive to R. The MF simulations indicate that the ideal capacity is a good approximation for the MF capacity for large M. Although the difference is not evident from the figure, the MF simulation results are within 1% of the ideal. It is worthy to repeat here that the MF policy is independent of changing average message length. Thus, the MF policy exhibits the quality of adapting not only to the traffic level, as it

is independent of the arrival probability, p, but its capacity is robust in the face of changing average message length. It may be expected that in real systems, the traffic characteristic would change in this fashion, showing time variance in both intensity and average message length. It should be noted though, that while the MF policy is independent of these traffic parameters, the accuracy of state estimation upon which the MF policy depends, almost certainly depends at least somewhat on these parameters. The extent of this dependance has not been investigated in this work.

5.1.5 Problems with the M1 model

The M1 model for multipacket messages is simple, allowing policy optimization and exact Markovian analysis of the policies that we have considered. This simplicity is only possible because of the following two key constraints put on the operation of the data subsystem:

- Reservation request packets do not contain information regarding the length of the message.
- Stations may only send one packet per frame.

For a real system, these constraints limit the performance of the protocol substantially. The first constraint limits the possible information that is available to the system for control of the allocation. Information on the length of the individual messages is possibly valuable for subframe allocation control. The second constraint is even more limiting. By allowing the stations to send only one packet per frame, we are introducing a large minimum delay for message transmission. Consider a situation where a system has a very low traffic level. If a station generates a message of length 4 packets, at best, the total transmission delay will be 5 frames, one for the reservation, and 4 for the message. This is indeed an inefficient use of the channel resource. When the system is very lightly loaded, the delay could be 2 frames, a 1 frame delay for reservation and another frame for data as all the packets could be sent in the next frame. At higher traffic levels, forcing the stations to share the data subframe in a

round-robin fashion seems more legitimate. To address these problems, we look at the more general multipacket model that does not impose these constraints on the protocol.

5.2 The M2 Model

5.2.1 Model and state structure

The M2 model is a more general one than the M1 model for multipacket message transmission. The system remains divided into a reservation subsystem and data subsystem but now the two constraints are loosened. Reservation request packets indicate the number of packets in the current message of the transmitting station. Furthermore, the stations in the global queue may send multiple packets in the data subframe. Note that for the M1 model, all stations in the global queue were identical from the perspective of queue service control and it was natural to serve them in a round robin fashion. Now, entries into the global queue are distinguished by their number of remaining packets. As such, to be entirely general, the system controller should choose not only the number of data slots available for packet transmission, but also which stations in the global queue should be served. As such, the state and decision space becomes very large in the completely general case.

To formulate this more general model as a Markov decision process is no longer a simple derivation and while it may be described as Markovian, the dimensionality of the state space grows large and the number of possible state grows without bound. The reservation subsystem is described as always by one state variable, the number of contending stations, N_1. The variables describing the state of the global queue may no longer be limited to a single number since not only the number of stations in the queue must be represented, but also the number of remaining packets for each. We may define a set of new state variables,

$$A_i \; ; \; i = 1, 2, \ldots M \quad , \tag{90}$$

that represent the number of unsent packets at the ith station in the global queue. There are M such variables since there may be up to M stations in the queue. We no longer need to

track the number of stations in the queue since that is simply the number of A_i which are non-zero. Thus the state vector becomes

$$S = (N_1, A_1, A_2, \ldots A_M) \ . \tag{91}$$

Although at normal unsaturated operation the vast majority of the A_i would be zero, all M+1 elements must be retained in the state vector for the process to be accurately described as Markovian. When message lengths are assumed to be geometrically distributed, the lengths may take on any positive integer value. Thus we are faced with a Markov process with a state space of dimension M+1 and an infinite number of states. While we could consider a truncated message length distribution to achieve a finite state Markov process, the state space dimensionality alone is imposing in its size.

