LEVEL

The Pennsylvania State University

The Graduate School

AD A105434

Control of Entry to a Queueing System

A Thesis in

AF17-CI-79-235D

Industrial Engineering and Operations Research

by

Robert Charles Rue

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

November 1979

The signatories below indicate that they have read and approved the thesis of Robert Charles Rue.

Date of Signature:	Signatories:
Sept 26, 1979	Matthew Rosenshine, Professor of Industrial Engineering, Chairman of Committee, Thesis Adviser
Sept. 26, 1979	William E. Biles, Professor of Industrial Engineering, Head of the Department of Industrial and Management Systems Engineering
Sept 28 1979	Milton C. Hallberg, Professor of Agricultural Economics, Chairman of the Committee on Operations Research
Sept 28, 1979	John J. Dinkel, Professor of Management Science, Member of Doctoral Committee
26 Sept. 1979	David L. Raphael, Professor of Industrial Engineering, Member of Doctoral Committee
Sept. 27 1979	Thomas A. Ryan, Jr., Associate Professor of Statistics, Member of Doctoral Committee

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER 79-231D	2. GOVT ACCESSION NO. AD-A105 434	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Control of Entry to a Queueing System		5. TYPE OF REPORT & PERIOD COVERED Dissertation
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Capt Robert Charles Rue		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS AFIT STUDENT AT: The Pennsylvania State University		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS AFIT/NR WPAFB OH 45433		12. REPORT DATE 1979
		13. NUMBER OF PAGES 237
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASS
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

29 SEP 1981

Fredric C. Lynch

FREDRIC C. LYNCH, Major, USAF
Director of Public Affairs
Air Force Institute of Technology (ATC)
Wright-Patterson AFB, OH 45433

18. SUPPLEMENTARY NOTES

APPROVED FOR PUBLIC RELEASE: IAW AFR 190-17

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

ATTACHED

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT

This thesis is concerned with the control of entry to queueing systems. An M/M/1 model with a single class of customers and an infinite time horizon studied by Naor (1969) provides the starting point for this work. Each customer receives a fixed reward for service and pays a holding cost at a fixed rate per unit of time he spends in the system. Each customer may choose to join the system or not. A self-optimizing customer decides whether or not to join by acting to maximize his own expected net benefit. A social optimizing customer decides by acting to maximize the gain rate of the system, the sum of the expected net benefits per unit time of all arrivals. Socially optimal control of the model is provided by establishing a balking point for the customers that is no greater than the balking point a self-optimizing customer would determine. Naor's approach is compared with the semi-Markov decision process formulation of Yechiali (1971). The results for this first model are extended to show that the gain rate can only increase as the arrival rate of the customers increases while the social balking point can only decrease as the arrival rate increases.

A semi-Markov approach is used to formulate an expanded model with several classes of customers, each with its own reward and holding cost rate. Socially optimal control of this model is shown to be provided by establishing a balking point for each class that is no greater than the balking point a member of the class would determine if he acted to maximize his own expected net benefit.

A semi-Markov approach is also used to extend the several class model to include Erlang service times. A heuristic solution technique based on policy iteration and a solution technique using mixed integer programming are presented.

The several class models are applied to the problem of determining an optimal policy for controlling the entry of commercial aircraft to the landing queue at the Greater Pittsburgh International Airport. A socially optimal control policy is found and analyzed.

Finally, the semi-Markov approach is used to formulate three other models:

1) A nonpreemptive priority model.
2) A class dependent service rate model.
3) A nonpreemptive priority model with class dependent service rates.

TABLE OF CONTENTS

Page

ABSTRACT iii

LIST OF TABLES viii

LIST OF FIGURES x

ACKNOWLEDGMENTS xi

I. INTRODUCTION 1

1.1 Problem Statement 1

1.2 Outline of the Dissertation 5

II. LITERATURE SURVEY 10

2.1 Control of Arrivals 10

2.2 Control of Server 17

2.3 General 19

III. THE SINGLE CLASS MODEL 21

3.1 Model I 21

3.2 Naor's Approach 22

3.3 Yechiali's Approach 24

3.4 Comparison of Results 28

3.5 Linear Programming Formulation of the Social Optimum Problem 32

3.6 g and n_o as Functions of λ 35

3.7 Tolls 42

3.8 Conclusion 44

IV. THE SEVERAL CLASS MODEL 45

4.1 Model II 45

4.2 Individual Optimum Problem 45

		Page
	4.3 Social Optimum Problem	48
	4.4 Equivalence of the Three Formulations	54
	4.5 The Form of the Optimal Solution	61
	4.6 Policy Iteration Solution of the Social Optimum Problem	65
	4.7 Linear Programming Solution of the Social Optimum Problem	73
	4.8 Conclusion	78
V.	ERLANG SERVICE TIMES	79
	5.1 The Erlang Density Function	79
	5.2 One Class Example Problem	82
	5.3 Model III	84
	5.4 The Individual Optimum Problem	85
	5.5 Social Optimum Using Policy Iteration	90
	5.6 Social Optimum Using Mixed Integer Programming	93
	5.7 Conclusion	100
VI.	AIRPORT LANDING QUEUE APPLICATION	102
	6.1 Customer Parameters	102
	6.2 Server Parameters	107
	6.3 Deterministic Service Time Model	116
	6.4 Results for Model II	122
	6.5 Results for Model III	128
	6.6 Conclusion	134
VII.	OTHER MODELS	137
	7.1 Nonpreemptive Priority Service Discipline Model	137
	7.2 Class Dependent Service Rate Model	151

Page

7.3 Nonpreemptive Priority Service Discipline Model with Class Dependent Service Rates 154

7.4 Conclusion 164

VIII. SUMMARY, CONCLUSIONS, AND SUGGESTIONS FOR FURTHER RESEARCH 166

8.1 Summary and Conclusions 166

8.2 Potential Uses of the Model 170

8.3 Suggestions for Further Research 172

CITED REFERENCES 175

APPENDIXES

A. GLOSSARY OF NOTATION 178

B. PROOFS OF SOME ASSERTIONS ABOUT MODEL I 184

B.1 Proof of Property One 184

B.2 Proof of Property Two 186

B.3 Proof of Property Three 190

B.4 Demonstration of Equation (3.5) 191

C. INTRODUCTION TO MARKOV DECISION PROCESS 192

C.1 Markov Decision Processes 192

C.2 Semi-Markov Decision Processes 196

C.3 Example Problem 198

D. USER'S GUIDE AND LISTING OF THE PROGRAM 205

D.1 Introduction to the Program 205

D.2 Use of the Program 208

D.3 Other Programs 210

D.4 Listing and Output 211

LIST OF TABLES

Table		Page
1.1	FAA Limits for High Density Traffice Airports IFR Operations Per Hour [FAA (1974)]	4
3.1	Policy Iteration Results for Various Values of λ for a One Class Example with $R = 5$, $C = 2$, and $\mu = 3$	40
4.1	Two Class Example Problem	76
6.1	Aircraft Categorized by Class	103
6.2	Percentage of Yearly Jet Traffic Represented by Each Class and Approximate Mean Arrival Rates for Each Class in a Busy Period	104
6.3	Estimates of Direct Operating Cost Per Hour of Flight for Each Class of Aircraft	106
6.4	Cost Per Hour of Flight for Each Class	106
6.5	Seating Capacity and Within Class Weights for the Aircraft in Each Class	108
6.6	Average Seating Capacity of Each Class	109
6.7	Reward Per Flight for Each Class	109
6.8	Observed Service Times (Pittsburgh)	111
6.9	Observed and Expected Class Frequencies When an Erlang ($\mu = 33.15$, $h = 19$) Distribution is Hypothesized	114
6.10	Observed and Expected Class Frequencies When an Erlang ($\mu = 33.15$, $h = 20$) Distribution is Hypothesized	115
6.11	Observed Service Times (Washington National	117
6.12	Individual Optimum Balking Points	118
6.13	Results for Several Erlang Models of the Example	122
6.14	Results for the Exponential ($\mu = 33.15$) Model	123
6.15	Results for the Exponential ($\mu = 33.15$) Model for Several Nonoptimum Policies	125
6.16	Balking Points and Gain Rate for Various Values of μ	126

Table		Page
6.17	Balking Points and Gain Rate for Various Values of λ_5	127
6.18	Balking Points and Gain Rate for Various Values of R_5	127
6.19	Phase Optimum Balking Points and Gain Rate for Several Low Erlang Number Models	129
6.20	Candidate Forced Balking Points for Several Linked Erlang Models	129
6.21	Best Results for Several Linked Erlang Models	131
6.22	Parameters for the Single Composite Class Model	132
6.23	Policy Iteration Phase Optimum Results for Several Models of Both a Single Composite Class and the Full Five Classes	133
7.1	Parameters for the Example	159

LIST OF FIGURES

Figure		Page
1.1	Flow Diagram of Dissertation	6
5.1	Erlang Density for Several Values of h	81
D.1	Flow Diagram of POLIT	207

ACKNOWLEDGMENTS

The author expreses his gratitude to Dr. Matthew Rosenshine, chairman of his committee, for inspiration and guidance during his work at The Pennsylvania State University.

The author is also grateful to Dr. J. J. Dinkel, D. L. Raphael, and Dr. Thomas A. Ryan, Jr. for their contributions to his education and to this thesis.

Support for the author's graduate work has been provided by the United States Air Force through the Air Force Institute of Technology.

CHAPTER I

INTRODUCTION

Until quite recently, queueing theory has been used primarily as a descriptive tool. In this mode, an existing or proposed queueing system is "allowed to operate" and its behavior is described by one or more measures such as expected waiting time and expected queue length. Currently, an increasing portion of the queueing literature is being devoted to the use of queueing theory to control and optimize the operation of a system. Here, queueing analyses are used to design a system or to develop an operating policy to control its operation so that certain standards are met. For instance, a policy might be sought to keep operating costs below some upper bound while maximizing throughput of the queue. This more recent approach of designing or controlling a queueing system based on an optimum operating policy is adopted throughout this work. The policies sought are those which maximize gain per unit time.[1]

1.1 Problem Statement

Suppose that customers from M homogeneous classes arrive at a service facility with a single server. Th classes are homogeneous in in the sense that each customer in a particular class, m, faces the same cost structure at the service facility, namely, a reward R_m for

[1]While gain can be given a broad range of interpretation, it is perhaps easiest to think of gain as financial gain, although the term is not restricted to dollars.

service and a cost C_m per unit time the customer spends in the system. Suppose also that each customer may either join the system or not. Many economic viewpoints or objectives can be used to make the join or balk decisions. Two viewpoints of interest are that of an individual customer and that of the group of customers or society acting as a whole.

The first viewpoint gives rise to the individual optimum problem in which each arriving customer makes his own decision whether or not to join the system so that his own expected net benefit is maximized. Arriving customers are assumed to be able to determine the state of the system, e.g., the number of customers in the system. A customer's expected net benefit for joining when i customers are in the system is his reward R_m minus C_m times his expected time in the system, given that i customers are in front of him. Unless otherwise stated, the service discipline used in the models is first come, first served. A customer's net benefit for balking is zero.

The other viewpoint produces the social optimum problem where the decision of whether or not to allow an arrival to join the system is made by a single decision maker who seeks to maximize the sum of the expected net benefits per unit time of all arriving customers. One way to view this decision maker is to think of him as an administrator who collects all rewards and pays all the costs for the customers who use the system, and then distributes the gains among all arrivals whether or not they joined the system. While solution of the social optimum problem is the major goal, the solution of the individual optimum problem is also obtained and used to limit the search for the solution to the social optimum problem.

The solution to the social optimum problem is an optimum policy (i.e., a join or balk decision for each class of customers, for each possible state of the system). Several assumptions about the arrival distribution of the customers and the service time distribution of the server are required to solve the problem in a straightforward manner. These assumptions are noted in the outline of the dissertation and are explained more fully in the various chapters.

For several of the models presented in this paper, optimal control of the entry of customers into the system is shown to be provided by a set of balking points $\bar{n} = (n_1, n_2, \ldots, n_M)$, where class m joins if the state of the system is less than n_m. For all the models presented in this paper, the balking point for a self-optimizing customer of class m is shown to be at least as large as that of customers of class m acting in a socially optimal manner. Thus, compared with socially optimal behavior, self-optimizing customers tend to overcongest the system.

The particular application to which the results obtained will be applied is the landing queue of an airport. It should be noted, however, that applications as diverse as deciding on the number of skiers to allow at a ski resort on a holiday and determining the number of terminals allowed to tie into a computer system can be molded into queueing system entry control problems. In the landing queue at an airport, various types of commercial aircraft are the classes of customers in the landing queue system. For each class, R_m is a measure of the gain when a class m aircraft lands. C_m is the cost per unit time of keeping an aircraft of class m in the landing queue, including such costs as crew wages and fuel.

Entry of aircraft into an airport's landing queue is currently being controlled in several manners for safety reasons. At all major airports, the air traffic controllers maintain an upper bound on the size of the landing queue. If the queue is full, arriving aircraft are denied entry into the landing queue and are kept under the command of controllers in another air traffic sector. Also, at certain airports designated as high density traffic airports, the Federal Aviation Administration (FAA) limits the number of instrument flight rule (IFR) operations (take offs and landings) per hour allowed during peak traffic periods. These limits which are set by negotiation within the industry are given in Table 1.1. It is not necessary that aircraft in flight which are not admitted to the landing queue be forced to land elsewhere. In practice, they could wait if they choose. The model, however, is useful in determining the capacity of an airport landing queue from an economic viewpoint. Thus, the entry of aircraft into the landing queue is controlled in the model for the purposes of defining an economical workload for the airport or, more appropriately, an economical airport for the workload. The former can be accomplished by schedule modifications, the latter by airport design.

TABLE 1.1

FAA Limits for High Density Traffic Airports
IFR Operations Per Hour [FAA (1974)]

	Airport				
Class of User	JFK	La Guardia	Newark	O'Hare	Wash. National
Air Carrier	70	48	40	115	40
Air Taxi	5	6	10	10	8
Other	5	6	10	10	12

The models developed in the first few chapters of this paper will be used to determine the capacity of the system (from an economic viewpoint). Since the aircraft are divided into several classes, it is possible that the capacity of the system may vary with the class of an arrival since, for certain states of the system, the administrator may admit members of some classes but not of others. When a decision concerning the direction the development of a model is to take is made, the appropriateness of the direction to the landing queue application is the criterion on which the choice is based.

1.2 Outline of the Dissertation

Following this chapter, Chapter II presents a literature survey of selected works in the area of control of entry to queues. A few works on the control of the server in a queueing system and other general topics that serve as background for this work are also included. Two of these topics are air traffic control and the estimation of the parameters of a probability distribution function. As indicated in the flow diagram for this dissertation, Figure 1.1, the flow of the dissertation is not interrupted by skipping Chapter II.

Model I, the simplest model used in this study, is presented in Chapter III. A single class of customers whose arrival forms a Poisson stream to a single server with exponentially distributed service times is modeled. Thus, this model is an M/M/1/n queueing system where the capacity of the system, n, is the decision variable. The capacity of the system, n, is also referred to as the balking point of the customers; that is, if n (or more) customers are in the system, an arrival balks at the opportunity to join.

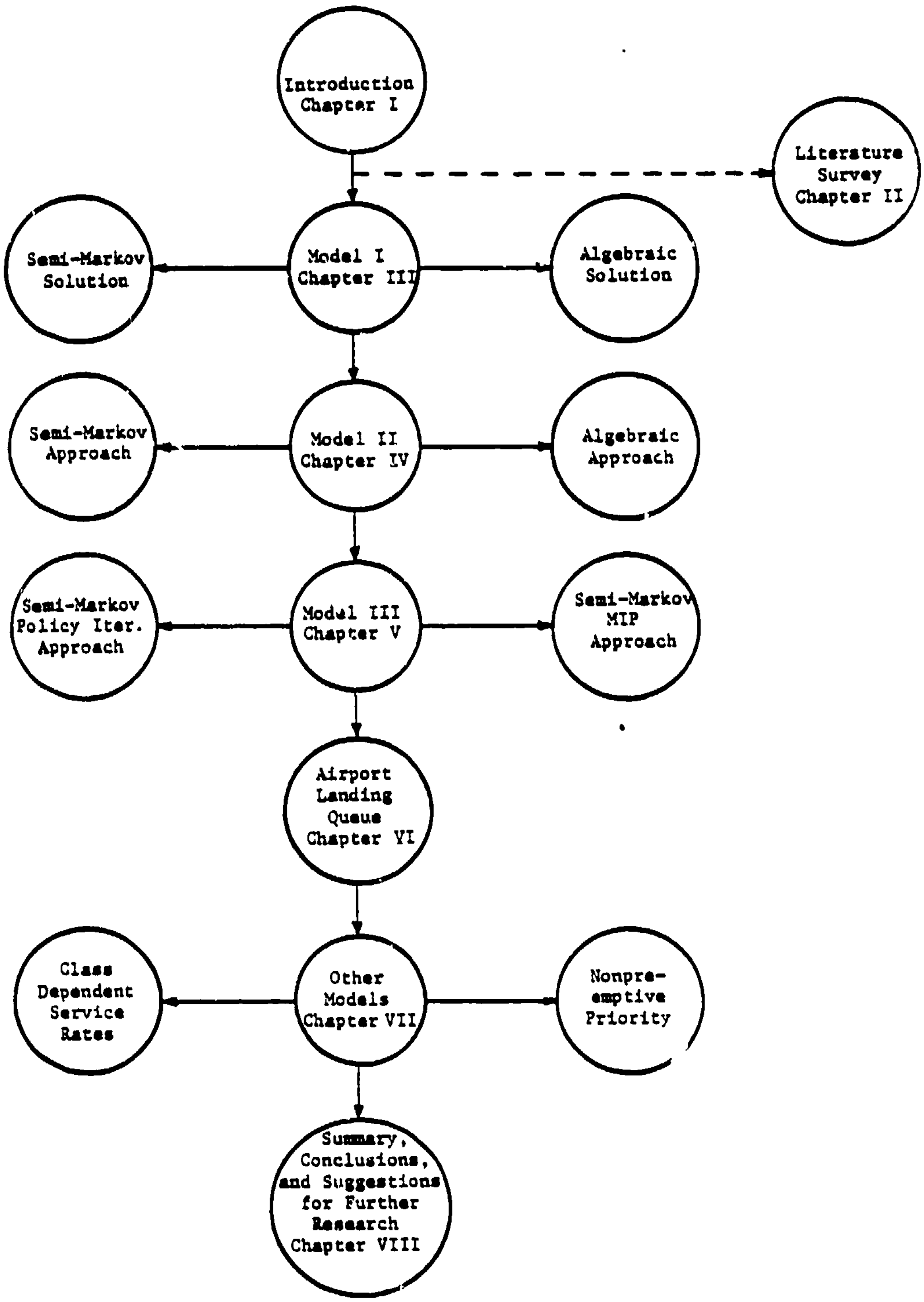

Figure 1.1 Flow Diagram of Dissertation

Model I has been solved previously by Naor (1969) and Yechiali (1971) for the individual and social optimum values of n. Naor developed his results using algebra and the properties of the model, while Yechiali developed his results by treating the problem as a semi-Markov decision process. Chapter III compares the work of the two authors and extends it somewhat. A few sidelights such as tolls (which both authors studied) are examined as one means of implementing the solution to the social optimum problem. The toll charged to a customer who joins the system alters his net benefit so that self-optimizing customers find it desirable to join the system only when a social optimizing administrator would have let them join.

In Chapter IV, M classes of customers are considered. Each class m has its own reward for service R_m, cost per unit time in the system C_m, and mean arrival rate λ_m, but all have the same service rate μ. The work of Naor (1969) and Yechiali (1971) is extended to this model. In particular, the form of the optimal policy carries over to this new model, Model II. The decision variable in this model is a policy, a set of join or balk decisions for each class of customers, for each possible state of the system. Recall that the number of customers in the system is the state of the system. The balking point of Model I now becomes a vector $\bar{n}$ of balking points for each class. This implies that some classes of customers may receive preferential treatment in the operation of the system. Policy iteration and linear programming solution techniques are presented for the semi-Markov decision process formulation of the problem.

The service time distribution of the M class model is generalized from an exponential to an Erlang distribution in Chapter V. This change allows more flexibility in modeling the service time distribution of an actual system. Again, the model is structured as a semi-Markov decision process. One solution method developed, based on policy iteration, is easy to implement but does not guarantee an optimal solution. It assumes that the optimal solution is again a vector of balking points and performs a limited search for a good solution. A second method permits an exhaustive search of all possible policies through a mixed integer programming formulation.

As indicated in Figure 1.1, Chapters III through V cover the three major models developed in this dissertation. Model II and the Erlang model, Model III, are applied to a sample airport landing queue problem in Chapter VI. Airline data, FAA data, and data taken at the Greater Pittsburgh International Airport and Washington National Airport are used in Chapter VI to estimate the parameters required for the models. The models are then used to determine how entry to the landing queue should be controlled to maximize the social optimum objective function. The sensitivity of the results to the input parameters is also examined.

In Chapter VII, three variations are developed for Model II. First, the service discipline is changed from first come, first served to a nonpreemptive priority service discipline. The second variation expands the flexibility of the model by allowing different mean service rates for each class of customers. The last variation combines the first two.

A summary of the entire dissertation and the conclusions reached is presented in Chapter VIII along with a list of potential uses for the models developed in the earlier chapters. Suggested areas for future research are also presented.

There are four appendices to this paper. A glossary of the notations used in the paper is presented in Appendix A to facilitate reading the more mathematical sections of this dissertation. Appendix B contains proofs of some of the work of Naor (1969) and an example problem solved using his work. A brief introduction to semi-Markov decision processes and policy iteration along with an example problem illustrating the use of policy iteration comprise Appendix C. Appendix D is a listing and user's guide for a computer program that solves Models I, II, and III.

CHAPTER II

LITERATURE SURVEY

This chapter is a survey of selected works that influenced the writing of this paper. The works are divided into three areas: 1) control of arrivals, 2) control of server, and 3) general. A short summary of the papers in each group is presented in alphabetical order by author's name.

2.1 Control of Arrivals

Balachandran and Schaefer (1975) examine an M/G/1 queueing model with several classes of customers. Each class has its own reward for service, cost per unit time in the system, mean service rate, and mean arrival rate which the class adjusts based on expected waiting time. The individual optimum policy allows the service facility to be dominated by a single class. The social optimum also admits only one class, but it may not be the same class as the individual optimum. Various techniques are explored for diversifying access to the facility.

Balachandran and Schaefer (1976) consider an M/M/1 queueing model with K classes of customers where each class adjusts its own arrival rate based on average waiting time in the queue. Each customer of class i receives a reward g_i for service and loses v_i for each unit of time spend in the system. If Λ^o is the optimal aggregate arrival rate, the average waiting time is $W = 1/(\mu - \Lambda^o)$. The service facility is used only by those classes for which $g_i \geq v_i/(\mu - \Lambda^o)$. The authors introduce admission prices to equalize the attractiveness of the facility among the classes.

Balachandran and Tilt (undated) consider decision making in queueing systems with customers given the choice of 1) joining, 2) balking, or 3) choosing a priority through payment made on arrival. Several models are examined from the standpoint of a noncooperative n person game. One of the models treated yields an individual optimum for the GI/M/s/n queue which is analogous to that obtained by Yechiali (1972).

Edelson and Hildebrand (1975) reexamine the work of Naor (1969) and extend it to several classes of customers. However, the authors are interested in the relationship between the social optimum and the revenue maximizer's optimum. They first examine conditions under which the two are the same. Then, they examine a generalized model which includes several classes of customers, each with its own reward and cost per unit time in the system. They give a computational technique for finding the expected number of each class of customers in the system. In addition, they show that for the generalized case, the revenue maximizer's optimum balking point is not necessarily less than the social optimum (which it was for Naor's model).

Emmons (1972) considers an M/M/s queueing system with the following cost structure:

a) A fixed running cost rate.

b) An expected revenue per customer, r.

c) An overtime running cost rate called K.

The system is run for a finite length of time with no customers admitted after closing. A policy of admitting customers as a function of number in the system and time to closing is sought which maximizes the operator's total expected profit. The optimal policy admits customers only if the number in the system, i, is such that $f_1(t) \leq i \leq f_2(t)$, where $f_1(t)$ and $f_2(t)$ are given. If μ is the service rate and $r \geq K/(c\mu)$, then $f_2(t) = \infty$, so that customers are always admitted when $f_1(t)$ or more are in the system. This rule can be transformed into an optimal rejection time rule. For $r < K/(c\mu)$, $f_2(t) \neq \infty$ and the rule cannot be transformed into an optimal rejection time rule.

Harrison (1975) studies an M/G/1 queueing system with K classes of customers. Each class has its own arrival rate, reward for service, holding cost, and service time distribution function. All arrivals are allowed to join the queue, but an administrator decides at the completion of each service which class, if any, to admit next to service. The objective of the administrator is to maximize the sum of the discounted net benefits of all customers in the system. A nonpreemptive priority discipline (that may ignore several classes) is shown to be optimal.

Knudsen (1972) considers an M/M/s/n queueing system. He generalizes the work of Naor (1969) to s servers and to a nonlinear waiting cost function. The author shows that if the net benefit to each arriving customer is a decreasing concave function of the number in the system, then Naor's results hold; that is, the revenue maximizer's optimum balking point is less than or equal to the social optimum which is in turn less than or equal to the individual optimum. The author also discusses the shadow price aspect of the tolls charged to customers to get them to act in a socially optimal manner. Finally, he discusses the difficulties of pricing resources in a stochastic situation.

Lippman and Stidham (1977) examine a queueing system with an arrival rate λ and a service rate μ_i that is a nondecreasing, concave, bounded above function of i, the number in the system. Typically, the sequence μ_i might arise from an M/M/s system where $\mu_i = \mu \cdot \min(s,i)$. Associated with each customer accepted is a reward r which is a random variable. An accepted customer joins the queue and incurs a waiting cost of h per unit time until he departs. The system may either be controlled by society as a whole or by the individual customers. Control of the system is exercised to maximize expected discounted net benefits over an infinite or finite horizon. The paper compares individual and social optima and shows that regardless of system state, remaining horizon, or discount rate, a customer left to his own devices will enter the system whenever the social optimum calls for him to enter. In addition, the customer may enter when the social optimum calls for him to balk. The authors claim that this discrepancy is caused by the failure of an individual customer to consider the shortfall in benefits to later arriving customers caused by his joining the queue. The paper also examines the behavior of socially and individually optimal policies and returns as functions of i, n, and the discount rate. Finally, the paper discusses tolls that should be charged to get individuals to act in a socially optimal manner.

Littlechild (1974) considers an M/M/1 queueing system with customers that have the same cost of waiting but different rewards for service. He develops a toll to charge all customers to reduce the arrival rate so that social benefit is maximized. In other words, the reduction in waiting cost due to the smaller arrival rate more than offsets the loss in benefits of service to the customers as a whole.

Miller (1969) examines an M/M/s/s queueing system with K classes of customers. Each class has its own reward for service and arrival rate, but all classes have the same service rate. When a customer arrives, a decision is made whether or not to serve him. If the decision is made not to serve, the customer departs; thus, there is no queue and no preemption. The objective of the paper is to find the admittance policy which maximizes the average value of rewards over an infinite planning horizon. When the system is in state j, j servers are free and a policy such as serve customers of classes {1, 2, 3} may be chosen. The problem is formulated as a semi-Markov process and policy iteration is used to solve for an optimal policy. The paper also describes two heuristic methods for examining a generalized problem that allows each class to have its own general service time distribution.

Mine and Ohno (1971) consider an M/G/1 queueing system in which the number of customers in the system is unavailable or too expensive to maintain. Arriving customers are accepted during the time interval (t_o, t'), starting time to rejection time, and rejected with compensation after time t'. The server runs at a cost rate r_o during normal hours, (t_o, T), where T is closing time (note that $t_o \leq t' \leq T$). The server runs at increased rate r_1 after T (overtime) and at reduced rate r_o' if he becomes idle after t' (goes home). The paper finds the rejection time that minimizes total expected cost.

Naor (1969) examines an M/M/1 queueing system in which customers receive a reward R for service and pay a cost of C per unit time in the system. The arrival rate is λ and the service rate is μ. Each arrival may choose either to join the system or not. Three types of objective functions are considered. In the first, the individual optimum problem, each customer seeks to maximize his own expected net gain. It is optimal to balk when n_s customers are in the system, where n_s is the greatest integer in $(R\mu/C)$. This is simply an $M/M/1/n_s$ system. The second optimization considers the collective good of all arrivals given by the sum of their expected net benefits per unit time. $n_o \leq n_s$ is now the maximum number allowed in the system. Reduction to n_o can be done either by administrative rule or by levying a toll on entering customers to reduce n_s to n_o. The third optimization maximizes the revenue of the toll collecting agency. $n_r \leq n_o \leq n_s$ is the maximum number allowed in the system. Thus, if left on its own, the toll collecting agency levies tolls that are too high for the social good.

Prabhu (1974) considers the problem of finding an optimal stopping time for an M/G/1 queueing system with a constant arrival rate. The stopping time is chosen to maximize expected discounted profit when the cost structure is as follows:

a) A revenue per unit time.

b) An operating cost per unit time in $(0, t_o)$, where t_o is the stopping time.

c) An operating cost rate in (t_o, ∞).

Finding the optimal stopping time is based on an infinitesimal look-ahead rule which can be described as follows. Suppose the system has been operating to time t and a profit $f(t, W_t)$ has been made, where W_t is the remaining workload. It is profitable to continue the operation up to time t + h if $E\{f(t + h, W_{t+h})\} > f(t, W_t)$.

Stidham (1978) considers the problem of accepting or rejecting arrivals at a GI/M/1 queueing system. The holding cost is convex in the number of customers in the system and the reward for service is a random variable. Finite and infinite horizon problems with and without discounting are considered. A terminal reward or cost is allowed in the finite horizon problems. The author shows that a socially optimal policy is less likely to accept a customer than an individually optimal policy.

Stidham and Prabhu (1974) examine the work that has been done in control of queueing systems. The authors show how several of the works are related and point out some generalizations that apply to most of the papers. For instance, they note that most research has sought to determine 1) when a stationary policy is optimal, 2) its form, and 3) the value of its parameters.

Yechiali (1971) considers a GI/M/1 queueing system. The cost structure includes a reward for service, a service charge for each customer served, a cost per unit time in the system, and a charge for balking. The author finds an individual optimum that is analogous to that of Naor (1969). He treats the social optimum problem as a semi-Markov decision process and shows that a control-limit rule is optimal; that is, the administrator will operate the queue as a finite capacity queue. As Naor did, he shows that the social optimum is less than or equal to the individual optimum balking point. He then formulates a linear program equivalent to the social optimum problem. The author also considers the revenue maximizer's optimum.

Yechiali (1972) extends the results of Yechiali (1971) to a GI/M/s queueing system. The author shows that the social optimum balking point is less than or equal to the individual optimum using the same techniques as in his earlier paper.

2.2 Control of Server

Balachandran (1971) considers an M/G/1 queueing system with two classes of arrivals. Class one customers are of higher priority than class two customers. When N class one customers are in the queue, the service of a class two customer is preempted until all N class one customers are served. Service of the class two customer is then resumed without loss of service accomplished. The value of N is sought which minimizes a linear cost function involving the expected number of customers of each type and the expected preemption rate.

Crabill (1972) examines an M/M/1 queueing system with a constant arrival rate and a service rate μ_i where $\mu_i \varepsilon \{\mu_1, \ldots, \mu_K\}$. A policy for choosing the service rate as a function of queue length is sought to minimize the long-run average expected cost rate of the queue. The cost expression involves a customer inconvenience cost rate which depends on the number in the system and a service cost rate which depends on the service rate used. The optimal policy is shown to be a set of k + 1 numbers which specify the range of values of number in the system for which each service rate is to be used.

Heyman (1968) considers an M/G/1 queueing system with the following cost structure:

a) Dormant cost rate.

b) Running cost rate.

c) Start up cost.

d) Shut down cost.

e) Holding cost rate.

The purpose of the paper is to find the policy that minimizes the total expected operating cost from among all possible policies of turning the server off or on during the operating horizon. For the undiscounted infinite horizon model, the optimal policy is shown to be either the server is always running or the server is turned on when n customers are in the system and off when the system is empty. For small interest rates, the optimal discounted policy is approximately the same as the optimal undiscounted policy. The author gives a recursive relation for determining the optimal policy for a finite horizon.

Zacks and Yadin (1969) examine an M/M/1 queueing system with arrival rate λ and service rate μ, where μ can be chosen from $(0,\mu^*)$. The following cost structure is considered:

a) Holding cost rate.

b) Cost of switching service rate.

c) Service cost rate.

The policy specifying service rates which minimizes discounted costs over an infinite horizon is sought. The optimal control times are shown to occur immediately after a change in queue size. For the special case of linear holding costs, switching costs that depend only on the absolute value of the change in service rate, and convex service cost rates with bounded derivatives, the optimal control policy is of the form:

a) Increase μ to $\underline{\eta}(x)$ if $\mu < \underline{\eta}(x)$.

b) No change if $\underline{\eta}(x) \leq \mu \leq \overline{\eta}(x)$.

c) Decrease μ to $\overline{\eta}(x)$ if $\mu > \overline{\eta}(x)$.

x is the number in the system and $\underline{\eta}(x)$ and $\overline{\eta}(x)$ are service rates associated with a given number, x, in the system.

Zacks and Yadin (1970) examine the special case of their earlier paper in a little more detail. Some additional properties of the optimal policy are established and a numerical example is provided.

2.3 General

Dear (1976) considers the problem of scheduling the landing of aircraft. He constrains the capability of the scheduler so that he can move an aircraft no more than a given number of positions from its first-come first-served position. These constraints keep the schedule feasible in that changes caused by a new arrival will be relatively minor and any given aircraft will land reasonably close to its first-come first-served time. Yet the constraints are loose enough to allow considerable improvement in runway utilization over the first-come first-served schedule.

Hodgson and Koehler (1978) examine the solution of Markov decision processes by policy iteration when the systems require large numbers of states. The authors give a procedure for transforming finite state, continuous time Markov or semi-Markov decision processes so that an approximation algorithm by White and Odoni can be used.

Koopman (1972) considers queues of aircraft awaiting landing at a single runway where there is a fixed maximum queue length. Arrivals are assumed to form a time dependent Poisson process and service times are either time dependent exponential random variables or time dependent constants. A periodic solution is assumed for the probability of n aircraft in the system; that is, $P_n(t + T) = P_n(t)$. Numerical results are given for several sets of values of the parameters. For the cases run, the author notes that results such as expected number in the system were remarkably insensitive to the form of the distribution assumed for service times.

White, Schmidt, and Bennett (1975) present in Chapter VII the statistical techniques necessary for modeling the components of queueing systems. Modeling the arrival and service distributions are of major interest. The authors give the theory and examples for:

a) Selecting candidate distributions.

b) Determining numerical values for the parameters of the distributions.

c) Testing the hypothesis that the chosen distribution is the true distribution.

CHAPTER III

THE SINGLE CLASS MODEL

This chapter defines Model I, the single class of customers model. The methods that Naor (1969) and Yechiali (1971) used to solve for the social optimum are presented and their approaches are compared. The model is also formulated as a linear program following Yechiali (1971). Then, the effect of changing the arrival rate on the solution is investigated. Finally, entry tolls are used to impose the social optimum solution on individual customers.

3.1 Model I

Model I has a single class of customers. The arrivals form a Poisson stream with mean rate λ. The service times of the single server are independent, identically distributed, exponential random variables with mean $1/\mu$. Thus, this model is an M/M/1 queueing model. The following cost structure is imposed on the operation of the queueing system:

a) Each customer served receives a reward of R dollars.

b) Each unit of time a customer spends in the system costs him C dollars.

Suppose that each arriving customer is given the choice of joining the queue and receiving reward R and paying C per unit time in the system or of not joining and not paying or receiving any money. In this and the other models to be considered, customers are assumed to decide by comparing the expected net gain associated with each decision and choosing the action with the larger gain. (In case of a tie, the customer joins the queue.) This model is considered by Naor (1969) and Yechiali (1971), although Yechiali allows a general arrival distribution and a slightly more general cost structure.

3.2 Naor's Approach

After detailing the assumptions and structure of the model, Naor argues that all reasonable strategies lead to a finite capacity queue. The task remaining then is to determine the optimal capacity, n. Naor first deals with determining $n = n_s$ under self-optimization where each customer considers only his own expected net gain in deciding whether or not to join the queue. The expected net gain for joining is $R - (i + 1)C/\mu$, where i is the number of customers the arrival finds in the system. Joining the queue serves the self-interest of a customer if i is less than n_s, where n_s is such that

$$R - (n_s + 1)C/\mu < 0 \leq R - n_s C/\mu \quad .$$

This strategy leads to an $M/M/1/n_s$ queueing system where

$$n_s = [R\mu/C] \quad . \tag{3.1}$$

The brackets indicate that n_s is the greatest integer in $R\mu/C$. Thus, the capacity of the queue or the balking point is n_s for the self-optimization problem.

If each customer or an administrator acts to maximize the sum of the individual net benefits, the problem becomes the social optimum problem. Considering an infinite horizon without discounting, Naor sets up an expected overall net benefit rate function. In notation based on Gross and Harris (1974), the expected net benefit rate is as follows:

$$g(n) = \lambda'(n)R - CL(n) \quad , \tag{3.2}$$

where $\lambda'(n)$ and $L(n)$ are, respectively, the effective arrival rate and the expected number in the system when a maximum of n customers is allowed in the system. $g(n)$ is the expected net benefit rate or expected rate of gain when a maximum of n customers is allowed in the system. The units of $g(n)$ are dollars per unit time. Setting $\rho = \lambda/\mu$ and using formulas for $\lambda'(n)$ and $L(n)$ that are available in Gross and Harris (1974) yields:

$$g(n) = \lambda R\{1 - \rho^n(1 - \rho)/(1 - \rho^{n+1})\} - C\{\rho/(1 - \rho) - (n + 1)\rho^{n+1}/(1 - \rho^{n+1})\} \quad . \tag{3.3}$$

The administrator wants to choose n to maximize $g(n)$. $g(n)$ is discretely unimodal in n (proof given in Appendix B), which implies that a local maximum is a global maximum. Thus, the administrator needs to find $n = n_o$, such that:

$$\Delta g(n_o + 1) < 0 \leq \Delta g(n_o) \quad ,$$

where $\Delta g(n) = g(n) - g(n - 1)$. Naor shows that n_o satisfies

$$\{n_o(1 - \rho) - \rho(1 - \rho^{n_o})\}/(1 - \rho)^2 \leq R\mu/C < \{(n_o + 1)(1 - \rho) - \rho(1 - \rho^{n_o+1})\}/(1 - \rho)^2 \; . \tag{3.4}$$

For continuous variables $v_s = R\mu/C$, and v_o related by

$$\{v_o(1 - \rho) - \rho(1 - \rho^{v_o})\}/(1 - \rho)^2 = v_s \quad , \tag{3.5}$$

Equation (3.4) leads to $v_o \leq v_s$ and, since v_s increases with v_o, $n_o = [v_o] \leq n_s = [v_s]$. (Proofs of these assertions also appear in Appendix B.)

Lippman and Stidham (1977) and others assert that $n_o \leq n_s$ is due to an individual optimizer's failure to consider the decrease in benefits to later arriving customers caused by his joining the queue. This type of effect is called an external economic effect. Considered another way, the difference between the social optimum formulation and the individual optimum formulation is that the social optimum formulation includes arrival rate information. This information allows the administrator to anticipate expected benefits from customers who have yet to arrive.

The queue capacity of the individual optimum problem serves as an upper bound on that of the social optimum problem. This bound is useful in limiting the number of possible solutions that need to be considered when approaching the problem as Yechiali does. The optimal capacity n_o of the social optimum problem is found by solving Equation (3.5) for v_o, then setting $n_o = [v_o]$. Thus, Naor presents solutions to both the individual and social optimum problems.