The range of available actions in the very general M2 model is no less imposing. Since the stations in the global queue are distinguished by their number of unsent packets, there may be advantage for the controller to allocate a specific number of slots in each data subframe to specific stations. For instance, there may be gains to be made by servicing stations with short messages to free them for new arrivals and wait to serve stations with long messages until the queue is less busy. We may define a set of new action variables,

$$B_i \ ; \ i = 1, 2, \ldots F \ , \tag{92}$$

specifying the stations that are serviced in the current data subframe. These variables form the action vector B. The length of the reservation subframe would be implicit in this action vector.

We provide here a very simple example to indicate how a controlling policy may work. Consider a model where the frame size is F=5, there are currently 4 contending stations and there are two stations in positions 1 and 2 of the global queue with 5 and 1 unsent packets respectively. The state would then be

$$\begin{aligned} S &= (N_1, A_1, A_2, \ldots A_M) \\ &= (4, 5, 1, 0, 0, \ldots) \ . \end{aligned} \tag{93}$$

A sample policy may allocate one slot for the reservation subframe, 1 slot to complete service of the station in position 2 and the remaining 3 slots to service the station in position 1. Thus the action vector would be

$$\begin{aligned} B &= (B_1, B_2, \ldots B_F) \\ &= (0, 1, 1, 1, 2) \ . \end{aligned} \tag{94}$$

A zero indicates that the slot is part of the reservation subframe.

To find an optimal state dependant policy for frame allocation it would be necessary to formulate this model as a Markov decision process, deriving the action dependant transition probabilities and state rewards. We stop short of attempting this due to the very large dimensionality and size of such a formulation. Indeed, we can expect that such an optimal policy for even quite small network model would be unattainable using the cumbersome techniques that we applied to the simple two dimensional state space and single action variable models discussed earlier in this thesis. This author has not seen any applications of Markov decision process theory of state dimension greater than 2 in the literature. Furthermore, when the Markov process has a countable number of possible states, there are no general techniques for deriving an optimal policy in the Markov decision process literature. In this most general model description, we leave finding the optimal policy as an open research problem and proceed in the next section to use insights gained in the single packet message model and the M1 model to find simple dynamic allocation policies that work quite well for the M2 model.

5.2.2 Simple allocation in the M2 model

Having abandoned the high road of Markov decision process formulation and optimization for the M2 model, we now make some simplifications to the problem that allows us to adapt the heuristic policies of the M1 model to the more general M2 model. First, let us reduce the actions to simply the allocation of the size of the reservation subframe, L, and assume that stations in the global queue are serviced to completion on a first come first served basis. Thus

we maintain generality in that stations may, and inevitably will transmit multiple packets per frame. (Essentially, this simplified service discipline is identical to that considered by [5] who does an approximate analysis of the performance in this case for a fixed allocation policy.) This simpler service discipline uses a simple subset of the state information specifying N_1 to be the number of contending stations and N_2 to be the total number of packets waiting for service in the global queue. The latter incorporates all global queue information by summing the true state variables

$$N_2 = A_1 + A_2 + \ldots + A_M \quad . \tag{95}$$

We will refer to these variables as the state although it must be noted that they contain only a portion (albeit a significant portion) of the state information in the Markov process. With these state variables, our system very closely resembles that of the M1 model. Since unsent packets in the global queue are effectively no different than single packet messages from the standpoint of the data subsystem, it seems logical to apply the MF and DP policies (which are independent of the average message lengths) with our two state variables N_1 and the redefined N_2. Since the process is no longer Markovian on these state variables, we cannot apply the Markovian techniques to determine throughput and delay so we incorporate these policies in simulation for the same model as was examined for the M1 model (F=5, V=3, M=20, R=3). A simulation over 60,000 frames was conducted, collecting delay and throughput statistics for different arrival probabilities. The large simulation length is chosen here simply because in this smaller model, the time taken for a simulation of this many frames is not as great as in the larger networks considered in earlier simulations. In any case, this simulation length provides us with results that enable comparison between the different models and the allocation policies. Figure 19 plots the average delay against the throughput for a fixed allocation policy and the MF policy with state estimation for the M2 model and M1 model simulations. We can see that having allowed stations to transmit multiple packets