3.3 Yechiali's Approach

Although Yechiali (1971) considers a general interarrival time distribution and a slightly more general cost structure than Model I, this discussion of his results is couched in the terminology and form of Model I. First, a few terms and concepts need to be defined. η_n is the number of customers in the system at the instant of the n^{th} arrival. $\{\Delta_n\}$ is the sequence of successive decisions made by arriving customers, where $\Delta_n = 0$ if the n^{th} customer balks, and $\Delta_n = 1$ if he joins the queue. $\{\eta_n, \Delta_n\}$, $n = 1, 2, \ldots$ is a semi-Markov decision process. (See Appendix C for a brief introduction to semi-Markov decision processes.) $H_m = \{\eta_1, \Delta_1, \ldots, \eta_m, \Delta_m\}(H_0 = \emptyset)$ is the history of the process through arrival m. A policy P for controlling the system is a set of decisions $\{D_k^P(H_{m-1}, \eta_m)\}$, $m = 1, 2, \ldots; k = 0, 1$. $D_k^P(H_{m-1}, \eta_m)$ is interpreted as the probability of implementing decision k (k = 0 if balk, 1 if join) at time m given history H_{m-1} and present state η_m when policy P is in effect.

Yechiali categorizes possible policies for controlling the system in the following manner. Let C_t be the class of all policies $\{D_k^P(H_{m-1}, \eta_m)\}$. Let C_s, a subclass of C_t, be the class of stationary Markovian policies; that is, for each arrival, only the state of the system at the instant of the arrival is used as a basis for making the decision of whether or not to join. Since, in Model I, the service times are exponential and the horizon is infinite, only stationary Markovian policies need to be considered. For convenience, let $D_i(1)$ be the stationary probability that $\Delta_n = 1$ given that $\eta_n = i$, and $\{1 - D_i(1)\} = D_i(0)$ be the stationary probability that $\Delta_n = 0$ given that $\eta_n = i$. S_k is the class of all stationary policies such that $0 < D_i(1) \leq 1$ for $i \leq k$ and $D_i(1) = 0$ for $i > k$. S_k is called the class of stationary control-limit policies of order k. Let S denote the class of stationary control-limit policies of infinite order; thus, $P \epsilon S$ if $P = \{D_i(1): D_i(1) > 0, i = 0, 1, \ldots\}$. Yechiali calls the union of S_k and S the class of control-limit policies, denoted by $C_{c\ell}$, which is a subclass of C_s. Let C_d, a subclass of C_t, be the class of nonrandomized policies, thus, $P \epsilon C_d$ implies $D_k^P(\cdot,\cdot) = 0$ or 1. The class of deterministic control-limit rules, $C_{dc\ell}$, is a subclass of C_d and $C_{c\ell}$. $P \epsilon C_{dc\ell}$ if $P = \{D_i(1): D_i(1) = 1, i \leq k; D_1(1) = 0, i > k\}$ for some k or $P = \{D_i(1): D_i(1) = 1, i = 0, 1, \ldots\}$.

Now, consider the individual optimum problem. A policy, P, or a set of joining probabilities $\{D_k^P(\cdot,\cdot)\}$ is sought that maximizes an arrival's expected net benefit. If the customer balks, his expected net benefit is zero. Since the service times are exponential, the expected time in the system for an arrival who finds i in the system is $(i + 1)/\mu$. Thus, the expected net benefit for joining is $R - (i + 1)C/\mu$. A policy P is sought such that

$$\sum_{i=o}^{\infty} D_1^P(H_{m-1},i)(R - (i + 1)C/\mu) \tag{3.6}$$

is maximized. If n_s is such that $R - n_s C/\mu \geq 0$, but $R - (n_s + 1)C/\mu < 0$, then clearly $P = \{D_1^P(H_{m-1},i) = 1,\ i < n_s;\ D_1^P(H_{m-1},i) = 0,\ i \geq n_s\}$ is the policy that maximizes Equation (3.6). The optimal policy is a deterministic control-limit rule which is indeed the same policy that Naor found.

Determining the optimal policy for the social optimum problem is more involved. The customers or administrators are assumed to act to maximize the expected net benefit per unit time of the arrivals as a whole. If g is the expected net benefit rate, two policies P and P' are called g-equivalent if $g_P = g_{P'}$. The expected net gain rate of the arrivals as a whole under policy P is:

$$g_P = \sum_{i=o}^{\infty} \phi_i^P D_i^P(1)\lambda\{R - C(i + 1)/\mu\} , \tag{3.7}$$

where ϕ_i^P is the steady state or stationary probability that the system is in state i if policy P is used. $D_i^P(1)$ is the stationary probability that customers are allowed to join under policy P when i customers are in the system. The main difference between the social optimum formulation and the individual optimum formulation is that the social optimum formulation includes information about the stationary probabilities of the various states. This information allows the administrator to anticipate expected benefits from customers who have yet to arrive.

Yechiali first establishes that for every $P \varepsilon C_s$, there is a g-equivalent rule $P' \varepsilon C_{c\ell}$. Thus, the search for the solution to the social optimum problem can be restricted to control-limit policies. He uses the fact that a nonrandomized policy is optimal for a finite state space to show that a deterministic control-limit policy, $P \varepsilon C_{dc\ell}$, is optimal for the finite state space social optimum problem. For the infinite state space social optimum problem, he considers two cases, an ergodic case and a nonergodic case. For the nonergodic case ($\lambda \geq \mu$), he argues that $L(n) \to \infty$ as $n \to \infty$; thus, $g(n) \to -\infty$ as $n \to \infty$. (Here, $g(n)$ denotes the expected gain rate under any control-limit policy with a maximum of n customers in the system.) Since $g(n)$ is finite when n is finite, the finite state-space result can be used to show that a finite deterministic control-limit policy is optimal for the nonergodic case. In the ergodic case ($\lambda < \mu$), Yechiali uses the limit of a sequence of finite state-space problems to show that again a deterministic control-limit policy is optimal. He then uses arguments based on the policy iteration algorithm of Howard (1960) to establish that the social control-limit, $n_o - 1$, is less than or equal to the individual control-limit, $n_s - 1$. (Howard's policy iteration algorithm is discussed in Appendix C.)

The solution to the social optimum problem can be found using Howard's policy iteration algorithm. (The solution of a small problem by policy iteration is given in Appendix C.) For each state i, only two choices exist in the policy iteration procedure, either reject the customer, $k = 0$, or accept the customer, $k = 1$. For later models, the range of alternatives will expand. When policy iteration is used to solve for n_o in Yechiali's formulation, n_s serves as an upper bound on the state space needed to describe the problem.

3.4 Comparison of Results

The two developments of n_s, the balking point in the self-optimum problem, given in Sections 3.2 and 3.3, lead to the same expression for determining n_s; namely, $n_s = [R\mu/C]$. This section establishes that the gain rate expressions for the social optimum problem (and thus, the solutions) in Sections 3.2 and 3.3 are the same. In fact, the expressions are also shown to be equivalent to a third gain rate expression.

Naor's work determines the capacity of the system, n_o, that maximizes Equation (3.2). Yechiali's work determines a control-limit policy that maximizes Equation (3.7). Control limits and system capacities are related in the following manner: a control limit of j leads to a capacity of $j + 1$. A third term used here, balking point or forced balking point, is equivalent to the capacity of the system.

Let Naor's form of the expected social gain rate be called g_N. From Equation (3.2), $g_N(n) = \lambda'(n)R - L(n)C$. $g_Y(n)$ is Yechiali's form of the expected gain rate expression. In this form of the gain rate expression, the expected net reward, i.e., the reward for service minus the expected waiting cost $\{R - C(i + 1)/\mu\}$ is given to an arrival the instant he joins the system. In the third form, $g_T(n)$, the waiting costs are charged over the length of the customer's stay in the system while his reward is given to him when he joins. $g_T(n)$ and $g_Y(n)$ are developed using notation from Howard (1971) and both are shown to be equal to $g_N(n)$.

$g_T(n)$ is developed first. n, the forced balking point, is the upper limit on the number in the system. Let y_i be the cost per unit time the system is in state i, where state i indicates that i customers are in the system. Here, y_i is simply Ci. Let b_{ij} be the reward for a transition from state i to state j.

$$b_{ij} = \begin{cases} R & j = i + 1 \\ 0 & \text{otherwise} \end{cases} .$$

Let $\bar{\tau}_{ij}$ be the expected holding time in state i given that the next transition is to state j. P_{ij} is the probability of a transition from state i to j. $\bar{\tau}_i = \sum_{j=o}^{n} P_{ij}\bar{\tau}_{ij}$ is the expected waiting time in state i. If r_i is defined to be the expected reward per occupancy of state i, then:

$$r_i = \sum_{j=o}^{n} P_{ij}(-y_i\bar{\tau}_{ij} + b_{ij})$$

$$= -Ci\bar{\tau}_i + P_{i,i+1}R \quad .$$

Define q_i to be the expected reward per unit time in state i.

$$q_i = r_i/\bar{\tau}_i = -Ci + P_{i,i+1}R/\bar{\tau}_i \quad .$$

Since n is the forced balking point, a state dependent arrival rate λ_i may be defined as follows:

$$\lambda_i = \begin{cases} \lambda & i < n \\ 0 & i \geq n \end{cases} .$$

Since the interarrival and service time distributions are exponential, the transition rate out of state i, T_i, is given by:

$$T_i = \begin{cases} \lambda_i & i = 0 \\ \lambda_i + \mu & i > 0 \end{cases} \quad .$$

Thus, $\bar{\tau}_i = 1/T_i$ or

$$\bar{\tau}_i = \begin{cases} 1/\lambda_i & i = 0 \\ 1/(\lambda_i + \mu) & i > 0 \end{cases} \quad .$$

Also, the probability of a transition from state i to i + 1 is

$$P_{i,i+1} = \begin{cases} 1 & i = 0 \\ \lambda_i/(\lambda_i + \mu) & 0 < i < n \\ 0 & i \geq n \end{cases} \quad .$$

Thus, the expected reward rate in state i becomes

$$q_i = - Ci + \lambda_i R \quad .$$

Using results from Howard (1960), $g_T(n)$ can be written as

$$g_T(n) = \sum_{i=o}^{n} \phi_i(n)\, q_i \quad , \qquad (3.8)$$

where $\phi_i(n)$ is the steady state probability that i customers are in the system given that the forced balking point is n. (This equation is similar to Equation (3.7) in Section 3.3.) Thus,

$$g_T(n) = \sum_{i=o}^{n} \phi_i(n)\lambda_i R - C \sum_{i=o}^{n} \phi_i(n)\, i$$

$$= \lambda R \sum_{i=o}^{n-1} \phi_i(n) - CL(n) \quad .$$

$\lambda \sum_{i=o}^{n-1} \phi_i(n)$ is the mean rate at which customers actually join the system which is the effective arrival rate, $\lambda'(n)$. Thus,

$$g_T(n) = \lambda'(n)R - CL(n) = g_N(n) \quad .$$

For $g_Y(n)$, $y_i = 0$ and

$$b_{ij} = \begin{cases} R - (i + 1)C/\mu & j = i + 1 \\ 0 & \text{otherwise} \end{cases} \quad .$$

Thus,

$$r_i = \sum_{j=o}^{n} P_{ij}b_{ij} = P_{i,i+1}\{R - (i + 1)C/\mu\} \quad .$$

The expected reward per unit time in state i is given by

$$q_i = r_i/\overline{\tau}_i = P_{i,i+1}\{R - (i + 1)C/\mu\}/\overline{\tau}_i \quad .$$

λ_i and $\overline{\tau}_i$ for $g_Y(n)$ are the same as they were for $g_T(n)$. Thus,

$$q_i = \begin{cases} 1\{R - (i + 1)C/\mu\}/(1/\lambda_i) & i = 0 \\ \{\lambda_i/(\lambda_i + \mu)\}\{R - (i + 1)C/\mu\}/\{1/(\lambda_i + \mu\} & i > 0 \end{cases}$$

Thus, for any i, $q_i = \lambda_i R - \lambda_i(i + 1)C/\mu$. As in Equation (3.8),

$$g_Y(n) = \sum_{i=o}^{n} \phi_i(n)q_i$$

which becomes

$$g_Y(n) = \lambda R\sum_{i=o}^{n-1} \phi_i(n) - C\sum_{i=o}^{n-1} (\lambda/\mu)\phi_i(n)(i + 1) \quad .$$

For an M/M/1/n queue, $\phi_{i+1}(n) = (\lambda/\mu)\phi_i(n)$ if $1 \leq i + 1 \leq n$. Thus,

$$\begin{aligned} g_Y(n) &= \lambda'(n)R - C\sum_{i=o}^{n-1} \phi_{i+1}(n)(i + 1) \\ &= \lambda'(n)R - CL(n) \\ &= g_N(n) \end{aligned}$$

and this establishes

<u>Theorem 3.1</u>: $g_N(n) = g_Y(n) = g_T(n)$.

(A result analogous to $g_Y(n) = g_T(n)$ for a discounted infinite horizon problem is established in Stidham and Prabhu (1974).)

3.5 Linear Programming Formulation of the Social Optimum Problem

Naor's formulation of the social optimum problem can be written as

$$\max_{n} g(n) = \lambda'(n)R - CL(n)$$

which is a nonlinear programming problem with an integer valued decision variable. However, this section shows that Yechiali's formulation can be transformed into a linear programming problem.

The following linear programming formulation is similar to that of Yechiali (1971) but is modified to maximize expected gain rate as in Fox (1966). Since $n_o \leq n_s$, only states 0 through n_s can have positive steady state probabilities. A policy $P^* \varepsilon C_s$ is sought such that

$$g_{P*} = \max_{P \varepsilon C_s} \{\sum_{i=o}^{n_s} \sum_{k=o}^{1} \emptyset_i^P D_i^P(k) r_i(k)\} / \{\sum_{i=o}^{n_s} \sum_{k=o}^{1} \emptyset_i^P D_i^P(k) \overline{\tau}_i(k)\}. \quad (3.9)$$

$\emptyset_i^P$ is the steady state probability that i are in the system under policy P. $D_i^P(k)$ is the probability of choosing action k (k = 0 if balk, 1 if join) in state i when policy P is used. $r_i(k)$ is the expected reward per occupancy of state i if action k is chosen. $\overline{\tau}_i(k)$ is the expected waiting time in state i when action k is chosen.

Equation (3.9) is just Equation (3.7) written in a slightly different manner. $D_i^P(1)\lambda\{R - (i + 1)C/\mu\}$ from Equation (3.7) is $q_i(1)$, the expected gain per unit time in state i when customers are allowed to join. However, $q_i(1) = r_i(1)/\overline{\tau}_i(1)$ and $q_i(0) = 0$. In Equation (3.9), the $r_i(k)$ and $q_i(k)$ terms are summed separately, then divided, as opposed to Equation (3.7) where $q_i(k)$ is formed first. In words, Equation (3.9) is the expected gain per transition divided by the expected time per transition which is indeed the expected gain per unit time.

The balance equations and normalizing equation which the steady state probabilities $\{\phi_i^P D_i^P(k)\}$ must satisfy are

$$\sum_{i=o}^{n_s} \sum_{k=o}^{1} \phi_i^P D_i^P(k) P_{ij}(k) - \sum_{k=o}^{1} \phi_j^P D_j^P(k) = 0, \; j = 0, \ldots, n_s - 1 \tag{3.10}$$

and

$$\sum_{j=o}^{n_s} \sum_{k=o}^{1} \phi_j^P D_j^P(k) = 1 \quad , \tag{3.11}$$

where $0 \leq \phi_j^P \leq 1$ and $0 \leq D_j^P(k) \leq 1$. $P_{ij}(k)$ is the probability of a transition from state i to j if action k is chosen. Note that Equation (3.10) consists of only n_s equations since the redundant balance equation for state n_s has been dropped from the formulation. Equations (3.9) to (3.11) constitute a linear program with a fractional objective function.

A linear program equivalent to Equations (3.9) to (3.11) but without the fractional objective function will be developed using the work of Fox (1966) and Charnes and Cooper (1962). First, let $x_i(k) = \phi_i D_i(k)$, so that Equations (3.9) to (3.11) become

$$\max_{P \epsilon C_s} \{\sum_{i=o}^{n_s} \sum_{k=o}^{1} x_i^P(k) r_i(k)\} / \{\sum_{i=o}^{n_s} \sum_{k=o}^{1} x_i^P(k) \overline{\tau}_i(k)\}$$

subject to

$$\sum_{i=o}^{n_s} \sum_{k=o}^{1} x_i^P(k) P_{ij}(k) - \sum_{k=o}^{1} x_j^P(k) = 0, \; j = 0, \ldots, n_s - 1$$

$$\sum_{j=o}^{n_s} \sum_{k=o}^{1} x_j^P(k) = 1 \qquad\qquad x_j^P(k) \geq 0 \quad .$$

Let $y_i(k) = tx_i(k)$, where $t \geq 0$, be chosen so that

$$\sum_{i=o}^{n_s} \sum_{k=o}^{1} y_i^P(k)\, \overline{\tau}_i(k) = 1 \quad .$$

Multiplication of the objective function by t/t yields

$$\max_{P \varepsilon C_s} \sum_{i=o}^{n_s} \sum_{k=o}^{1} y_i^P(k)\, r_i(k) \quad .$$

After multiplication of Equations (3.10) and (3.11) by t, the linear program becomes

$$\max_{P \varepsilon C_s} \sum_{i=o}^{n_s} \sum_{k=o}^{1} y_i^P(k)\, r_i(k)$$

subject to

$$\sum_{i=o}^{n_s} \sum_{k=o}^{1} y_i^P(k)\, P_{ij}(k) - \sum_{k=o}^{1} y_j^P(k) = 0, \; j = 0, \ldots, n_s - 1$$

$$\sum_{j=o}^{n_s} \sum_{k=o}^{1} y_j^P(k) = t$$

$$\sum_{j=o}^{n_s} \sum_{k=o}^{1} y_j^P(k)\overline{\tau}_j(k) = 1$$

$$y_j^P(k) \geq 0, \; t \geq 0 \quad .$$

The last constraint is included to maintain the transformation. The first n_s constraints together with the last allow at most $n_s + 1$ of the $y_j(k)$'s to be positive. The next to last constraint merely adjusts the value of t. Since t is of no interest, it and the next to last constraint can be dropped from the formulation. Thus, Equations (3.9) to (3.11) can be written as the following equivalent linear program:

$$\max \sum_{i=o}^{n_s} \sum_{k=o}^{1} y_i(k) r_i(k) \tag{3.12}$$

subject to

$$\sum_{i=o}^{n_s} \sum_{k=o}^{1} y_i(k) P_{ij}(k) - \sum_{k=o}^{1} y_j(k) = 0, \; j = 0, \ldots, n_s - 1 \tag{3.13}$$

$$\sum_{j=o}^{n_s} \sum_{k=o}^{1} y_j(k) \overline{\tau}_j(k) = 1 \tag{3.14}$$

$$y_j(k) \geq 0 \quad , \tag{3.15}$$

The superscript P has been dropped since the optimal policy can be recovered from the solution to Equations (3.12) to (3.15) through

$$D_i^{P*}(k) = y_i(k) / \sum_{k=o}^{1} y_i(k), \; i = 0, \ldots, n_s; \; k = 0, 1. \tag{3.16}$$

Fox also shows that the $D_i^{P*}(k)$ are either zero or one and at most $n_s + 1$ of them are one. ϕ_i^{P*} can be found from

$$\phi_i^{P*} = \begin{cases} y_i(k) \overline{\tau}_i(k), & y_i(k) > 0 \\ 0 \; . & \text{otherwise} \end{cases}$$

The linear program yields the social balking point through

$$n_o = \max\{i: \; y_i(1) > 0\} + 1 \quad . \tag{3.17}$$

3.6 <u>g and n_o as Functions of λ</u>

It seems reasonable to conjecture that the expected gain rate can only increase as the arrival rate increases. This is now established.

<u>Theorem 3.2</u>: g is a nondecreasing function of λ.

Proof: Let $\lambda' > \lambda''$. Define $g(P,\lambda)$ to be the expected gain rate under policy P when the arrival rate is λ. Let $g^*(\lambda)$ denote the optimal expected gain rate when the arrival rate is λ. Since Yechiali (1971) shows that a deterministic control-limit policy is optimal, let $P^* = \{D_i(1): D_i(1) = 1, i \leq n_o - 1; D_i(1) = 0, i > n_o - 1\}$ be the optimal policy when the arrival rate is λ''. Let $P' \varepsilon C_{c\ell}$ (the set of stationary control-limit policies) be such that $P' = \{D_i(1): D_i(1) = \lambda''/\lambda', i \leq n_o - 1;$ $D_i(1) = 0, i > n_o - 1\}$. Under policy P', an arrival who finds less than n_o in the system is allowed to join with probability λ''/λ' and is forced to balk with probability $1 - (\lambda''/\lambda')$. Thus, P' is not a deterministic policy. Since no penalty is assessed for rejecting a customer, $g(P', \lambda') = g(P^*, \lambda'')$. Finally, since $g^*(\lambda') \geq g(P',\lambda')$, $g^*(\lambda') \geq g^*(\lambda'')$.

Although policy iteration gives a method for determining the social optimum, n_o, for a given λ, the following section presents a method for determining the range of values of λ over which a given n_o is optional.

Define $\{f(i)\}$ to be the sequence of expected net rewards of joining customers, where $f(i) = R - C(i + 1)/\mu$ is the expected net reward if i customers are in the system. In Yechiali's formulation, an arriving customer receives his expected net reward upon entering the system. Thus, for Yechiali's formulation, the expected net gain of the system per occupancy of state i is

$$r_i = \begin{cases} f(i) & i = 0 \\ \{\lambda_i/(\lambda_i + \mu)\}f(i), & i > 0 \end{cases} .$$

Since

$$\overline{\tau}_i = \begin{cases} 1/\lambda_i , & i = 0 \\ 1/(\lambda_i + \mu) , & i > 0 \end{cases} ,$$

the expected reward per unit time in state i is

$$q_i = r_i/\overline{\tau}_i = \lambda_i f(i) ,$$

where

$$\lambda_i = \begin{cases} \lambda, & i < n_o \\ 0, & i \geq n_o \end{cases} .$$

Also, from Equation (3.8),

$$g(n) = \sum_{i=o}^{n} \phi_i(n)\, q_i ,$$

where the Y subscript has been dropped since the three equations for gain rate were shown to be equal in Theorem 3.1. Naor states that g(n) is discretely unimodal in n. Thus, if $\Delta g(n) = g(n) - g(n - 1)$, then n_o such that $\Delta g(n_o + 1) < 0 \leq \Delta g(n_o)$ is optimal. Note that

$$\Delta g(n) = \sum_{i=o}^{n} \phi_i(n) q_i(n) - \sum_{i=o}^{n-1} \phi_i(n - 1) q_i(n - 1) , \quad (3.18)$$

where $q_i(n)$ denotes the expected gain rate in state i when the forced balking point is n. Thus, $q_n(n) = q_{n-1}(n - 1) = 0$ since no entry is allowed at the forced balking point.

$$q_i(n) = q_i(n - 1) = \lambda f(i) , \quad \text{for } i = 0, \ldots, n - 2;$$

$$q_{n-1}(n) = \lambda f(n - 1) .$$

Substitution for $q_i(.)$ in Equation (3.18) yields

$$\Delta g(n) = \sum_{i=o}^{n-2} \{\phi_i(n) - \phi_i(n-1)\}\lambda f(i) + \phi_{n-1}(n)\lambda f(n-1) \quad .$$

Since $n_o \leq n_s$, $f(i) \geq 0$. Also, $\phi_{n-1}(n) > 0$ and $\phi_i(n) - \phi_i(n-1) < 0$. Thus, $\Delta g(n) \geq 0$ if

$$\phi_{n-1}(n)\lambda f(n-1) \geq \sum_{i=o}^{n-2} \{\phi_i(n-1) - \phi_i(n)\}\lambda f(i) \quad . \quad (3.19)$$

If $\rho = \lambda/\mu \neq 1$, then Equation (3.19) can be written as

$$(1-\rho)\rho^{n-1} f(n-1)/(1-\rho^{n+1}) \geq$$
$$\sum_{i=o}^{n-2} \{(1-\rho)\rho^i/(1-\rho^n) - (1-\rho)\rho^i/(1-\rho^{n+1})\} f(i) \quad . \quad (3.20)$$

After formation of a common denominator on the right-hand side and division by $\{(1-\rho)/(1-\rho^{n+1})\} > 0$, Equation (3.20) becomes

$$f(n-1) \geq \{\rho(1-\rho)/(1-\rho^n)\} \sum_{i=o}^{n-2} \rho^i f(i) \quad .$$

Since $\{(1-\rho^n)/(1-\rho)\} = \sum_{i=o}^{n-1} \rho^i$, this leads to $\Delta g(n) \geq 0$ if

$$f(n-1) \sum_{i=o}^{n-1} \rho^i \geq \rho \sum_{i=o}^{n-2} \rho^i f(i) \,, \qquad n \geq 2 \quad . \quad (3.21)$$

Now consider the case where $\rho = 1$. If $\rho = 1$, $\phi_i(n) = 1/(n+1)$, so Equation (3.19) becomes

$$f(n-1)/(n+1) \geq \sum_{i=o}^{n-2} \{1/n - 1/(n+1)\} f(i) \quad .$$

After some algebra, this can be written as

$$f(n-1) \geq \{\sum_{i=o}^{n-2} f(i)\}/n \quad .$$

Since this is the limit $\rho \to 1$ of Equation (3.21), Equation (3.21) can be used for all values of ρ.

Finding the range of λ over which $n_o = 1$ requires finding λ such that $\Delta g(1) \geq 0$, but $\Delta g(2) < 0$. If $g(1) < 0$ $(R < C/\mu)$, the system is trivial since no customer ever enters. Thus, $\Delta g(1) \geq 0$ for all λ since $g(0) = 0$. For $\Delta g(2) < 0$, Equation (3.21) is a linear function of λ. Finding the range of λ over which $n_o = 2$ requires finding λ such that $\Delta g(2) \geq 0$ but $\Delta g(3) < 0$. Again, for $\Delta g(2) \leq 0$, Equation (3.21) is a linear function of λ. For $\Delta g(3) < 0$, Equation (3.21) is a quadratic function of λ. This pattern continues so that finding λ such that $\Delta g(n + 1) \geq 0$ or $\Delta g(n + 1) < 0$ requires finding the roots of an n^{th} degree polynomial. Use of Equation (3.21) is demonstrated with an example.

Suppose the reward for service of the members of a single class of customers is $R = 5$. Also, suppose that the cost per unit time in the system is $C = 2$, and that the service rate of the single server is $\mu = 3$. $\{f(i)\} = \{R - C(i + 1)/\mu\} = \{13/3, 11/3, 3, 7/3, 5/3, 1, 1/3. -2/3, \ldots\}$. (Note that $n_s = 7$.) First, find the range of values of λ for which $n_o = 1$. $\Delta g(1) \geq 0$ for all λ. $\Delta g(2) < 0$ if from Equation (3.21)

$$f(1)\sum_{i=o}^{1} \rho^i < \rho\sum_{i=o}^{0} \rho^i f(i) \qquad \text{or} \qquad 16.5 < \lambda \quad .$$

Thus, $n_o = 1$ if $\lambda > 16.5$. Now, find the range of λ over which $n_o = 2$. $\Delta g(2) \geq 0$ if $\lambda \leq 16.5$. $\Delta g(3) < 0$ if from Equation (3.21)

$$f(2) \sum_{i=o}^{2} \rho^i > \rho \sum_{i=o}^{1} \rho^i f(i) \qquad \text{or} \qquad -2\lambda^2 - 12\lambda + 81 > 0 \quad .$$

Solution of the above for λ yields $\Delta g(3) < 0$ if $\lambda > 4.035$. Therefore,

$n_o = 2$ if $4.035 < \lambda \leq 16.5$. Further use of Equation (3.21) yields $n_o = 3$ if $2.1 < \lambda \leq 4.035$. Table 3.1 presents the results of the policy iteration program of Appendix C for this example. These results are given to confirm Equation (3.21), to illustrate Theorem 3.2, and to introduce the next idea.

TABLE 3.1

Policy Iteration Results for Various Values of λ for a One Class Example with $R = 5$, $C = 2$, and $\mu = 3$

λ	n_o	g
0.1	7	0.431
1.0	5	4.003
2.1	4	6.944
2.2	3	7.128
4.02	3	8.993
4.05	2	9.011
16.4	2	10.998
16.6	1	11.010
100.00	1	12.621

The value of n_o, the forced balking point for the social optimum problem, appears to decrease as the arrival rate increases in Table 3.1. This result is established as Theorem 3.3.

<u>Theorem 3.3</u>: n_o is a nonincreasing function of λ.

Proof: Since Naor proved that g is discretely unimodal in n, n_o is optimal if $\Delta g(n_o + 1) < 0 \leq \Delta g(n_o)$. Let $n_o(\lambda)$ denote the optimal forced balking point when the arrival rate is λ. Let $n = n_o(\lambda'')$, where λ'' is a fixed value of λ. From Equation (3.21),

$$g(n) \geq 0 \quad \text{if}$$

$$f(n-1) \sum_{i=o}^{n-1} \rho^i \geq \rho \sum_{i=o}^{n-2} \rho^i f(i) \quad ,$$

for $n \geq 2$, which can be written as

$$\Delta g(n) \geq 0 \quad \text{if}$$

$$f(n-1)(1 + \rho + \ldots + \rho^{n-1}) \geq \rho f(0) + \rho^2 f(1) + \ldots + \rho^{n-1} f(n-2) \quad . \tag{3.22}$$

As λ increases, ρ increases. Since $\{f(i)\}$ decreases as i increases, $f(0) > f(1) > \ldots > f(n-2) > f(n-1)$. Thus, as λ increases, the right-hand side of Equation (3.22) increases faster than the left-hand side. This eventually leads to $\Delta g(n) < 0$ for λ greater than or equal to some $\lambda' > \lambda''$. Therefore, $n_o(\lambda') < n_o(\lambda'')$. Since $\Delta g(1) \geq 0$ for all λ, continuation of the above argument leads to the existence of λ''' such that $n_o(\lambda) = 1$ for all $\lambda \geq \lambda'''$. Thus, as λ increases, n_o decreases until it becomes equal to one.

Consider again the value of g as λ increases in Table 3.1. Theorem 3.2 established that indeed g can only increase as λ increases. However, note that g appears to approach a limit as $\lambda \to \infty$. From Theorem 3.3, n_o decreases as λ increases, so that eventually $n_o = 1$. If $n_o = 1$,

$$\phi_o(1) = \mu/(\mu + \lambda) \quad \text{and} \quad \phi_1(1) = \lambda/(\mu + \lambda) \quad .$$

From Equation (3.8),

$$g(n_o) = \sum_{i=o}^{n_o} \phi_i(n_o) q_i(n_o)$$

so

$$g(1) = \mu\lambda f(0)/(\mu + \lambda) \quad .$$

Dividing numerator and denominator by λ and then letting $\lambda \to \infty$, yields

$$\lim_{\lambda\to\infty} g(1) = \mu f(0) \quad . \tag{3.23}$$

For this example,

$$\lim_{\lambda\to\infty} g(1) = 13 \quad .$$

3.7 Tolls

Lippman and Stidham (1977) define an optimal congestion toll as "an entrance fee that induces customers acting individually to behave in a socially optimal way." Two types of optimal congestion tolls[1] are considered. The first form of the toll is analogous to the tolls of Naor (1969) and Yechiali (1971). The second form illustrates what Lippman and Stidham (1977) call the monotonicity of a toll that is a function of the number of customers in the system.

---

[1]The "tolls" developed here are actually payments for not joining but have the same effect as charges made for joining.

Suppose that the assumptions of Model I are modified so that an additional term, Q, is introduced into the cost structure, where Q is a fixed payment to any customer who does not join the queue. If each customer is allowed to decide whether or not to join the queue, then each will maximize his own expected net benefit. Expected net benefit for joining is still $R - (i + 1)C/\mu$ if i customers are in the system, but is now Q for not joining. Thus, joining serves the customer's self-interest if $R - (i + 1)C/\mu \geq Q$. This leads to an $M/M/1/n_s'$ queueing system, where n_s' is such that

$$R - (n_s' + 1)C/\mu < Q \leq R - n_s'C/\mu \quad .$$

Thus,

$$n_s' = [(R - Q)\mu/C]$$

which is just Equation (3.1) with R replaced by R - Q.

If an administrator decides who joins the queue and he wants to impose a limit of n_o from the original cost structure of Model I, he can do so by changing the cost structure to include Q. Since $(R - Q)\mu/C \leq R\mu/C$, $n_s' \leq n_s$. The administrator needs to choose Q such that $n_s' = n_o$. Thus, he must find Q such that

$$R - C(n_o + 1)/\mu < Q \leq R - Cn_o/\mu \qquad (3.24)$$

which corresponds to the tolls of Naor (1969) and Yechiali (1971).

Suppose now that the assumptions of Model I are modified so that again a term is added to the cost structure. Any customer who does not join the queue is paid $q \cdot i$, where q is a fixed amount of money and i is the number of customers in the system. Again, assume that each customer acts to maximize his own expected net benefit. The expected net benefit for joining is still $R - (i + 1)C/\mu$; however, the expected net benefit for not joining is $q \cdot i$. Joining serves the self-interest of the customers if $R - (i + 1)C/\mu \geqq q \cdot i$. This leads to an $M/M/1/n''_s$ queueing system, where

$$n''_s = [\mu(R + q)/(C + \mu q)] \quad .$$

Again, if an administrator decides who joins the queue and he wants to impose the social limit from the cost structure of Model I, he can do so by changing the cost structure to include $q \cdot i$. Since $R - (i + 1)C/\mu \geqq R - (i + 1)C/\mu - q \cdot i$, $n_s \geqq n''_s$. The administrator wants to find q such that $n''_s = n_o$. Such a value of q is determined by

$$\{\mu R - C(n_o + 1)\}/(\mu n_o) < q \leqq (\mu R - Cn_o)/\{\mu(n_o - 1)\}. \quad (3.25)$$

3.8 Conclusion

The simplest model of a controlled queue considered is Model I. The semi-Markov decision process and linear programming formulations of the model that Yechiali (1971) presents can easily be solved for n_o by use of a policy iteration algorithm and a simplex algorithm, respectively. Although Model I is not a particularly realistic representation of a system such as an airport landing queue, it serves as a point of departure in the development of more realistic models.

CHAPTER IV

THE SEVERAL CLASS MODEL

This chapter defines Model II in which several classes of customers are considered. The formulations of Model I presented by Naor (1969) and Yechiali (1971) are extended to Model II. The form of the optimal policy is shown to be a set of forced balking points, one for each class of customers. The Naor type of formulation of Model II is not easy to solve; however, the formulation of this model as a semi-Markov decision process following Yechiali (1971) lends itself to solution by both policy iteration and linear programming.

4.1 Model II

This model allows M classes of customers. The arrivals from each class, m, form a Poisson stream with mean rate λ_m, where $m = 1, 2, \ldots, M$. The service times of the single server are independent, identically distributed, exponential random variables with mean $1/\mu$. Like Model I, Model II is also an M/M/1 queueing system but now the customers come from M separate groups or classes with varying costs and rewards. A member of class m receives a reward for service of R_m and pays C_m per unit time he spends in the system. To avoid trivialities, R_m is required to be greater than or equal to C_m/μ for all m. Again, two objective functions are considered, one for self-optimizing customers and the other, the one of primary interest, for customers acting "socially." Each arrival or the administrator is assumed to be able to determine the state of the system at the time of each arrival, where again the state of the system is the number of customers in the system. An optimal policy is sought for each objective function; that is, a set of join or balk decisions is sought for each class for each possible state of the system to maximize each objective function.

A self-optimizing customer decides whether or not to join by choosing the larger of the expected net benefit for joining and the expected net benefit for not joining (which is zero). For a member of class m arriving to find i customers ahead of him, the expected net benefit of joining is $R_m - (i + 1)C_m/\mu$. For the social optimum problem, an administrator decides whether or not a customer of class m can join when i customers are ahead of him. The sum of the expected net benefits per unit time of all arrivals of all classes is shown in Section 4.5 to be maximized by a policy that imposes a vector $\bar{n}_o$ of forced balking points on the customers. $\bar{n}_o = (n_{o_1}, n_{o_2}, \ldots, n_{o_M})$, where members of class m are allowed to join if the state of the system is less than n_{o_m} but must balk otherwise.

4.2 Individual Optimum Problem

The solution of the individual optimum problem is investigated from two points of view. The first development is an extension of the Naor approach to Model I. If an arrival from class m finds i customers ahead of him, his expected net benefit for joining is $R_m - (i + 1)C_m/\mu$. His expected time in the system is $(i + 1)/\mu$ because:

1) his expected service time and that of the i - 1 waiting customers ahead of him is $1/\mu$, and

2) the expected remaining service time of the customer in service is $1/\mu$ since the service time distribution is memoryless.

Since the net benefit for balking is zero, joining the queue is in the arrival's self-interest if the number of customers in the system is less than n_{s_m}, where n_{s_m} is such that

$$R_m - (n_{s_m} + 1)C_m/\mu < 0 \leq R_m - n_{s_m}C_m/\mu \quad .$$

Thus, self-optimizing customers of class m determine a balking point n_{s_m} such that

$$n_{s_m} = [R_m\mu/C_m] \quad , \qquad m = 1, 2, \ldots, M \quad . \tag{4.1}$$

(The brackets indicate the greatest integer function.) This is analogous to Equation (3.1) for the single class model. Thus, for the individual optimum problem, the M/M/1 queueing system becomes a finite capacity system with the capacity given by n_s^*, where

$$n_s^* = \max_m \{n_{s_m}\} \quad . \tag{4.2}$$

However, members of class m treat the system as if it had capacity n_{s_m}; that is, they balk when n_{s_m} or more customers are in the system.

Equation (4.1) can also be developed by extending the Yechiali approach to Model I. Since customers only use the state of the system to decide whether or not to join the system,[1] only the subclass of stationary Markovian policies, C_s, needs to be examined for the optimal

[1]Using the state of the system is the best a customer can do since the transitions are memoryless and the horizon is infinite.

policy. Thus, a policy $P^*\varepsilon C_s$ is sought such that P^* yields

$$\max_{P\varepsilon C_s} \sum_{i=o}^{\infty} D_i^P(\overline{k}) \sum_{m=1}^{M} k_m(i)\{R_m - (i + 1)C_m/\mu\} \qquad . \qquad (4.3)$$

Expression (4.3) represents the maximum expected gain per customer for self-optimizing customers. $D_i^P(\overline{k})$ is the probability that action $\overline{k}$ is chosen under policy P when the state of the system is i. Action $\overline{k} = (k_1, k_2, \ldots, k_M)$ accepts class m if $k_m = 1$ and rejects class m if $k_m = 0$. A policy P is a set of join or balk decisions for each class for every possible state of the system. The notation $k_m(i)$ is used to emphasize that the join or balk decision for class m is a function of i, the state of the system. Again, $R_m - (i + 1)C_m/\mu$ is the expected net benefit to an arrival from class m for joining if i customers are ahead of him. Equation (4.3) can be maximized by setting $D_i^P(\overline{k}) = 1$ for all i and all $\overline{k}$ and setting

$$k_m(i) = \begin{cases} 1, & \text{if } R_m - (i + 1)C_m/\mu \geq 0 \\ 0, & \text{otherwise} \end{cases} \quad .$$

The same arguments that were used earlier in this section to develop Equation (4.1) can now be used again to arrive at Equation (4.1). $k_m(i)$ can then be found from Equation (4.1) by setting

$$k_m(i) = \begin{cases} 1, & \text{if } i < n_{s_m} \\ 0, & \text{otherwise,} \quad \text{for } m = 1, 2, \ldots, M \end{cases} \quad .$$

4.3 Social Optimum Problem

Three different formulations of the social optimum problem are presented in this section. As in Model I, an infinite time horizon without discounting is considered. The units of the objective function for the social optimum problem are dollars per unit time. Thus, the administrator wants to maximize the sum of the expected net benefits per unit time of all arrivals. The formulation that extends Naor (1969) is presented first.

In Section 4.5, the optimal policy is shown to specify a vector $\bar{n}_o$ of forced balking points, $\bar{n}_o = (n_{o_1}, n_{o_2}, \ldots, n_{o_M})$, where class m joins if the state of the system is less than n_{o_m} and balks otherwise. The sum of the expected net benefits of all arrivals per unit time when the forced balking points given by $\bar{n}$ are chosen is

$$g(\bar{n}) = \sum_{m=1}^{M} \lambda_m'(\bar{n}) R_m - \sum_{m=1}^{M} C_m L_m(\bar{n}) \quad . \tag{4.4}$$

Here, $\lambda_m'(\bar{n})$ and $L_m(\bar{n})$ are, respectively, the effective arrival rate of class m and the contribution of class m to the expected number of customers in the system when forced balking points $\bar{n}$ are employed. Equation (4.4) is Equation (3.2) extended to M classes of customers. Analytic solution of Equation (4.4) for the optimal $\bar{n}$ is a formidable task. (See Edelson and Hildebrand (1975) for the case M = 2.) Although a heuristic search procedure could be used to find a good solution to Equation (4.4), this will not be done here since the next formulation of the problem readily lends itself to solution.

The second and third developments formulate the social optimum problem as a semi-Markov decision process. This second formulation extends the method of Yechiali (1971) for Model I. Here, the state of the system is again the number of customers in the system. An arrival who joins when i are in the system is given his expected net benefit for joining, $R_m - (i + 1)C_m/\mu$, upon entry into the system. The decision to be made by the administrator for each state of the system is which classes if any to admit. This decision is represented by $\overline{k} = (k_1, k_2, \ldots, k_M)$, where class m is admitted if $k_m = 1$ and rejected if $k_m = 0$. In Section 4.5, n_{o_m}, the forced social balking point, is shown to be less than or equal to n_{s_m}, the self-optimizer's balking point, for each class m. Thus, n_s^* from Equation (4.2) serves as a bound on the state space required for the semi-Markov formulation. A policy $P^* \varepsilon C_s$, the class of stationary Markovian policies, is sought such that P* yields

$$\max_{P \varepsilon C_s} g_P = \max_{P \varepsilon C_s} \sum_{i=o}^{n_s^*} D_i^P(\overline{k}) \; \phi_i^P \sum_{m=1}^{M} k_m(i)\lambda_m\{R_m - (i + 1)C_m/\mu\} \quad . \tag{4.5}$$

The units of Equation (4.5) are dollars per unit time, the same units as Equation (4.4). Comparison of Equation (4.5) with Equation (4.3) yields some insight into the difference between the social optimum problem and the individual optimum problem for Model II. The main difference is that the social optimum formulation given by Equation (4.5) includes the steady state probability that the system is in state i, ϕ_i^P, when policy P is used. This information allows the administrator to anticipate net benefits from future arrivals. The other differences between Equations (4.5) and (4.3) are that n_s^* bounds the state space in Equation (4.5) and the inclusion of λ_m in Equation (4.5) makes its units dollars per unit time rather than dollars per customer as in Equation (4.3).

Equation (4.5) represents the optimum gain rate for a semi-Markov decision process with a finite state space and a finite policy space.

The policy space is finite since, at most, 2^M actions are possible for each of at most $n_s^* + 1$ states. Derman (1962) shows that a nonrandomized rule, $P \epsilon C_d$, is optimal for such a process. Thus, $D_i^P(\bar{k})$ can be dropped from Equation (4.5) since it will be one for the action given by the optimal $\bar{k}(i)$ and zero otherwise. In other words, $\bar{k}(i)$ is the only action chosen in state i under the optimal policy. Equation (4.5) then becomes

$$\max_{P \epsilon C_s \cap C_d} g_P = \max_{P \epsilon C_s \cap C_d} \sum_{i=o}^{n_s^*} \emptyset_i^P \sum_{m=1}^{M} k_m(i)\lambda_m\{R_m - (i + 1)C_m/\mu\} \quad . \tag{4.6}$$

Solution of Equation (4.6) by policy iteration and linear programming is examined in Sections 4.6 and 4.7.

The third and final formulation of the social optimum problem for Model II also formulates it as a semi-Markov decision process and extends the third form of the social optimum problem presented in Chapter III. The difference between this model and that which extends Yechiali's formulation is that now a joining customer receives his reward for service upon entering the system but pays his costs per unit time in the system throughout his stay in the system. Since the social optimum problem of Model II uses an infinite horizon and no discounting, the timing of the payments does not affect the gain rate of a given policy. The state of the system must indicate the number of each class of customers in the system so that the cost rate can be computed at all times. That is, the charges for a unit of time with three customers of class one and two of class two in the system are different than those for a unit of time with two of class one and three of class two. That still more information is needed is demonstrated by the following example.

Let the number of classes be two and the state space be (i,j), where (i,j) indicates that i customers of class one and j customers of class two are in the system. If an arrival from class one occurs and the customer joins, the state becomes (i + 1,j); conversely, if an arrival from class two occurs and the customer joins, the state becomes (i,j + 1). However, if a service occurs, the state of the system is unknown since the state space does not indicate the class of the customer in service. If the state space is amended to include the class of the customer in service, it becomes (i,j,k), where k = 1 if a class one customer is in service and k = 2 if a class two customer is in service. This definition of the state space still does not contain enough information to keep track of the number of each class of customers in the system. For instance, suppose that the state of the system is (2,2,1). Thus, two customers of each class are in the system and one of the class one customers is in service. If a service occurs, the state becomes (1,2,?). Since the service discipline is first come first served and the state space does not indicate which customer is first in line, the class of the customer moving into service is unknown. The system occupation costs can be calculated for the present state, but if another service occurs, the number of each class of customers in the system is unknown. Thus, the state space must be expanded further. In fact, the state space must indicate exactly what the queue looks like. A state space like $(m_1, m_2, \ldots, m_j, \ldots)$ is needed where m_j indicates the class of the customer in position j. For example, state (1,2,1,1,2) indicates that a customer of class one is in service followed by a customer of class two, two of class one, and finally a class two customer. Although this state space becomes large and complicated when the number of customers in the system is large, the social optimum problem for Model II can still be formulated in this manner.

Again, only the class of stationary Markovian policies needs to be considered. For each state of the system, an action $\bar{k} = (k_1, k_2, \ldots, k_M)$ is sought to maximize the gain rate of the system. As before, class m is admitted if $k_m = 1$ and must balk if $k_m = 0$. Let $\bar{m} = (m_1, m_2, \ldots, m_j, \ldots)$ indicate the state of the system, where m_j gives the class of the customer in position j. Also, let $\sigma_m(\bar{m})$ be the number of customers of class m present in state $\bar{m}$. Note that only the number of customers of each class in the system is required to compute the occupation costs at any time, but the position of the customers in the system is required to keep track of how these numbers change as the system goes through various transition. n_s^* from Equation (4.2) still serves as a bound on the number of customers allowed in the system. If $\bar{m}'$ denotes the set of all $\bar{m}$ satisfying

$$\sum_{m=1}^{M} \sigma_m(\bar{m}) \leq n_s^* \quad , \tag{4.7}$$

the objective function for this third formulation of the social optimum problem is

$$\max_{P\epsilon C_s} g_P = \max_{P\epsilon C_s} \sum_{\bar{m}\epsilon\bar{m}'} D_{\bar{m}}^P(\bar{k}) \; \phi_{\bar{m}}^P \sum_{m=1}^{M} \{k_m(\bar{m})\lambda_m R_m - \sigma_m(\bar{m}) C_m\} \quad . \tag{4.8}$$

Since this is again a finite state-space, finite policy-space problem, a deterministic policy is optimal according to the results of Derman

(1962) and thus, $D^{P}_{\underline{m}}(\overline{k})$ can be eliminated from the formulation. Equation (4.8) then becomes

$$\max_{P\epsilon C_s \cap C_d} g_P = \max_{P\epsilon C_s \cap C_d} \sum_{\underline{m}\epsilon\overline{m}'} \phi^{P}_{\underline{m}} \sum_{m=1}^{M} \{k_m(\overline{m})\lambda_m R_m - \sigma_m(\overline{m})C_m\} \quad . \tag{4.9}$$

The units of Equation (4.9) are dollars per unit time. This formulation can be solved using either policy iteration or linear programming. However, since all three formulations are shown to be equivalent in the next section, this formulation will later be dropped because it is more difficult to solve than the previous one.

4.4 Equivalence of the Three Formulations

The equivalence of the two formulations of the individual optimum problem has already been shown in Section 4.2. This section establishes the equivalence of the three formulations of the social optimum problem given in Section 4.3. As such, it extends the equivalence demonstrated in Section 3.4 for the one class model. For convenience, only two classes of customers are considered, although the method used generalizes to any finite number of classes. The expected gain rate, g_N, for the Naor-type extension is, from Equation (4.4),

$$g_N(\overline{n}) = \sum_{m=1}^{2} \lambda'_m(\overline{n})R_m - \sum_{m=1}^{2} C_m L_m(\overline{n}) \quad .$$

$g_Y(\overline{n})$ denotes the gain rate for the Yechiali-type extension and $g_T(\overline{n})$ denotes the gain rate for the third formulation. (Forced balking points given by $\overline{n} = (n_1, n_2)$ correspond to control limits $n_1 - 1$ and $n_2 - 1$.) $g_T(\overline{n})$ and $g_Y(\overline{n})$ are developed using the notation of Howard (1971) and both are shown to be equal to $g_N(\overline{n})$.

$g_T(\bar{n})$ is developed first. Let y_{ij} be the cost per unit time in state (i,j),[1] where (i,j) indicates that i customers of class one and j of class two are in the system. Thus,

$$y_{ij} = iC_1 + jC_2 \quad .$$

Let $b_{ij,i'j'}$ be the reward for a transition from state (i,j) to state (i',j'). Here,

$$b_{ij,i'j'} = \begin{cases} R_1 , & \text{if } i' = i + 1,\ j' = j \\ R_2 , & \text{if } i' = i,\ j' = j + 1 \\ 0 , & \text{otherwise} \end{cases} \quad .$$

Let $\bar{\tau}_{ij,i'j'}$ be the expected holding time in state (i,j) given the next transition is to state (i',j'). $\bar{\tau}_{ij}$ denotes the unconditional expected waiting time in state (i,j). Let $\lambda_m(i,j)$ be the mean arrival rate of class m customers when the state of the system is (i,j). Here,

$$\lambda_m(i,j) = \begin{cases} \lambda_m , & \text{if } i + j < n_{o_m} \\ 0 , & \text{otherwise} \end{cases} \quad .$$

With this definition, $\bar{\tau}_{ij}$ can be found from

$$\bar{\tau}_{ij} = \begin{cases} 1/\{\sum_{m=1}^{2} \lambda_m(i,j)\} , & i + j = 0 \\ \\ 1/\{\mu + \sum_{m=1}^{2} \lambda_m(i,j)\} , & i + j \neq 0 \end{cases} \quad .$$

Also, the probability of a transition from state (i,j) to state (i',j') is

[1]For notational convenience, all states, $\bar{m}$, with $\sigma_1(\bar{m}) = i$ and $\sigma_2(\bar{m}) = j$ are combined into state (i,j).

$$P_{ij,i'j'} = \begin{cases} \lambda_1(i,j)/\{\sum_{m=1}^{2} \lambda_m(i,j)\} , & i+j=0;\ i'=1,\ j'=0 \\ \lambda_2(i,j)/\{\sum_{m=1}^{2} \lambda_m(i,j)\} , & i+j=0;\ i'=0,\ j'=1 \\ \lambda_1(i,j)/\{\mu + \sum_{m=1}^{2} \lambda_m(i,j)\} , & \\ & i+j>0;\ i'=i+1,\ j'=j \\ \lambda_2(i,j)/\{\mu + \sum_{m=1}^{2} \lambda_m(i,j)\} , & \\ & i+j>0;\ i'=i,\ j'=j+1 \\ \mu/\{\mu + \sum_{m=1}^{2} \lambda_m(i,j)\} , & i+j>0;\ i'+j'=i+j-1 \\ 0 \quad , & \text{otherwise} \quad . \end{cases}$$

r_{ij} is the expected reward per occupancy of state (i,j).

$$\begin{aligned} r_{ij} &= \sum_{i'j'} P_{ij,i'j'}(-\,y_{ij}\bar{\tau}_{ij,i'j'} + b_{ij,i'j'}) \\ &= -\,y_{ij}\sum_{i'j'} P_{ij,i'j'}\bar{\tau}_{ij,i'j'} + \sum_{i'j'} P_{ij,i'j'}b_{ij,i'j'} \\ &= -\,y_{ij}\bar{\tau}_{ij} + P_{ij,(i+1)j}\, R_1 + P_{ij,i(j+1)}\, R_2 \quad . \end{aligned}$$

Let q_{ij} denote the expected reward per unit time in state (i,j).

$$\begin{aligned} q_{ij} &= r_{ij}/\bar{\tau}_{ij} = -\,y_{ij} + \{P_{ij,(i+1)j}\, R_1 + P_{ij,i(j+1)} R_2\}/\bar{\tau}_{ij} \\ &= -\,(iC_1 + jC_2) + \lambda_1(i,j)\, R_1 + \lambda_2(i,j)\, R_2 \quad . \end{aligned}$$

From Howard (1960), $g_T(\bar{n})$ can be written as

$$g_T(\bar{n}) = \sum_{i+j\le n^*_s} \phi_{ij}(\bar{n})\, q_{ij} \quad , \tag{4.10}$$

where $\phi_{ij}(\bar{n})$ is the steady state probability that i customers of class one and j customers of class two are in the system given that the forced balking points are given by $\bar{n}$. Thus,

$$g_T(\bar{n}) = \sum_{i+j \leq n_s^*} \phi_{ij}(\bar{n})\{-(iC_1 + jC_2) + \lambda_1(i,j)R_1 + \lambda_2(i,j)R_2\}$$

$$= -\sum_{i+j \leq n_s^*} \phi_{ij}(\bar{n})(iC_1 + jC_2) + \lambda_1 R_1 \sum_{i+j<n_{o_1}} \phi_{ij}(\bar{n})$$

$$+ \lambda_2 R_2 \sum_{i+j<n_{o_2}} \phi_{ij}(\bar{n}) \quad . \quad (4.11)$$

Equation (4.11) is just Equation (4.9) expressed in slightly different notation. Equation (4.11) can be rewritten as

$$g_T(\bar{n}) = -C_1 \sum_{i+j \leq n_s^*} i\phi_{ij}(\bar{n}) - C_2 \sum_{i+j \leq n_s^*} j\phi_{ij}(\bar{n}) + \lambda_1'(\bar{n})R_1 + \lambda_2'(\bar{n})R_2$$

$$= -C_1 L_1(\bar{n}) - C_2 L_2(\bar{n}) + \lambda_1'(\bar{n})R_1 + \lambda_2'(\bar{n})R_2$$

$$= \sum_{m=1}^{2} \lambda_m'(\bar{n})R_m - \sum_{m=1}^{2} C_m L_m(\bar{n}) = g_N(\bar{n}) \quad .$$

Next, $g_Y(\bar{n})$ is shown to be equivalent to $g_N(\bar{n})$. For this formulation, the state space simply gives the number of customers in the system. The cost per unit time in state i is $y_i = 0$. The reward for a transition from state i to j is

$$b_{i,i+1} = \begin{cases} R_1 - (i+1)C_1/\mu & \text{with probability } \lambda_1(i)/\{\sum_{m=1}^{2} \lambda_m(i)\} \\ R_2 - (i+1)C_2/\mu & \text{with probability } \lambda_2(i)/\{\sum_{m=1}^{2} \lambda_m(i)\} \end{cases}$$

and

$$b_{i,j} = 0, \quad j \neq i+1 \quad .$$

Let $\overline{\tau}_{i,j}$ be the expected holding time in state i given that the next transition is to state j. $\overline{\tau}_i$ denotes the unconditional expected waiting time in state i. Let $\lambda_m(i)$ be the mean arrival rate of class m when the state of the system is i.

$$\lambda_m(i) = \begin{cases} \lambda_m , & \text{if } i < n_{o_m} \\ 0 , & \text{otherwise} \quad . \end{cases}$$

$\overline{\tau}_i$ is given by

$$\overline{\tau}_i = \begin{cases} 1/\{\sum_{m=1}^{2} \lambda_m(i)\} , & i = 0 \\ 1/\{\mu + \sum_{m=1}^{2} \lambda_m(i)\}, & i \neq 0 \quad . \end{cases}$$

The probability of a transition from state i to j is

$$p_{i,j} = \begin{cases} 1 , & i = 0,\ j = 1 \\ \sum_{m=1}^{2} \lambda_m(i)/\{\mu + \sum_{m=1}^{2} \lambda_m(i)\} , & i \neq 0,\ j = i+1 \\ \mu/\{\mu + \sum_{m=1}^{2} \lambda_m(i)\} , & i \neq 0,\ j = i-1 \\ 0 , & \text{otherwise} \quad . \end{cases}$$

Thus, r_i, the expected reward per occupancy of state i is

$$r_i = \sum_{j=o}^{n_s^*} P_{i,j} b_{i,j} = P_{i,i+1} b_{i,i+1} \quad .$$

The expected reward per unit time in state i is given by

$$\begin{aligned} q_i = r_i/\overline{\tau}_i &= (P_{i,i+1} b_{i,i+1})/\overline{\tau}_i \\ &= \lambda_1(i)\{R_1 - (i+1)C_1/\mu\} \\ &\quad + \lambda_2(i)\{R_2 - (i+1)C_2/\mu\} \quad . \end{aligned}$$

Assume, without loss of generality, that $n_{o_1} \geq n_{o_2}$. Again, from Howard (1960),

$$\begin{aligned} g_Y(\overline{n}) &= \sum_{i=o}^{n_s^*} \phi_i(\overline{n})\, q_i \\ &= \sum_{i=o}^{n_{o_1}-1} \phi_i(\overline{n})\lambda_1\{R_1 - (i+1)C_1/\mu\} \\ &\quad + \sum_{i=o}^{n_{o_2}-1} \phi_i(\overline{n})\lambda_2\{R_2 - (i+1)C_2/\mu\} \\ &= \lambda_1 R_1 \sum_{i=o}^{n_{o_1}-1} \phi_i(\overline{n}) + \lambda_2 R_2 \sum_{i=o}^{n_{o_2}-1} \phi_i(\overline{n}) \\ &\quad - C_1 \sum_{i=o}^{n_{o_2}-1} (\lambda_1/\mu)(i+1)\phi_i(\overline{n}) \\ &\quad - C_1 \sum_{i=n_{o_2}}^{n_{o_1}-1} (\lambda_1/\mu)(i+1)\phi_i(\overline{n}) \end{aligned}$$

$$- C_2\sum_{i=o}^{n_{o_2}-1}(\lambda_2/\mu)(i + 1)\phi_i(\bar{n})$$

$$= \lambda_1'(\bar{n})R_1 + \lambda_2'(\bar{n})R_2$$

$$- \{\lambda_1/(\lambda_1 + \lambda_2)\}C_1 \sum_{i=o}^{n_{o_2}-1}\{(\lambda_1 + \lambda_2)/\mu\}(i + 1)\phi_i(\bar{n})$$

$$- \{\lambda_2/(\lambda_1 + \lambda_2)\}C_2 \sum_{i=o}^{n_{o_2}-1}\{(\lambda_1 + \lambda_2)/\mu\}(i + 1)\phi_i(\bar{n})$$

$$- C_1 \sum_{i=n_{o_2}}^{n_{o_1}-1}(\lambda_1/\mu)(i + 1)\phi_i(\bar{n}) \quad . \tag{4.12}$$

The term $\{-C_1\sum_{i=n_{o_2}}^{n_{o_1}-1}(\lambda_1/\mu)(i + 1)\phi_i(\bar{n})\}$ only appears if $n_{o_1} \geq n_{o_2} + 1$.

For a finite capacity M/M/1 queue, $\phi_{i+1}(\bar{n}) = (\lambda_i/\mu_{i+1})\phi_i(\bar{n})$ if $0 \leq i<n_o^*$; here, $\lambda_i = \sum_{m=1}^{2} k_m(i)\lambda_m$, $\mu_{i+1} = \mu$, and $n_o^* = \max_m\{n_{o_m}\}$. Thus,

$$g_Y(\bar{n}) = \sum_{m=1}^{2}\lambda_m'(\bar{n})R_m - \{\lambda_1/(\lambda_1 + \lambda_2)\}C_1 \sum_{i=o}^{n_{o_2}-1}(i + 1)\phi_{i+1}(\bar{n})$$

$$- \{\lambda_2/(\lambda_1 + \lambda_2)\}C_2 \sum_{i=o}^{n_{o_2}-1}(i + 1)\phi_{i+1}(\bar{n})$$

$$- C_1 \sum_{i=n_{o_2}}^{n_{o_1}-1}(i + 1)\phi_{i+1}(\bar{n}) \quad . \tag{4.13}$$

$\sum_{i=o}^{n_{o_1}-1}(i + 1)\phi_{i+1}(\bar{n}) = L(\bar{n})$, the expected number in the system. Thus,

$$\{\lambda_1/(\lambda_1+\lambda_2)\}\sum_{i=o}^{n_{o_2}-1}(i+1)\emptyset_{i+1}(\bar{n}) + \sum_{i=n_{o_2}}^{n_{o_1}-1}(i+1)\emptyset_{i+1}(\bar{n})$$

is the contribution of class one to $L(\bar{n})$ and

$$\{\lambda_2/(\lambda_1+\lambda_2)\}\sum_{i=o}^{n_{o_2}-1}(i+1)\emptyset_{i+1}(\bar{n})$$

is the contribution of class two to $L(\bar{n})$. Finally,

$$g_Y(n) = \sum_{m=1}^{2}\lambda'_m(\bar{n})R_m - C_1L_1(\bar{n}) - C_2L_2(\bar{n})$$

$$= \sum_{m=1}^{2}\lambda'_m(\bar{n})R_m - \sum_{m=1}^{2}C_mL_m(\bar{n}) \quad .$$

This establishes

<u>Theorem 4.1</u>: $g_N(\bar{n}) = g_Y(\bar{n}) = g_T(\bar{n})$.

Since the three formulations of the social optimum problem are equivalent, the easiest to solve formulation, $g_Y(\bar{n})$ with its simple state space and the availability of policy iteration and linear programming solution techniques, is chosen for use in this paper. The solution techniques for this problem are presented in Sections 4.6 and 4.7.

4.5 The Form of the Optimal Solution

For the social optimum problem with one class of customers, Yechiali (1971) proves that a control-limit policy is optimal. Also, both Naor (1969) and Yechiali (1971) prove that the forced balking point, n_o, for the social optimum problem is less than or equal to the balking point, n_s, for the individual optimum problem. The purpose of this section is to prove that these two properties of the optimal solution carry over to Model II. The following lemma is established first and then used to prove that a control-limit policy is optimal for several classes of customers.

Lemma 4.1: For the social optimum problem, reneging is not optimal.

Proof: Suppose customer A of class m arrives at time T_A and joins the system. Later, at time T_D, customer A departs the system before he is served. The actions of customer A affect no customer who arrived before him. The contribution of customer A to the net gain of the system over the interval (T_A, T_D) is $-C_m(T_D - T_A) < 0$, his holding cost for the time he is in the system. If no other customer arrives before customer A departs, his actions have no effect on those customers arriving after him. If other customers arrive during (T_A, T_D), the only effect of customer A's temporary presence in the system is to possibly keep a profitable customer from being allowed to join the system. Thus, the contribution of customer A to the system is negative when compared with his not joining the system at all.

Theorem 4.2: A control-limit policy is optimal for each class.

Proof: Suppose that there exists a class m and a state i such that in the optimal policy $k_m(i) = 1$, but $k_m(i - 1) = 0$;

that is, the optimal policy for class m is not a control-limit policy. The optimal gain rate is denoted by g^*. If $\emptyset_i = 0$, then a policy which is the same as the optimal except that $k_m(i) = 0$ also yields g^*. The case $\emptyset_i \neq 0$ is considered next. Consider the following modification to Model II. At the completion of a service, each customer returns his expected net benefit plus his occupation costs for the service just completed. The customers remain in the order they arrived, but the administrator recomputes the expected net benefit of each based on the number now ahead of him and uses the given policy to decide whether or not the customers can stay in the system. Because there is no discounting, the gain rate of this system is equivalent to that of Model II for the same policy. Since the transitions are Markovian, all relevant information about the future of the system is contained in the current state of the system. (The state of the system must give the position of every customer as in the third formulation.) A customer of class m who had joined when i customers were ahead of him would be forced to depart when the number of customers ahead of him dropped to $i - 1$, since $k_m(i - 1) = 0$. Thus, the customer would be forced to renege. By Lemma 4.1, the given policy cannot be optimal for the modified system or for Model II.

The next theorem establishes the relationship between the balking point for a class in the individual optimum problem and the forced balking point for the same class in the social optimum problem.

<u>Theorem 4.3</u>: For each class m, $n_{o_m} \leq n_{s_m}$.

Proof: First, the decision made by the administrator for a given arrival does not affect the times of arrival of any customer yet to arrive. As previously shown, n_{s_m} satisfies

$$R_m - (n_{s_m} + 1)C_m/\mu < 0 \leq R_m - n_{s_m} C_m/\mu . \qquad (4.14)$$

Suppose a customer, customer A, of class m arrives to find the state of the system $i \geq n_{s_m}$. Let T_A be the time of arrival of customer A, and T_B be the time of arrival of the next customer, customer B. In view of Equation (4.14), let $\alpha < 0$ be the expected net benefit of customer A joining the system. If all expected costs and rewards are assigned to a customer upon arrival, the contribution of the interval (T_A, T_B) to the expected net gain is $\alpha < 0$ if customer A is allowed to join. The decision regarding customer A does not affect the expected net gain of customers who arrived before him, but it does affect the expected net gain of those arriving after him. If customer A does not join, the state of the system found by customer B and all others after him is less than or equal to the state of the system if customer A joins. Since $f_m(i) = R_m - (i + 1)C_m/\mu$, the expected gain for a customer of class m joining when i are in the system, is a strictly decreasing function of i, the contribution to the net expected gain of the interval (T_B, T_C) for any $T_C > T_B$ is at least as large when customer A balks as when he joins. Thus, if $i \geq n_{s_m}$, forcing a customer of class m to balk yields a larger net expected gain or gain rate than allowing him to join.

Theorems 4.2 and 4.3 suggest a simple-minded approach to finding the optimal policy. Since a control-limit policy is optimal for each class and since the control limit is bounded by $n_{s_m}-1$ for each class m, a multidimensional search technique can be used to find the optimal policy from among the $\prod_{m=1}^{M} (n_{s_m})$ possible policies. In each of the next two sections, a more efficient solution technique than the suggested multidimensional search is described.

4.6 Policy Iteration Solution of the Social Optimum Problem

Equation (4.6) represents the optimum gain rate for a continuous time, infinite horizon, undiscounted, semi-Markov decision process with a finite state space. The policy iteration algorithm of Howard (1971) (see Algorithm C.1 in Appendix C) will be applied to a small two class problem to illustrate the method. A semi-Markov decision process is a semi-Markov process over which a decision maker, here, the administrator, has some control. The control arises from the ability at each transition to change the probability distribution for the next transition of the process. Here, the control is carried out by choosing the classes of customers that will be admitted to the system. Ignoring, for the moment, the form of the optimal policy demonstrated in the previous section, the administrator must choose from among 2^M

alternatives at each transition. However, since the transitions are memoryless and since a steady-state exists and is of interest, the decision as to which classes to admit only depends on the number of customers in the system. The expected net benefit rate of the system is influenced in two ways by these decisions:

1) The expected net benefit of the next transition depends on the classes of customers allowed to join the queue.
2) The expected time before the next transition depends on the classes of customers allowed to join the queue.

A solution to a two class example is now obtained by policy iteration to illustrate the use of the technique. In this example, class one customers receive a reward for service of $R_1 = 3$ and pay $C_1 = 4$ for each unit of time spent in the system. The arrival rate for class one is $\lambda_1 = 2$ customers per unit time. Class two customers receive $R_2 = 2$ for service and pay $C_2 = 3$ for each unit of time spent in the system. The arrival rate of class two customers is $\lambda_2 = 4$ customers per unit time. The single exponential server has a service rate capability of $\mu = 4$ customers per unit time.

From Equation (4.1), $n_{s_1} = 3$ and $n_{s_2} = 2$. Thus, from Equation (4.2), $n_s^* = 3$. The set of all possible actions or alternatives consists of $\{\overline{k}: \overline{k} = (0,0), (1,0), (0,1), \text{ or } (1,1)\}$. To avoid a trivial system, $\overline{k}(0) \neq (0,0)$; that is, if the system empties out, customers must be allowed back into it. Also, Theorem 4.3 implies that $\overline{k}(3) = (0,0)$ and $\overline{k}(2) = (k_1,0)$.

P(0,0) is the matrix of transition probabilities for alternative $\overline{k} = (0,0)$.

$$
\begin{array}{ccc}
 & & \text{To State} \\
 & & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\
P(0,0) = & \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{pmatrix} - & - & - & - \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \\
\text{From State} & &
\end{array}
$$

The dashes indicate that action $\overline{k} = (0,0)$ cannot be chosen when the state of the system is 0. For alternative (0,0), the next transition is sure to be the completion of a service. Transition matrices for the other alternatives follow.

$$
\begin{array}{ccc}
 & & \text{To State} \\
 & & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\
P(1,0) = & \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0.67 & 0 & 0.33 & 0 \\ 0 & 0.67 & 0 & 0.33 \\ - & - & - & - \end{pmatrix} \\
\text{From State} & &
\end{array}
$$

$$
\begin{array}{ccc}
 & & \text{To State} \\
 & & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\
P(0,1) = & \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0.5 & 0 & 0.5 & 0 \\ - & - & - & - \\ - & - & - & - \end{pmatrix} \\
\text{From State} & &
\end{array}
$$

$$
P(1,1) = \begin{array}{c|cccc} & \multicolumn{4}{c}{\text{To State}} \\ & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 1 & 0 & 0 \\ 1 & 0.4 & 0 & 0.6 & 0 \\ \text{From State } 2 & - & - & - & - \\ 3 & - & - & - & - \end{array}
$$

To see where the entries come from, consider the row corresponding to state 1 of P(1,0). The transition rate from state 1 to 2 is given by $\lambda_1 = 2$. The transition rate from state 1 to state 0 is given by the service rate $\mu = 4$. Thus, the total rate out of state 1 is 6 and the rate from state 1 to state 2 provides 0.33 of the total. Thus, the probability that the competing rates yield a transition from state 1 to state 2 is 0.33. Similarly, the probability that the competing rates yield a transition from state 1 to state 0 is 0.67.

Let b(0,0) be the matrix of expected rewards for the various transitions under alternative $\overline{k} = (0,0)$.

$$
b(0,0) = \begin{array}{c|cccc} & \multicolumn{4}{c}{\text{To State}} \\ & 0 & 1 & 2 & 3 \\ \hline 0 & - & - & - & - \\ \text{From State } 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 \end{array}
$$

Alternative (0,0) yields no benefits since no customers are allowed to join the system. The expected rewards for the various transitions for the other alternatives follow.

$$
b(1,0) = \begin{array}{c c} & \text{To State} \\ & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ - & - & - & - \end{bmatrix} \end{array}
$$

From State

$$
b(0,1) = \begin{array}{c c} & \text{To State} \\ & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 1.25 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \\ - & - & - & - \\ - & - & - & - \end{bmatrix} \end{array}
$$

From State

$$
b(1,1) = \begin{array}{c c} & \text{To State} \\ & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 1.5 & 0 & 0 \\ 0 & 0 & 0.67 & 0 \\ - & - & - & - \\ - & - & - & - \end{bmatrix} \end{array}
$$

From State

The entries $b_{i,i+1}$ of $b(1,1)$ are found from

$$\{\lambda_1/(\lambda_1 + \lambda_2)\}\{R_1 - (i + 1)C_1/\mu\} + \{\lambda_2/(\lambda_1 + \lambda_2)\}\{R_2 - (i+1)C_2/\mu\} .$$

The components of the vector of expected rewards per transition for alternative $\overline{k}$ are found from

$$r_i(\overline{k}) = \sum_{j=o}^{n_s^*} P_{i,j}(\overline{k})\, b_{i,j}(\overline{k}) \quad .$$

For this example, the expected reward per transition vectors are

$$\bar{r}(0,0) = \begin{bmatrix} - \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \bar{r}(1,0) = \begin{bmatrix} 2 \\ 0.33 \\ 0 \\ - \end{bmatrix}, \quad \bar{r}(0,1) = \begin{bmatrix} 1.25 \\ 0.25 \\ - \\ - \end{bmatrix}, \quad \bar{r}(1,1) = \begin{bmatrix} 1.5 \\ 0.4 \\ - \\ - \end{bmatrix} .$$

The components of the vector of expected rewards per unit time in state i under alternative $\bar{k}$ are found from

$$q_i(\bar{k}) = r_i(\bar{k})/\bar{\tau}_i(\bar{k}) .$$

Since $\hat{\bar{\tau}}(\bar{k})$, the vector of expected waiting times under action $\bar{k}$ is required, it is given below for the various alternatives.

$$\hat{\bar{\tau}}(0,0) = \begin{bmatrix} - \\ 0.25 \\ 0.25 \\ 0.25 \end{bmatrix}, \quad \hat{\bar{\tau}}(1,0) = \begin{bmatrix} 0.5 \\ 0.167 \\ 0.167 \\ - \end{bmatrix},$$

$$\hat{\bar{\tau}}(0,1) = \begin{bmatrix} 0.25 \\ 0.125 \\ - \\ - \end{bmatrix}, \quad \hat{\bar{\tau}}(1,1) = \begin{bmatrix} 0.167 \\ 0.1 \\ - \\ - \end{bmatrix} .$$

Thus, the expected gain per unit time vectors are

$$\bar{q}(0,0) = \begin{pmatrix} - \\ 0 \\ 0 \\ 0 \end{pmatrix}, \ \bar{q}(1,0) = \begin{pmatrix} 4 \\ 2 \\ 0 \\ - \end{pmatrix}, \ \bar{q}(0,1) = \begin{pmatrix} 5 \\ 2 \\ - \\ - \end{pmatrix}, \ \bar{q}(1,1) = \begin{pmatrix} 9 \\ 4 \\ - \\ - \end{pmatrix} .$$

A sequence of constants, $\{v_i\}$, for $i = 0, \ldots, n_s^*$, referred to by Howard (1971) as the relative value of the system starting out in state i, is required to start the policy iteration algorithm. As he suggests, take $v_i = 0$ for all i. These values are then used in the policy determination portion of the algorithm which is given by Equation (4.15).

For $i = 0, 1, \ldots, n_s^*$,

$$u_i(h) = \max_{\bar{k}} \left(q_i(\bar{k}) + (1/\bar{\tau}_i(\bar{k}))\left(\sum_{j=0}^{n_s^*} p_{i,j}(\bar{k})v_j - v_i\right)\right) . \quad (4.15)$$

The vector $\bar{d}(h)$ gives the policy chosen for application h of the algorithm. For this example,

$$\bar{d}(1) = \begin{pmatrix} (1,1) \\ (1,1) \\ (1,0) \\ (0,0) \end{pmatrix} .$$

In one sense, the relative value of starting out in state i, v_i, allows the administrator to anticipate net benefits from future arrivals in that state. If all v_i are set equal to zero, this information is lost and the administrator can do no better than the solution to the individual optimum problem which is given by $\bar{d}(1)$. As the policy iteration algorithm is applied again and again, better knowledge of the v_i's results so the administrator can do better than the individual optimum policy.

The gain rate and relative starting values associated with policy $\bar{d}(1)$ are then computed in the policy evaluation portion of the algorithm given by Equation (4.16).

For $i = 0, 1, \ldots, n_s^*$,

$$g\bar{\tau}_i\{d_i(h)\} + v_i = r_i\{d_i(h)\} + \sum_{j=0}^{n_s^*} P_{i,j}\{d_i(h)\}v_j \quad . \qquad (4.16)$$

In Equation (4.16), g is the expected gain rate of the system under policy $\bar{d}(h)$. Setting $v_{n_s^*}$ to zero reduces the number of unknowns in Equation (4.16) to the number of equations and yields a solution vector $\bar{v}$ whose entries are values relative to $v_{n_s^*} = 0$. Solution of this system of linear equation when policy $\bar{d}(1)$ is used yields

$$g = 2.553 \quad \text{and} \quad \{v_i\} = \{2.67,\ 1.60,\ 0.64,\ 0\} \quad .$$

This new set of v_i's is used to begin the second application or stage of the policy iteration algorithm. For this new set, Equation (4.15) yields

$$\bar{d}(2) = \begin{pmatrix} (1,1) \\ (1,0) \\ (0,0) \\ (0,0) \end{pmatrix} \quad .$$

Evaluation of policy $\bar{d}(2)$ gives

$$g = 3.692 \quad \text{and} \quad \{v_i\} = \{2.73, 1.85, 0.92, 0\} \quad .$$

This new set of v_i's leads to termination of the algorithm because $\bar{d}(3) = \bar{d}(2)$. Since the maximum i for which $k_1(i) = 1$ is one, $n_{o_1} = 2$. Similarly, $n_{o_2} = 1$.

4.7 Linear Programming Solution of the Social Optimum Problem

The formulation of the social optimum problem given by Equation (4.6) is transformed into a linear programming problem in this section. As in Section 3.5, the approaches of Yechiali (1971) and Fox (1966) are combined to yield the formulation given here. The linear programming formulation will be developed first. Then, the example of Section 4.6 will be set up as a linear program.

Since $n_{o_m} \leq n_s^*$ for all m, only states 0 through n_s^* can have positive steady state probabilities. A policy $P^* \in C_s$ is sought such that

$$g_{P^*} = \max_{P \in C_s} \frac{\sum_{i=o}^{n_s^*} \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} \phi_i^P D_i^P(\bar{k})\, r_i(\bar{k})}{\sum_{i=o}^{n_s^*} \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} \phi_i^P D_i^P(\bar{k})\, \bar{\tau}_i(\bar{k})} \quad . \tag{4.17}$$

Equation (4.17) is just Equation (4.6) written in a slightly different form. First, recall that in Equation (4.6), $D_i^P(\bar{k})$ was dropped because $k_m(i)$ carried the same information. Conversely, $k_m(i)$ is dropped here in favor of $D_i^P(\bar{k})$. Second, in Equation (4.6),

$$\sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} D_i^P(\bar{k})\, \lambda_m\{R_m - (i+1)C_m/\mu\}$$

is just $q_i(\bar{k})$, the expected gain per unit time in state i when action $\bar{k}$ is chosen under policy P. However, $q_i(\bar{k}) = r_i(\bar{k})/\bar{\tau}_i(\bar{k})$. In Equation

(4.17), the $r_i(\bar{k})$ and $\bar{\tau}_i(\bar{k})$ terms are summed separately, then divided, as opposed to Equation (4.6), where $q_i(\bar{k})$ is formed first. Equation (4.17) is the expected gain per transition divided by the expected time per transition which is indeed the expected gain per unit time. The balance equations and normalizing equation which the steady state probabilities $\{\emptyset_i D_i^P(\bar{k})\}$ must satisfy are

$$\sum_{i=o}^{n_s^*} \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} \emptyset_j^P \, D_i^P(\bar{k}) \, P_{i,j}(\bar{k}) - \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} \emptyset_i^P D_j^P(\bar{k}) = 0 \quad ,$$

$$\text{for } j = 0, 1, \ldots, n_s^* - 1 \tag{4.18}$$

and

$$\sum_{j=o}^{n_s^*} \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} \emptyset_j^P \, D_j^P(\bar{k}) = 1 \quad , \tag{4.19}$$

where $\emptyset_i^P \geq 0$ and $D_i^P(\bar{k}) \geq 0$. With the results of Fox (1966), as demonstrated in Section 3.5, the linear program with fractional objective function given by Equations (4.17) through (4.19) is equivalent to the following linear program:

$$\max \sum_{i=o}^{n_s^*} \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} y_i(\bar{k}) \; r_i(\bar{k}) \tag{4.20}$$

subject to

$$\sum_{i=o}^{n_s^*} \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} y_i(\bar{k}) P_{i,j}(\bar{k}) - \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} y_j(\bar{k}) = 0 \quad ,$$

$$j = 0, \ldots, n_s^* - 1 \tag{4.21}$$

and

$$\sum_{i=o}^{n_s^*} \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} y_i(\overline{k})\overline{\tau}_i(\overline{k}) = 1 \quad , \tag{4.22}$$

where

$$y_i(\overline{k}) = \phi_i^P D_i^P(\overline{k}) \geq 0 \quad . \tag{4.23}$$

Once Equations (4.20) to (4.23) are solved, $D_i^{P*}(\overline{k})$ can be found from

$$D_i^{P*}(\overline{k}) = y_i(\overline{k})/\{\sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} y_i(\overline{k})\} \quad . \tag{4.24}$$

Fox (1966) shows that $D_i^{P*}(\overline{k}) = 0$ or 1 and at most $n_s^* + 1$ of the $D_i^{P*}(\overline{k})$ are 1. Thus, ϕ_i^{P*} can be found from

$$\phi_i^{P*} = y_i(\overline{k})\overline{\tau}_i(\overline{k}) \quad \text{for} \quad y_i(\overline{k}) > 0 \quad . \tag{4.25}$$

The social balking point for each class m can be found from

$$n_{o_m} = 1 + \max\{i: y_i(k_1, k_2, \ldots, k_{m-1}, 1, k_{m+1}, \ldots, k_M) > 0\} \quad . \tag{4.26}$$

The optimal gain rate of the system is given by the maximum value of Equation (4.20). Thus, Equations (4.20) to (4.23) constitute the formulation of an alternate solution technique to policy iteration for the social optimum problem of Model II.