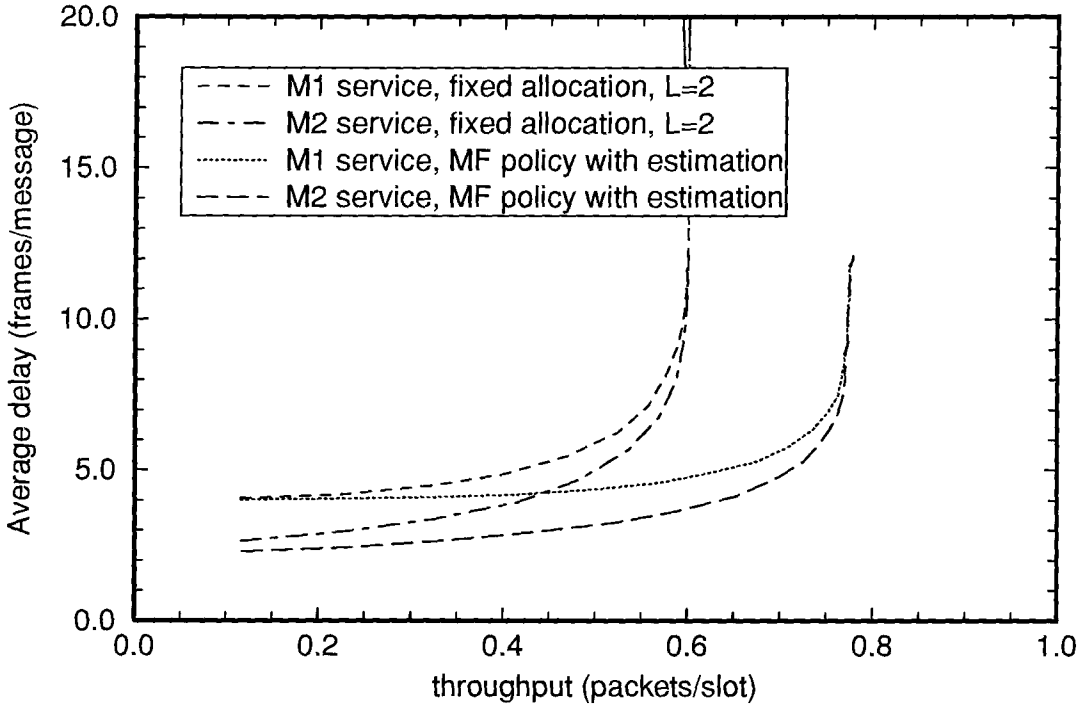

Figure 19. Average delay vs. throughput for a fixed policy and the MF policy with estimation in the M1 and M2 models with parameters (M=15, F=5, V=3, R=3)

per frame has produced the expected reduction in average message delay at low throughput levels. In this case the delay is reduced from approximately 4 frames/message for the M1 model (1 frame for reservation and 3 for data packets) to near 2 frames for the M2 model (1 frame for reservation and 1 frame for data packets). This reduction is significant and we may expect the savings to increase with increasing message length. For low traffic levels, we may expect the M1 model delay to be $1+R$ frames while the M2 model will produce delays near $1+R/F$. As throughput reaches capacity, the difference in average delay between identical policies in the M1 and M2 models decreases and seems to disappear at capacity. Capacity for the same policies for the two models appears the same. This is not unexpected since we expect the capacity approximation of the M1 model to be applicable also to the M2 model.

In summary, loosening the constraints on the M1 model has rendered optimization and Markovian formulation of the general model inachievable. We do find though, that we can still improve results with dynamic subframe allocation using a portion of the state

information and a simple service discipline that does not take advantage of its full range of options. Capacity remains the same, and the average delay is significantly reduced for low and medium throughput levels. Finding optimal or improved policies that incorporate complete state information and the full range of possible global queue service disciplines remains an open research problem.