The data for the example problem of Section 4.6 are repeated below in Table 4.1.

TABLE 4.1

Two Class Example Problem

Class	R_m	C_m	λ_m
1	3	4	2
2	2	3	4

The single server provides exponentially distributed service times at a mean rate capability of four customers per unit time. n_s^* was found to be three. $P(\bar{k})$, $\bar{r}(\bar{k})$, and $\hat{\bar{\tau}}(\bar{k})$ which were also found in the previous section are repeated below for convenience.

$$P(0,0) = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{pmatrix} - & - & - & - \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array} \qquad P(1,0) = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0.67 & 0 & 0.33 & 0 \\ 0 & 0.67 & 0 & 0.33 \\ - & - & - & - \end{pmatrix} \end{array}$$

$$P(0,1) = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0.5 & 0 & 0.5 & 0 \\ - & - & - & - \\ - & - & - & - \end{pmatrix} \end{array} \qquad P(1,1) = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0.4 & 0 & 0.6 & 0 \\ - & - & - & - \\ - & - & - & - \end{pmatrix} \end{array}$$

$$\bar{r}(0,0) = \begin{pmatrix} - \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \bar{r}(1,0) = \begin{pmatrix} 2 \\ 0.33 \\ 0 \\ - \end{pmatrix} \quad \bar{r}(0,1) = \begin{pmatrix} 1.25 \\ 0.25 \\ - \\ - \end{pmatrix} \quad \bar{r}(1,1) = \begin{pmatrix} 1.5 \\ 0.4 \\ - \\ - \end{pmatrix}$$

$$\hat{\bar{\tau}}(0,0) = \begin{bmatrix} - \\ 0.25 \\ 0.25 \\ 0.25 \end{bmatrix} \quad \hat{\bar{\tau}}(1,0) = \begin{bmatrix} 0.5 \\ 0.167 \\ 0.167 \\ - \end{bmatrix} \quad \hat{\bar{\tau}}(0,1) = \begin{bmatrix} 0.25 \\ 0.125 \\ - \\ - \end{bmatrix} \quad \hat{\bar{\tau}}(1,1) = \begin{bmatrix} 0.167 \\ 0.1 \\ - \\ - \end{bmatrix}$$

Using this information, the linear programming formulation in Equations (4.20) to (4.23) for the example problem is

$$\max \quad 2y_0(1,0) + 1.25y_0(0,1) + 1.5y_0(1,1) + 0.33y_1(1,0) + 0.25y_1(0,1) + 0.4y_1(1,1)$$

subject to

$$y_1(0,0) + 0.67y_1(1,0) + 0.5y_1(0,1) + 0.4y_1(1,1) - y_0(1,0) - y_0(0,1) - y_0(1,1) = 0$$

$$y_0(1,0) + y_0(0,1) + y_0(1,1) + y_2(0,0) + 0.67y_2(1,0) - y_1(0,0) - y_1(1,0) - y_0(0,1) - y_0(1,1) = 0$$

$$0.33y_1(1,0) + 0.5y_1(0,1) + 0.6y_1(1,1) + y_3(0,0) - y_2(0,0) - y_2(1,0) = 0$$

$$0.5y_0(1,0) + 0.25y_0(0,1) + 0.167y_0(1,1) + 0.25y_1(0,0) + 0.167y_1(1,0) + 0.125y_1(0,1) + 0.1y_1(1,1) + 0.25y_2(0,0) + 0.167y_2(1,0) + 0.25y_3(0,0) = 1$$

where $y_i(k) \geq 0$.

The optimal solution of this linear program is $y_0(1,1) = 1.85$, $y_1(1,0) = 2.27$, $y_2(0,0) = 0.92$, and all other $y_i(\overline{k}) = 0.0$. The maximum value of the objective function is 3.692. From Equation (4.24), $D_0^{P*}(1,1) = D_1^{P*}(1,0) = D_2^{P*}(0,0) = 1$ and all other $D_i^{P*}(\overline{k}) = 0.0$. From Equation (4.25), $\phi_0^{P*} = 0.308$, $\phi_1^{P*} = 0.462$, and $\phi_2^{P*} = 0.23$. Finally, from Equation (4.26), $n_{o_1} = 2$ and $n_{o_2} = 1$.

4.8 Conclusion

Model I has been extended to allow for several classes of customers. The characteristics of the optimal policy are found to carry over to this extended model, Model II. A control-limit policy for each class of customers maximizes the expected net benefits per unit time. Compared with the social optimum, self-optimizing customers of each class tend to overcongest the system.

Of the three formulations of the social optimum problem that are presented, the one that assigns all expected costs and rewards to a customer at the time of his entry into · system is the easiest to solve. Two solution techniques, policy iteration and linear programming, are presented and illustrated.

CHAPTER V

ERLANG SERVICE TIMES

Now that the restriction on the number of classes of customers has been eliminated (see Chapter IV), the assumption of exponential service times which is not very satisfying in the context of an airport landing queue will be relaxed. The Erlang density which allows more flexibility than the exponential in approximating an airport's service time density is introduced. Model III extends Model II to include Erlang service times. Finally, the chapter explores two methods for solving Model III, one through policy iteration, and the other through mixed integer programming.

5.1 The Erlang Density Function

The following introduction to the Erlang density is adapted from Gross and Harris (1974). The Erlang density function is a subset of the gamma density. Recall that

$$f(x) = \{1/\{\Gamma(\alpha)\beta^{\alpha}\}\}\{x^{\alpha-1}\exp(-x/\beta)\} \quad \text{for } \alpha,\beta > 0 \text{ and } 0 \leq x < \infty$$

is the gamma density function. $\Gamma(\alpha) = \int_0^\infty x^{\alpha-1}\exp(-x)dx$ and $\exp(.)$ represents e raised to the power (.). If $f(x)$ is the density function of a random variable X, then $E(X) = \alpha\beta$ and $Var(X) = \alpha\beta^2$. The Erlang density is a gamma with $\alpha = h$ and $\beta = 1/(h\mu)$, where $h = 1, 2, \ldots$ and $\mu > 0$ is a constant. Thus, if Z has an Erlang density, then,

$$f(z) = \{1/(h-1)!\}(h\mu)^h z^{h-1}\exp(-h\mu Z), \quad Z \geq 0 \quad .$$

Also, $E(Z) = 1/\mu$ and $Var(Z) = 1/(h\mu^2)$. The graphs in Figure 5.1 demonstrate the increase in modeling flexibility of the Erlang density over the exponential density function. (Note that if $h = 1$, the Erlang density is an exponential density.) The effect on the Erlang distribution of increasing the value of h while holding the mean constant is to reduce the variance of the distribution.

To extend Model II in a straightforward manner requires a service time distribution with the Markovian or memoryless property. Of the continuous distributions, only the exponential is memoryless, so the usefulness of the Erlang distribution may at first appear doubtful. However, the moment generating function of the sum of h independent, identically distributed, exponential random variables with parameter θ is $\{\theta/(\theta - t)\}^h$. This can be written as $\{h\mu/(h\mu - t)\}^h$ which is the moment generating function of an Erlang random variable Z with $E(Z) = 1/\mu$ and $Var(Z) = 1/(h\mu^2)$. Thus, an Erlang random variable Z with parameters h and μ can be generated by the sum of h independent, identically distributed, exponential random variables each with mean $1/h\mu$. The technique used to introduce Erlang service times while maintaining the memoryless property is to artificially break a service up into h independent, identically distributed, exponential phases. Of course, only one customer is allowed in service at any one time. The phases are imposed for the convenience of mathematical tractability and do not necessarily reflect any attributes of the actual service process. To accommodate Erlang service times, the state space of the model is expanded to indicate the number of phases of service in the system. If the system contains one customer, the state of the system

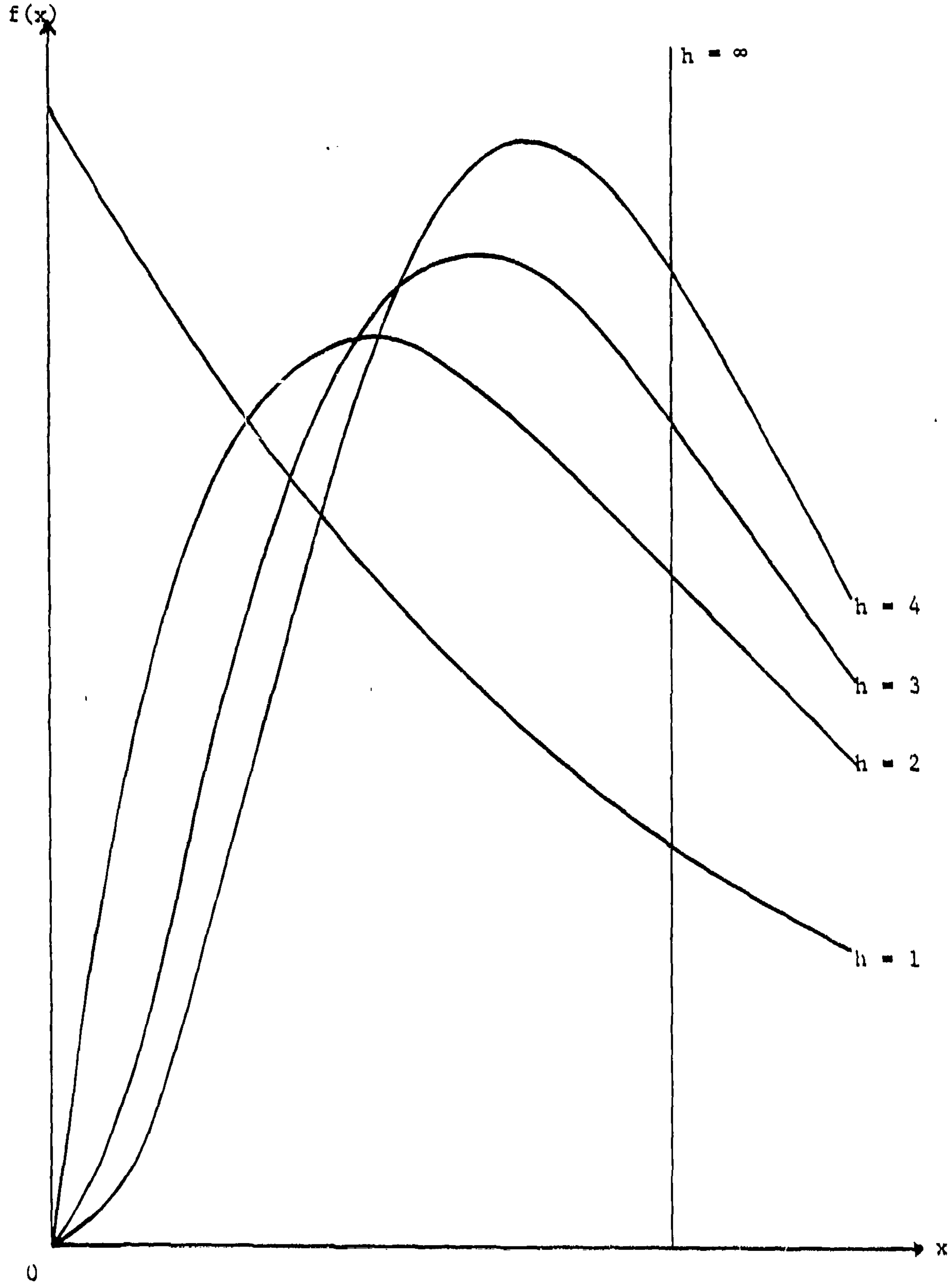

Figure 5.1 Erlang Density for Several Values of h

can be 1 to h. If the system contains two customers, the state can be h + 1 to 2h. In general, if the system contains i customers, the state can be (i - 1)h + 1 to ih.

5.2 One Class Example Problem

A simple problem is presented prior to the presentation of Model III to illustrate a potential problem that may arise so that the problem can be avoided. Consider a single class of customers with the reward for service $R = 3.5$, the cost per unit time in the system $C = 4$, the arrival rate $\lambda = 2$, and the mean service rate capability $\mu = 4$. Suppose that an Erlang 2 distribution provides a close fit to the actual service time distribution. Then, each service can be assumed to consist of 2 independent exponential phases, each with mean time $1/(h\mu) = 1/8$.

The individual optimum problem for this example is examined first. Each customer's expected service time only depends on the mean service rate which, in turn, does not depend on the form of the service time distribution. However, the calculation used previously for the expected time a customer spends in the system depends on the memoryless property of the service time distribution. Although this difficulty precludes finding the balking point for Erlang distributed service times in terms of the number of customers in the system as easily as before, the balking point can be found in terms of the number of phases of service in the system since the length of each phase of service is an exponential random variable. Let n_s' be the individual optimum balking point in terms of number of phases of service in the system. n_s' is such that

$$R - (n_s' + h)C/(h\mu) < 0 \leq R - (n_s' - 1 + h)C/(h\mu). \qquad (5.1)$$

Thus,

$$n_s' = [Rh\mu/C] - (h - 1) \quad , \tag{5.2}$$

where the brackets represent the greatest integer function. For this example, $n_s' = 6$. This policy implies that a self-optimizing arrival will balk if he finds three customers in the system and the customer in service is in the first phase of service. If three were in the system and the customer in service were in the second phase of service, the arrival would join the system. Since the phases of service are purely artificial and indistinguishable, this policy cannot be implemented. Thus, while the individual optimum solution may turn out to be one that can be implemented, the possibility exists that it may not be implementable. This uncertainty casts doubt upon the usefulness of this formulation of the problem. The same difficulty can occur in the solution to the social optimum problem.

Solution of the example for the social balking point in terms of phases of service by policy iteration or linear programming leads to the forced balking point occurring when four phases of service are present in the system. Since the service time distribution is Erlang two, the forced balking point occurs when two customers are in the system and the customer in service is in his first phase of service. Again, since the phases are not actually present in the service itself, this policy, though optimal for the model, is unrealistic and impossible to implement.

This example emphasizes the fact that an implementable policy must be one that does not force an arrival or the administrator to determine the service phase of the customer in service. Recall that a policy is a set of join or balk decisions for each class of customers for each state of the system. The set of all possible policies, the policy space, contains policies that cannot be implemented. The policy space can be reduced to only those policies that choose the same action (the same join or balk decision for the classes) for all states representing the presence of the same number of customers. Such policies are implementable because they do not require an arrival or the administrator to identify the service phase of the customer in service. Implementable policies will also be referred to as linked policies since the actions chosen for all states representing the presence of the same number of customers must be the same.

5.3 Model III

Model III has M classes of customers. The arrivals for each class m form a Poisson stream with mean rate λ_m. The service times of the single server are independent, identically distributed, Erlang random variables with mean $1/\mu$ and variance $1/(h\mu^2)$. Each service time is generated by the sum of h independent, identically distributed, exponential random variables, each with mean $1/(h\mu)$. The state space for Model III is expanded to indicate the number of phases of service in the system. Phases $(i - 1)h + 1$ to ih represent the presence of i customers, where $i = 1, 2, \ldots$. The following cost structure is imposed on the operation of the queueing system:

1) Each customer of class m that is served receives a reward of R_m dollars.
2) Each unit of time a class m customer spends in the system costs him C_m dollars.

To avoid triviality, $R_m \geq C_m/\mu$.

Again, for this model, each arriving customer is given the choice of joining and being subjected to the cost structure or balking and not paying or receiving any money. Both the individual and social optimum problems are considered. Since only the state of the system upon arrival is used to decide whether or not to join the system,[1] only the class of stationary Markovian policies, C_s, is examined for the optimum. To avoid coming up with a policy that cannot be implemented, the decisions for all of the states corresponding to the presence of the same number of customers must be identical. Therefore, as in Model II, let $D_i^P(\bar{k})$ be the probability that decision $\bar{k}$ is chosen under policy P when i customers are in the system. Decision $\bar{k} = (k_1, k_2, \ldots, k_M)$ accepts class m if $k_m = 1$ and rejects class m if $k_m = 0$.

5.4 The Individual Optimum Problem

The solution of the individual optimum problem will be investigated first. The method of phases will be used to learn as much information as possible about the solution to the problem. When this method runs into difficulty, other methods will be used to finally determine $\{n_{s_m}\}$ for $m = 1, 2, \ldots, M$.

First, remove the restriction that all decisions made for the states representing a single customer be identical. As in Section 5.2, n'_{s_m}, the optimal individual balking point for class m in terms of

---

[1]Again, using the state of the system is the best a customer can do since the transitions are memoryless and the horizon is infinite.

phases is given by Equation (5.2) as

$$n'_{s_m} = [R_m h\mu/C_m] - (h - 1) \quad .$$

This balking point in terms of phases may translate into a balking point in terms of customers that cannot be implemented. If this happens, n_{s_m}, the individual balking point for class m in terms of the number of customers in the system, is either $[(n'_{s_m} + h - 1)/h]$ or $[(n'_{s_m} + h - 1)h] + 1$. That is, n_{s_m} is either the last complete customer represented by n'_{s_m} or one more than that. This is really enough information if the individual optimum is only used to bound the state space for the social optimum problem. (In Section 5.5, n_{o_m}, the social balking point for class m is shown to be less than or equal to n_{s_m}.) The larger number of customers, $[(n'_{s_m} + h -1)/h] + 1$, can serve as the bound for each class. The individual balking point for each class can be found, if desired, by different means.

One method of obtaining the individual balking points is an extension of the approach used in Naor (1969) for Model I. A self-optimizing customer will join the system if his expected net benefit for joining is greater than zero which is his expected net benefit for balking. Since $R_m \geq C_m/\mu$, $[(n'_{s_m} + h - 1)/h] \geq 1$, so there must be at least one customer in the system for an arrival to consider balking. For each customer in the system but not in service, the expected service time is $1/\mu$. The difficulty with Erlang service times is that the expected remaining service time for the customer in service is not $1/\mu$ since the Erlang distribution is not memoryless. Thus, the problem is to find the expected remaining service time of the customer in service. Since the arrivals for each class form a Poisson stream, an arrival is equally likely to occur at any point during a service. Following Kleinrock (1975), the expected remaining service time found by an arbitrary arrival is

$$E(\text{Remaining Service Time}) = E(RST) = \{E^2(S) + Var(S)\}/\{2E(S)\} \quad , \tag{5.3}$$

where S is a random variable representing the complete service time of a customer. For Erlang service times with $E(S) = 1/\mu$ and $Var(S) = 1/h\mu^2)$, Equation (5.3) becomes

$$E(RST) = (h + 1)/(2h\mu) \quad .$$

The individual balking point in terms of customers can be found from

$$R_m - C_m\{n_{s_m}/\mu + (h + 1)/(2h\mu)\} < 0 \leq R_m - C_m\{(n_{s_m} - 1)/\mu + (h + 1)/(2h\mu)\} \quad . \tag{5.4}$$

This leads to

$$n_{s_m} = [\mu R_m/C_m + (h - 1)/(2h)] \quad , \tag{5.5}$$

where the brackets indicate the greatest integer function. The set of individual balking points determined for the classes by Equation (5.5) solves the individual optimum problem.

Equation (5.5) can also be developed using an extension of the approach of Yechiali (1971) for Model I. Since each customer considers only his own net benefit in deciding whether or not to join the system and since all customers of a given class face the same cost structure, each member of a class makes the same decision when a given number of customers is in the system. Thus, a policy $P^* \epsilon C_s$ is sought such that P^* yields

$$\max_{P \epsilon C_s} \sum_{i=o}^{\infty} D_i^P(\bar{k}) \sum_{m=1}^{M} k_m(i)\{R_m - iC_m/\mu - C_m\, E(RST)\}$$

$$+ D_0^P(k) \sum_{m=1}^{M} k_m(0)(R_m - C_m/\mu) \quad . \qquad (5.6)$$

The units of Equation (5.6) are dollars per customer so it maximizes the expected gain per customer for self-optimizing customers. Equation (5.6) is analogous to Equation (4.3) in the extension of Yechiali's formulation to Model II. Again, $D_i^P(\bar{k})$ is the probability that decision $\bar{k}$ is chosen under policy P when i customers are in the system. $k_m(i) = 0$ if class m balks when i customers are in the system, and $k_m(i) = 1$ if class m joins when i are in the system. Equation (5.6) can be maximized by setting $D_i^P(\bar{k}) = 1$ for all i and all $\bar{k}$ and setting

$$k_m(i) = \begin{cases} 1 & \text{if } R_m - iC_m/\mu - C_m\, E(RST) \geq 0 \\ 0 & \text{otherwise,} \quad \text{for } m = 1, 2, \ldots, M. \end{cases}$$

The arguments in the previous development can now be used to develop Equation (5.5). Then,

$$k_m(i) = \begin{cases} 1 & \text{if } i < n_{s_m} \\ 0 & \text{otherwise,} \quad \text{for } m = 1, 2, \ldots, M. \end{cases}$$

An interesting twist to the normal pattern of development used in this paper provides numerical results to confirm Equation (5.5); in particular, to demonstrate that n_{s_m} can indeed be $[(n'_{s_m} + h - 1)/h] + 1$. As noted in Lippman and Stidham (1977), the difference between the self-optimizer and the social optimizer is that the self-optimizer fails to consider the decrease in benefits to later arriving customers caused by his joining the queue. This effect is called an external economic effect. Although Lippman and Stidham investigate systems like the M/M/s system, the external economic effect applies equally well to Model III. As $\lambda_m \to 0$ for all m = 1, 2, ..., M, the effect of a given customer's decision to join on later arriving customers declines since the expected interarrival time between customers is large compared with $1/\mu$, the expected service time. Thus, it seems reasonable to assume that as $\lambda_m \to 0$ for all m = 1, 2, ..., M, n_{o_m} approaches n_{s_m} from below (the next section establishes $n_{o_m} \leq n_{s_m}$ for all m). The twist is to find a social optimum problem for which $n_{o_m} = [(n'_{s_m} + h - 1)/h] + 1$, which implies that n_{s_m} is also $[(n'_{s_m} + h - 1)/h] + 1$.

Using the policy iteration method (see Section 5.5) for solving the social optimum problem, a simple example was solved for which $n_{s_m} = [(n'_{s_m} + h - 1)/h] + 1$ for some m. In the example, a single class of customers is given a reward R = 3.99 for service and charged C = 4 per unit time spent in the system. The service time distribution is modeled as an Erlang 2 distribution with each phase of service having a mean service time of 1/8. The arrival rate of the customers is 0.4. From Equation (5.2),

$$n'_s = [Rh\mu/C] - (h - 1) = 6.$$

Thus,

$$[(n'_s + h - 1)/h] = 3$$

and

$$[(n'_s + h - 1)/h] + 1 = 4.$$

From Equation (5.5),

$$n_s = [\mu R/C + (h - 1)/(2h)] = 4 .$$

The policy iteration method of Section 5.5 applied to this problem yields $n_o = 4$ which implies that $n_s = 4$ and confirms the result of Equation (5.5).

5.5 Social Optimum Using Policy Iteration

A policy iteration approach to the social optimum problem is considered next. With a few minor modifications, the proof of Theorem 4.3 can be adapted to establish that $n_{o_m} \leq n_{s_m}$, for all m, for Model III.[1] Thus, a bound on the state space required for policy iteration can be determined from Equation (5.5) through

$$n_s^* = \max_m \{n_{s_m}\} . \qquad (5.7)$$

A policy $P^* \varepsilon C_s$, the class of stationary Markovian policies, is sought such that P* yields

$$\max_{P\varepsilon C_s} g_P = \max_{P\varepsilon C_s} \left\{ \sum_{i=1}^{n_s^*} D_i^P(\bar{k}) \emptyset_i^P \sum_{m=1}^{M} k_m(i)\lambda_m\{R_m - iC_m/\mu - C_m\ E(RST)\} \right.$$

$$\left. + D_0^P(\bar{k})\emptyset_0^P \sum_{m=1}^{M} k_m(0)\lambda_m(R_m - C_m/\mu) \right\} . \qquad (5.8)$$

The units of Equation (5.8) are dollars per unit time. As in Models I and II, the main difference between the formulation of the individual and social optimum problems is that the social optimum formulation

---

[1]Since the optimal policy has not been shown to be a control-limit policy, n_{o_m} needs to be redefined as $n_{o_m} = 1 + \max\{i: k_m(i) = 1\}$. n_{o_m} will still be referred to as a balking point.

includes information, here, $\emptyset_i^P$, that allows the administrator to anticipate net benefits from customers who have yet to arrive.

According to Derman (1962), a nonrandomized rule $P\varepsilon C_d$ maximizes Equation (5.8). Thus, $D_i^P(\bar{k})$ can be dropped from Equation (5.8) since it will be one for the decision chosen for each i and zero otherwise. Equation (5.8) then becomes

$$\max_{P\varepsilon C_s \cap C_d} g_P = \max_{P\varepsilon C_s \cap C_d} \left\{ \sum_{i=1}^{n_s^*} \emptyset_i^P \sum_{m=1}^{M} k_m(i)\lambda_m \{R_m - iC_m/\mu - C_m\, E(RST)\} + \emptyset_0^P \sum_{m=1}^{M} k_m(0)\lambda_m(R_m - C_m/\mu) \right\}. \quad (5.9)$$

It should be obvious that if Equation (5.9) works for Erlang service times, it works for any general service time distribution with a finite mean and variance since E(RST) can be found from Equation (5.3). However, Equation (5.9) cannot be solved by the techniques at hand since in the formulation in Equation (5.9), the problem is not represented as a semi-Markov (or Markov) decision process. To demonstrate this, consider that if the system enters state i from state i - 1, an arrival has occurred and $\bar{\tau}_i$, the expected waiting time in state i, is E(RST), while if the system enters state i from state i + 1, a service occurred and the expected waiting time in state i is $1/\mu$, the expected service time. Unless the service times are exponentially distributed, $E(RST) \neq 1/\mu$. Thus, another approach is required to solve the social optimum problem.

The inability to solve Equation (5.9) using the techniques developed leads back to the method of phases. The social optimum problem without the restriction requiring the same decision be made for every state representing the presence of the same number of customers can be written as

$$\max_{P \in C_s} g_P = \max_{P \in C_s} \sum_{j=o}^{hn_s^*} D_j^P(\overline{k}) \phi_j^P \sum_{m=1}^{M} k_m(j)\lambda_m\{R_m - (j + h)C_m/(h\mu)\} \quad . \tag{5.10}$$

Again, the results of Derman (1962) imply that $D_j^P(\overline{k})$ can be dropped from the formulation since a nonrandomized rule is optimal so that Equation (5.10) becomes

$$\max_{P \in C_s \cap C_d} g_P = \max_{P \in C_s \cap C_d} \sum_{j=o}^{hn_s^*} \phi_j^P \sum_{m=1}^{M} k_m(j)\lambda_m\{R_m - (j + h)C_m/(h\mu)\} \quad . \tag{5.11}$$

This is the same formulation that led to difficulties in Section 5.2; that is, the optimal policy may be one that cannot be implemented since it may require arrivals to identify the service phase of the customer in service.

Since linking decisions places additional constraints on the problem, the optimal solution to the "linked" problem yields a gain rate that is no greater than that found by solving Equation (5.11). A simple way to find a good linked policy is to use the following algorithm:

Algorithm 5.1:

a) Use policy iteration to solve Equation (5.11) for g and n'_{o_m} for all m, where n'_{o_m} is the optimal forced balking point in terms of phases of service in the system.

b) If the optimal solution can be implemented, stop. If not, go to Step (c).

c) For each class whose balking point occurs at the last phase of a customer's service, fix $n_{o_m} = \left[(n'_{o_m} + h - 1)/h\right]$. For the other classes, n_{o_m} is either $\left[(n'_{o_m} + h - 1)h\right]$ or $\left[(n'_{o_m} + h - 1)/h\right] + 1$.

d) Among all possible sets generated from Step (c), find the set $\{n_{o_m}\}$, $m = 1, 2, \ldots, M$, of balking points that yields the maximum gain rate.

This procedure is fairly easy to implement, although it may require use of the policy evaluation portion of the policy iteration algorithm on 2^M policies in Step (d). However, it may not find the optimal solution since it searches only a few linked policies in the region of the phase optimum. In fact, the few policies examined are control-limit policies. (Although the proof of Theorem 4.2 does not directly apply to Erlang service times, it seems reasonable to assume that the optimal policy will again be a control-limit policy.) A more thorough search of the linked policies lends itself to a mixed integer programming formulation.

5.6 Social Optimum Using Mixed Integer Programming

This section extends the phases of service formulation of Model III so that all possible linked policies, policies that can be implemented, are examined to determine the optimal policy. Recall that a linked policy makes the same decision for all states that represent the presence of the same number of customers.

Equation (5.11) represents the social optimum problem in terms of phases of service in the system with no restriction on the decisions made. Equation (5.11) can be formulated as a linear program similar to that developed in Chapter IV [see Equations (4.20) to (4.23)].

$$\max \sum_{i=o}^{hn_s^*} \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} y_i(\bar{k}) r_i(\bar{k})$$

subject to

$$\sum_{i=o}^{hn_s^*} \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} y_i(\bar{k}) P_{i,j}(\bar{k}) - \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} y_j(\bar{k}) = 0 \ ,$$

$$j = 0, 1, \ldots, (hn_s^* - 1)$$

$$\sum_{i=o}^{hn_s^*} \sum_{k_1=o}^{1} \cdots \sum_{k_M=o}^{1} y_i(\bar{k}) \bar{\tau}_i(\bar{k}) = 1$$

$$y_i(\bar{k}) \geq 0 \quad . \qquad (5.12)$$

hn_s^* bounds the total number of phases of service required for the social optimum problem, where h is the number of phases per service and n_s^* is found from Equations (5.5) and (5.7). $r_i(\bar{k})$ is the expected reward per occupancy of state i when decision $\bar{k}$ is chosen. $\bar{\tau}_i(\bar{k})$ is the expected waiting time in state i if action $\bar{k}$ is chosen. $P_{i,j}(\bar{k})$ is the probability of a transition from state i to j if action $\bar{k}$ is chosen. $y_i(\bar{k})$ can best be explained by observing that the stationary probability of the system being in state i under policy P is $\phi_i^P = y_i(\bar{k})\bar{\tau}_i(\bar{k})$ for $y_i(\bar{k}) > 0$. Thus, $y_i(\bar{k})$ is in essence a weighted probability that the system is in state i under policy P. Policy P is a set of decisions $\{\bar{k}(i)\}$, $i = 0, 1, \ldots, hn_s^*$.

The only remaining task is to include in the formulation the requirement that the decision for each class be the same for the states of the system that represent the presence of the same number of customers. In general, states $(i - 1)h + 1$ to ih represent the presence of i customers, where $i = 1, 2, \ldots, n_s^*$. If the policy is to be one that can be implemented, the decisions $\overline{k}(j) = \{k_1(j), k_2(j), \ldots, k_M(j)\}$ made for states $j = (i - 1)h + 1$ to ih must be the same. If $i = 1, 2, \ldots, n_s^* - 1$, 2^M actions are possible. [The number of actions for some states could be reduced by using Equation (5.5) to set $k_m(j) = 0$ for all $j = (i - 1)h + 1$ to ih, where $i \geq n_{s_m}$.] Thus, for a policy to be implementable, the actions chosen for the states representing each of $i = 1, 2, \ldots, n_s^* - 1$ customers must be either

$$\overline{k}\{(i - 1)h + 1\} = \overline{k}\{(i - 1)h + 2\} = \ldots = \overline{k}(ih) = (0,0,\ldots,0)$$

or

$$\overline{k}\{(i - 1)h + 1\} = \overline{k}\{(i - 1)h + 2\} = \ldots = \overline{h}(ih) = (1,0,\ldots,0)$$

or

$$\vdots$$

or

$$\overline{k}\{(i - 1)h + 1\} = \overline{k}\{(i - 1)h + 2\} = \ldots = \overline{k}(ih) = (1,1,\ldots,1) \quad .$$

These either/or constraints do not fit the usual form of linear programming constraints. The method suggested in Taha (1971) will be used to convert the either/or constraints to constraints that can be used in a mixed integer program.

For $i = 1, 2, \ldots, n_s^* - 1$, let

$$Z_i(1,0,\ldots,0) = \begin{cases} 1 & \text{if } \overline{k}\{(i - 1)h + 1\} = \ldots = \overline{k}(ih) = (1,0,\ldots,0) \\ 0 & \text{otherwise.} \end{cases}$$

$$\vdots$$

$$Z_i(1,1,\ldots,1) = \begin{cases} 1 & \text{if } \overline{k}\{(i - 1)h + 1\} = \ldots = \overline{k}(ih) = (1,1,\ldots,1) \\ 0 & \text{otherwise.} \end{cases}$$

$Z_i(0,0,\ldots,0)$ is not needed since $\bar{k}$ must be $\bar{0}$ if $\bar{k} \neq (1,0,\ldots,0)$, and ...., and $\bar{k} \neq (1,1,\ldots,1)$. The $\{Z_i(\bar{k})\}$ are binary variables which indicate whether or not action $\bar{k}$ is chosen when i customers are in the system. If B is a large positive number, the either/or constraints are equivalent to

$$-BZ_i(1,0,\ldots,0) + \sum_{j=(i-1)h+1}^{ih} y_j(1,0,\ldots,0) \leq 0$$

$$-B\{1 - Z_i(1,0,\ldots,0)\} + \sum_{j=(i-1)h+1}^{ih} \sum_{\bar{k}\neq(1,0,\ldots,0)} y_j(\bar{k}) \leq 0$$

$$\vdots$$

$$-BZ_i(1,1,\ldots,1) + \sum_{j=(i-1)h+1}^{ih} y_j(1,1,\ldots,1) \leq 0$$

$$-B\{1 - Z_i(1,1,\ldots,1)\} + \sum_{j=(i-1)h+1}^{ih} \sum_{\bar{k}\neq(1,1,\ldots,1)} y_j(\bar{k}) \leq 0$$

$$\text{where } i = 1, 2, \ldots, n_s^* - 1. \qquad (5.13)$$

To illustrate the equivalence of Equation (5.13) to the either/or constraints, consider that if action $(1,0,\ldots,0)$ is chosen for each state from $(i - 1)h + 1$ to ih, then $Z_i(1,0,\ldots,0) = 1$ and all other $Z_i(\bar{k}) = 0$. Thus,

$$-BZ_i(1,0,\ldots,0) + \sum_{j=(i-1)h+1}^{ih} y_j(1,0,\ldots,0) = -B$$

$$+ \sum_{j=(i-1)h+1}^{ih} y_j(1,0,\ldots,0) \leq 0$$

since B is a large positive number. Since $y_j(\bar{k}) \geq 0$ and $-B\{1 - Z_i(1,0,\ldots,0)\} = 0$,

$$-B\{1 - Z_i(1,0,\ldots,0)\} + \sum_{j=(i-1)h+1}^{ih} \sum_{\bar{k}\neq(1,0,\ldots,0)} y_j(\bar{k}) \leq 0$$

only if all $y_j(\bar{k})$, $\bar{k} \neq (1,0,\ldots,0)$ and $j = (i-1)h+1, \ldots, ih$, are zero. Since action $(1,0,\ldots,0)$ is chosen for all states $(i-1)h+1$ to ih, these $y_j(\bar{k})$ are indeed zero. The other constraints in Equation (5.13) are also satisfied. Thus, choosing action $\bar{k} = (1,0,\ldots,0)$ for all states representing customer i satisfies Equation (5.13). Similar arguments can be made for any other action chosen for all the states representing a single customer. If different actions are chosen for the states representing a single customer, all $Z_i(\bar{k})$ are zero and at least one of the

$$-BZ_i(\bar{k}) + \sum_{j=(i-1)h+1}^{ih} y_j(\bar{k}) \leq 0$$

constraints is violated. Thus, only those policies that choose the same action for all states representing a single customer satisfy Equation (5.13).

Equation (5.13) allows at most h of the $\{y_j(\bar{k})\}$, $j = (i-1)h+1, \ldots, ih$; all $\bar{k}$, to be positive. Since $y_j(\bar{k}) = \phi_j^P/\bar{\tau}_j(\bar{k})$ and ϕ_j^P is a probability,

$$y_j(\bar{k}) \leq 1/\left(\min_{j,\bar{k}} \{\bar{\tau}_j(\bar{k})\}\right) \quad .$$

Thus, the large positive number, B, required in Equation (5.13) is determined by

$$B \geq h/\left(\min_{j,\bar{k}}\{\bar{\tau}_j(\bar{k})\}\right) \quad .$$

Equation (5.13) does not constrain the decision when zero or n_s^*

customers are in the system. When n_s^* customers are in the system, action $\overline{k} = (0,0,\ldots,0)$ must be chosen since $n_{c_m} \leq n_s^*$ for all m. Thus,

$$\overline{k}\{(n_s^* - 1)h + 1\} = \ldots = \overline{k}(n_s^* h) = (0,0,\ldots,0) \quad . \qquad (5.14)$$

Zero customers are in the system only if zero phases of service are in the system, so only one state represents the condition of having zero customers in the system. Thus, the problem of linking decisions does not arise when zero customers are in the system. Equations (5.12) to (5.14) constitute a mixed integer programming formulation of Model III. The first example of Section 5.2 will be formulated as a mixed integer program to illustrate the method.