Chapter 6 CONCLUSION

6.1 Summary

In this thesis we have investigated how the reservation multiple access protocol first proposed by Roberts[1] may be improved by dynamically allocating the size of the reservation subframe based on state feedback information. It was shown that an optimal allocation policy exists when complete state information is available and using infinite horizon dynamic programming techniques, optimal policies were found for specific networks. Numerical analysis and simulation results demonstrated that the capacity of the broadcast channel is dramatically improved over previously considered fixed allocation schemes when the optimal policy is implemented and average message delays are decreased for the same system throughput.

While the optimal policy is valuable as it produces an upper bound on the system throughput, the complexity of its derivation and implementation prompts us to investigate two heuristic allocation policies that show performance results close to optimal while holding some implementation advantages over the optimal. The MF policy shows the best performance of the two, almost matching the optimal performance. The DP policy is attractive in that it depends only on the fully observed state variable for its implementation. Both these policies are simple to derive, and continue to perform well when the average message length changes or the system traffic intensity changes. Capacity approximations show that the heuristic policies can produce capacities close to the ideal capacity for the reservation protocol while fixed allocation schemes often fall far short of this ideal.

Initial analysis was done on the model with the assumption that complete state information is available for control. This is an unrealistic assumption, and, in loosening the assumption it is necessary to find for policies dependant on full state information a suitable technique

to estimate the number of contending stations using ternary feedback from the reservation subframe. We have proposed such an estimator, using a variation on Rivest's [2] pseudo-Bayesian estimator to provide a recursive estimate of the unknown state. Simulation results have shown that the implementation of this estimator does not significantly depreciate the performance of the dynamic policies where estimation is necessary.

Finally, we considered the extension of the model to allow multipacket messages. Two service disciplines were considered. The first depended on severe constraints to allow us to maintain our Markovian formulation, derive optimal policies and generate exact performance results by analysis of the Markov process. The second multipacket model, while being more applicable to real systems, prevents valuable Markovian analysis with its complex state structure. Thus we were forced to use insights gained from our results for the simpler models to propose a heuristic dynamic allocation scheme for this more general model and test it with simulation. For both the multipacket message models, numerical and simulation results show that the performance of the dynamic allocation policies is improved significantly over simple fixed allocation.

6.2 Proposals for Further Research

In this research we have demonstrated that altering Roberts model to allow dynamic subframe allocation can significantly improve the performance of the broadcast channel. The results are indeed encouraging and it is likely that follow-up research could broaden the advances made in this work. In this section we propose areas of further work that may prove beneficial.

In our analysis, the user population in the network had to be finite to allow us to do finite-state Markovian analysis. Indeed, due to the multiple dimensions of the problem, the user population had to be quite small to produce analytical results. Since network populations in a real world application may be quite large and indeed may fluctuate as users go on-line

and off-line, it would be more realistic to use an infinite population model with a true Poisson traffic assumption to achieve a more general result. While derivation of the true optimal policy may be inachievable with this model, use of approximate methods to propose extensions to and analyze the DP and MF policies for the infinite population model would be beneficial.

The optimal policies in this thesis were derived based on the full state information. A state estimator was then proposed independently to provide an estimate of the system state to implement the optimal policy. While we have treated these as separate problems, derivation of the true optimal policy for the partially observed Markov process would integrate the two problems, basing the allocation on the aposteriori distribution of the state rather than the state estimate (the mean of the aposteriori). While in many systems for controlling a partially observed Markov process these problems may be shown to be separable, it is unknown if this is the case for the reservation protocol that we have considered. Techniques for analyzing such control problems are given in [26]. An investigation on these lines would be complex but would possibly produce a theoretically appealing result.

The improvement in channel performance using dynamic subframe allocation reveals that state information is indeed valuable for channel access control. While this fact has been recognized by previous researchers for the stabilization of a simple slotted ALOHA network [2], [11], [12], [18], this work is the first to use this information explicitly to optimize subframe allocation. The dynamic programming techniques that we have used rely on the fact that the model may be reasonable formulated as a Markovian decision process. In the model that we have considered this has been the case and it is very likely that such Markovian decision process formulation can be performed for other reservation-type protocols to allow subframe allocation optimization. These are some similar protocols that may benefit from such analysis.