A single class of customers receives a reward $R = 3.5$ for service and is charged $C = 4$ for every unit of time spent in the system. The mean arrival rate of the Poisson stream of customers is $\lambda = 2$ customers per unit time. The mean service rate capability of the single server is $\mu = 4$ customers per unit time. Suppose that an Erlang two distribution provides a reasonable model of the service time distribution. From Equations (5.5) and (5.7),

$$n_s^* = n_s = [\mu R/C + (h - 1)/(2h)] = 3 \quad .$$

Thus, $hn_s^* = 6$ is a bound on the number of phases of service required for the social optimum problem. The mean service time of each phase is $1/(h\mu) = 1/8$. The transition probabilities are

$$P_{j,j-1}(0) = 1,\ j = 1,\ 2,\ \ldots,\ 6 \quad ,$$

$$P_{0,2}(1) = 1$$

and

$$P_{i,j}(1) = \begin{cases} 0.2, & j = i + 2 \\ 0.8, & j = i - 1 \\ 0\ , & \text{otherwise}, i = 1, 2, \ldots, 4 \end{cases} .$$

(Recall that decision 0 cannot be chosen if $i = 0$, and decision 1 cannot be chosen if $i = 5$ or 6.) Calculation of the expected reward per occupancy of a state yields

$$r_j(0) = 0, \quad j = 1, 2, \ldots, 6$$

and

$$r_j(1) = P_{j,j+h}(1)\{R - (j + h)C/(h\mu)\}; \quad j = 0, 1, \ldots, 4 \quad .$$

The expected waiting times are

$$\bar{\tau}_j(0) = 0.125, \quad j = 1, 2, \ldots, 6$$

and

$$\bar{\tau}_j(1) = \begin{cases} 0.5, & j = 0 \\ 0.1, & j = 1, 2, 3, 4 \end{cases} .$$

Also, $B \geq h/\left(\min\{\bar{\tau}_j(k)\}\right) = 2/0.1 = 20$; let $B = 25$. With this information, Equation (5.12) to (5.14) yield

$$\max 2.5y_0(1) + 0.4y_1(1) + 0.3y_2(1) + 0.2y_3(1) + 0.1y_4(1)$$

subject to

$$y_0(1) - y_1(0) - 0.8y_1(1) = 0$$

$$y_1(0) + y_1(1) - y_2(0) - 0.8y_2(1) = 0$$

$$y_2(0) + y_2(1) - y_0(1) - y_3(0) - 0.8y_3(1) = 0$$

$$y_3(0) + y_3(1) - 0.2y_1(1) - y_4(0) - 0.8y_4(1) = 0$$

$$y_4(0) + y_4(1) - 0.2y_2(1) - y_5(0) = 0$$

$$y_5(0) - 0.2y_3(1) - y_6(0) = 0$$

$$0.5y_0(1) + 0.125y_1(0) + 0.1y_1(1) + 0.125y_2(0) + 0.1y_2(1)$$
$$+ 0.125y_3(0) + 0.1y_3(1) + 0.125y_4(0) + 0.1y_4(1)$$
$$+ 0.125y_5(0) + 0.125y_6(0) = 1$$

$$-25Z_1(1) + y_1(1) + y_2(1) \leqq 0$$

$$-25\{1 - Z_1(1)\} + y_1(0) + y_2(0) \leqq 0$$

$$-25Z_2(1) + y_3(1) + y_4(1) \leqq 0$$

$$-25\{1 - Z_2(1)\} + y_3(0) + y_4(0) \leqq 0 \quad ,$$

where

$$Z_1(1),\ Z_2(1) = 0 \text{ or } 1;\ y_0(1) \geqq 0,\ y_j(k) \geqq 0 \text{ for } j = 1, \ldots, 4;$$
$$k = 0,1;\ y_5(0) \geqq 0,\ y_6(0) \geqq 0 \quad .$$

The optimal solution to this problem is

$$y_0(1) = 1.12,\ y_1(1) = 1.4,\ y_2(1) = 1.75,\ y_3(0) = 0.63,$$
$$y_4(0) = 0.35,\ Z_1(1) = 1 \quad .$$

All other $y_j(k)$ and $Z_2(1)$ are 0. The optimal gain rate is $g = 3.895$. This solution gives an optimal social balking point of $n_o = 2$.

5.7 Conclusion

In this chapter, the model was generalized to provide an Erlang service distribution for the server. This was done because the flexibility of the Erlang distribution will be needed in Chapter VI to develop a realistic model of the actual service time distribution for an airport landing system. For the policy iteration solution procedure, a control-limit solution was assumed; however, no restrictions were placed on the form of the solution found by mixed integer programming.

The policy iteration procedure is a heuristic algorithm that is easy to implement. On the other hand, the mixed integer programming formulation allows identification of the optimal solution but becomes cumbersome for large problems. The policy iteration procedure requires the inversion of a matrix that has as many rows and columns as there are states. The mixed integer program has an equality constraint for each state. These problems may hamper the solution of problems with several hundred states or more; however, the user of these methods should be able to alleviate such problems by taking advantage of the sparseness of the matrix (constraints). Since the policy iteration procedure is easy to implement, the user should consider trying it first. If its results are unsatisfactory, the user can then try mixed integer programming.

CHAPTER VI

AIRPORT LANDING QUEUE APPLICATION

In this chapter, Models II and III are applied to airport landing queues. Data from several sources are used to develop the parameters of the models; however, the particular airport that is modeled is the Greater Pittsburgh International Airport. The work of Adler and Naor (1969) is used to help refine the results obtained for the Pittsburgh Airport.

6.1 Customer Parameters

The purpose of this chapter is to determine how entry to the landing queue should be controlled during peak traffic periods when a single runway is being used for landings only. The use of a peak traffic period to determine the control policy makes the benefits of a control policy apparent. In light traffic, little or no control is required. Thus, a peak traffic period gives the best indication of an airport's capability to land aircraft effectively. Only commercial jet aircraft are considered in this study, although other traffic uses the Pittsburgh Airport. The other traffic generally uses another runway during periods of heavy use of the Pittsburgh Airport so that the models can be applied realistically to commercial jets only.

Commercial jet aircraft are divided into the following five classes:

1) Class One - Three-engine, wide body
2) Class Two - Four-engine, wide body
3) Class Three - Three-engine, regular body
4) Class Four - Four-engine, regular body
5) Class Five - Two-engine

The aircraft that make up the various classes are listed in Table 6.1.

TABLE 6.1

Aircraft Categorized by Class

Class	Manufacturer	Aircraft
1	Lockheed	L-1011
	McDonnell-Douglas	DC-10
2	Boeing	747
3	Boeing	727
4	Boeing	707
	McDonnell-Douglas	DC-8
5	British Aircraft Corp.	BAC-111
	McDonnell-Douglas	DC-9

The mean arrival rate of each class, λ_m, is determined first. The Transportation System Center (1978) lists the hourly number of landings on 4 August 1978 for the Pittsburgh Airport. A peak number of 32 landings occurred from 1600-1700 hours and again from 1900-2000 hours. Thus, the overall mean arrival rate used in this study is 32 aircraft per hour. Rather than find class arrival rates by attempting to determine which aircraft arrived during these periods, class arrival rates are approximated by first determining the proportion of total yearly passenger jet traffic at the Pittsburgh Airport that is represented by each class. The mean arrival rate of each class during a peak period is approximated by multiplying 32 by the proportion of yearly jet traffic that each class represents. The Civil Aeronautics Board and Federal Aviation Administration (1978) list the total number of departures (and thus, arrivals) from the Pittsburgh Airport for each type of aircraft for a 12-month period ending 30 June 1978. The percentage of the total number of commercial jet departures and the computed mean arrival rate of each class are given in Table 6.2.

TABLE 6.2

Percentage of Yearly Jet Traffic Represented by Each Class and Approximate Mean Arrival Rate for Each Class in a Busy Period

Class	% of Yearly Jet Traffic	λ_m (Aircraft/Hr)
1	2.1	0.672
2	0.2	0.064
3	26.2	8.384
4	6.2	1.984
5	65.3	20.896

Noah et al. (1977) provide three estimates of the direct operating cost to the airlines of each hour of flying time due to delay. The costs included in the estimates are the costs of fuel, airline crew, aircraft maintenance, and depreciation of the aircraft. The estimates derived from three different sources [Civil Aeronautics Board (CAB) (1975); Reck et al. (1975); and Rogers et al. (1975)] are given in Table 6.3. For each class m, C_m is approximated by averaging the three estimates given in Table 6.3. These approximations are shown in Table 6.4.

The computation of R_m, the reward for the service of an aircraft of class m, is rather involved. R_m is found by multiplying the average profit per passenger by the average number of passengers on a flight of class m. The average profit per passenger is obtained by dividing the sum of the profits of all domestic airlines for a year by the number of passengers that flew in that year. 1974 figures are used because the estimates of costs used for C_m were made in 1975 and thus were probably based on 1974 data. Moles and Wimbush (1976) report the sum of the profits of all domestic airlines in 1974 as \$799,298,000 and list 189,733,000 as the number of passengers for that year. Thus, the average profit per passenger is \$4.21. Moles and Wimbush (1976) also give the fraction of seating that was occupied, the passenger load factor, as 0.555 for 1974. The average number of passengers on a flight of class m is the average capacity of class m aircraft times the load factor. The average capacity of aircraft of class m is determined by a weighted average of the seating capacities of the aircraft that make up the class. The weight for a given type of aircraft in class m is the proportion of yearly flights into the Pittsburgh Airport by class m aircraft that is accounted for by the particular aircraft type. These proportions can be computed for the Pittsburgh Airport from data given by the CAB and FAA (1978). The

TABLE 6.3

Estimates of Direct Operating Cost Per Hour of Flight for Each Class of Aircraft

Class	CAB ($/Hr)	Reck ($/Hr)	Rogers ($/Hr)
1	1729.20	1718.40	1620.00
2	2415.60	2295.60	1980.00
3	871.20	860.40	780.00
4	1112.40	1093.20	1080.00
5	694.80	671.40	660.00

TABLE 6.4

Cost Per Hour of Flight for Each Class

Class	C_m ($/Hr)
1	1689.00
2	2230.00
3	837.00
4	1095.00
5	675.00

maximum seating capacities of the various aircraft in the classes are taken from Aviation Week (1977). The weights and seating capacities are given in Table 6.5. The average capacity for each class can be found by summing the products of the last two columns of Table 6.5. These results are given in Table 6.6. For each class m, R_m is the average profit per passenger ($4.21) times the load factor (0.555) times the average capacity of an aircraft of the class (Table 6.6). The values of R_m are given in Table 6.7.

6.2 Server Parameters

The first task that must be accomplished is to define a service. In actual practice, the control of an aircraft approaching the Pittsburgh Airport begins with the Cleveland center, is then transferred to approach control, and is finally passed on to the control tower. Each of these control sectors is really a group of air traffic controllers. In turn, each controller may handle from one to about six aircraft. During busy periods at the Pittsburgh Airport, most commercial jet traffic is landed on Runway 28 center, while departures operate from Runway 28 right. If Runway 28 center and the airspace near it are defined as the service facility, then a single server model is appropriate.[1] Since FAA rules require a minimum separation between aircraft, on approach and on the runway, the service time of an aircraft is defined to be the period of time that the runway is cleared for use by the aircraft. This definition is easiest to

---

[1] If an airport uses several runways for landings, a single server model could still be used for a given runway if the class arrival rates are adjusted so they are class arrival rates at the particular runway rather than at the whole airport.

TABLE 6.5

Seating Capacity and Within Class Weights for the Aircraft in Each Class

Class	Aircraft Type	Seating (maximum)	Weight
1	DC-10	380	0.46
	L-1011	400	0.54
2	Boeing 747	500	1.00
3	Boeing 727-100	131	0.65
	Boeing 727-200	189	0.35
4	Boeing 707-100B	181	0.63
	Boeing 707-300, 300B, 300C	189	0.20
	DC-8-20	176	0.01
	DC-8-50, 62	189	0.01
	DC-8-61	259	0.15
5	BAC-111	89	0.22
	DC-9-10	90	0.14
	DC-9-30	115	0.53
	DC-9-50	139	0.11

TABLE 6.6

Average Seating Capacity of Each Class

Class	Average Capacity
1	391
2	500
3	151
4	194
5	108

TABLE 6.7

Reward Per Flight for Each Class

Class	R_m (\$)
1	914
2	1168
3	353
4	453
5	252

implement when the airport is busy; then, the service time of Aircraft B (which follows Aircraft A) is defined to be the time from Aircraft A's crossing the end of the runway (or some other easily defined point) to Aircraft B's crossing the same point. Thus, the service time distribution is obtained from data taken during busy periods at the airport. This service time distribution is assumed to apply whether or not the airport is busy.

Data were taken at the Pittsburgh Airport on 8 June 1979 under good weather conditions. Observations of the service times were made at a radar scope similar to that used by the air traffic controllers. The radar scope was used instead of direct visual observation for the following reasons:

1) The information given on the scope included the altitude of the aircraft which provided an alternate definition of the end of a service; in this case, 1200 feet, the elevation of the runway, was used rather than the end of the runway.
2) The information given on the scope included the type of aircraft so that service time observations not involving aircraft in the five classes (or aircraft with similar landing speeds) could be dropped.
3) The pattern of aircraft on the scope facilitated the identification of busy periods.

The service times which were found using a decimal minute stopwatch are given in Table 6.8.

The sample mean of the Pittsburgh data is $\bar{x} = 1.81$ minutes and the sample standard deviation is $s = 0.413$ minutes. The parameters of

TABLE 6.8

Observed Service Times (Pittsburgh)

Observation	Time (Min)	Observation	Time (Min)	Observation	Time (Min)
1	2.00	24	1.68	46	1.34
2	2.15	25	1.00	47	1.53
3	1.76	26	2.39	48	2.34
4	2.09	27	2.27	49	2.50
5	2.65	28	1.61	50	1.94
6	2.48	29	1.57	51	1.79
7	1.94	30	2.33	52	1.32
8	2.10	31	1.69	53	1.18
9	2.01	32	1.55	54	2.57
10	1.56	33	1.41	55	1.87
11	2.18	34	1.48	56	1.93
12	1.33	35	1.87	57	1.79
13	1.10	36	2.58	58	1.94
14	2.02	37	1.73	59	1.32
15	1.93	38	1.80	60	1.57
16	2.42	39	2.05	61	1.62
17	1.85	40	1.27	62	1.63
18	1.55	41	1.41	63	1.32
19	1.98	42	1.48	64	1.41
20	2.87	43	1.87	65	1.48
21	2.27	44	1.56	66	1.16
22	1.70	45	1.72	67	1.62
23	1.85				

the Erlang distribution, service rate μ and Erlang number h, are estimated by both the method of moments and the method of maximum likelihood. [See White, Schmidt, and Bennett (1975).]

For the method of moments, the point estimates, $\hat{\mu}$ and $\hat{h}$, of the parameters of the Erlang distribution are found using Equations (6.1) and (6.2).

$$\hat{\mu} = 1/\bar{x} \tag{6.1}$$

and

$$\hat{h} = 1/(s^2 \hat{u}^2) \quad . \tag{6.2}$$

For the data in Table 6.8, the estimates are $\hat{u}$ = 0.552 aircraft per minute and $\hat{h}$ = 19.3. Since the unit of time used in the calculation of the customer's parameters is an hour, the estimate of μ needs to be converted to $\hat{\mu}$ = 33.15 aircraft per hour. Since h has to be an integer, 19.3 is infeasible. This problem will be resolved shortly.

The point estimates, $\hat{u}$ and $\hat{h}$, of the parameters of the Erlang distribution using the method of maximum likelihood are found using Equations (6.3) and (6.4).

$$\hat{\mu} = 1/\bar{x} \tag{6.3}$$

and

$$\hat{h} = \left(\prod_{i=1}^{n} x_i\right)^{-1/n} \hat{\mu}^{-1} \exp\{\Psi(\hat{h})\} \quad , \tag{6.4}$$

where x_i is observation i, n is the number of observations, exp(.) represents e raised to the power (.), and $\Psi(\hat{h})$ is given by

$$\Psi(\hat{h}) \cong \ell n(\hat{h} - 0.5) + 1/\{24(\hat{h} - 0.5)^2\} \quad . \tag{6.5}$$

The approximation in Equation (6.5) is only valid if $\hat{h} \geq 2$. Since $\hat{h}$ appears on both sides of Equation (6.4), a few iterations using trial and error or bisection are required to solve for it. For the data in Table 6.8, the estimates are $\hat{\mu} = 0.552$ aircraft per minute or $\hat{\mu} = 33.15$ aircraft per hour and $\hat{h} = 19.6$.

Both methods yield estimates of the Erlang number that are infeasible. White, Schmidt, and Bennett recommend choosing an integer on either side of $\hat{h}$ when this occurs. The estimate for the data in Table 6.8 is thus either $\hat{h} = 19$ or $\hat{h} = 20$.

The final step in fitting the data in Table 6.8 is to test the hypothesis that the sample data could have come from an Erlang distribution with the estimated parameters. A chi-square goodness of fit test will be used. White, Schmidt, and Bennett suggest that maximum likelihood estimates be used for the parameters of the Erlang distribution to guarantee that the asymptotic distribution of the test statistic is chi-square. The test is:

H_0: The data come from an Erlang (μ,h) population.

H_1: The data come from some other population.

Test Statistic: $\chi^2 = \sum_{i=1}^{k} (O_i - E_i)^2/E_i$.

Rejection Region: $\chi^2 > \chi^2_{1-\alpha}(d)$.

k is the number of class intervals into which the data are divided. O_i is the number of observations that fall in interval i, while E_i is the expected number of observations that would fall in interval i under H_0. α is the level of significance of the test. The number of degrees of freedom of the test, d, is k - 1 minus the number of estimated parameters of the distribution. Since two parameters are estimated for the Erlang distribution, d is k - 3. White, Schmidt, and Bennett recommend that:

1) The number of intervals be chosen such that $O_i \geq 5$ and $E_i \geq 5$ for all i.

2) The intervals be chosen so that the probabilities of an observation falling in each are nearly equal under H_0.

For the data in Table 6.8, k was arbitrarily chosen to be six. Since there are 67 observations, such a value of k should allow the intervals to be chosen so that O_i and E_i are at least five for each. A computer program was written to evaluate the integral of the Erlang density so that the location of the intervals could be chosen to satisfy both recommendations. The location of the intervals used and the values of O_i and E_i when the hypothesized distribution is an Erlang ($\mu = 33.15$, $h = 19$) distribution are given in Table 6.9. If a level of significance $\alpha = 0.05$ is used, the critical value, $\chi^2_{0.95}(3)$, is 7.81. Since the value of the test statistic is 0.805 for this test, the hypothesized distribution cannot be rejected. In addition, the extremely small value of the test statistic indicates a very good fit.

TABLE 6.9

Observed and Expected Class Frequencies When an Erlang ($\mu = 33.15$, $h = 19$) Distribution Is Hypothesized

	Interval					
	0.0 to 1.385	1.385 to 1.585	1.585 to 1.735	1.735 to 1.905	1.905 to 2.205	2.205 to ∞
O_i	10	13	9	9	14	12
E_i	10.02	10.92	9.72	10.77	14.36	11.21

The location of the intervals used and the values of 0_i and E_i when the hypothesized distribution is an Erlang ($\mu = 33.15$, $h = 20$) distribution are given in Table 6.10. The value of the test statistic for this data if 0.998. Again, if the level of significance is 0.05, the hypothesized distribution cannot be rejected and a very good fit is indicated.

TABLE 6.10

Observed and Expected Class Frequencies When an Erlang ($\mu = 33.15$, $h = 20$) Distribution Is Hypothesized

	Interval					
	0.0 to 1.385	1.385 to 1.585	1.585 to 1.735	1.735 to 1.905	1.905 to 2.205	2.205 to ∞
0_i	10	13	9	9	14	12
E_i	9.54	11.00	9.94	11.04	14.62	10.86

Thus, both 19 and 20 are reasonable estimates of the Erlang number for the data in Table 6.8. Since the number of states needed for policy iteration is $n_s^*h + 1$, there is an incentive to choose the smaller h. Recall that each step of the policy iteration algorithm or the evaluation of each policy requires the inversion of a matrix that is $(n_s^*h + 1)$-square. In light of this requirement, it may be even more practical to use an estimate of h that is smaller than the maximum likelihood estimate but which passes the goodness of fit test. While in theory, this procedure may be hard to justify, it should not introduce any appreciable errors. To illustrate, an Erlang ($\mu = 33.15$, h = 8) distribution yields χ^2 = 8.65 which fails the chi-square test at α = 0.05, but passes for α = 0.01. For the purpose of this chapter, however, the Erlang (μ = 33.15, h = 19) distribution is chosen. When Model II is used in this chapter, an exponential distribution with μ = 33.15 is assumed.

Data were also taken at the Washington National Airport on 10 July 1979, again when the weather was good. During the period of observation, Runway 18 was being used for both arrivals and departures. Some of the observations of service time were made using a radar scope, but the majority of them were made visually. The Washington data given in Table 6.11 is presented primarily to support the general applicability of the Pittsburgh data. At a level of significance of 0.05, an Erlang (μ = 37.48, h = 16) distribution provides a good fit of the 34 observations taken.

6.3 Deterministic Service Time Model

n_{s_m}, the individual optimum balking point for class m (in terms of customers), is found using Equation (5.5). The value of n_{s_m} is given for each class, m, in Table 6.12. These results imply that for the Erlang 19 model, n_s^* is 18, so that the number of states required is 343. The solution techniques based on policy iteration, Algorithm 5.1, will require the inversion of a matrix that is 343 rows by 343 columns. Since the matrix generated will be sparse, special techniques could be used to invert such a large matrix while maintaining a reasonable degree of numerical precision. However, another approach is pursued in this paper.

TABLE 6.11

Observed Service Times (Washington National)

Observation	Time (min)	Observation	Time (Min)	Observation	Time (Min)
1	1.78	13	2.12	24	2.37
2	1.40	14	1.60	25	2.15
3	1.63	15	1.12	26	1.77
4	1.60	16	1.52	27	1.22
5	1.95	17	1.12	28	2.18
6	1.70	18	1.48	29	2.32
7	1.66	19	2.05	30	1.62
8	1.15	20	0.95	31	1.16
9	1.30	21	1.97	32	1.82
10	1.15	22	1.57	33	1.29
11	1.30	23	1.00	34	1.95
12	1.48				

TABLE 6.12

Individual Optimum Balking Points (h = 19)

Class	n_{s_m} (Customers)
1	18
2	17
3	14
4	14
5	12

The optimal policy and corresponding gain rate for the Erlang 19 model will be estimated based on lower Erlang number models. The next two sections develop a pattern to the results for the lower Erlang number models which suggests that this estimation procedure is reasonable. Erlang 1 through Erlang 8 models will be used for this process. Furthermore, since the Erlang ∞ service time distribution represents deterministic service times, a deterministic service time model can be used as one bound for the Erlang 19 results, while the Erlang 8 results represent the other bound (the Erlang 19 results lie between these bounds). Adler and Naor (1969) study a single class model like Model I but with deterministic service times. Their results are presented here and then used in Section 6.5 to approximate the results for a multiclass deterministic service time model.

Let T be the length of a service. ϕ_0 is the steady state probability that the system is empty, while ϕ_c is the steady state probability that the service station is closed. ($\emptyset_c$ corresponds to $\emptyset_{n_0}$ in Model I.) The average rate at which customers depart the service station is $(1 - \emptyset_0)/T$. The average rate at which customers are admitted to the service station is $\lambda(1 - \emptyset_c)$. In steady state, these two rates must be equal; that is,

$$(1 - \emptyset_0)/T = \lambda(1 - \emptyset_c) \quad . \tag{6.6}$$

The social optimum objective function that Adler and Naor use is

$$g = \{R(1 - \emptyset_0)/T\} - CL \quad . \tag{6.7}$$

From Equation (6.6), $(1 - \emptyset_0)/T = \lambda(1 - \emptyset_c) = \lambda'$, the effective arrival rate. With this substitution, Equation (6.7) becomes

$$g = \lambda' R - CL \tag{6.8}$$

which is Equation (3.2), ignoring the dependence of n. Thus, the model of Adler and Naor is indeed the same as Model I except for the deterministic nature of the service times.

Adler and Naor define a state space that can assume nonintegral values. Recall that, in Model I, the state of the system is the number of customers present in the system. Adler and Naor define the state of the system to be the number of whole service times, T, present in the system. (They assume an arrival can determine how much service time remains for the customer in service.) Since an arrival can find the service of the customer in service partially completed, the state of the system can take on nonintegral values.

The balking point for the individual optimum problem is $v_s - 1$, where v_s is determined by

$$v_s = R/(CT) \quad . \tag{6.9}$$

The authors show that the forced balking point for the social optimum problem is given by $v_o - 1$, where v_o satisfies

$$v_s \emptyset_0(v_o,\rho) - v_o + L(v_o, \rho) = 0 \quad . \tag{6.10}$$

dler and Naor also show that

$$\emptyset_0(v,\rho) = 1/\left(1 + \sum_{j=1}^{n} (-1)^{j-1} \frac{\{(v - j)\rho\}^{j-1}}{(j - 1)!} \rho \exp\{(v - j)\rho\}\right) \tag{6.11}$$

and

$$\begin{aligned} L(v,\rho) = \; & n - \emptyset_0(v,\rho)\Bigg\{ \exp\{(v - 1)\rho\}(1 + n\rho - v\rho) \\ & + \sum_{j=2}^{n} \Bigg(\exp\{(v - j)\rho\} \Bigg\{\sum_{k=1}^{j-1} (-1)^{k-1}\frac{\{(v - j)\rho\}^{k-1}}{(k - 1)!} \\ & + (-1)^{j-1} \frac{\{(v - j)\rho\}^{j-1}}{(j - 1)!} (1 + n\rho - v\rho)\Bigg\}\Bigg)\Bigg\} \quad , \end{aligned} \tag{6.12}$$

where n is the greatest integer in v. One way to solve for v_o is to guess a starting value such as $v = v_s$, then use a technique like bisection to search for a value of v that satisfies Equation (6.10), where Equations (6.11) and (6.12) are used to evaluate $\emptyset_0$ and L required in Equation (6.10). Once a value of v_o is determined, the corresponding values of $\emptyset_0$ and L can be used in Equation (6.7) to determine the maximum gain rate of the model.

To illustrate the above solution technique, let R = 5, C = 2, $\lambda = 1$, and T = 1/3 be the parameters of the model. The traffic intensity, ρ, is given by $\rho = \lambda T = 1/3$. Equation (6.9) yields $v_s = 7.5$. After a few iterations, bisection yields $v_o = 5.4167$, which implies that the forced balking point is 4.4167. Equations (6.11) and (6.12) yield ϕ_0 = 0.6667 and L = 0.4164. Finally, Equation (6.7) gives g = 4.1667.

To illustrate the suggested method for bounding the results for an Erlang 19 model, the deterministic results will be compared with the results of several Erlang models with low values of h. Since Adler and Naor assume that an arrival knows how much service time remains for the customer in service, the most appropriate Erlang model for comparison with the deterministic model is one in which the customer knows the service phase of the customer in service. Table 6.13 gives the social optimum gain rate and forced balking points for several Erlang models of the example. The forced balking point in Table 6.13 is given in terms of customers in line plus phases of service left for the customer in service. For instance, 4 cust + 3 ph means that customers are forced to balk if the state of the system represents at least four customers in line plus three phases of service remaining for the customer in service. The deterministic results plus the data in Table 6.13 indicate that the optimal policy for an Erlang 19 model is probably a balking point of four customers in line plus several phases of service left for the customer in service which should yield a gain rate between 4.146 and 4.1667. The actual phase optimum solution of the Erlang 19 model provides a gain rate of 4.158 with a balking point of four customers in line plus eight phases of service left for the customer in service.

TABLE 6.13

Results for Several Erlang Models
of the Example

Erlang Number	Gain Rate	Forced Balking Point
1	4.003	4 cust + 1 ph
2	4.084	4 cust + 1 ph
6	4.139	4 cust + 3 ph
8	4.146	4 cust + 4 ph

6.4 Results for Model II

For Model II, applied to the Pittsburgh Airport problem, the service rate of the exponential server is 33.15 aircraft per hour. The computer program in Appendix D, POLIT, was used to solve for the optimal policy, the resulting gain rate, and the probability of rejection of an arrival from each class. The probability of rejection of class m is found by summing the steady state probabilities of the system occupying states n_{o_m} through n_s^*. The results from POLIT are summarized in Table 6.14.

Since the first iteration of the policy iteration algorithm chooses the individual optimum policy (see Table 6.12), its gain rate can be found ($4905 per hour). Thus, the model indicates that for a peak traffic hour, the average net benefit to the airline companies can be increased $1784 by implementing the social optimum policy rather than the individual optimum policy.

TABLE 6.14

Results for the Exponential (μ = 33.15) Model

Class	n_{o_m}	Rejection Probability
1	13	< 0.01
2	13	< 0.01
3	6	< 0.01
4	7	< 0.01
5	3	0.316
Gain Rate ($/Hr)	6689	

Although the model assigns no cost to rejecting an aircraft, the probability of rejection of a class indicates how the optimal policy would affect the current operations of that class at the airport. It is possible to use the model so that a "cost" of rejection is introduced by setting controls on the maximum probability of rejection allowed for any class. This approach might consist of a procedure to search various combinations of class arrival rates using POLIT to evaluate each combination until a combination is achieved that produces both an acceptable gain rate and an acceptable maximum probability of rejection. The schedule used in the Pittsburgh example leads to a rejection probability of 0.316 for the class that provides 65% of the arrivals at the airport. The procedure mentioned above could be used to improve on this situation. Although such use of the model is beyond the scope of this chapter, the sensitivity of the gain rate and rejection probabilities to deviations from the optimal policy will be checked to provide some insight into how this use of the model might proceed.

A straightforward approach to checking the sensitivity of the model to deviations from the optimal policy is to use the policy evaluation portion of the policy iteration algorithm to evaluate several policies that are in some sense close to the optimum. The results for several such policies are given in Table 6.15. For convenience, a balking point of 14 rather than 13 is used for class one. Since the steady state probability of the system occupying state 14 is essentially zero, such a change from $n_{o_1} = 13$ has little effect on the results of the model. Results like those presented in Table 6.14 provide an administrator some capability to trade off a reduction in the gain rate for a reduction in the maximum probability of rejection for any class.

The sensitivity of the results to changes in a single parameter was tested by determining the range of parameter values for which the optimal policy remained the same. The service rate, μ, was varied both up and down until the optimal policy changed. The results are given in Table 6.16. Since the change of n_{o_2} from 14 to 13 is insignificant due to the low probability (less than 10^{-4}) of the system occupying state 13 or 14, the optimal policy is stable for μ in the range 31^+ to 34^-. The location of the change could be found more precisely with additional evaluations.

TABLE 6.15

Results for the Exponential (μ = 33.15) Model for Several Nonoptimum Policies

Class	Balking Point	Rejection Probability	Balking Point	Rejection Probability
1	14	< 0.01	14	< 0.01
2	14	< 0.01	14	< 0.01
3	6	0.021	6	0.052
4	7	< 0.01	7	< 0.01
5	4	0.250	5	0.197
Gain Rate ($/Hr)	6674		6564	

Class	Balking Point	Rejection Probability	Balking Point	Rejection Probability
1	14	< 0.01	14	< 0.01
2	14	< 0.01	14	< 0.01
3	6	0.131	7	0.112
4	6	0.131	7	0.112
5	6	0.131	7	0.112
Gain Rate ($/Hr)	6396		6214	

TABLE 6.16

Balking Points and Gain Rate
for Various Values of μ

Class	μ 31	32	33.15	34
1	13	13	13	14
2	13	13	14	14
3	5	6	6	6
4	7	7	7	8
5	3	3	3	4
Gain Rate (\$/Hr)	6367	6521	6689	6811

The value of λ_5, the arrival rate of class five, was also varied in both directions until the optimal policy changed. The results are given in Table 6.17. These results indicate that the optimal policy is stable for λ_5 in the range 19^+ to 27^-.

Finally, the value of R_5, the reward for the service of a class five customer, was varied until the optimal policy changed. The results given in Table 6.18 indicate that the optimal policy is stable for R_5 in the range 230^+ to 270^-.

If desired, the stability of the optimal policy to changes in the other parameters, including the C_m's, can also be tested. The technique used for μ, λ_5, and R_5 can be applied to these determinations as well.

TABLE 6.17

Balking Points and Gain Rate
for Various Values of λ_5

Class	λ_5 19	20	20.90	26	27
1	13	14	13	13	13
2	13	14	14	14	14
3	6	6	6	6	5
4	7	7	7	7	7
5	4	3	3	3	3
Gain Rate ($/Hr)	6568	6631	6689	6958	7000

TABLE 6.18

Balking Points and Gain Rate
for Various Values of R_5

Class	R_5 230	240	252	260	270
1	14	14	13	13	13
2	14	14	14	14	14
3	6	6	6	6	6
4	8	7	7	7	7
5	3	3	3	3	4
Gain Rate ($/Hr)	6375	6518	6689	6803	6956

6.5 Results for Model III

For Model III, the service time distribution is modeled as an Erlang (μ = 33.15, h = 19) distribution as developed in Section 6.2. As this distribution leads to a large number of states (343), the solution presented in this section is found in a somewhat roundabout manner. First, Erlang models with μ = 33.15 and h = 2, 4, 6, and 8 are solved using Algorithm 5.1. These results are used to hypothesize the optimal policy for the Erlang 19 model. The gain rate of the Erlang 8 model serves as a lower bound for that of the Erlang 19 model. Then, the deterministic model of Section 6.3 is used to approximate the gain rate for a deterministic service time model of the Pittsburgh Airport. This gain rate serves as an upper bound on the gain rate for the Erlang 19 model.

The results for the lower Erlang number models will be presented first. Recall that the policy iteration-based algorithm, Algorithm 5.1, uses as a starting point the results from an Erlang model, the phase optimum model, in which customers can determine the service phase of the customer in service. The phase optimum results for Erlang models with μ = 33.15 and h = 2, 4, 6, and 8 are given in Table 6.19. As in Table 6.13, the forced balking points are given in terms of customers in line plus phases of service left for the customer in service.

Recall that an implementable or linked policy does not require the administrator to determine the service phase of the customer in service. The set of candidate forced balking points, found using Step (c) of Algorithm 5.1, are given in Table 6.20 for each of the four linked Erlang models considered. Except for class three in the

TABLE 6.19

Phase Optimum Balking Points and Gain Rate for Several Low Erlang Number Models

Class	h: 2	4	6	8
1	12 cust + 2 ph	12 cust + 3 ph	12 cust + 5 ph	12 cust + 6 ph
2	13 cust + 1 ph	13 cust + 1 ph	13 cust + 1 ph	13 cust + 1 ph
3	5 cust + 1 ph	4 cust + 4 ph	4 cust + 5 ph	4 cust + 7 ph
4	6 cust + 2 ph	6 cust + 2 ph	6 cust + 3 ph	6 cust + 3 ph
5	2 cust + 2 ph	2 cust + 2 ph	2 cust + 3 ph	2 cust + 4 ph
Gain Rate ($/Hr)	6975	7141	7203	7235

TABLE 6.20

Candidate Forced Balking Points for Several Linked Erlang Models

Class	h: 2	4	6	8
1	13,14	13,14	13,14	13,14
2	14	14	14	14
3	6	5,6	5,6	5,6
4	7,8	7,8	7,8	7,8
5	3,4	3,4	3,4	3,4

Erlang two model, the candidate forced balking points for each class are the same for all four Erlang models. Since the steady state probability of any of those systems occupying 13 or 14 is insignificant, the forced balking point for class one is arbitrarily chosen to be 14. Policies that use all possible combinations of this reduced set of candidate balking points were evaluated to determine the best linked policy for each model. The results for the best policy for each model are given in Table 6.21. For each of these models, several policies tied (within roundoff error) for the best policy. Of these, the policy with the largest values of the n_{o_m} was chosen in each case as the best policy. The ties were the result of low probabilities of the systems occupying states beyond state six. Note that for each model, the best linked policy found using Algorithm 5.1 achieves a gain rate that is better than 99.8% of the phase optimum gain rate. Depending on the use of the model, such a result probably precludes the need for a more lengthy search for a best policy. Since the same linked policy provides the best results for all four models, it seems reasonable to assume that the policy will also provide the best results for the Erlang 19 model. Although the gain rate increases with the Erlang number h, the incremental jump decreases with higher values of h. Thus, the gain rate of the Erlang eight model, $7223 per hour, provides a good lower bound on the gain rate of the Erlang 19 model.

A deterministic service time model provides the capability to get an upper bound on the gain rate for the Erlang 19 model; however, the deterministic model of Adler and Naor (1969) presented in Section 6.3 only applies to a single class of customers. No attempt is made here to extend their work. Instead, the single class deterministic

TABLE 6.21

Best Results for Several Linked Erlang Models

Class	h = 2 Balking Point	h = 2 Rejection Probability	h = 4 Balking Point	h = 4 Rejection Probability
1	14	< 0.01	14	< 0.01
2	14	< 0.01	14	< 0.01
3	6	< 0.01	6	< 0.01
4	8	< 0.01	8	< 0.01
5	3	0.2796	3	0.2557
Gain Rate ($/Hr)	6963		7129	

Class	h = 6 Balking Point	h = 6 Rejection Probability	h = 8 Balking Point	h = 8 Rejection Probability
1	14	< 0.01	14	< 0.01
2	14	< 0.01	14	< 0.01
3	6	< 0.01	6	< 0.01
4	8	< 0.01	8	< 0.01
5	3	0.2464	3	0.2414
Gain Rate ($/Hr)	7191		7223	

model is used to approximate the gain rate for the five class Pittsburgh Airport model with deterministic service times. The five classes of aircraft are replaced by a single composite class whose reward for a service is determined by a weighted average of the rewards for the five classes. The weight for each class can be determined from Table 6.2 as the proportion of the total arrival rate that is accounted for by the class. Analogously, the cost per hour of flight for the composite class is a weighted average of the costs for each of the five classes. The arrival rate for the composite class is the sum of the arrival rates of all five classes. Since the services are deterministic, the service rate is a constant whose value is the estimate of μ used in Models II and III. The parameters of the composite class are given in Table 6.22.

TABLE 6.22

Parameters for the Single Composite Class Model

Parameter	Value
R	306.7
C	767.9
λ	32.0
μ	33.15

The approach used here to estimate the gain rate for a deterministic five class model from the deterministic single composite class model begins with the comparison of the gain rates found by policy iteration for several phase optimum Erlang models applied first to the composite class and then to the full five classes. Calculation of the ratio of the composite class gain rate to the five class gain rate for Models II and III with h = 2, 4, 6, and 8 provides a means of estimating the ratio of the composite class deterministic gain rate to the five class deterministic gain rate. Table 6.23 gives the composite class gain rate, the five class gain rate, and the ratio of the two rates for each of the five models mentioned.

TABLE 6.23

Policy Iteration Phase Optimum Results for Several Models of Both a Single Composite Class and the Full Five Classes

Model	Composite Class Gain Rate ($/Hr)	Five Classes Gain Rate ($/Hr)	Ratio (Composite/Five)
II	6506	6689	0.973
III, h = 2	6800	6975	0.975
h = 4	6977	7141	0.977
h = 6	7042	7203	0.978
h = 8	7076	7235	0.978

The deterministic gain rate for the single composite class can be determined using the results of Adler and Naor presented in Section 6.3. Their method gives $\emptyset_0 = 0.1504$ and $L = 1.890$ which, when substituted into Equation (6.7) yields $g = \$7186$ per hour. Since Table 6.23 suggests that 0.97 is a good lower bound on the ratio of the deterministic composite class gain rate to the deterministic five class gain rate, a reasonable upper bound on the deterministic five class gain rate is $7408 per hour. This gain rate serves as an upper bound on the gain rate of the Erlang 19 model.

The results of this section suggest that the socially optimal policy for the Erlang 19 model of the Pittsburgh Airport is to implement the following set of forced balking points: $\bar{n}_0 = (14, 14, 6, 8, 3)$. This policy should provide a gain rate of somewhere between $7223 and $7408 per hour to the arrivals during a busy period, while incurring a rejection probability of less than 0.2414 for class five customers and less than 0.01 for any other class. Analysis of the sensitivity of these results to changes in the input parameters could be tested in the same manner as was done for Model II.

6.6 Conclusion

This chapter was devoted to applying the techniques developed to determine the optimum control policy for the landing queue at the Pittsburgh Airport. Use was made of Models II and III. The parameters of the models were determined from published FAA data, the literature, and direct observation. Computational simplifications were investigated and sensitivity of solutions examined. All indications are that the simplified analyses are practical and sensitivities easy to determine.

A few suggestions for refining the estimates of the model parameters follow; however, before such refinements are incorporated into the model, it is suggested that the sensitivity of the model to the changes be checked. It might be possible to estimate the effect of the changes based on model sensitivities already determined.

The R_m values can be refined by using an average passenger load factor for each type of aircraft and thus for each class. Further improvements in the estimates of the R_m's also affect the estimates of the C_m's and λ_m's through expansion of the number of classes. (Each new class m would require its own estimated R_m, C_m, and λ_m.) For instance, the existing classes can be subdivided based on a range of values of the passenger load factor. Thus, class one might become four classes, one for a 0 - 25% loading factor, another for a 26 - 50% loading, and so on. R_m might be improved even further by again subdividing the classes based on individual aircraft type.

The estimates of the λ_m can be improved by basing them on actual or proposed schedules of arrivals during representative busy periods rather than yearly statistics.

The estimates of the parameters of the service time distribution were based on observations made at the Pittsburgh Airport. The estimates were confirmed by a second set of observations made at the Washington National Airport. Several additional days of observations from the Pittsburgh Airport may result in better estimates of the parameters of the service time distribution.

Several areas for further study were broached in this chapter. One of these is the difficulty in solving problems involving a large number of states. Special techniques for solving large sparse systems of linear equations could be examined to permit direct solution of models like the Erlang 19 model. Another problem is the extension of the work of Adler and Naor (1969) to several classes of customers. While this might require substantial analytical work, it would provide a means of quickly bounding the results for any Erlang model. The final problem introduced is perhaps the most interesting. The problem is to design a schedule of arrivals that achieves an acceptable social gain rate while keeping the maximum rejection probability for any class of customers below some upper bound. Models II and III can be used to solve this problem as a part of a search technique or a new expanded model can be developed.

CHAPTER VII

OTHER MODELS

In this chapter, three new models are developed to illustrate that the semi-Markov approach used for Models I, II, and III can be adapted to solve other related models. Although all three models have M classes of customers and an exponential server, the models could easily be generalized to an Erlang server. The three models considered are:

1) a nonpreemptive priority service discipline model,
2) a class dependent service rate model, and
3) a nonpreemptive service discipline model with class dependent service rates.

7.1 Nonpreemptive Priority Service Discipline Model

The assumptions and structure of this model are those of Model II except that the service discipline utilizes nonpreemptive priorities rather than position in line (first come first served) to determine the order of service. Let class one have highest priority, class two next highest, and so on. Within a class, the customers are served first come first served. Since the priorities are nonpreemptive, the customer in service is allowed to finish regardless of the priority of an arrival.

The state space for this model is a vector $\overline{a}$, where a_0 is the class of the customer in service and a_m is the number of class m customers in the system. To illustrate the state space consider a two class example in state (2,1,2). Thus, there are two customers of class two and one of class one in the system with one of the class two customers in service. If the next event is the completion of a service, the state becomes (1,1,1) since the class one customer has priority over the remaining class two customer.

Self-optimizing customers of class m will join the system if their expected net benefit for joining is greater than zero. When the state of the system is $\bar{a}$, the expected net benefit of joining for a customer of class m is

$$R_m - t_m(\bar{a})\, C_m \quad , \tag{7.1}$$

where $t_m(\bar{a})$ is the time the customer expects to spend in the system when he finds the system in state $\bar{a}$. A class one arrival must wait for all class one customers in the system to be served before he can be served. In addition, he must wait for the customer in service to finish; however, if the customer in service is a member of class one, he has already been counted with the class one customers. Equation (7.2) gives $t_1(\bar{a})$.

$$t_1(\bar{a}) = (a_1 + 1 + \sum_{m=2}^{M} \delta_{m,a_0})/\mu \quad , \tag{7.2}$$

where

$$\delta_{m,a_0} = \begin{cases} 1, & \text{if } a_0 = m \\ 0, & \text{otherwise} \end{cases} \quad .$$

A class two arrival must wait for all class one and class two customers in the system to be served before he can be served. If the customer in service is not a member of class one or two, one more customer must be added to the list of customers who must be served before the class two arrival. Thus,

$$t_2(\bar{a}) = (a_1 + a_2 + 1 + \sum_{m=3}^{M} \delta_{m,a_0})/\mu \quad . \tag{7.3}$$

In general, a class m customer must wait for the service of customers in the system of his class and all classes of higher priority. In addition, if the customer in service is of lower priority, he has not been counted and must also be served before the class m arrival. Thus,

$$t_m(\bar{a}) = (\sum_{i=1}^{m} a_i + 1 + \sum_{j=m+1}^{M} \delta_{j,a_0})/\mu \quad . \tag{7.4}$$

Self-optimizing customers of class m will join the system if

$$R_m - t_m(\bar{a})C_m = R_m - (\sum_{i=1}^{m} a_i + 1 + \sum_{j=m+1}^{M} \delta_{j,a_0})C_m/\mu \geq 0 \quad . \tag{7.5}$$

If n_{s_m} is such that

$$R_m - n_{s_m} C_m/\mu \geq 0 > R_m - (n_{s_m} + 1)C_m/\mu \quad ,$$

then,

$$n_{s_m} = [R_m \mu / C_m] \quad , \tag{7.6}$$

where the brackets indicate the greatest integer function. n_{s_m} is the individual optimum balking point for class m customers. If an arrival from class m finds

$$\sum_{i=1}^{m} a_i + \sum_{j=m+1}^{M} \delta_{j,a_0} < n_{s_m} \quad , \tag{7.7}$$

he joins; otherwise, he balks.

Since customers are served on the basis of their priority, the expected net reward for a customer of class m who is in the system but not in service is altered by later arriving customers of higher priority who join the system. This may result in customers desiring to renege. For instance, if a class m customer joined the system when

$$\sum_{i=1}^{m} a_i + \sum_{j=m+1}^{M} \delta_{j,a_0} = n_{s_m} - 1 \quad ,$$

and the next event were the arrival of a higher priority customer, the class m customer would want to leave the system to cut his expected loss. Even though his time in the system has cost him money, he now expects to lose more money by remaining in the system. If customers are allowed to renege, another problem arises. The decision to renege by a class m customer affects the reneging decision of a lower priority customer. Thus, the presence or absence of information concerning the reneging decisions of customers of higher priority can lead a customer to make different decisions concerning reneging. Several individual optimum models would result from different combinations of assumptions about

1) whether or not to allow reneging, and
2) if reneging is allowed, whether or not to assume that each customer knows the decision of every other customer when he makes his own decision about reneging.

Since the social optimum problem is of principal interest, these ideas will not be pursued here.

As will be discussed later, the individual optimum balking point for each class serves as an upper bound on the social optimum balking point for the class.[1] This again provides a bound on the size of the state space required for the semi-Markov decision process formulation of the social optimum problem. Since self-optimizing customers of class m can join if as many as $n_{s_m} - 1$ customers of their own class are in the system (this can occur if no customers of higher priority are in the system and a class m customer is in service), a bound on the size of the state space for the social optimum problem is the number of states such that $a_m \leq n_{s_m}$, for all m. Let a* be the set of all such states.

Recall that Section 5.4 points out that the social optimum problem of Model III (as well as Models I and II) differs from the self-optimum problem in that a self-optimizing customer fails to consider the decrease in expected benefits to later arriving customers caused by his joining the queue. This external economic effect also occurs in this model but manifests itself in two ways. Not only does the self-optimizing customer fail to consider the decrease in benefits to later arriving customers of lower priority that his joining causes, but he also fails to consider the decrease in benefits to customers of lower priority who are already in the system (but not in service). The formulation of the social optimum problem considers both forms of the external economic effect.

Consider an administrator who charges a joining customer for the expected costs his joining causes customers present in the system as

---

[1]Since the optimal policy has not been shown to be a control-limit policy, n_{o_m} needs to be redefined as one plus the maximum number of customers in the system for which a class m arrival joins the queue.

well as for his own expected costs. Since the joining of a lowest priority (class M) arrival affects no customers already in the system, the expected system gain in state $\bar{a}$ due to his joining is

$$R_M - (\sum_{i=1}^{M} a_i + 1)C_m/\mu \quad . \tag{7.8}$$

The joining of a customer from class M-1 causes all customers from class M who are in the queue to remain in the system for an additional service. Thus, the expected system gain in state $\bar{a}$ due to a class M-1 customer joining is

$$R_{M-1} - (\sum_{i=1}^{M-1} a_i + 1 + \delta_{M,a_0})C_{M-1}/\mu - (a_M - \delta_{M,a_0})C_M/\mu \quad . \tag{7.9}$$

In general, the expected system gain in state $\bar{a}$ due to a class m customer joining is

$$R_m - (\sum_{i=1}^{m} a_i + 1 + \sum_{j=m+1}^{M} \delta_{j,a_0})C_m/\mu - \sum_{j=m+1}^{M} (a_j - \delta_{j,a_0})C_j/\mu \quad . \tag{7.10}$$

This approach offers a means of internalizing the cost to customers already in the system and avoids reneging since a customer's expected costs are known at the time of his arrival and do not change with later arriving customers.

The other portion of the external economic effect, the decrease in benefits to later arriving customers of lower priority caused by a customer joining the system, is internalized as in Models I, II and III through the social optimum objective function which is

$$\max_{P \in C_s \cap C_d} g_P = \max_{P \in C_s \cap C_d} \sum_{\bar{a} \in a^*} \phi_{\bar{a}}^{P} \, q_{\bar{a}}^{P} \quad . \tag{7.11}$$

Again, only stationary Markovian decisions are considered and the work of Der ~ (1962) is used to limit the search to a nonrandomized rule. $\phi^P_{\bar{a}}$ is the steady state probability of the system occupying state $\bar{a}$ under policy P. $q^P_{\bar{a}}$ is the expected gain per unit time the system is in state $\bar{a}$ when policy P is in effect. The inclusion of the steady state probabilities of occupying the various states allows the administrator to consider the expected decrease in benefits to later arriving customers of lower priority when deciding whether or not to admit a customer. Equation (7.11) can be rewritten as

$$\max_{P \varepsilon C_s \cap C_d} g_P = \max_{P \varepsilon C_s \cap C_d} \sum_{\bar{a} \varepsilon a*} \phi^P_{\bar{a}} \sum_{m=1}^{M} k_m(\bar{a}) \lambda_m f_m(\bar{a}) \quad , \qquad (7.12)$$

where $k_m(\bar{a})$ is an indicator that is one if a class m arrival is admitted to the system when the system is in state $\bar{a}$ and zero if not. $f_m(\bar{a})$, which is the expected system gain in state $\bar{a}$ if a class m customer is admitted, is given by Equation (7.10),

$$f_m(\bar{a}) = R_m - (\sum_{i=1}^{m} a_i + 1 + \sum_{j=m+1}^{M} \delta_{j,a_0})C_m/\mu$$

$$- \sum_{j=m+1}^{M} (a_j - \delta_{j,a_0})C_j/\mu \quad . \qquad (7.13)$$

The formulation expressed in Equation (7.12) can be solved by policy iteration as will be demonstrated by example. Theorem 4.3 can be adapted to show that $n_{o_m} \leq n_{s_m}$. No attempt is made here to establish the form of the optimal policy.

Consider a two class example with $\mu = 4$ and $\lambda_1 = 2$, $R_1 = 2$, and $C_1 = 3$. Also, $\lambda_2 = 4$, $R_2 = 1$, and $C_2 = 2.5$. From Equation (7.6), $n_{s_1} = 2$ and $n_{s_2} = 1$. Thus,

$$a^* = \{(0,0,0),\ (1,1,0),\ (2,0,1),\ (1,2,0),\ (1,1,1),\ (2,1,1),\ (1,2,1),\ (2,2,1)\}\ .$$

a* may contain several unnecessary states, states whose steady state probability will be zero under the optimal policy. Although Equations (7.7) and (7.13) can be used to eliminate some of these states, for small examples like this one, it is probably easier to carry them along.

Since there are two classes, there are four possible actions, (0,0), (1,0), (0,1), and (1,1). The transition matrix for each action follows:

$$P(0,0) = \begin{array}{c} \\ \text{From State} \end{array} \begin{array}{c} \\ (0,0,0) \\ (1,1,0) \\ (2,0,1) \\ (1,2,0) \\ (1,1,1) \\ (2,1,1) \\ (1,2,1) \\ (2,2,1) \end{array} \begin{array}{c} \text{To State} \\ \begin{array}{cccccccc} (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (1,1,1) & (2,1,1) & (1,2,1) & (2,2,1) \end{array} \\ \left[\begin{array}{cccccccc} - & - & - & - & - & - & - & - \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}$$

A row of dashes indicates that the action cannot be chosen when the system is in the state represented by the row. For action (0,0), the next transition is sure to be the completion of a service.

To State

P(1,0) =

From State	(0,0,0)	(1,1,0)	(2,0,1)	(1,2,0)	(1,1,1)	(2,1,1)	(1,2,1)	(2,2,1)
(0,0,0)	0	1	0	0	0	0	0	0
(1,1,0)	0.67	0	0	0.33	0	0	0	0
(2,0,1)	0.67	0	0	0	0	0.33	0	0
(1,2,0)	-	-	-	-	-	-	-	-
(1,1,1)	0	0	0.67	0	0	0	0.33	0
(2,1,1)	0	0.67	0	0	0	0	0	0.33
(1,2,1)	-	-	-	-	-	-	-	-
(2,2,1)	-	-	-	-	-	-	-	-

If action (1,0) is chosen in state (0,0,0), the next transition is sure to be the joining of a class one customer. For the other states in which action (1,0) may be chosen, the entries are the result of the service rate competing with the arrival rate.

To State

P(0,1) =

From State	(0,0,0)	(1,1,0)	(2,0,1)	(1,2,0)	(1,1,1)	(2,1,1)	(1,2,1)	(2,2,1)
(0,0,0)	0	0	1	0	0	0	0	0
(1,1,0)	0.5	0	0	0	0.5	0	0	0
(2,0,1)	-	-	-	-	-	-	-	-
(1,2,0)	0	0.5	0	0	0	0	0.5	0
(1,1,1)	-	-	-	-	-	-	-	-
(2,1,1)	-	-	-	-	-	-	-	-
(1,2,1)	-	-	-	-	-	-	-	-
(2,2,1)	-	-	-	-	-	-	-	-

$$
P(1,1) = \begin{array}{cc}
 & \text{To State} \\
 & \begin{array}{cccccccc} (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (1,1,1) & (2,1,1) & (1,2,1) & (2,2,1) \end{array} \\
\text{From State} \begin{array}{c} (0,0,0) \\ (1,1,0) \\ (2,0,1) \\ (1,2,0) \\ (1,1,1) \\ (2,1,1) \\ (1,2,1) \\ (2,2,1) \end{array} &
\begin{bmatrix}
0 & 0.33 & 0.67 & 0 & 0 & 0 & 0 & 0 \\
0.4 & 0 & 0 & 0.2 & 0.4 & 0 & 0 & 0 \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & -
\end{bmatrix}
\end{array}
$$

If action (1,1) is chosen in state (1,1,0), the arrival rates for each class compete with each other as well as with the service rate to cause the next transition.

$\bar{\tau}_{\bar{a}}(\bar{k})$, the unconditional expected waiting time in state $\bar{a}$ under action $\bar{k}$, is simply the reciprocal of the transition rate out of state $\bar{a}$ under action $\bar{k}$. Thus,

$$
\hat{\tau}(0,0) = \begin{bmatrix} - \\ 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{bmatrix}, \quad
\hat{\tau}(1,0) = \begin{bmatrix} 0.5 \\ 0.167 \\ 0.167 \\ - \\ 0.167 \\ 0.167 \\ - \\ - \end{bmatrix}, \quad
\hat{\tau}(0,1) = \begin{bmatrix} 0.25 \\ 0.125 \\ - \\ 0.125 \\ - \\ - \\ - \\ - \end{bmatrix}, \quad
\hat{\tau}(1,1) = \begin{bmatrix} 0.167 \\ 0.1 \\ - \\ - \\ - \\ - \\ - \\ - \end{bmatrix} .
$$

The components of all vectors for this example correspond to the states as listed in a*; that is, component one corresponds to state (0,0,0), component two corresponds to state (1,1,0), and so on.

If b(0,0) is the matrix of expected rewards for transitions between the various states under alternative (0,0), b(0,0) is the null matrix since no customers are allowed to join under alternative (0,0). The following matrices of expected rewards for transitions under the other alternatives were found using Equation (7.13).

$$
b(1,0) = \begin{array}{c}
\text{From} \\ \text{State}
\end{array}
\begin{array}{cc}
 & \text{To State} \\
 & \begin{array}{cccccccc} (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (1,1,1) & (2,1,1) & (1,2,1) & (2,2,1) \end{array} \\
\begin{array}{c} (0,0,0) \\ (1,1,0) \\ (2,0,1) \\ (1,2,0) \\ (1,1,1) \\ (2,1,1) \\ (1,2,1) \\ (2,2,1) \end{array} &
\left[\begin{array}{cccccccc}
0 & 1.25 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
- & - & - & - & - & - & - & - \\
0 & 0 & 0 & 0 & 0 & 0 & -0.125 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.25 \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & -
\end{array}\right]
\end{array}
$$

If the only nonzero entries in a row of $b(\bar{k})$ are negative, action $\bar{k}$ can be eliminated from consideration when the system is in the state represented by the row because action (0,0) will dominate $\bar{k}$ in the policy improvement section of the policy iteration algorithm.

$$
\begin{array}{lll}
 & & \text{To State} \\
 & & \begin{array}{cccccccc} (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (1,1,1) & (2,1,1) & (1,2,1) & (2,2,1) \end{array} \\
\begin{array}{l} b(0,1) = \\ \\ \\ \\ \text{From} \\ \text{State} \\ \\ \\ \end{array} & \begin{array}{c} (0,0,0) \\ (1,1,0) \\ (2,0,1) \\ (1,2,0) \\ (1,1,1) \\ (2,1,1) \\ (1,2,1) \\ (2,2,1) \end{array} & \left[\begin{array}{cccccccc}
0 & 0 & 0.375 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -0.25 & 0 & 0 & 0 \\
- & - & - & - & - & - & - & - \\
0 & 0 & 0 & 0 & 0 & 0 & -0.875 & 0 \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & -
\end{array}\right]
\end{array}
$$

$$
\begin{array}{lll}
 & & \text{To State} \\
 & & \begin{array}{cccccccc} (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (1,1,1) & (2,1,1) & (1,2,1) & (2,2,1) \end{array} \\
\begin{array}{l} b(1,1) = \\ \\ \\ \\ \text{From} \\ \text{State} \\ \\ \\ \end{array} & \begin{array}{c} (0,0,0) \\ (1,1,0) \\ (2,0,1) \\ (1,2,0) \\ (1,1,1) \\ (2,1,1) \\ (1,2,1) \\ (2,2,1) \end{array} & \left[\begin{array}{cccccccc}
0 & 1.25 & 0.375 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.5 & -0.25 & 0 & 0 & 0 \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & - \\
- & - & - & - & - & - & - & -
\end{array}\right]
\end{array}
$$

The components of the vectors of expected rewards per transition are found from

$$
r_{\bar{a}'}(\bar{k}) = \sum_{\bar{a}''\varepsilon a^*} P_{\bar{a}',\bar{a}''}(\bar{k})\, b_{\bar{a}',\bar{a}''}(\bar{k}) \qquad ,
$$

where $\bar{a}'$ and $\bar{a}''$ are used to denote the state before and after a transition, respectively. Recall that a* is a set that contains all states that are required for the social optimum problem. For this example,

$$\bar{r}(0,0) = \begin{pmatrix} - \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad \bar{r}(1,0) = \begin{pmatrix} 1.25 \\ 0.167 \\ 0.167 \\ - \\ -0.042 \\ -0.083 \\ - \\ - \end{pmatrix}, \quad \bar{r}(0,1) = \begin{pmatrix} 0.375 \\ -0.125 \\ - \\ -0.438 \\ - \\ - \\ - \\ - \end{pmatrix}, \quad \bar{r}(1,1) = \begin{pmatrix} 0.67 \\ 0 \\ - \\ - \\ - \\ - \\ - \\ - \end{pmatrix} .$$

The components of the vector of expected rewards per unit time in state $\bar{a}$ under action $\bar{k}$ are found from

$$q_{\bar{a}}(\bar{k}) = r_{\bar{a}}(\bar{k})/\tau_{\bar{a}}(\bar{k}) \quad .$$

Thus,

$$\bar{q}(0,0) = \begin{pmatrix} - \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad \bar{q}(1,0) = \begin{pmatrix} 2.5 \\ 1.0 \\ 1.0 \\ - \\ -0.25 \\ -0.5 \\ - \\ - \end{pmatrix}, \quad \bar{q}(0,1) = \begin{pmatrix} 1.5 \\ -1.0 \\ - \\ -3.5 \\ - \\ - \\ - \\ - \end{pmatrix}, \quad \bar{q}(1,1) = \begin{pmatrix} 4 \\ 0 \\ - \\ - \\ - \\ - \\ - \\ - \end{pmatrix} .$$

The optimal policy found using Algorithm C.1 (Appendix C) is

$$P^* = \begin{bmatrix} (1,0) \\ (1,0) \\ (1,0) \\ (0,0) \\ (0,0) \\ (0,0) \\ (0,0) \\ (0,0) \end{bmatrix}$$

which yields g = 1.714 and

$$\bar{\phi} = \begin{bmatrix} 0.57 \\ 0.29 \\ 0.00 \\ 0.14 \\ 0.00 \\ 0.00 \\ 0.00 \\ 0.00 \end{bmatrix} .$$

For this example, it is socially optimal for class one to dominate the system to the exclusion of class two. Even if a class two customer arrives to find an empty system, he is turned away since the administrator expects the customers to achieve a higher gain rate by leaving the server idle until a class one customer arrives.

7.2 Class Dependent Service Rate Model

The assumptions and structure of this model are the same as for Model II except each class, m, has its own service rate, μ_m. For an arrival to be able to compute his expected time in the system, he must know the number of each class of customers in the system. As illustrated in Section 4.3, this requirement together with a first come first served queue discipline leads to a state space that gives the class and position of each customer in the system. Thus, the state of the system is given by $\overline{m} = (m_1, m_2, \ldots, m_j, \ldots)$, where m_j is the class of the customer in position j. The customer in position one is in service, the customer in position two is first in line, and so on.

Self-optimizing customers of class m will join the system if their expected net benefit for joining is greater than zero which is their expected net benefit for balking. The expected net benefit of a class m customer joining when the system is in state $\overline{m}$ is

$$R_m - \left(\left\{ \sum_{i=1}^{M} \sigma_i(\overline{m})/\mu_i \right\} + (1/\mu_m)\right) C_m \quad , \tag{7.14}$$

where $\sigma_i(\overline{m})$ is the number of class i customers present in state $\overline{m}$.

Since

$$\sigma_i(\overline{m}) = \sum_{j=1}^{\infty} \delta_{i,m_j} \quad , \tag{7.15}$$

where

$$\delta_{i,m_j} = \begin{cases} 1, & \text{if } m_j = i \\ 0 & \text{otherwise,} \end{cases}$$

a class m customer will join the system if $\overline{m}$ is such that Equation (7.14) is nonnegative or

$$\sum_{i=1}^{M} \{\sigma_i(\bar{m})/\mu_i\} \overset{<}{=} R_m/C_m - 1/\mu_m \quad . \tag{7.16}$$

Since the service rate varies with customer class, Equation (7.16) cannot be reduced to a balking point in terms of the number in the system like n_{s_m}. Rather, for each class m, let S_m be the set of all $\bar{m}$ such that Equation (7.16) holds. Thus, if the system is in state $\bar{m}$, a self-optimizing customer of class m joins if $\bar{m}\varepsilon S_m$ and balks otherwise.

It will be argued later that the set of states in which a class m customer acting in a socially optimal manner joins the queue is a subset of the set of states in which he would join if he were acting in an individually optimal manner. Thus, if 0_m is the set of all states such that customers of class m who are acting in a socially optimal manner join the system, then $0_m \subseteq S_m$. A little additional work must be done to bound the state space required for the social optimum problem. $\bigcup_{m=1}^{M} S_m$ may not contain enough states since the states that can be reached from states belonging to S_m through the joining of a class m customer must also be included to bound the social optimum state space. Let S'_m contain all states in S_m plus all states that can be reached from the states of S_m through the joining of a class m customer. The required state space for the social optimum problem can then be bounded by $\bigcup_{m=1}^{M} S'_m$.

Again, a self-optimizing customer fails to consider the decrease in benefits to later arriving customers caused by his joining the queue. This external economic effect is internalized in the formulation of the social optimum problem which is

$$\max_{P\varepsilon C_s \cap C_d} g_P = \max_{P\varepsilon C_s \cap C_d} \sum_{\bar{m}\varepsilon \bigcup_{m=1}^{M} S'_m} \emptyset^P_{\bar{m}} \; q^P_{\bar{m}} \quad . \tag{7.17}$$

Again, only stationary Markovian policies are considered and the work of Derman (1962) is used to limit the search to a nonrandomized policy. $\phi^P_{\bar{m}}$ is the steady state probability of the system occupying state $\bar{m}$ under policy P. $q^P_{\bar{m}}$ is the expected gain per unit time the system is in state $\bar{m}$ while policy P is employed. Equation (7.17) can be rewritten as

$$\max_{P\epsilon C_s \cap C_d} g_P = \max_{P\epsilon C_s \cap C_d} \sum_{\bar{m} \in \bigcup_{m=1}^{M} S'_m} \phi^P_{\bar{m}} \sum_{m=1}^{M} k_m(\bar{m}) \lambda_m f_m(\bar{m}) \quad , \qquad (7.18)$$

where $k_m(\bar{m})$ is one if class m is admitted under policy P when the system is in state $\bar{m}$ and zero if not. $f_m(\bar{m})$, the expected system gain if a class m customer is admitted when the system occupies state $\bar{m}$, follows from Equation (7.14) as

$$f_m(\bar{m}) = R_m - \Big(\sum_{i=1}^{M} \{\sigma_i(\bar{m})/\mu_i\} + (1/\mu_m)\Big) C_m \quad . \qquad (7.19)$$

Although the state space is quite cumbersome, the semi-Markov formulation given by Equation (7.18) can be solved by policy iteration.

To show that $0_m \subseteq S_m$ consider first the quantity t_{s_m}, where $t_{s_m} = R_m/C_m - 1/\mu_m$. S_m contains all states for which the sum of the expected service times of all customers in the system is no greater than t_{s_m} {see Equation (7.16)}. If t_{o_m} is defined to be the maximum amount of expected service time in the system for which a social optimizing customer of class m joins, then, $t_{o_m} \leq t_{s_m}$ is analogous to $n_{o_m} \leq n_{s_m}$. Theorem 4.3 can be adapted to show that $t_{o_m} \leq t_{s_m}$. Since the definition of 0_m implies that 0_m is the set of all states for which the sum of the expected service times is no greater than t_{o_m}, $0_m \subseteq S_m$.

7.3 Nonpreemptive Priority Service Discipline Model with Class Dependent Service Rates

This model is a combination of the two previous models. The assumptions and structure are the same as Model II except that:

a) a nonpreemptive priority service discipline is used, and

b) the mean service rate varies with customer class.

Class one is of highest priority, class two of next highest, and so on. Within a class, the service discipline is first come first served. Since the priorities are nonpreemptive, the customer in service is allowed to finish no matter what the priority of an arrival. The mean service rate of class m is μ_m, $m = 1, \ldots, M$. Since the model has a nonpreemptive priority service discipline, the state space can again be a vector $\bar{a}$, where a_0 is the class of the customer in service and a_m is the number of class m customers in the system.

Self-optimizing customers of class m will join the system if their expected net benefit for joining is greater than their expected net benefit for balking which is zero. For a class m arrival, the expected net benefit of joining when the system is in state $\bar{a}$ is

$$R_m - t_m(\bar{a})\, C_m \quad , \qquad (7.20)$$

where $t_m(\bar{a})$ is the time the arrival expects to spend in the system if he joins when it is in state $\bar{a}$. Since class one is of highest priority, a class one arrival must only wait for the customer in service plus any class one customers already in the system. If a class one customer is in service, he should not be counted twice. Thus, the time a class one arrival will expect to spend in the system is given by

$$t_1(\bar{a}) = (a_1 + 1)/\mu_1 + \sum_{m=2}^{M} \delta_{m,a_0}/\mu_m \quad , \tag{7.21}$$

where $\delta_{m,a_0} = \begin{cases} 1 & \text{if } a_0 = m \\ 0 & \text{otherwise} \end{cases}$.

A class two arrival must wait for all class one and class two customers present in the system plus the customer in service if he has not already been counted. Thus,

$$t_2(\bar{a}) = a_1/\mu_1 + (a_2 + 1)/\mu_2 + \sum_{m=3}^{M} \delta_{m,a_0}/\mu_m \quad . \tag{7.22}$$

In general, a class m arrival must wait for all customers of equal or greater priority who are present in the system plus the customer in service if he has not already been counted. Thus, for any class m,

$$t_m(\bar{a}) = \sum_{i=1}^{m-1} a_i/\mu_i + (a_m + 1)/\mu_m + \sum_{j=m+1}^{M} \delta_{j,a_0}/\mu_j \quad . \tag{7.23}$$

Self-optimizing customers of class m will join the system if

$$R_m - t_m(\bar{a})C_m = R_m - \{\sum_{i=1}^{m-1} a_i/\mu_i + (a_m + 1)/\mu_m$$

$$+ \sum_{j=m+1}^{M} \delta_{j,a_0}/\mu_j\}C_m \geq 0 \quad . \tag{7.24}$$

Equation (7.24) can be written as

$$(\sum_{i=1}^{m} a_i/\mu_i + \sum_{j=m+1}^{M} \delta_{j,a_0}/\mu_j) \leq R_m/C_m - 1/\mu_m \quad . \tag{7.25}$$

Since the mean service rate varies with customer class, Equation (7.25) cannot be reduced to a balking point in terms of customers in the system like n_{s_m}; however, it does express a maximum amount of expected service time that can be ahead of a class m arrival and still have him join the system. $t_{s_m} = R_m/C_m - 1/\mu_m$ is the maximum expected time a class m customer can wait for his service to begin and still find it profitable to join the system.

Since the customers are served on the basis of their priority, the expected net reward for a class m member of the queue (i.e., in the system but not in service) is altered by the joining of a customer of higher priority. This again raises the problem of reneging and the various assumptions that could be made (see Section 7.1).

Let S_m be the set of all states such that Equation (7.25) holds. It will be argued later that if t_{o_m} is the maximum expected time a class m arrival can wait for his service to begin and still find it socially optimal to join the system, then, $t_{o_m} \leq t_{s_m}$. Thus, if O_m is the set of all states that a class m arrival expects to wait no more than t_{o_m} for his service to begin, then, $O_m \subseteq S_m$. Let S'_m contain the states that are elements of S_m together with the states that can be reached from the elements of S_m through the joining of a class m customer. $\bigcup_{m=1}^{M} S'_m$ provides a bound on the state space required for the social optimum problem.

Self-optimizing customers ignore the decrease in net benefits that their joining causes customers of lower priority either present in the system or yet to arrive. Both of these costs will be incorporated into the formulation of the social optimum problem. The first will be charged to the arrival and the second will be accounted for in the objective function for the social optimum problem.

Consider an administrator who charges a joining customer for the expected costs he causes other customers already in the system as well as for his own expected costs. Since the joining of a class M arrival affects no customers present in the system, the expected gain in state $\bar{a}$ due to his joining is

$$R_M - \{(\sum_{i=1}^{M-1} a_i/\mu_i) + (a_M + 1)/\mu_M\}C_M \quad . \tag{7.26}$$

The joining of a class M-1 customer affects only class M customers who are in the queue. Thus, if the system is in state $\bar{a}$, the expected system gain due to a class M-1 customer joining is

$$R_{M-1} - \{\sum_{i=1}^{M-2} a_i/\mu_i + (a_{M-1} + 1)/\mu_{M-1} + \delta_{M,a_0}/\mu_M\}C_{M-1}$$

$$- (a_M - \delta_{M,a_0})C_M/\mu_{M-1} \quad . \tag{7.27}$$

In general, if the system is in state $\bar{a}$, the expected system gain due to a class m customer joining is

$$R_m - \{\sum_{i=1}^{m-1} a_i/\mu_i + (a_m + 1)/\mu_m + \sum_{j=m+1}^{M} \delta_{j,a_0}/\mu_j\}C_m$$

$$- \sum_{j=m+1}^{M} \{(a_j - \delta_{j,a_0})C_j/\mu_m\} \quad . \tag{7.28}$$

This method of assigning costs avoids reneging since a customer's expected cost is not changed by later arrivals joining the queue.

The objective function for the social optimum problem is

$$\max_{P \in C_s \cap C_d} g_P = \max_{P \in C_s \cap C_d} \sum_{\bar{a} \in \bigcup_{m=1}^{M} S'_m} \phi^P_{\bar{a}} \, q^P_{\bar{a}} \quad . \tag{7.29}$$

As before, only nonrandomized, stationary Markovian policies need to be considered. $\phi_{\bar{a}}^{P}$ and $q_{\bar{a}}^{P}$ are, respectively, the steady state probability of occupying state $\bar{a}$ and the expected gain per unit time in state $\bar{a}$ under policy P. The inclusion of the steady state probabilities of occupying the various states considers the expected decrease in benefits to later arriving customers of lower priority when deciding whether or not to admit a customer. Equation (7.29) can be written as

$$\max_{P \in C_s \cap C_d} g_P = \max_{P \in C_s \cap C_d} \sum_{\bar{a} \in \bigcup_{m=1}^{M} S'_m} \phi_{\bar{a}}^{P} \sum_{m=1}^{M} k_m(\bar{a}) \lambda_m f_m(\bar{a}) \quad , \tag{7.30}$$

where again, $k_m(\bar{a})$ is one if class m is admitted under policy P when the system is in state $\bar{a}$ and zero if not. $f_m(\bar{a})$, the expected system gain due to a class m customer joining when the system is in state $\bar{a}$, follows from Equation (7.28) as

$$\begin{aligned} f_m(\bar{a}) = R_m - \{\sum_{i=1}^{m-1} a_i/\mu_i + (a_m + 1)/\mu_m + \sum_{j=m+1}^{M} \delta_{j,a_0}/\mu_j\}C_m \\ - \sum_{j=m+1}^{M} (a_j - \delta_{j,a_0})C_j/\mu_m \quad . \end{aligned} \tag{7.31}$$

The solution of the semi-Markov process formulation in Equation (7.30) by policy iteration will be illustrated by an example. Theorem 4.3 can be adapted to show that, for all m, $t_{o_m} \leq t_{s_m}$ and thus, $O_m \subseteq S_m$.

Consider the same two class example used in Section 7.1, but let each class have its own service rate. The mean service rates and other parameters for the example are given in Table 7.1. The maximum expected service time a self-optimizing customer of class one can wait for his service to begin and still join the system is

$t_{s_1} = R_1/C_1 - 1/\mu_1 = 0.344$. Similarly, $t_{s_2} = 0.208$. Thus, $S_1 = \{(0,0,0), (1,1,0), (2,0,1)\}$, and $S_1' = \{(0,0,0), (1,1,0), (2,0,1), (1,2,0), (2,1,1)\}$. Also, $S_2 = \{(0,0,0), (2,0,1)\}$, and $S_2' = \{(0,0,0), (2,0,1), (2,0,2)\}$. The state space for the social optimum problem is a subset of $\bigcup_{m=1}^{2} S_m' = \{(0,0,0), (1,1,0), (2,0,1), (1,2,0), (2,1,1), (2,0,2)\}$.

TABLE 7.1

Parameters for the Example

Class	R_m	C_m	λ_m	μ_m
1	2	3	2	3.1
2	1	2.5	4	5.2

The transition matrices for the four possible actions follow:

$$
P(0,0) = \quad \text{From State} \quad
\begin{array}{c|cccccc|}
 & \multicolumn{6}{c}{\text{To State}} \\
 & (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (2,1,1) & (2,0,2) \\
\hline
(0,0,0) & - & - & - & - & - & - \\
(1,1,0) & 1 & 0 & 0 & 0 & 0 & 0 \\
(2,0,1) & 1 & 0 & 0 & 0 & 0 & 0 \\
(1,2,0) & 0 & 1 & 0 & 0 & 0 & 0 \\
(2,1,1) & 0 & 1 & 0 & 0 & 0 & 0 \\
(2,0,2) & 0 & 0 & 1 & 0 & 0 & 0
\end{array}
$$

Again, a row of dashes indicates the action cannot be chosen when the system occupies the state represented by the row. For action (0,0), the next transition is sure to be the completion of a service.

$$P(1,0) = \begin{array}{c|cccccc} & \multicolumn{6}{c}{\text{To State}} \\ & (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (2,1,1) & (2,0,2) \\ \hline (0,0,0) & 0 & 1 & 0 & 0 & 0 & 0 \\ (1,1,0) & 0.61 & 0 & 0 & 0.39 & 0 & 0 \\ (2,0,1) & 0.72 & 0 & 0 & 0 & 0.28 & 0 \\ \text{From State } (1,2,0) & - & - & - & - & - & - \\ (2,1,1) & - & - & - & - & - & - \\ (2,0,2) & - & - & - & - & - & - \end{array}$$

If action (1,0) is chosen in state (0,0,0), the next transition is sure to be the joining of a class one customer. For other states in which the action may be chosen, the entries are the result of competition between the rate of service for the customer in service and the arrival rate of class one.

$$P(0,1) = \begin{array}{c|cccccc} & \multicolumn{6}{c}{\text{To State}} \\ & (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (2,1,1) & (2,0,2) \\ \hline (0,0,0) & 0 & 0 & 1 & 0 & 0 & 0 \\ (1,1,0) & - & - & - & - & - & - \\ (2,0,1) & 0.57 & 0 & 0 & 0 & 0 & 0.43 \\ \text{From State } (1,2,0) & - & - & - & - & - & - \\ (2,1,1) & - & - & - & - & - & - \\ (2,0,2) & - & - & - & - & - & - \end{array}$$

To State

From State

$$
\begin{array}{ll}
 & \begin{array}{cccccc} (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (2,1,1) & (2,0,2) \end{array} \\
P(1,1) = \begin{array}{c} (0,0,0) \\ (1,1,0) \\ (2,0,1) \\ (1,2,0) \\ (2,1,1) \\ (2,0,2) \end{array} &
\begin{bmatrix}
0 & 0.33 & 0.67 & 0 & 0 & 0 \\
- & - & - & - & - & - \\
0.46 & 0 & 0 & 0 & 0.18 & 0.36 \\
- & - & - & - & - & - \\
- & - & - & - & - & - \\
- & - & - & - & - & -
\end{bmatrix}
\end{array}
$$

If action (1,1) is chosen in state (2,0,1), the class arrival rate of each class and the service rate of the customer in service compete to cause the next transition.

$\bar{\tau}_{\bar{a}}(\bar{k})$, the unconditional expected waiting time in state $\bar{a}$ under action $\bar{k}$, is the reciprocal of the transition rate out of state $\bar{a}$ under action $\bar{k}$. Thus,

$$
\bar{\tau}(0,0) = \begin{bmatrix} - \\ 0.32 \\ 0.19 \\ 0.32 \\ 0.19 \\ 0.19 \end{bmatrix}, \quad
\bar{\tau}(1,0) = \begin{bmatrix} 0.5 \\ 0.20 \\ 0.14 \\ - \\ - \\ - \end{bmatrix}, \quad
\bar{\tau}(0,01) = \begin{bmatrix} 0.25 \\ - \\ 0.11 \\ - \\ - \\ - \end{bmatrix}, \quad
\bar{\tau}(1,1) = \begin{bmatrix} 0.17 \\ - \\ 0.09 \\ - \\ - \\ - \end{bmatrix}.
$$

The components of all vectors for this example correspond to the states as listed in $\bigcup_{m=1}^{2} S'_m$.