- R-ALOHA [3] is a reservation protocol that allows users to send multipacket messages by reserving future slots on the frame by successfully sending their first data packet by

slotted ALOHA on an unreserved slot. The allocation of slots as reserved or unreserved is done by a simple heuristic in the R-ALOHA protocol and the use of state feedback for optimal allocation has not been considered in the literature.

- Leung [8] proposes multichannel reservation protocols for multipacket messages which use successful first packet transmission by slotted ALOHA to gain reserved data slots for the rest of the packets. Thus the multiple channels are at each slot allocated to be reserved for data packets or free for contention. This partition of a fixed number of channels for access by stations in two distinct modes is not very dissimilar to the single channel reservation model that we have considered.
- Wong and Yum [33] propose a combined random access/reservation protocol for single packet messages that allows both packet transmission by slotted ALOHA and by minislot reservation request transmission followed by reserved access. Essentially, time on the channel is divided between minislots, slotted ALOHA slots and reserved data slots. Optimizing this three way allocation based on state feedback may improve the system performance.

Bibliography

[1] L. Roberts, "Dynamic allocation of satellite capacity through packet reservation," *AFIPS Conf. Proc.*, vol. 42, pp. 711 – 716, 1973.

[2] R. Rivest, "Network control by Bayesian broadcast," *IEEE Trans. Information Theory*, vol. IT-33, pp. 323 – 328, May 1987.

[3] S. S. Lam, "Packet broadcast networks - a performance analysis of the R-ALOHA protocol," *IEEE Trans. Comput.*, vol. COM-29, pp. 596 – 603, July 1980.

[4] R. Binder, "A dynamic packet-switching system for satellite broadcast channels," in *Proc. of the IEEE International Conf. on Communications*, (San Francisco, CA), pp. 41–1 to 41–5, June 1975.

[5] S. Tasaka and Y. Ishibashi, "A reservation protocol for satellite packet communication — a performance analysis and stability considerations," *IEEE Trans. Commun.*, vol. COM-32, pp. 920 – 927, Aug. 1984.

[6] I. Rubin, "Access control disciplines for multiaccess communication channels: Reservation and TDMA schemes," *IEEE Trans. Inf. Th.*, vol. IT-25, pp. 516 – 536, Sept. 1979.

[7] F. A. Tobagi and L. Kleinrock, "Packet switching in radio channels: Part III - polling and (dynamic) split channel reservation multiple access," *IEEE Trans. Communications*, vol. 24, pp. 832 – 845, Aug. 1976.

[8] V. C. M. Leung, "A multichannel reservation protocol for satellite multiple access networks," in *Proc. of the IEEE Pacific Rim Conf. on Commun., Computers and Sig. Proc.*, (Victoria, BC), pp. 437 – 440, May 1991.

[9] S. F. W. Ng and J. W. Mark, "Multiaccess model for packet switching with a satellite having processing capability : Delay analysis," *IEEE Trans. Communications*, vol. 26, pp. 283 – 290, Feb. 1978.

[10]W. Szpankowski, "Analysis and stability considerations in a reservation multiaccess system," *IEEE Trans. Commun.*, vol. COM-31, pp. 684 – 692, May 1983.

[11]J. Mosely and P. Humblet, "A class of efficient contention resolution algorithms for multiple access channels," *IEEE Trans. Commun.*, vol. COM-33, pp. 145 – 151, Feb. 1985.

[12]E. Fainberg, Y. Kogan, and A. Smirnov, "Optimal control by the restransmission probability in slotted aloha systems," *Performance Evaluation*, vol. 5, pp. 85 – 96, 1985.

[13]J.-B. Suk and C. G. Cassandras, "Optimal scheduling of two competing queues with blocking," *IEEE Trans. Automatic Control*, vol. 36, pp. 1086 – 1091, Sept. 1991.

[14]K. R. Krishnan, "Joining the right queue : a state dependant decision rule," *IEEE Trans. Automatic Control*, vol. 35, pp. 104 – 108, Jan. 1990.