The matrix of expected rewards for transitions between states under alternative (0,0), b(0,0), is the null matrix since no customers may join the system. Equation (7.31) is used to find the entries in the matrices of expected rewards for transitions under the other alternatives. For this example,

$$
b(1,0) = \begin{array}{cc} & \text{To State} \\ & \begin{array}{cccccc} (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (2,1,1) & (2,0,2) \end{array} \\ \begin{array}{c} (0,0,0) \\ (1,1,0) \\ (2,1,0) \\ (1,2,0) \\ (2,1,1) \\ (2,0,2) \end{array} & \left[\begin{array}{cccccc} 0 & 1.03 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.06 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.46 & 0 \\ - & - & - & - & - & - \\ - & - & - & - & - & - \\ - & - & - & - & - & - \end{array}\right] \end{array}
$$

From State

$$
b(0,1) = \begin{array}{cc} & \text{To State} \\ & \begin{array}{cccccc} (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (2,1,1) & (2,0,2) \end{array} \\ \begin{array}{c} (0,0,0) \\ (1,1,0) \\ (2,0,1) \\ (1,2,0) \\ (2,1,1) \\ (2,0,2) \end{array} & \left[\begin{array}{cccccc} 0 & 0 & 0.52 & 0 & 0 & 0 \\ - & - & - & - & - & - \\ 0 & 0 & 0 & 0 & 0 & 0.04 \\ - & - & - & - & - & - \\ - & - & - & - & - & - \\ - & - & - & - & - & - \end{array}\right] \end{array}
$$

From State

$$
b(1,1) = \begin{array}{c c} & \text{To State} \\ & \begin{array}{cccccc} (0,0,0) & (1,1,0) & (2,0,1) & (1,2,0) & (2,1,1) & (2,0,2) \end{array} \\ \text{From State} \begin{array}{c} (0,0,0) \\ (1,1,0) \\ (2,0,1) \\ (1,2,0) \\ (2,1,1) \\ (2,0,2) \end{array} & \begin{bmatrix} 0 & 1.03 & 0.52 & 0 & 0 & 0 \\ - & - & - & - & - & - \\ 0 & 0 & 0 & 0 & 0.46 & 0.04 \\ - & - & - & - & - & - \\ - & - & - & - & - & - \\ - & - & - & - & - & - \end{bmatrix} \end{array}
$$

The components of $\overline{r}(\overline{k})$, the vector of expected rewards per transition under action $\overline{k}$, are found from

$$r_{\overline{a}'}(\overline{k}) = \sum_{\overline{a}'' \in \bigcup_{m=1}^{2} S'_m} P_{\overline{a}',\overline{a}''}(\overline{k}) b_{\overline{a}',\overline{a}''}(\overline{k}) \quad ,$$

where $\overline{a}'$ and $\overline{a}''$ are, respectively, the state before and after a transition. Here,

$$\overline{r}(0,0) = \begin{pmatrix} - \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \ \overline{r}(1,0) = \begin{pmatrix} 1.03 \\ 0.03 \\ 0.13 \\ - \\ - \\ - \end{pmatrix}, \ \overline{r}(0,1) = \begin{pmatrix} 0.52 \\ - \\ 0.02 \\ - \\ - \\ - \end{pmatrix}, \ \overline{r}(1,1) = \begin{pmatrix} 0.69 \\ - \\ 0.10 \\ - \\ - \\ - \end{pmatrix} \quad .$$

The components of the vector of expected rewards per unit time in state $\overline{a}$ under action $\overline{k}$ are found using

$$q_{\overline{a}}(\overline{k}) = r_{\overline{a}}(\overline{k}) / \tau_{\overline{a}}(\overline{k}) \quad .$$

Hence,

$$\bar{q}(0,0) = \begin{pmatrix} - \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \ \bar{q}(1,0) = \begin{pmatrix} 2.06 \\ 0.13 \\ 0.91 \\ - \\ - \\ - \end{pmatrix}, \ \bar{q}(0,1) = \begin{pmatrix} 2.08 \\ - \\ 0.15 \\ - \\ - \\ - \end{pmatrix}, \ \bar{q}(1,1) = \begin{pmatrix} 4.14 \\ - \\ 1.06 \\ - \\ - \\ - \end{pmatrix} .$$

The optimal policy found using Algorithm C.1 is as follows:

$$P^* = \begin{pmatrix} (1,1) \\ (0,0) \\ (0,0) \\ (0,0) \\ (0,0) \\ (0,0) \end{pmatrix} ,$$

which yields $g = 1.715$ and $\bar{\emptyset} = \begin{pmatrix} 0.414 \\ 0.267 \\ 0.319 \\ 0.0 \\ 0.0 \\ 0.0 \end{pmatrix}$.

For this example, it is socially optimal for class one and class two customers tr join the system only if it is empty.

7.4 Conclusion

The models of this chapter require a more complex state space than Models I, II, and III. The state space for the nonpreemptive priority models is tractable as long as the number of classes remains small. However, the state space for the model of Section 7.2 is tractable only for extremely small problems.

The thrust of this chapter has been to demonstrate that the approach used in Models I, II, and III can be extended to other models. That is, once a state space has been defined that carries the necessary information for memoryless transitions and allows computation of the expected net benefit of joining, the social optimum problem can be formulated as a semi-Markov decision process. Also, the individual optimum problem can be used to bound the number of states required for the policy iteration solution technique. No attempt has been made to draw any conclusions about the form of the optimal policy.

CHAPTER VIII

SUMMARY, CONCLUSIONS, AND SUGGESTIONS FOR FURTHER RESEARCH

In this chapter, the work presented in this paper is summarized and a few conclusions are drawn about the models that have been developed. Also, a few potential uses of the models beyond that presented in Chapter VI are listed. Finally, some suggestions are offered for further research.

8.1 Summary and Conclusions

The models of this paper are extensions of the model studied by Naor (1969). One of the major contributions of Naor's work was the suggestion that not every arrival at the service facility would want to be or be allowed to be served. Naor's work was complemented by that of Yechiali (1971) who solved a slightly more general problem. Yechiali's contribution was the formulation of the elementary problem as a semi-Markov decision process. Some other authors [e.g., Stidham (1978)] have subsequently proposed that the choice facing an arrival be the option of joining a system with the cost structure following Naor or joining an alternate system. The alternate system might be either a self-service facility or a system in which the cost structure is simply a fixed fee for service. This proposal is most appealing when all customers eventually require service as is the case with landing aircraft that are in flight. Such an alternate system can easily be incorporated into the models of this paper. (The effect of the alternate system is to change the expected net benefit of balking from zero to the expected net benefit of service in the alternate system.)

The semi-Markov approach has been used in this paper to extend the model of Naor to several classes of customers, Model II. For Model II, the optimal policy for social optimizing customers was shown to be a vector of forced balking points, $\bar{n}_o$. Thus, a class m arrival joins the system only if he finds the number of customers in the system to be less than n_{o_m}, the balking point for his class. The optimal policy for self-optimizing customers was also shown to be a vector of balking points, $\bar{n}_s$. Finally, it was shown that the social and self-optimum balking points for each class are related by $n_{o_m} \leq n_{s_m}$ which facilitates solution of the social optimum problem by providing a bound on the required state space. Both a policy iteration and a linear programming approach were presented as solutions of Model II.

The semi-Markov formulation of Model II was generalized in Chapter V to Model III which incorporates Erlang service times. This generalization enhances the usefulness of the model by providing more flexibility in representing the distribution of service times. The Erlang problem was solved by employing the method of phases which replaces a single Erlang h service time with the mathematically equivalent sum of h independent, identically distributed, exponential service times. Although this approach enables the policy iteration and linear programming solution techniques of Model II to be applied to Model III, it can lead to an optimal policy that forces the administrator to identify the various phases of service. Since, in general, the phases are not physical attributes of the system, the solution found may not be implementable. A heuristic procedure was suggested for determining a good solution that does not require the identification of the phases of service. In addition, a mixed integer programming formulation was presented that, if solved, would identify the optimal implementable solution. The set of states required for the social optimum problem was again shown to be a subset of the set of states required for the individual optimum problem.

In Chapter VI, Models II and III were applied to the landing queue at the Greater Pittsburgh International Airport. Data available from the FAA and other easily accessible sources were used to specify the required parameters for five classes of customers. Data taken at the Pittsburgh Airport led to a rather cumbersome Erlang 19 distribution of service times. This pointed out that Model III can create difficulties for both the policy iteration and mixed integer programming solution methods by requiring a large number of states. Although special techniques for solving such large problems exist, this particular problem was solved by bounding the Erlang 19 results between the results of an Erlang 8 model and those of a deterministic service time model. The deterministic service time model used is an approximate model based on the work of Adler and Naor (1969).

The semi-Markov approach was extended to three new models in Chapter VII to illustrate the versatility of the approach. The nonpreemptive priority discipline models (one with and one without class dependent service rates) require a state space that gives the number of each class of customers in the system together with the class of the customer in service. This new state space requires more states than that of Model II for the same size problem and thus, will tend to cause more computational difficulties for both the policy iteration and linear programming solution technique. The other model, a first come, first served model with class dependent service rates, requires a state space that specifies the class of the customer in every position in the system. This state space compounds the computational problems even further. For all three models, the state space required for the social optimum problem is again a subset of the state space required for the individual optimum problem.

The semi-Markov decision process formulation used in this paper applies to a broad range of queueing control problems as illustrated by the variety of models solved in this paper. The semi-Markov formulation of Models II and III and each of the models in Chapter VII can be solved using a technique based on policy iteration or linear programming. The solution technique may have to be tailored to the problem as was the case with Model III. When a large number of states is involved, the solution technique may encounter computational problems; however, special techniques such as the bounding approach of Chapter VI can often be applied to get useful results even in these cases.

For all models considered, the social optimum balking point (in terms of customers or expected service time in the system) for each class, m, is no greater than the self-optimum balking point. This characteristic of the models is useful in implementing the solution techniques for the social optimum problem since it provides a bound on the state space required for the social optimum problem. The control-limit property of the optimal solution to Model I carries over to Model II. The form of the optimal solution was not investigated thoroughly for the other models.

Models II and III can be successfully applied to the problem of deciding which classes of aircraft to admit to the landing queue of an airport. The results are useful in planning the schedule of arrivals at the airport.

8.2 Potential Uses of the Models

In Chapter VI, Models II and III were used to determine an optimal policy for admitting aircraft to the landing queue of an airport. The models can be applied to other similar problems. For instance, the service might be the unloading of an oil tanker (or a merchant ship or a truck). The tankers could be broken into classes based on capacity or type of crude oil. A policy of admitting tankers to the queue at the port would be sought so that the gain rate of all tankers acting as a group is maximized.

Systems modeled need not have customers that physically line up. Consider the problem of using a single train or truck fleet to haul grain from elevators throughout Nebraska to Kansas City. In this problem, the server moves to the customers and the queue is a schedule of elevators the train is committed to serve. The classes of customers might be based on elevator capacity, location, or type of grain. The problem remains that of determining a policy of admitting customers to the queue (list) so that the gain rate of all customers taken as a whole is maximized.

Some problems that cannot be solved directly using Models II and III can be solved by combining the models with other techniques.

For instance, Chapter VI suggested combining Model II or III with a search technique to determine a schedule of arrivals at an airport which achieves an acceptable gain rate for the customers while keeping the probability of rejection for each class below some specified bound.

A slightly different example is the problem of choosing the number of computer terminals allowed to tie into a computer system. The classes of customers might be based in type of user account or on type of service demanded. Choosing the number of terminals allowed for each class does not directly determine queue size but rather determines the size of the population of each class of customers. If the class arrival rates can be determined as a function of class population, then a search technique could be used in combination with Model II or III to find a socially good policy for allowing terminals to tie into a computer system.

Still another similar application is that of choosing the number of skiers to admit to a ski resort on a holidy or weekend. The customers might be classified by skiing ability which would indicate which parts of the ski area they would be likely to use. The service demanded is the use of a ski lift to carry the skier to the top of the slope. Since most ski areas have more than one lift, class arrival rates must be determined for each lift as a function of population. Again, a search tehcnique could be combined with Model II or III to find socially good policy of admitting skiers to the ski area.

8.3 Suggestions for Further Research

The work described in this paper can be carried forward in several different directions. The heuristic procedure introduced to provide a good solution to Model III assumes that the optimal policy is a control-limit policy for each class. The assumption is based on the fact that the expected net benefit for joining decreases as the number in the system increases. Implementing a policy that is not a control-limit policy would force a customer to balk when i are in the system but allow him to join when i + 1 are in the system which yields the customer a smaller expected net benefit than i. Although this argument supports the control-limit assumption, a proof of the control-limit property would lend more credibility to the heuristic procedure.

The expected net benefit of joining for customers in the first come, first served, class dependent service rate model of Chapter VII decreases as the expected service time ahead of the arrival increases. Arguments similar to those used above for Model III can also be used to support the assumption that the optimal policy for each class establishes a control limit in terms of the expected service time ahead of an arrival. The nonpreemptive priority models of Chapter VII charge an arrival for both his own expected costs and the increase in expected costs he causes customers of lower priority already present in the system. This complicates the definition of a control limit even further since the expected net benefit of joining decreases with an increase in the number of customers of lower priority in the system. The form of the optimal policy needs to be investigated further for each of the three models of Chapter VII. In addition, use of the three models would be facilitated by adapting the program of Appendix D to handle them.

On a broader scope, additional work should be done to determine the range or variety of models to which the semi-Markov approach applies. Subsets of these models should be sought for which the social optimum state space is contained in the individual optimum state space. Also, the subsets should be identified for which a control-limit policy is optimal. Further, the results of the models of this paper should be compared with results of other similar models such as those of Balachandran and Schaefer (1975, 1976) that use expected queue length rather than number of customers in the system to determine the expected costs of joining.

Some refinements and extensions that have been applied to Model I can also be applied to Models II and III. For example, tolls similar to those used in Model I to implement the socially optimal policy should be developed for Model II by using Equation (3.25) to find a toll for each class of customers. Also, the approaches used by Knudsen (1972) and Yechiali (1972) to extend single class models to include multiple servers could be tried on the multiclass models as well.

Finally, the work presented in this paper has touched several other interesting problems that should be investigated. One such problem is the expansion of the deterministic service time model of Adler and Naor (1969) to several classes of customers. Another is the development of a model in which the administrator can control customer arrival rates through population size rather than queue size. One final problem is the use of objective functions other than the individual and social optimum ones used in this paper. One such objective function used by Naor (1969) assumes that the administrator is trying to maximize the sum of the tolls he collects from joining customers.

Certainly, many other areas for further research exist. However, these few are listed to give the researcher some idea as to where he might start.

REFERENCES CITED

Adler, I. and Naor, P. (1969). Social optimization versus self-optimization in waiting lines. Tech. Rept. No. 4, Dept. of Operations Research, Stanford University.

Aviation Week (1977). Specifications: U. S. commercial transports. Leading international aircraft. Aviation Week and Space Technology, March 21, pp. 95, 99.

Balachandran, K. R. (1971). Queue length dependent priority queues. Management Sci. 17, pp. 463-471.

Balachandran, K. R. and Schaefer, M. E. (1975). Public and private optimization at a service facility with approximate information on congestion. College of Industrial Management, Georgia Institute of Technology.

Balachandran, K. R. and Schaefer, M. E. (1976). Regulation by price of arrivals to a congested facility. Working Paper in Business Administration, No. BA5, University of Kentucky.

Balachandran, K. R. and Tilt, B. (undated). Stable and superstable customer policies in queues with balking and priority options. Forthcoming in European Journal of Operations Research.

CAB and FAA (1978). Airport Activity Statistics of Certified Route Air Carriers, Dept. of Transportation: FAA, p. 243.

Charnes, A. and Cooper, W. W. (1962). Programming with linear fractional functionals. NRLQ 9, pp. 181-186.

Crabill, T. B. (1972). Optimal control of a service facility with a variable exponential service time and constant arrival rate. Management Sci. 18, pp. 560-566.

Dear, R. G. (1976). The Dynamic Scheduling of Aircraft in the Near Terminal Area, MIT, FTL Report R76-9.

Derman, C. (1962). On sequential decisions and Markov chains. Management Sci. 9, pp. 16-24.

Edelson, N. M. and Hildebrand, D. K. (1975). Congestion tolls for Poisson queueing processes. Econometrica 43, pp. 81-92.

Emmons, H. (1972). Optimal admission policy to a multi-server queue with finite horizons. J. Appl. Prob. 9, pp. 103-116.

FAA (1974). Federal Air Regulations, Dept. of Transportation: FAA, Part 93, Subpart k.

Fox, B. (1966). Markov renewal programming by linear fractional programming. Siam J. Appl. Math. 14, pp. 1418-1432.

Gross, D. and Harris, C. M. (1974). Fundamentals of Queueing Theory, New York: Wiley.

Harrison, J. M. (1975). Dynamic scheduling of a multiclass queue: discount optimality. Operations Research 23, pp. 270-282.

Heyman, D. P. (1968). Optimal operating policies for M/G/1 queueing systems. Operations Research 16, pp. 362-382.

Hodgson, T. J. and Koehler, G. J. (1978). Computation techniques for large scale undiscounted Markov decision processes. Forthcoming in NRLQ.

Howard, R. A. (1960). Dynamic Programming and Markov Processes. Cambridge, MA: MIT Press.

Howard, R. A. (1971). Dynamic Probabilistic Systems, Volume II. New York: Wiley.

IBM (1970). System/360 Scientific Subroutine Package Version III Programmer's Manual, Program No. 360A-CM-03X, White Plains, NY: IBM, p. 120.

IBM (1972). Mathematical Programming System-Extended (MPSX), and Generalized Upper Bounding (GUB) Program Description, Program No. 5734-XM4, White Plains, NY: IBM.

Ilgen, J. L. (1978). LPFREE, The Pennsylvania State University Computation Center.

Kleinrock, L. (1975). Queueing Systems, Volume I. New York: Wiley, pp. 169-174.

Knudsen, N. C. (1972). Individual and social optimization in a multiserver queue with a general cost-benefit structure. Econometrica 40, pp. 515-528.

Koopman, B. O. (1972). Air-terminal queues under time-dependent conditions. Operations Research 20, pp. 1089-1114.

Lippman, S. A. and Stidham, S., Jr. (1977). Individual versus social optimization in exponential congestion systems. Operations Research 25, pp. 233-247.

Littlechild, S. C. (1974). Optimal arrival rate in a simple queueing system. Int. J. Prod. Res. 12, pp. 391-397.

Miller, B. I. (1969). A queueing reward system with several customer classes. Management Sci. 16, pp. 234-245.

Mine, H. and Ohno, K. (1971). An optimal rejection time for an M/G/1 queueing system. Operations Research 7, pp. 194-207.

Moles, C. and Wimbush, V. (1976). FAA Statistical Handbook of Aviation, Dept. of Transportation: FAA, pp. 76, 86.

Naor, P. (1969). The regulation of queue size by levying tolls. Econometrica 37, pp. 15-24.

Noah, J. W. et al. (1977). Cost-Benefit Analysis and the National Aviation System - A Guide, Dept. of Transportation, Report No. FAA-AVP-77-15, p. 66.

Prabhu, N. U. (1974). Stochastic control of queueing system. NRLQ 21, pp. 411-418.

Ross, S. M. (1970). Applied Probability Models with Optimization Applications, San Francisco: Holden-Day.

Stidham, S., Jr. (1978). Socially and individually optimal control of arrivals to a GI/M/1 queue. Management Sci. 15, pp. 1598-1610.

Stidham, S., Jr. and Prabhu, N. U. (1974). Optimal control of queueing systems. In Mathematical Methods in Queueing Theory, Lecture Notes in Economics and Math. Systems, Vol. 98, New York: Springer-Verlag, pp. 263-294.

Taha, H. A. (1971). Operations Research, New York: MacMillan, pp. 345-346.

Transportation System Center (1978). Profiles of Scheduled Air Carrier Departure and Arrival Operations for Top 100 U. S. Airports, Dept. of Transportation: FAA, p. 211.

White, J. S., Schmidt, J. W. and Bennett, G. K. (1975). Analysis of Queueing Systems, New York: Academic Press, pp. 297-338.

Yechiali, U. (1971). On optimal balking rules and toll charges in the GI/M/1 queueing process. Operations Research 19, pp. 349-370.

Yechiali, U. (1972). Customer's optimal joining rules for the GI/M/s queue. Management Sci. 18. pp. 434-443.

Zacks, S. and Yadin, M. (1969). The optimal control of a queueing process. Technical Report No. 175, NSF Project GP 9007, Dept. of Mathematics and Stat., University of New Mexico.

Zacks, S. and Yadin, M. (1970). Analytic characterization of the optimal control of a queueing system. J. Appl. Prob. 7, pp. 617-633.

APPENDIX A

GLOSSARY OF NOTATION

This glossary defines notation used throughout this dissertation. Mathematical symbols and operators, such as the greatest integer function $[\cdot]$, are not included. Also missing is some notation that is used only locally for one idea or development. The symbols are listed in alphabetical order with Greek symbols appearing at the end of the list for their English equivalent; for example, ρ is listed at the end of the r's.

$\bar{a}$	$(a_0, a_1, \ldots, a_M)$, state space for the nonpreemptive priority models, where a_0 is the class of the customer in service and a_m is the number of class m customers in the system
B	Large numerical value required to develop the either/or constraints in the mixed integer program
$b_{ij}(\bar{k})$	Expected reward for a transition from state i to j when action $\bar{k}$ is chosen
C	Cost per unit time in the system for customers in a single class problem
$C_{c\ell}$	Class of control-limit policies
C_d	Class of nonrandomized policies
$C_{dc\ell}$	Class of deterministic control-limit policies
C_m	Cost per unit time in the system for customers of class m

C_s	Class of stationary Markovian policies
C_t	Class of all policies
$\overline{d}(h)$	Policy chosen in iteration h of the policy iteration altorithm
$D_i(\overline{k})$	Stationary probability that action $\overline{k}$ is chosen when the system is in state i
$D_i^P(\overline{k})$	Stationary probability that action $\overline{k}$ is chosen under policy P when the system is in state i
$D_k^P(H_{m-1},\eta_m)$	Probability that action k is chosen under policy P at time m given history H_{m-1} and present state η_m
Δ_n	Decision made by the n^{th} arrival ($\Delta_n = 0$ if balk, 1 if join)
η_n	Number of customers in the system at the instant of the n^{th} arrival
$f_m(i)$	Expected net reward to a joining customer of class m if i customers are in the system
g	Expected net benefit rate or gain rate of the system
g_N	Expected gain rate using the formulation given by Naor (1969)
g_P	Expected gain rate of the system under policy P
g_T	Expected gain rate using the formulation which delivers the reward when a customer joins the system but distributes the charges for time in the system throughout the customer's stay

g_Y	Expected gain rate using the formulation given by Yechiali (1971)
h	Erlang number
H_m	History of the process through arrival m
$\overline{k}$	$(k_1, k_2, \ldots, k_M)$, a possible action, where $k_m = 0$ if class m balks and 1 if it joins
$k_m(i)$	Indicator which is zero if class m balks when the state of the system is i and one if class m joins when the state is i
L	Expected number of customers in the system
L_m	Contribution of class m to the expected number of customers in the system
λ	Arrival rate of customers in the single class problem
λ_m	Arrival rate of class m customers
λ'	Effective arrival rate of customers in the single class problem
λ'_m	Effective arrival rate of class m customers
$\lambda(i)$	Arrival rate of customers in the single class problem when the system is in state i
$\lambda_m(i)$	Arrival rate of class m customers when the system is in state i
M	Number of classes of customers
$\overline{m}$	$(m_1, m_2, \ldots, m_j, \ldots)$, state space which gives the position of customers in the system, where m_j indicates the class of the customer occupying position j

μ	Service rate of the single server
μ_m	Service rate for class m customers in the class dependent service rate models
$\bar{n}$	$(n_1, n_2, \ldots, n_M)$, vector of balking points
$\bar{n}_o$	$(n_{o_1}, n_{o_2}, \ldots, n_{o_M})$, vector of forced balking points for the social optimum problem
$\bar{n}_s$	$(n_{s_1}, n_{s_2}, \ldots, n_{s_M})$, vector of balking points for the individual optimum problem
n_s^*	Largest component of $\bar{n}_s$
O_m	Set of states for which a social optimizing arrival from class m joins the system
P	Policy for controlling the system
$P_{ij}(\bar{k})$	Probability of a transition from state i to j under action $\bar{k}$
$\emptyset_i^P$	Steady state probability that i customers are in the system under policy P
Q	Fixed payment to an arrival who does not join the system
q	Amount paid per customer in the system to an arrival who does not join the system
$q_i(\bar{k})$	Expected net reward per unit time in state i under action $\bar{k}$
R	Reward for service of a customer in the single class problem
$r_i(\bar{k})$	Expected reward per occupancy of state i under action $\bar{k}$

R_m	Reward for service of a member of class m
RST	Remaining service time for the customer in service
ρ	λ/μ, the traffic intensity
S	Class of stationary control-limit policies of infinite order
S_k	Class of stationary control-limit policies of order k
S_m	Set of states for which a self-optimizing arrival from class m joins the system
S'_m	State space required for class m in the individual optimum problem
$\sigma_m(\overline{m})$	Number of customers of class m present in state $\overline{m}$
T	Length of a service for the deterministic model
T_i	Transition rate out of state i
$t_m(\overline{a})$	Expected time in the system for a class m arrival who finds the system in state $\overline{a}$
t_{o_m}	Maximum amount of expected service time ahead of a social optimizing class m customer for which he will join the svstem
t_{s_m}	Maximum amount of expected service time ahead of a self-optimizing class m customer for which he will join the system
$\hat{\tau}(\overline{k})$	Vector of unconditional expected waiting times under action $\overline{k}$

$\bar{\tau}_i(\bar{k})$	Unconditional expected waiting time in state i under action $\bar{k}$
$\bar{\tau}_{ij}(\bar{k})$	Expected holding time in state i under action $\bar{k}$ given the next transition is to state j
v_i	Relative value of the system starting in state i
v_o	Continuous variable representing the forced balking point for the single class social optimum problem
v_s	Continuous variable representing the balking point for the single class self-optimum problem
y_i	Cost per unit time of operating the system in state i
$y_i(\bar{k})$	$\emptyset_i/\bar{\tau}_i(\bar{k})$, decision variable in the linear programming formulation of a semi-Markov decision process
$Z_i(\bar{k})$	Binary variable that indicates whether or not action $\bar{k}$ is chosen when i customers are in the system (used in the mixed integer programming formulation of the Erlang service time problem)

APPENDIX B

PROOFS OF SOME ASSERTIONS ABOUT MODEL I

Naor (1969) states some properties of Model I but does not provide supporting proofs. The following three properties are established in this appendix:

1) $g(n)$, the socially optimal gain rate is discretely unimodal in n, the forced balking point.
2) v_s increases with v_o, where $v_s = R\mu/C$ and v_o and v_s are related by Equation (3.5).
3) $v_o \leq v_s$, where again v_o and v_s are related by Equation (3.5).

B.1 Proof of Property One

Property (1) is important because it guarantees that a local optimum is a global optimum.

Theorem B.1: $g(n)$ is discretely unimodal in n.

Proof: Define $\Delta g(n)$ to be $g(n) - g(n - 1)$. Since $R \geq C/\mu$, $g(1) \geq 0$. This together with $g(0) = 0$ implies that $\Delta g(1) \geq 0$. Thus, to establish the theorem, it is sufficient to show that if $\Delta g(n) \leq 0$, then $\Delta g(n + 1) < 0$. Equation (3.3) is provided below for convenience.

$$\Delta g(n) = \lambda R\{1 - \rho^n(1 - \rho)/(1 - \rho^{n+1})\} - C\{\rho/(1 - \rho) - (n + 1)\rho^{n+1}/(1 - \rho^{n+1})\}$$

Use of Equation (3.3) for g(n) and g(n - 1) yields

$$g(n) = \lambda R\{\rho^{n-1}(1 - \rho)/(1 - \rho^{n}) - \rho^{n}(1 - \rho)/(1 - \rho^{n+1})\}$$

$$- C\{n\rho^{n}/(1 - \rho^{n}) - (n + 1)\rho^{n+1}/(1 - \rho^{n+1})\}$$

$$= \lambda R\rho^{n-1}(1 - \rho)^{2}/\{(1 - \rho^{n})(1 - \rho^{n+1})\}$$

$$- C\rho^{n}\{n(1 - \rho) - \rho + \rho^{n+1}\}/\{(1 - \rho^{n})(1 - \rho^{n+1})\} \tag{B.1}$$

Similarly,

$$\Delta g(n + 1) = \lambda R\rho^{n}(1 - \rho)^{2}/\{(1 - \rho^{n+1})(1 - \rho^{n+2})\}$$

$$- C\rho^{n+1}\{(n + 1)(1 - \rho) - \rho + \rho^{n+2}\}/$$

$$\{(1 - \rho^{n+1})(1 - \rho^{n+2})\} \quad . \tag{B.2}$$

With the addition and subtraction of

$$C\rho^{2n+2}/\{(1 - \rho^{n+1})(1 - \rho^{n+2})\},$$

Equation (B.2) can be written as

$$\Delta g(n + 1) = \{\lambda R\rho^{n}(1 - \rho)^{2} - Cn\rho^{n+1}(1 - \rho)$$

$$+ C\rho^{n+2} - C\rho^{2n+2}\}/\{(1 - \rho^{n+1})(1 - \rho^{n+2})\}$$

$$+ \{-C\rho^{n+1}(1 - \rho) + C\rho^{2n+2} - C\rho^{2n+3}\}/$$

$$\{(1 - \rho^{n+1})(1 - \rho^{n+2})\} \; . \tag{B.3}$$

Multiplication of Equation (B.3) by

$$\rho(1 - \rho^{n})(1 - \rho^{n+2})/\{\rho(1 - \rho^{n})(1 - \rho^{n+2})\}$$

yields

$$\Delta g(n+1) = \{(\lambda R\rho^{n-1}(1-\rho)^2 - Cn\rho^n(1-\rho) + C\rho^{n+1} - C\rho^{2n+1})/\{(1-\rho^n)(1-\rho^{n+1})\}\} \times \{\rho(1-\rho^n)/(1-\rho^{n+2})\}$$

$$+ C\rho^{n+1}(1-\rho)(\rho^{n+1}-1)/\{(1-\rho^{n+1})(1-\rho^{n+2})\} \quad . \qquad (B.4)$$

Equation (B.4) can be written as

$$\Delta g(n+1) = \Delta g(n)\rho(1-\rho^n)/(1-\rho^{n+2}) - C\rho^{n+1}(1-\rho)/(1-\rho^{n+2}) \quad . \qquad (B.5)$$

For $\rho > 0$, $\rho(1-\rho^n)/(1-\rho^{n+2}) > 0$ and $C\rho^{n+1}(1-\rho)/(1-\rho^{n+2}) > 0$. Thus, if $\Delta g(n) \leq 0$, Equation (B.5) yields $\Delta g(n+1) < 0$ and establishes the theorem.

B.2 Proof of Property Two

v_s and v_o are continuous variables that satisfy the equalities in the formulas that define n_s and n_o, respectively. Naor (1969) shows that $n_s \leq [v_s]$, where the brackets denote the greatest integer in v_s. If Property (2) holds, that is, if v_s increases as v_o increases, then $n_o = [v_o]$ specifies the integer that satisfies Equation (3.4) and is thus, the balking point for the social optimum problem.

One relationship (given by Naor) that is used in the proof of Property ( ) is $v_o \geq 1$. This relationship arises from Equation (3.5) which is

$$\{v_o(1-\rho) - \rho(1-\rho^{v_o})\}/(1-\rho)^2 = v_s \quad .$$

Substitution of $v_o = 1$ into Equation (3.5) yields $v_s = 1$. Substitution of $v_o < 1$ yields $v_s < 1$, but this leads to a trivial system since $n_s = 0$. Since trivial systems are avoided by requiring that $R \geq C/\mu$, the relationship is validated.

<u>Theorem B.2</u>: $\partial v_s/\partial v_o > 0$.

Proof: The proof of this theorem is divided up into proofs for each of three cases,

1) $0 < \rho < 1$,
2) $\rho = 1$,
3) $\rho > 1$.

<u>Case 1</u>. $0 < \rho < 1$.

First, Equation (3.5) can be written as

$$v_s = v_o/(1 - \rho) - \rho/(1 - \rho)^2 + \rho^{v_o+1}/(1 - \rho)^2.$$

Thus,

$$\partial v_s/\partial v_o = 1/(1 - \rho) + \{1/(1 - \rho)^2\}(\partial \rho^{v_o+1}/\partial v_o)$$

$$= 1/(1 - \rho) + \ln\rho \, \rho^{v_o+1}/(1 - \rho)^2. \quad \text{(B.6)}$$

Since $v_o \geq 1$ and $\ln\rho < 0$,

$$\partial v_s/\partial v_o = 1/(1 - \rho) + \rho^2 \ln\rho/(1 - \rho)^2$$

$$= (1 - \rho + \rho^2 \ln\rho)/(1 - \rho)^2 .$$

Since $(1 - \rho)^2 > 0$, it must be shown that

$$1 - \rho + \rho^2 \ln\rho > 0 . \quad \text{(B.7)}$$

Equation (B.7) can be written as

$$(\rho - 1)/(\rho^2 \ell n\rho) > 1. \tag{B.8}$$

First, examine Equation (B.8) at the end points of the region; that is, at $\rho = 0$ and $\rho = 1$.

$$\lim_{\rho \to 0} \{(\rho - 1)/(\rho^2 \ell n\rho)\} = \lim_{\rho \to 0} \left(\{(\rho - 1)/\rho^2\}/\ell n\rho\right)$$

which is indeterminate. Application of L'Hospital's rule yields

$$\lim_{\rho \to 0} \left(\{(\rho - 1)/\rho^2\}/\ell n\rho\right) = \lim_{\rho \to 0} (2 - \rho)/\rho^2$$

which is undefined ($+\infty$). Also,

$$\lim_{\rho \to 1} \{(\rho - 1)/(\rho^2 \ell n\rho)\} = \lim_{\rho \to 1} \{1/(2\rho\ell n\rho + \rho)\} = 1,$$

with the help of L'Hospital's rule. Thus, if

$$d\{(\rho - 1)/(\rho^2 \ell n\rho)\}/d\rho < 0 \text{ for } 0 < \rho < 1,$$

then, Equation (B.8) holds and the theorem is established for Case 1. Differentiation yields

$$\begin{aligned} d\{(\rho - 1)/(\rho^2 \ell n\rho)\}/d\rho &= \{\rho^2 \ell n\rho - (\rho - 1)(2\rho\ell n\rho + \rho)\}/ \\ &\quad (\rho^2 \ell n\rho)^2 \\ &= \{\rho\ell n\rho(1 - \rho) + \\ &\quad + \rho(\ell n\rho - \rho + 1)\}/(\rho^2 \ell n\rho)^2. \end{aligned}$$

Since the denominator is greater than zero, the numerator must be shown to be less than zero.

For $0 < \rho < 1$,

$$\rho \ln\rho (1 - \rho) < 0 \quad .$$

Also, from the series expansion of $\ln\rho$,

$$\ln\rho < \rho - 1 \ .$$

Thus,

$$\rho(\ln\rho - \rho + 1) < \rho(\rho - 1 - \rho + 1) = 0.$$

This demonstrates that

$$d\{(\rho - 1)/(\rho^2 \ln\rho)\}/d\rho < 0 \quad \text{for } 0 < \rho < 1$$

and establishes the theorem for Case 1.

<u>Case 2</u>. $\rho = 1$.

Substitution of $\rho = 1$ into Equation (3.5) yields an indeterminate form. Evaluation of the limit as ρ approaches one of Equation (3.5) is somewhat complicated by the fact that v_o is a function of ρ. However, two applications of L'Hospital's rule yield

$$\lim_{\rho \to 1} v_s = (v_o^2 + v_o)/2 \quad . \tag{B.9}$$

Thus,

$$\lim_{\rho \to 1} \partial v_s/\partial v_o = v_o + \tfrac{1}{2} \tag{B.10}$$

which is greater than zero since $v_o = 1$. This establishes the theorem for Case 2.

<u>Case 3</u>. $\rho > 1$.

From Equation (B.6),

$$\begin{aligned} \partial v_s/\partial v_o &= (1 - \rho + \rho^{v_o+1} \ln\rho)/(1 - \rho)^2 \\ &\geq (1 - \rho + \rho^2 \ln\rho)/(1 - \rho)^2 \quad . \end{aligned}$$

Since $\rho > 1$, $\ln\rho > (\rho - 1)/\rho$. Therefore,

$$\partial v_s/\partial v_o > 1/(1 - \rho) + \rho^2(\rho - 1)/\{(1 - \rho)^2\rho\}$$

$$= 1/(1 - \rho) - \rho/(1 - \rho)$$

$$= 1. \qquad \text{(B.11)}$$

This establishes the theorem for Case 3.

B.3 Proof of Property Three

Since Property (2) gives $n_o = [v_o]$ and Naor (1969) gives $n_s = [v_s]$, showing that $v_o \leqq v_s$ also implies that $n_o \leqq n_s$. Theorem 3.1 states the equivalence of Yechiali's formulation of Model I to Naor's. Since Yechiali proved that $n_o \leqq n_s$, Property (3) will be shown by starting with $n_o \leqq n_s$. A more straightforward approach would be to use Equation (3.5) and again attack the proof in three cases,

1) $0 < \rho < 1$

2) $\rho = 1$

3) $\rho > 1$.

The proof of Theorem B.2 can be used to show $v_o \leqq v_s$ for Cases two and three. For Case two, Equation (B.9) gives

$$\lim_{\rho \to 1} v_s = (v_o^2 + v_o)/2 \geqq (v_o + v_o)/2 = v_o ,$$

since $v_o \geqq 1$. For Case three, Equation (B.11) gives $\partial v_s/\partial v_o > 1$. This result together with $\lim_{\rho \to 1} v_s \geqq v_o$ from Case two proves $v_o \leqq v_s$ for Case three. Since Case one apparently does not follow so easily from Theorem B.2, Yechiali's result is used to establish Property (3).

Theorem B.3: $v_o \leq v_s$.

Proof: Theorem 3.1 establishes that n_o and n_s in Naor (1969) are the same as in Yechiali (1971). To prove the theorem, assume $v_s < v_o$. Since $n_o = [v_o]$ and $n_s = [v_s]$, $n_o \geq n_s$. This contradicts Yechiali's result and establishes the theorem.

B.4 Demonstration of Equation (3.5)

Use of Equation (3.5) is demonstrated on the one class example problem of Table 3.1 with $\lambda = 1$. The other parameters of the model are $R = 5$, $C = 2$, $\mu = 3$, $\rho = 1/3$. $v_s = R\mu/C = 7.5$. Substitution of this result into Equation (3.5) yields

$$7.5 = \{v_o(2/3) - (1/3)\{1 - (1/3)^{v_o}\}\}/(4/9)$$

or

$$(2/3)v_o + (1/3)^{v_o+1} = 11/3 \quad .$$

$v_o = 5$ yields 3.3347 which is too small, but $v_o = 6$ yields 4.0005 which is too large. Since v_o lies between 5 and 6, $n_o = 5$ which agrees with Table 3.1.

APPENDIX C

INTRODUCTION TO MARKOV DECISION PROCESS

This appendix is a brief introduction to Markov and semi-Markov decision processes including the use of the policy iteration algorithm. The reader who desires a more thorough development is referred to Howard (1971) and Ross (1970).

C.1 Markov Decision Processes

The first process considered is a discrete time, finite horizon, finite state space, undiscounted, Markov decision process. A Markov process can occupy any of a number of states. In this paper, the states usually correspond to the number of customers in the system. Some mechanism causes the process to jump from one state to another according to a probability distribution. The jumps or transitions are equally spaced in time. The process is Markovian because knowing the present state of the process is as good as knowing the entire history of the process when trying to predict the next state. The process is a decision process because at each transition, an administrator can choose an action from among a set of actions. The administrator's choice affects the probability distribution of the next transition. Since the transitions are equally spaced in time, the process is called a discrete time process. The process has a finite horizon because it stops at some given point of time in the future. In this paper, a bound exists for the maximum number of states required for the process; thus, the process has a finite state space. The process is undiscounted because it is assumed that a dollar now is worth a dollar at any point in the future.

Define q_i to be the expected reward for the next transition given that the system is in state i. $v_i(n)$ is defined to be the expected reward of operating the system over the last n periods (jumps) given that the process is in state i, n periods from the end. $v_i(n)$ can be expressed recursively as:

$$v_i(n) = q_i + \sum_{j=1}^{N} P_{ij} v_j(n-1), \quad i = 1, \ldots, N. \qquad (C.1)$$

N is the number of states, P_{ij} is the probability of a transition from state i to j, q_i is the expected reward over the next period, and $\sum_{j=1}^{N} P_{ij} v_j(n-1)$ is the expected reward over the last n - 1 periods. To convert this process to an infinite horizon process, assume that

$$\lim_{n\to\infty} v_i(n) = \lim_{n\to\infty} (v_i + ng) \quad , \qquad (C.2)$$

where v_i is the reward for starting in state i, and g is a single reward per period or reward rate for the process. Substitution of Equation (C.2) into (C.1) yields

$$v_i + g = q_i + \sum_{j=1}^{N} P_{ij} v_j , \quad i = 1, \ldots, N. \qquad (C.3)$$

This system of N equations contains N + 1 unknowns which are g and v_i, for i = 1, ..., N.