[15]Z. Rosberg and I. S. Gopal, "Optimal hop-by-hop flow control in computer networks," *IEEE Trans. Automatic Control*, vol. 31, pp. 813 – 822, Sept. 1986.

[16]Z. Rosberg and A. M. Makowski, "Optimal routing to parallel heterogeneous servers - small arrival rates," *IEEE Trans. Automatic Control*, vol. 35, pp. 789 – 796, July 1990.

[17]A. Segall, "Recursive estimation from discrete-time point processes," *IEEE Trans. Information Theory*, vol. 22, pp. 422 – 431, July 1976.

[18]L. Kleinrock and S. S. Lam, "Packet-switching in a multi-access broadcast channel: dynamic control procedures," *IEEE Trans. Commun.*, vol. COM-23, pp. 891 – 904, Sept. 1975.

[19]S. Thomopolous, "A simple and versatile decentralized control for slotted ALOHA, reservation ALOHA, and local area networks," *IEEE Trans. Communications*, vol. 36, pp. 662 – 674, March 1990.

[20]G. A. Cunningham, "Delay versus throughput comparisons for stabilized slotted ALOHA," *IEEE Trans. on Comm.*, vol. COM-38, pp. 1932 – 1934, Nov. 1990.

[21]L. Kleinrock and S. S. Lam, "Packet-switching in a multi-access broadcast channel: performance evaluation," *IEEE Trans. Commun.*, vol. COM-23, pp. 410 – 423, Apr. 1975.

[22]J. T. Lim and S. M. Meerov, "Performance of markovian access protocols in satellite channels," *IEEE Trans. Communications*, vol. 38, pp. 273 – 276, March 1990.

[23]M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York, NY: J. Wiley and Sons, 1993.

[24]M. Puterman, "Dynamic programming," *Encyclopedia of Physical Science and Technology*, vol. 4, pp. 438 – 463, 1987.

[25]C. Derman, *Finite State Markovian Decision Processes*. New York, NY: Academic Press, 1970.

[26]D. P. Bertsekas, *Dynamic Programming and Stochastic Control*. New York, NY: Academic Press, 1976.

[27]R. Howard, *Dynammic Programming and Markov Processes*. Cambridge, Ma: MIT Press, 1960.

[28]R. Gallager and D. Bertzekas, *Data Networks*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1987.

[29]B. Hajek and T. VanLoon, "Decentralized dynamic control of a multiaccess broadcast channel," *IEEE Trans. Automatic Control*, vol. AC-27, pp. 559 – 569, June 1982.

[30]L. Merakos and D. Kazakos, "On retransmission control policies in multiple-access communications networks," *IEEE Trans. Automatic Control*, vol. AC-30, pp. 109 – 117, Feb. 1985.

[31]T. S. H. Muyahara and T. Hasegawa, "Performance evaluation of an integrated access schemes in a satellite communications channel," *IEEE Journal on Selected Areas in Communications*, vol. SAC-1, pp. 153 – 164, Jan. 1983.

[32]S. S. Rappaport, "Demand assigned multiple access schemes using collision type request channels - traffic capacity comparisons," *IEEE Trans. Commun.*, vol. COM-27, pp. 1325 – 1331, Sept. 1979.

[33]E. W. M. Wong and T.-S. Yum, "A controlled multiaccess protocol for packet satellite communication," *IEEE Trans. Commun.*, vol. COM-39, pp. 1133 – 1140, July 1991.

[34]S. S. Lam, "Satellite packet communication - multiple access protocols and performance," *IEEE Trans. Commun.*, vol. COM-27, pp. 1456 – 1466, Oct. 1979.

[35]F. A. Tobagi, "Multi-access protocols in packet communications systems," *IEEE Trans. Commun.*, vol. COM-28, pp. 468 – 488, April 1980.

[36]S. Tasaka, "Stability and performance of the R-ALOHA packet broadcast system," *IEEE Trans. Comput.*, vol. C-32, pp. 717 – 726, Aug. 1983.

[37]S. Tasaka, "Multiple-access protocols for satellite packet communication networks - a comparison," *IEEE Proceedings*, vol. 72, pp. 1573 – 1582, Nov. 1984.

[38]W. Feller, *An Introduction to Probability Theory and Its Applications*. New York, NY: J. Wiley and Sons, 1970.

[39]X. Wang, J. Kaniyil, Y. Onozato, J. Lui, S. Shimamoto, and S. Noguchi, "Performance analysis of a combined random-reservation access scheme," *IEEE Trans. Commun.*, vol. 39, pp. 478 – 481, Apr. 1991.

Appendix A Derivation of Conditions for a Policy to Produce an Ergodic Markov Process

In chapter 3 it was demonstrated that the network model under consideration may be formulated as a Markov decision process and thus a Markov process when a specific decision policy is implemented. We wish to show here that for the policies of interest, the Markov process is unichain. This condition must be met for the regular optimality equations to hold and for the modified policy iteration algorithm to be guaranteed to converge to an optimal solution.

A unichain Markov process is one that is composed of a single recurrent class and any number of transient states. A fairly simple way to show that a process is unichain is to show that there exists a state that can be reached from all other states within a finite number of steps. We show in this appendix that the empty state $S_k = (0, 0)$ is such a state for a set of policies that abide by a couple of conditions.

Let us first set the condition that $p < 1$. (Although this condition is not actually necessary for the process to be unichain, the system with $p = 1$ is really not relevant as obviously a reservation protocol would not be appropriate in that circumstance.) Let us say that there occurs a finite consecutive series of frames wherein no new packets arrive. Since the probability of no arrivals in a frame is non-zero, such a series will occur within a finite amount of time. At the beginning of such a series, let the system state be $S = (N_1, N_2)$.

We need simply now to show that the system will empty out in a finite amount of steps when there are no arrivals. Obviously this is the case if at any state,

$$E\left[Y^k\right] + E\left[W^k\right] > 0 \ . \tag{96}$$

From eqn. (37) repeated here,

$$E\left[Y^k\right] = p_s N_1 \left(1 - \frac{p_s}{LV}\right)^{N_1 - 1} \ , \tag{97}$$

we can see that $E[Y^k] > 0$ if $N_1 > 0$ and $L > 0$. Also, since

$$E[W^k] = \min(N_2, F - L) \ , \tag{98}$$

we may say that $E[W^k] > 0$ if $N_2 > 0$ and $L < F$. The sum then is greater than zero for any non-empty state is the following two conditions hold for all non-empty states :

1. If $N_1 = 0$ then $L < F$.
2. If $N_2 = 0$ then $L > 0$.

These conditions do not seriously restrict our choice of policies, and indeed, only exclude degenerate policies which would lock the system into undesirable absorbing states. The vast majority of possible policies and certainly all policies of interest to us fit these criteria and thus produce unichain processes.

Although we have looked here at only the single packet message model, it is not difficult to extend the same argument to the multipacket message model for message length distributions that have finite mean values.

Appendix B Joint Probability Distribution for Observations in the Reservation Subframe when n' Requests are Transmitted

In chapter 4, the problem is encountered of determining the probability that h holes and t successes are observed when n' request packets are transmitted over LV minislots. This problem bears close resemblance to the classic occupancy problem [38] where n' objects are randomly placed in LV urns. Feller provides the probability of exactly h empty urns in [38] as

$$P(h \mid LV, n') = (-1)^h \binom{LV}{h} \sum_{i=h}^{LV} (-1)^i \binom{LV-h}{i-h} \left(1 - \frac{i}{LV}\right)^{n'} . \tag{99}$$

We are left to find joint probability of observing exactly h empty urns and t urns containing exactly 1 object. Applying Bayes rule as follows,

$$P(t, h \mid LV, n') = P(t \mid h, LV, n') \times P(h \mid LV, n') , \tag{100}$$

and using eqn. (99), we are left to find the conditional probability of observing t singly occupied urns given that exactly h urns are empty.

If we consider that when there are exactly h empty urns, we know with certainty that the remaining $LV - h$ urns all have at least one object in them. Discounting the guaranteed single object in each remaining urn from the total number of objects, the problem of finding the conditional probability becomes one of finding the probability of observing t holes when $n' - (LV - h)$ objects are distributed over $LV - h$ urns. This probability may be found from eqn. (99) directly to be

$$P(t \mid h, LV, n') =$$
$$(-1)^t \binom{LV-h}{t} \sum_{i=t}^{LV-h} (-1)^i \binom{LV-h-t}{i-t} \left(1 - \frac{i}{LV-h}\right)^{n'-(LV-h)} . \tag{101}$$

Now using eqns. (99), (100) and (101), and with some manipulations, we find the joint distribution

$$P(t,h \mid LV,n') == (-1)^{h+t}\binom{LV}{h}\binom{LV-h}{t}\sum_{i=t}^{LV-h}\sum_{j=h}^{LV} \left[(-1)^{i+j}\binom{LV-h}{j-h}\binom{LV-h-t}{i-t}\left(1-\frac{j}{LV}\right)^{n'}\left(1-\frac{i}{LV-h}\right)^{n'-(LV-h)}\right] . \tag{102}$$

Appendix C Glossary of Symbols, Acronyms and Abbreviations

$\mathbf{A}$: set of all possible reservation allocations

A_i : the number of unsent packets at the ith station of the global queue (M2 multipacket message model)

a : subframe allocation in frame k

B : action vector for the M2 multipacket message model

C_k : number of minislots containing reservation packet collisions in frame k

C : ideal system capacity

D : overall expected message delay

D_1 : reservation request packet delay

D_2 : data packet delay

DP policy : "Data priority" policy

EOM flag : "End-of-message" flag

F : number of slots per frame

F^k : "flow" in frame k

G : round trip propagation delay

H_k : number of idle minislots in frame k

L : allocation policy mapping the system state, S_k to the allocation L_k

L_{DP} : DP policy

L_{MF} : MF policy

L_f : fixed reservation subframe allocation

L_k : reservation subframe allocation in frame k

L^* : optimal reservation subframe allocation policy

L' : ideal fixed subframe allocation

LAN : local area network

M : number of user stations in the network

m_1 : number of stations in reservation mode at the start of frame k+1

m_2 : number of stations in the global queue at the start of frame k+1

MDP : Markov decision process

MF policy : "Maximum flow" policy

N_1^k : number of stations in reservation mode at the start of frame k

N_2^k : number of stations in the global queue at the start of frame k

n_1 : number of stations in reservation mode at the start of frame k

n_2 : number of stations in the global queue at the start of frame k

$\hat{n}_k$: estimated number of contending stations in frame k

O_k : reservation subframe observation vector for frame k

P^L : state transition probability matrix when allocation policy L is implemented

p : probability that a new message arrives at an idle user in a frame

p_s : reservation request packet transmission probability

$\bar{p}_k$: a priori distribution of the number of contending stations in frame k

$\bar{p}_k'$: a posteriori distribution of the number of contending stations in frame k

R : average message length

$\bar{r}_L$: vector indicating the reward for each state when the action policy L is implemented

S_k : system state at the beginning of frame k

$\mathbf{S}$: state space

T_k : number of minislots containing successful reservation packet transmissions in frame k

V : number of minislots per slot

$\bar{v}$: relative value function vector

$v(i)$: element i of the relative value function vector

VSAT : Very small aperture terminal

W^k : number of message transmissions that are successfully completed in frame k

X^k : number of new messages generated in frame k

Y^k : number of successful reservation requests sent in frame k

Z^k : number of successful packet transmissions in frame k

α : weighting coefficient for derivation of the MF policy

Π^* : limiting distribution of the Markov process

π_S^* : probability for state S in the limiting distribution of the Markov process

$\hat{\lambda}_k$: estimated number of newly arrived packets in frame k

ψ : set of all stationary, non-randomized policies

η^L : system throughput when allocation policy L is implemented

η^* : optimal system throughput