The number of unknowns in Equation (C.3) needs to be reduced by one if the system is to be solved. Let $v_i = w_i + z$. Substitution into Equation (C.3) yields

$$w_i + z + g = q_i + \sum_{j=1}^{N} P_{ij} w_j + z\sum_{j=1}^{N} P_{ij}, \quad i = 1, \ldots, N. \tag{C.4}$$

Since $\sum_{j=1}^{N} P_{ij} = 1$, Equation (C.4) becomes

$$w_i + g = q_i + \sum_{j=1}^{N} P_{ij} w_j, \quad i = 1, \ldots, N. \tag{C.5}$$

Since Equation (C.5) is of the same form as Equation (C.3), the v_i's can be treated as relative values. This allows one of the v_i's, say v_N, to be set to zero, thereby reducing the number of variables to N.

At this point, it is convenient to develop another expression for g, the gain rate of the system. If P is the matrix of transition probabilities and $\overline{\emptyset}$ is the vector of steady state probabilities of occupying the states of the system, then,

$$\overline{\emptyset} = \overline{\emptyset} P \quad . \tag{C.6}$$

Multiplication of the i^{th} equation of Equation (C.1) by $\emptyset_i$ and summation of the N equations yield

$$\sum_{i=1}^{N} \emptyset_i g + \sum_{i=1}^{N} \emptyset_i v_i = \sum_{i=1}^{N} \emptyset_i q_i + \sum_{i=1}^{N} \sum_{j=1}^{N} P_{ij} v_j \emptyset_i \quad . \tag{C.7}$$

However,

$$\sum_{i=1}^{N} \sum_{j=1}^{N} P_{ij} v_j \emptyset_i = \sum_{j=1}^{N} v_j \sum_{i=1}^{N} P_{ij} \emptyset_i = \sum_{j=1}^{N} v_j \emptyset_j \quad ,$$

utilizing Equation (C.6). Thus, Equation (C.7) becomes

$$g = \sum_{i=1}^{N} \emptyset_i q_i \tag{C.8}$$

since $\sum_{i=1}^{N} \phi_i = 1$.

The development thus far has just considered the evaluation of the gain rate of the process. The decisions available to the administrator have not been considered. If $k = 1, \ldots, K$ indexes the alternatives available to the administrator and if he wants to maximize the expected reward of the finite horizon problem, Equation (C.1) becomes

$$v_i(n) = \max_k \{q_i(k) + \sum_{j=1}^{N} P_{ij}(k)\, v_j(n-1)\}\ , \quad i = 1, \ldots, N. \tag{C.9}$$

For the infinite horizon problem, substitution of Equation (C.2) into Equation (C.9) yields

$$\begin{aligned} v_i + ng &= \max_k \{q_i(k) + \sum_{j=1}^{N} P_{ij}(k)\, v_j + (n-1)g \sum_{j=1}^{N} P_{ij}(k)\} \\ &= \max_k \{q_i(k) + \sum_{j=1}^{N} P_{ij}(k)\, v_j + (n-1)g\}\ , \\ &\qquad\qquad i = 1, \ldots, N. \end{aligned}$$

This can be written as

$$g = \max_k \{q_i(k) + \sum_{j=1}^{N} P_{ij}(k)\, v_j - v_i\}\ , \quad i = 1, \ldots, N. \tag{C.10}$$

v_i could be dropped from the right-hand side of Equation (C.10) since it is unaffected by k.

C.2 Semi-Markov Decision Processes

A discrete time semi-Markov process[1] differs from a Markov process in that the transitions do not necessarily take place at evenly spaced intervals of time. Let $h_{ij}(m)$ be the holding time probability mass function. $h_{ij}(m)$ is the probability of holding m periods in state i given that the next transition is to state j. $\bar{\tau}_{ij} = \sum_{m=1}^{\infty} m h_{ij}(m)$ is the expected holding time in state i given that the next transition is to state j. $\bar{\tau}_i = \sum_{j=1}^{N} p_{ij} \bar{\tau}_{ij}$ is the unconditional expected waiting time in state i.

For a finite horizon, discrete time, semi-Markov process, Equation (C.1) becomes

$$v_i(n) = q_i\bar{\tau}_i + \sum_{m=1}^{n} \sum_{j=1}^{N} p_{ij}\, v_j(n-m)\, h_{ij}(m), \quad i = 1, \ldots, N. \tag{C.11}$$

As n becomes large, substitution from Equation (C.2) leads to

$$\begin{aligned} v_i + ng &= q_i\bar{\tau}_i + \sum_{j=1}^{N} p_{ij} \sum_{m=1}^{\infty} \{v_j + (n-m)g\}\, h_{ij}(m) \\ &= q_i\bar{\tau}_i + \sum_{j=1}^{N} p_{ij}\, (v_j + ng - g\bar{\tau}_{ij}) \\ &= q_i\bar{\tau}_i + \sum_{j=1}^{N} p_{ij}\, v_j + ng - g\bar{\tau}_i\,, \quad i = 1, \ldots, N. \end{aligned}$$

Thus, for an infinite horizon, Equation (C.11) becomes

$$v_i + g\bar{\tau}_i = q_i\bar{\tau}_i + \sum_{j=1}^{N} p_{ij}\, v_j\,, \quad i = 1, \ldots, N. \tag{C.12}$$

---

[1]All processes considered in the rest of this appendix will be undiscounted and will have a finite state space.

A continuous time semi-Markov process allows a continuous distribution of holding times. For a finite horizon, Equation (C.1) becomes

$$v_i(t) = q_i\bar{\tau}_i + \sum_{j=1}^{N} P_{ij} \int_0^{\infty} h_{ij}(\tau)\, v_j(t-\tau)d\tau \;, \quad i = 1, \ldots, N. \tag{C.13}$$

Continuous variables t and τ are used to index time and q_i is the expected reward per unit time in state i. $\bar{\tau}_{ij}$ becomes an integral, namely, $\int_0^{\infty} \tau h_{ij}(\tau)d\tau$, while $\bar{\tau}_i$ remains $\sum_{j=1}^{N} P_{ij}\,\bar{\tau}_{ij}$. As t becomes large, substitution from Equation (C.2) (with t replacing n) yields

$$\begin{aligned} v_i + gt &= q_i\bar{\tau}_i + \sum_{j=1}^{N} P_{ij}(v_j + gt - g\bar{\tau}_{ij}) \\ &= q_i\bar{\tau}_i + \sum_{j=1}^{N} P_{ij}\, v_j + gt - g\bar{\tau}_i, \quad i = 1, \ldots, N. \end{aligned}$$

Here, g is the gain per unit time rather than gain per period. Thus, for an infinite horizon, Equation (C.13) becomes

$$v_i + g\bar{\tau}_i = q_i\bar{\tau}_i + \sum_{j=1}^{N} P_{ij}\, v_j \;, \quad i = 1, \ldots, N, \tag{C.14}$$

which is the same as Equation (C.12) except for the new definitions of terms.

If after each transition, an administrator can choose an action from among a set of actions indexed by k, the process becomes a continuous time, infinite horizon, semi-Markov decision process. The action chosen affects the transition probability mass function and the holding time density function. If the administrator wants to maximize g, the expected reward rate of the process, Equation (C.14) becomes

$$g = \max_k \{q_i(k) + 1/\bar{\tau}_i(k)\}\{\sum_{j=1}^{N} P_{ij}(k)\, v_j - v_i)\} ,$$

$$i = 1, \ldots, N. \quad (C.15)$$

Equation (C.15) suggests the following algorithm which can be shown to converge to the optimal solution for problems like those of this paper:

Algorithm C.1 (Policy Iteration Algorithm)

Step 1: Set $v_i = 0$ for all i. $\bar{d}(0) = \bar{0}$, where $\bar{d}(h)$ is a vector of actions chosen for each state for iteration h of the algorithm. Set $h = 0$.

Step 2: $h = h + 1$. For each state i, find $d_i(h)$, the index of the action that yields

$$\max_k \left(q_i(k) + \{1/\bar{\tau}_i(k)\} \{\sum_{j=1}^{N} P_{ij}(k)\, v_j - v_i\}\right) .$$

Step 3: If $\bar{d}(h) = \bar{d}(h - 1)$, stop.

Step 4: Solve

$$v_i + g\bar{\tau}_i\{d_i(h)\} = q_i\{d_i(h)\}\bar{\tau}_i\{d_i(h)\} + \sum_{j=1}^{N} P_{ij}\{d_i(h)\}v_j ,$$

$$i = 1, \ldots, N,$$

for g and v_1 through v_{N-1} ($v_N = 0$). Go to Step 2.

Algorithm C.1 is the version of policy iteration used in this paper.

C.3 Example Problem

As in Section B.4, the one class example from Table 3.1 with $\lambda = 1$ is used to demonstrate Algorithm C.1. The remaining parameters of the model are R = 5, C = 2, and μ = 3. Since $n_o \leq n_s$ and $n_s = [R\mu/C] = 7$, a bound on the size of the state space is seven. The set of possible actions is indexed by k = 0 or 1 (this indexing scheme agrees with the body of the paper rather than with Section C.1). k = 0 if an arrival is rejected, and k = 1 if an arrival is accepted into the system. To avoid a trivial system, action zero cannot be chosen when the state of the system is zero; that is, if the system empties out, customers must be allowed back into it. $n_o \leq 7$ implies that action one cannot be chosen when the state of the system is seven.

P(0) is the matrix of transition probabilities for alternative k = 0.

$$
P(0) = \begin{array}{c} \\ \text{From State} \end{array}
\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array}
\overset{\text{To State}}{\begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array}}
$$

$$
\begin{array}{c|cccccccc}
 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
\hline
0 & - & - & - & - & - & - & - & - \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
5 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
6 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
7 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{array}
$$

The dashes in a row indicate that the action cannot be selected when the system is in the state represented by the row. For alternative zero, the next transition is sure to be the completion of a service. P(1) is the matrix of transition probabilities for alternative one.

$$
P(1) = \begin{array}{c c} & \text{To State} \\ & \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{array}{r} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ \text{From State } 5 \\ 6 \\ 7 \end{array} & \left[\begin{array}{cccccccc} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.75 & 0 & 0.25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.75 & 0 & 0.25 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.75 & 0 & 0.25 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.75 & 0 & 0.25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.75 & 0 & 0.25 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.75 & 0 & 0.25 \\ - & - & - & - & - & - & - & - \end{array}\right] \end{array}
$$

$P_{01}(1) = 1$ because the next transition is sure to be an arrival if alternative one is chosen when the state of the system is zero. For states other than zero, the service rate competes with the arrival rate. Since the total rate out of a state other than zero is $\lambda + \mu = 4$ and since μ provides 3/4 of the total rate and λ provides 1/4 of the total, the probability that the next transition is the completion of a service is 3/4 while the probability that the next transition is an arrival is 1/4.

$\overline{\tau}_i(k)$ is simply the reciprocal of the transition rate out of state i under action k. Thus,

$$
\hat{\overline{\tau}}(0) = \begin{bmatrix} - \\ 0.33 \\ 0.33 \\ 0.33 \\ 0.33 \\ 0.33 \\ 0.33 \\ 0.33 \end{bmatrix} \quad \text{and} \quad \hat{\overline{\tau}}(1) = \begin{bmatrix} 1 \\ 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \\ - \end{bmatrix} .
$$

Let b(0) be the matrix of expected rewards for the various transitions under alternative zero. b(0) is the null matric because no customers are allowed to join the system under action zero. b(1) is the matrix of expected rewards for the various transitions under alternative one.

To State

$$
\begin{array}{ll}
 & \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\
b(1) = & \begin{array}{r} 0 \\ 1 \\ 2 \\ 3 \\ \text{From State } 4 \\ 5 \\ 6 \\ 7 \end{array}
\left[\begin{array}{cccccccc}
0 & 4.33 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 3.67 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 3.00 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 2.33 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1.67 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1.00 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.33 \\
- & - & - & - & - & - & - & -
\end{array}\right]
\end{array}
$$

The entries $b_{i,i+1}(1)$ are

$$b_{i,i+1}(1) = R - (i + 1)C/\mu \quad .$$

The components of the vector of expected rewards per transition, $r_i(k)$, are

$$r_i(k) = \sum_{j=0}^{n_s} P_{ij}(k)\ b_{ij}(k) \quad .$$

Here,

$$\bar{r}(0) = \begin{pmatrix} - \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \text{and} \quad \bar{r}(1) = \begin{pmatrix} 4.333 \\ 0.917 \\ 0.750 \\ 0.583 \\ 0.417 \\ 0.250 \\ 0.083 \\ - \end{pmatrix} .$$

The expected rewards per unit time in state i under action k are

$$q_i(k) = r_i(k)/\bar{\tau}_i(k) .$$

Thus,

$$\bar{q}(0) = \begin{pmatrix} - \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \text{and} \quad \bar{q}(1) = \begin{pmatrix} 4.33 \\ 3.67 \\ 3.00 \\ 2.33 \\ 1.67 \\ 1.00 \\ 0.33 \\ - \end{pmatrix}$$

The information required for Algorithm C.1 is now at hand. Application of the algorithm proceeds as follows:

Step 1: $v_0 = v_1 = \ldots v_7 = 0$. $\overline{d}(0) = 0$.

Step 2: h = 1.

$$\overline{d}(1) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}.$$

Step 3: Continue.

Step 4: g = 4.001.

$$\overline{v} = \begin{pmatrix} 7.84 \\ 7.50 \\ 6.84 \\ 5.85 \\ 4.56 \\ 3.00 \\ 1.33 \\ 0.00 \end{pmatrix}$$

Step 2: h = 2.

$$\overline{d}(2) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Step 3: Continue.

Step 4: $g = 4.003.$

$$\bar{v} = \begin{pmatrix} 7.18 \\ 6.85 \\ 6.19 \\ 5.23 \\ 4.00 \\ 2.67 \\ 1.33 \\ 0.00 \end{pmatrix}$$

Step 2: $h = 3$

$$\bar{d}(3) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Step 3: Stop.

$\bar{d}(3)$ indicates that $n_o = 5$ which agrees with Table 3.1 as does the optimal value of g. Section 4.6 contains another example of policy iteration and gives some insight into the relationship between $\bar{d}(1)$ and the optimal policy for the individual optimum problem.

APPENDIX D

USER'S GUIDE AND LISTING OF COMPUTER PROGRAM

Several different computer programs were used in testing the models developed in Chapters III through V. The program that is presented here was used for many of the tests and was also used as a starting point for the other programs that were developed. The program is written in FORTRAN for the WATFIV compiler on the IBM 370-3033 computer at The Pennsylvania State University. After the program and its use are described, more will be said about how the program was adapted for other uses.

D.1 Introduction to the Program

The program is called POLIT to reflect the fact that it performs policy iteration. As discussed in Appendix C, the particular type of process this program is designed to solve is a continuous time, infinite horizon, finite state space, undiscounted, semi-Markov decision process. This program is capable of solving Models I, II, and III. However, the optimal solution for Model III, the Erlang service time model is the phase optimum solution which cannot always be implemented.

The program consists of two parts. The first builds the $P(\overline{k})$ matrices, $\overline{q}(\overline{k})$ vectors, and $\overline{\hat{\tau}}(\overline{k})$ vectors from the input data. The second part performs policy iteration to solve for the optimal policy and maximum gain rate. Other information provided for the optimal policy is the set of v_i's and the vector of steady state probabilities of the system occupying the various states, $\overline{\emptyset}$.

The MAIN routine reads the input data and prints it for user verification. MAIN also controls the flow of the program. The subroutines are described in the order in which they are called. See Figure D.1 for a flow diagram of the program.

Subroutine PROB generates a transition matrix, $P(\overline{k})$, for each possible action $\overline{k}$. 2^M transition matrices are generated since there are 2^M possible actions. The program will handle up to five classes ($M = 5$). Since transitions can only be made to the next higher or lower state, each row of a transition matrix contains only two elements. The first element is the probability of a transition to the next higher state while the second is the probability of a transition to the next lower state. Subroutine PROB1 is used by PROB to fill in the transition matrices.

Subroutine PROFIT generates the matrices of expected rewards of a transition, $b(\overline{k})$, for each possible action $\overline{k}$. Since only transitions to the next higher state can yield a nonzero expected reward, each row of the $b(\overline{k})$ matrices contains only one element. Thus, the $b(\overline{k})$ matrices are dimensioned and used as vectors.

Subroutine EXPECT first generates the vectors of expected rewards per transition, $\overline{r}(\overline{k})$, from

$$r_i(\overline{k}) = \sum_{j=o}^{n_s^*} p_{ij}(\overline{k})\, b_{ij}(\overline{k}) \quad .$$

EXPECT then calculates the vectors of unconditional expected waiting times, $\hat{\overline{\tau}}(\overline{k})$. Also, using

$$q_i(\overline{k}) = r_i(\overline{k})/\overline{\tau}_i(\overline{k}) \quad ,$$

EXPECT computes the vectors of expected rewards per unit time, $\overline{q}(\overline{k})$.

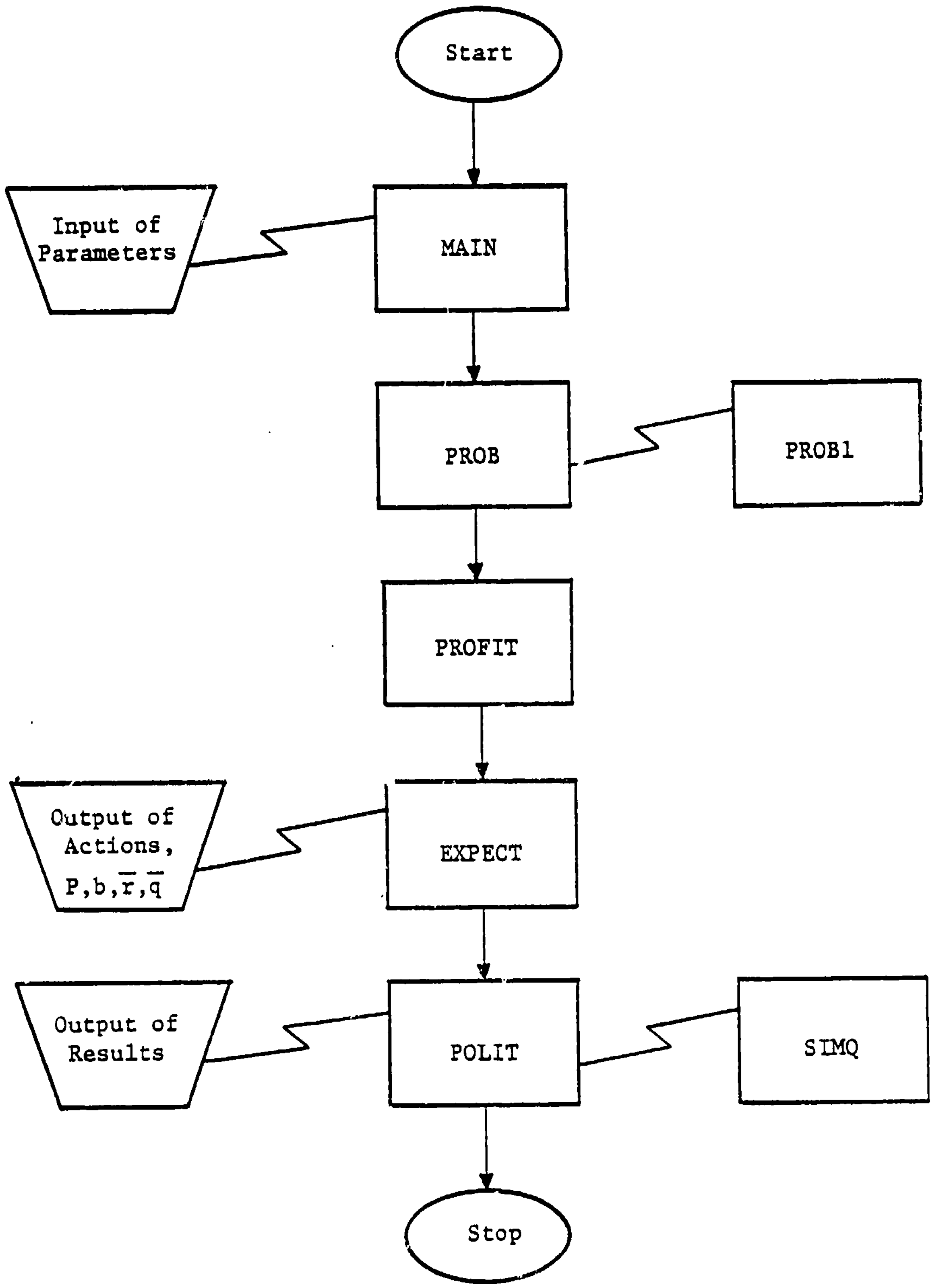

Figure D.1 Flow Diagram of POLIT

EXPECT prints the code which relates the number of an action to the classes that action admits. If the input parameter IFLAG is set to one, EXPECT prints the condensed matrices $P(\overline{k})$ and $b(\overline{k})$ along with the vectors $\overline{r}(\overline{k})$ and $\overline{q}(\overline{k})$.

Subroutine POLIT implements the policy iteration algorithm. It utilizes subroutine SIMQ from IBM (1970) to solve the system of linear equations generated by Step (4) of the algorithm. POLIT outputs g, $\overline{d}$, and $\overline{v}$ for each iteration and prints the vector of steady state probabilities, $\overline{\emptyset}$, for the last iteration (which yields the optimal policy). In addition, POLIT prints the vector of optimal expected rewards per unit time. Finally, POLIT uses

$$g_{P*} = \sum_{i=o}^{n_s^*} \emptyset_i^{P*} q_i^{P*} ,$$

where P* represents the optimal policy, to again compute g. The comparison of the two values of g should give the user a feel for the roundoff error that exists in the solution of his problem.

D.2 Use of the Program

The program input is described first and then illustrated for the example described in Appendix C. Output for this problem is given with the listing of the program.

Card one of the input gives the number of classes of customers in the problem, the number of phases of service (Erlang number, h), and mean service rate for each phase of service. The user is reminded that if the overall service rate is μ, then the service rate for each phase of service is $h\mu$. The format of this information is 2I5, F10.2. If the user wants to see the $P(\overline{k})$ and $b(\overline{k})$ matrices and the $\overline{r}(\overline{k})$ and

$\overline{q}(\overline{k})$ vectors that are generated, he should also include a 1 in column 25 of this card.

Card two of the input gives the reward for service for each class of customer. The reward for class one comes first followed by the reward for class two and so on. The format for the rewards is 8F10.2.

Card three of the input lists the cost per unit time in the system for each class of customer. Again, the costs are input in class order. The format for the costs is also 8F10.2.

Card four of the input contains the self-optimum balking points, n_{s_m}, for each class m. These are computed from

$$n_{s_m} = [R_m \mu / C_m + (h - 1)/2h] \quad .$$

The self-optimum balking points are input in class order using a format of 16I5. The classes must be numbered so that the class with the largest n_{s_m} is class one.

Card five contains the arrival rate of each class of customers. The arrival rates are input in class order with format 8F10.2.

The input for the one class example of Appendix C follows:

Card One

Column	5	10	11-20	25
Input	1	1	3.0	1

Card Two

Column	1-10
Input	5.0

Card Three

Column 1-10

Input 2.0

Card Four

Column 5

Input 7

Card Five

Column 1-10

Input 1.0

The output for this example is given with the listing of the program. Since each section of output includes descriptive information, the output will not be described here.

D.3 Other Programs

The arrays of POLIT had to be enlarged to run the Erlang service time model for the airport example in Chapter VI. The number of states required for the Erlang 8 model is 145, so the dimensions representing the number of states (50 in the listing) must be at least 145. Also, since the optimal policy for the Erlang model was not implementable, POLIT was modified to check policies that could be implemented that were near the phase optimum policy. The modifications were to provide for the input of the policies to be tested and to alter subroutine POLIT so that only the policy evaluation portion of Algorithm C.1, the determination of g and $\overline{v}$, was performed for each input policy.

Two programs that are on the IBM 370-3033 system at The Pennsylvania State University were also used. LPFREE [see Ilgen (1978)] was used to test the linear programming formulation of Models I and II.

MPSX [see IBM (1972)] was used to test the mixed integer programming formulation of Model III.

D.4 Listing and Output

A listing of POLIT and output for the example follow.

```
.. HASP-II***.........START JOB 1501....K3289646......RUE R            ......
.. HASP II***.........START JOB 1501....K3289646......RUE R            .....
.. HASP-II***.........START JOB 1501....K3289646......RUE P            .....
.. HASP-II***.........START JOB 1501....K3289646......RUE R            ......
.. HASP-II***.........START JOB 1501....K3289646......RUE P            .....
```

```
5:01 DATE:  06/12/79              H A S P   S Y S T E M   L O G

 JOB 'U3764,T=0020,R=05000,S=280,       ','RUE R              '        AA
TTAF,TRAIN=TN,FORMS=16                                           00000025
STEP WAS EXECUTED - COND CODE 0000
EP /SETTAF  / START 79163.1115
EP /SETTAF  / STOP  79163.1115 CPU   0MIN 00.03SEC MAIN   8K LCS   0K
CG                                                               00000050
 *
UT DD *
 RXR03.DATA1                                                     00011950
STEP WAS EXECUTED - COND CODE 0000
EP /DATA    / START 79163.1115
EP /DATA    / STOP  79163.1115 CPU   0MIN 00.52SEC MAIN 280K LCS   0K
OB /K3289646/ START 79163.1115
OB /K3289646/ STOP  79163.1115 CPU   0MIN 00.55SEC
```

```
ATFIV *** 19-3 (05/26/79--1049)

      C     MAIN PROGRAM
      C*********************************************************************
      C        CLASS ONE SHOULD HAVE LARGEST REWARD/COST, CLASS 2 NEXT LARGEST
      C        AND SO ON.
      C             DEFINITIONS
      C        IFLAG - IF.NE.1, SHORT O/P; IF 1, O/P INCLUDES P,B,R,Q MATRICES
      C        LCLASS - NUMBER OF CLASSES OF CUSTOMERS
      C        REWARD(L) - CLASS REWARDS FOR SERVICE
      C        COST(L) - CLASS TIME COSTS
      C        NLIM(L) - FORCED CLASS BALKING POINTS - CUSTOMERS IN SYSTEM
      C                 AS OF 18JAN79 ONLY NLIM(1) IS USED
      C        U - MEAN SERVICE RATE
      C        KPHASE - NUMBER OF PHASES OF SERVICE
      C        FLAMDA(L) - CLASS ARRIVAL RATES GIVEN STATE < NLIM(L)
      C        I - STATE OF SYSTEM - NUMBER OF PHASES OF SERVICE IN THE SYSTEM
      C        J - 1 IF STATE BECOMES I + KPHASE AFTER TRANS, 2 IF I-1
      C        P(L,I,J) - CONDENSED TRANSITION MATRICES
      C        R(L,I) - EXPECTED PROFIT / OCCUPANCY
      C        Q(L,I) - EXPECTED PROFIT / UNIT TIME
      C        T(L,I) - EXPECTED WAITING TIME IN STATE
      C        V(I) - RELATIVE VALUE OF STARTING IN STATE I
      C        G - GAIN RATE - EXPECTED PROFIT / TIME
      C        ID(I) - POLICY CHOSEN THIS ITERATION
      C        B(L, I) - VECTORS OF EXPECTED PROFIT OF TRANS FROM I - I+KPHASE
      C*********************************************************************
  1         COMMON LCLASS, REWARD( 50), COST( 50), NLIM( 50), U, KPHASE,
           1 FLAMDA( 50), P( 32, 50,  2), R( 32, 50),Q( 32, 50), T( 32, 50),
           2 V( 50), G, ID( 50), B( 32, 50), NSTATE
      C*********************************************************************
      C        READ INPUT
      C*********************************************************************
  2         IN = 5
```

```
      IOUT = 6
      READ(IN, 100) LCLASS, KPHASE, U, IFLAG
  100 FORMAT(2I5, F10.2, I5)
      READ(IN,200) (REWARD(L), L = 1, LCLASS)
  200 FORMAT(8F10.2)
      READ(IN, 200) (COST(L), L = 1, LCLASS)
      READ(IN, 300) (NLIM(L), L = 1, LCLASS)
  300 FORMAT(16I5)
      READ(IN, 200) (FLAMDA(L), L = 1, LCLASS)
C*********************************************************************
C        ECHOCHECK OF INPUT
C*********************************************************************
      WRITE(IOUT, 400)
  400 FORMAT(1H1)
      WRITE(IOUT, 500) LCLASS, KPHASE, U
  500 FORMAT(1H0, 10X, 'THIS QUEUE CONTROL PROBLEM HAS',I5, ' CLASSES O'
     1, 'F CUSTOMERS. THE MEAN SERVICE RATE OF'//20X,' THE ERLANG',I5,
     2 ' SERVER  IS ', F10.2)
      WRITE(IOUT, 600)
  600 FORMAT(1H0, 10X, 'INPUT VALUES OF THE PARAMETERS FOR EACH CLASS',
     1 ' ARE GIVEN IN THE FOLLOWING TABLE')
      WRITE(IOUT, 700)
  700 FORMAT(1H0, 10X, 'CLASS', 10X, 'REWARD', 15X, 'COST', 16X,
     1 'NLIM', 15X, 'FLAMDA')
      DO 5010 L = 1, LCLASS
 5010 WRITE(IOUT, 800) L, REWARD(L), COST(L),NLIM(L),FLAMDA(L)


  800 FORMAT(1H0, 8X,I5,  9X, F10.2, 10X, F10.2, 12X, I5, 13X, F10.2)
      NSTATE = 1 + KPHASE * NLIM(1)
C*********************************************************************
C        GENERATE TRANSITION MATRICES
C*********************************************************************
      CALL PROB
C*********************************************************************
C        GENERATE EXPECTED PROFIT OF NEXT TRANSITION MATRICES
C*********************************************************************
      CALL PROFIT
      IF(IFLAG.NE.1) GO TO 5047
C*********************************************************************
C        OUTPUT P AND B
C*********************************************************************
      LCLP1 = 2 ** LCLASS
      DO 5045 L = 1, LCLP1
      WRITE(IOUT, 2100) L
 2100 FORMAT(1H0, 20X, 'P AND B MATRICES FOLLOW FOR ACTION',I5)
      WRITE(IOUT, 2200)
 2200 FORMAT(1H0, 30X, 'TRANSITION PROBABILITIES')
      DO 5030 I = 1, NSTATE
 5030 WRITE(IOUT, 2300) (P(L,I,J), J = 1, 2)
 2300 FORMAT(8F10.6)
      WRITE(IOUT, 2400)
 2400 FORMAT(1H0, 30X, 'EXPECTED REWARDS FOR TRANSITION')
      WRITE(IOUT, 2300) (B(L,I), I = 1, NSTATE)
 5045 CONTINUE
C*********************************************************************
C        GENERATE EXPECTED PROFIT/OCCUPANCY AND EXPECTED REWARD/TIME
C*********************************************************************
 5047 CALL EXPECT
      IF(IFLAG.NE.1) GO TO 5060
C*********************************************************************
C        OUTPUT R AND Q
C*********************************************************************
      DO 5050 L = 1, LCLP1
      WRITE(IOUT, 2500) L
 2500 FORMAT(1H0, 20X, 'R AND Q VECTORS FOLLOW FOR ACTION', I5)
```

```
      WRITE(IOUT, 2600)
 2600 FORMAT(1H0, 30X, 'R(I)')
      WRITE(IOUT, 2700) (R(L,I), I = 1, NSTATE)
 2700 FORMAT(8F10.3)
      WRITE(IOUT, 2800)
 2800 FORMAT(1H0, 30X, 'Q(I)')
      WRITE(IOUT, 2700) (Q(L,I), I = 1, NSTATE)
 5050 CONTINUE
C***********************************************************************
C         PERFORM POLICY ITERATION
C***********************************************************************
 5060 CALL POLIT
      STOP
      END


      SUBROUTINE PROB
      COMMON LCLASS, REWARD( 50), COST( 50), NLIM( 50), U, KPHASE,
     1 FLAMDA( 50), P( 32, 50,  2), R( 32, 50), Q( 32, 50), T( 32, 50),
     2 V( 50), G, ID( 50), B( 32, 50), NSTATE
C***********************************************************************
C         2 ** LCLASS TRANSITION MATRICES WILL BE GENERATED
C         1ST - LCLASS CLASSES TAKEN 0 AT A TIME
C         2ND - LCLASS CLASSES TAKEN 1 AT A TIME
C         .
C         .
C         LAST - LCLASS CLASSES TAKEN LCLASS AT A TIME
C         NOTE - SO FAR 5 IS UPPER LIMIT ON LCLASS
C***********************************************************************
      LCLP1 = 2 ** LCLASS
      DO 10 L = 1, LCLP1
      DO 10 I = 1, NSTATE
      DO 10 J = 1, 2
   10 P(L,I,J) = 0.0
C***********************************************************************
C         GENERATE TRANSITION MATRIX FOR LCLASS CLASSES TAKEN 0 AT A TIME
C***********************************************************************
      DO 20 I = 2, NSTATE
      J = 2
   20 P(1,I,J) = 1.0
C***********************************************************************
C         GENERATE TRANSITION MATRICES FOR LCLASS CLASSES TAKEN 1
C         AT A TIME
C***********************************************************************
      LL = 1
      DO 100 L = 1, LCLASS
      TLAMDA = FLAMDA(L)
      LL = LL + 1
  100 CALL PROB1(LL,TLAMDA)
C***********************************************************************
C         GENERATE TRANSITION MATRICES FOR LCLASS CLASSES TAKEN 2
C         AT A TIME
C***********************************************************************
      IF(LCLASS.LT.2) GO TO 1000
      LCLM1 = LCLASS - 1
      DO 200 L = 1, LCLM1
      LP1 = L + 1
      DO 200 L1 = LP1, LCLASS
      LL = LL + 1
      TLAMDA = FLAMDA(L) + FLAMDA(L1)
      CALL PROB1(LL, TLAMDA)
  200 CONTINUE
C***********************************************************************
C         GENERATE TRANSITION MATRICES FOR LCLASS CLASSES TAKEN 3
C         AT A TIME
C***********************************************************************
```

```
      IF(LCLASS.LT.3) GO TO 1000
      LCLM2 = LCLASS - 2
      DO 300 L = 1, LCLM2
      LP1 = L + 1
      DO 300 L1 = LP1, LCLM1
      L1P1 = L1 + 1
      DO 300 L2 = L1P1, LCLASS
      LL = LL + 1
      TLAMDA = FLAMDA(L) + FLAMDA(L1) + FLAMDA(L2)
      CALL PROB1(LL, TLAMDA)

  300 CONTINUE
C*******************************************************************
C        GENERATE TRANSITION MATRICES FOR LCLASS CLASSES TAKEN 4
C        AT A TIME
C*******************************************************************
      IF(LCLASS.LT.4) GO TO 1000
      LCLM3 = LCLASS - 3
      DO 400 L = 1, LCLM3
      LP1 = L + 1
      DO 400 L1 = LP1, LCLM2
      L1P1 = L1 + 1
      DO 400 L2 = L1P1, LCLM1
      L2P1 = L2 + 1
      DO 400 L3 = L2P1, LCLASS
      LL = LL + 1
      TLAMDA = FLAMDA(L) + FLAMDA(L1) + FLAMDA(L2) + FLAMDA(L3)
      CALL PROB1(LL, TLAMDA)
  400 CONTINUE
C*******************************************************************
C        GENERATE TRANSITION MATRICES FOR LCLASS CLASSES TAKEN 5
C        AT A TIME. NOTE FOR THIS PROGRAM, MAX VALUE OF LCLASS IS 5.
C*******************************************************************
      IF(LCLASS.LT.5) GO TO 1000
      TLAMDA = 0.0
      DO 500 L = 1, 5
  500 TLAMDA = TLAMDA + FLAMDA(L)
      LL = LL + 1
      CALL PROB1(LL, TLAMDA)
      IOUT = 6
      IF(LCLASS.GT.5) WRITE(IOUT, 2000)
 2000 FORMAT(1H0, 20X, 'PROBLEM HAS MORE THAN 5 CLASSES. ONLY CLASSES',
     1 ' TO 5 WILL BE USED.')
 1000 RETURN
      END

      SUBROUTINE PROB1(LL, TLAMDA)
      COMMON LCLASS, REWARD( 50), COST( 50), NLIM( 50), U, KPHASE,
     1 FLAMDA( 50), P( 32, 50,  2), R( 32, 50), Q( 32, 50), T( 32, 50),
     2 V( 50), G, ID( 50), B( 32, 50), NSTATE
C*******************************************************************
C        PROB1 FILLS IN P MATRIX FOR SUBROUTINE PROB
C*******************************************************************
      NSM1 = NSTATE - 1
      TOTAL = TLAMDA + U
      PUP = TLAMDA / TOTAL
      PDOWN = U / TOTAL
      P(LL, 1, 1) = 1.0
      DO 40 I = 2, NSM1
      P(LL, I, 1) = PUP
   40 P(LL, I, 2) = PDOWN
      RETURN
      END
```

```
      SUBROUTINE PROFIT
      COMMON LCLASS, REWARD( 50), COST( 50), NLIM( 50), U, KPHASE,
     1 FLAMDA( 50), P( 32, 50,  2), R( 32, 50), Q( 32, 50), T( 32, 50),
     2 V( 50), G, ID(50), B( 32, 50), NSTATE
C***********************************************************************
C          2 ** LCLASS EXPECTED PROFIT OF THE NEXT TRANSITION MATRICES
C          WILL BE GENERATED.
C          1ST - LCLASS CLASSES TAKEN 0 AT A TIME
C          2ND - LCLASS CLASSES TAKEN 1 AT A TIME
C          .
C          .
C          LAST - LCLASS CLASSES TAKEN LCLASS AT A TIME
C***********************************************************************
      NSM1 = NSTATE - KPHASE
      LCLP1 = 2 ** LCLASS
      DO 10 L = 1, LCLP1
      DO 10 I = 1, NSTATE
   10 B(L, I) = 0.0
C***********************************************************************
C          FOR LCLASS CLASSES TAKEN 0 AT A TIME, B = 0.
C
C          GENERATE PROFIT MATRIX FOR LCLASS CLASSES TAKEN 1 AT A TIME.
C***********************************************************************
      LL = 1
      DO 100 L = 1, LCLASS
      LL = LL + 1
      TRATE = FLAMDA(L)
      DO 50 I = 1, NSM1
      J = I + KPHASE
      FJ = J - 1
      SUM = FLAMDA(L) * (REWARD(L) - FJ * COST(L) / U)


   50 B(LL, I) = SUM / TRATE
  100 CONTINUE
C***********************************************************************
C          GENERATE PROFIT MATRICES FOR LCLASS CLASSES TAKEN 2 AT A TIME
C***********************************************************************
      IF(LCLASS.LT.2) GO TO 1000
      LCLM1 = LCLASS - 1
      DO 200 L = 1, LCLM1
      LP1 = L + 1
      DO 200 L1 = LP1, LCLASS
      LL = LL + 1
      TRATE = FLAMDA(L) + FLAMDA(L1)
      DO 150 I = 1, NSM1
      J = I + KPHASE
      FJ = J - 1
      SUM = FLAMDA(L) * (REWARD(L) - FJ * COST(L) / U) +
     1 FLAMDA(L1) * (REWARD(L1) - FJ * COST(L1) / U)
  150 B(LL, I) = SUM / TRATE
  200 CONTINUE
C***********************************************************************
C          GENERATE PROFIT MATRICES FOR LCLASS CLASSES TAKEN 3 AT A TIME
C***********************************************************************
      IF(LCLASS.LT.3) GO TO 1000
      LCLM2 = LCLASS - 2
      DO 300 L = 1, LCLM2
      LP1 = L + 1
      DO 300 L1 = LP1, LCLM1
      L1P1 = L1 + 1
      DO 300 L2 = L1P1, LCLASS
      LL = LL + 1
      TRATE = FLAMDA(L) + FLAMDA(L1) + FLAMDA(L2)
      DO 250 I = 1, NSM1
```

```
      J = I + KPHASE
      FJ = J - 1
      SUM = FLAMDA(L) * (REWARD(L) - FJ * COST(L) / U)
     1 + FLAMDA(L1) * (REWARD(L1) - FJ * COST(L1) / U)
     2 + FLAMDA(L2) * (REWARD(L2) - FJ * COST(L2) / U)
  250 B(LL, I) = SUM / TRATE
  300 CONTINUE
C***********************************************************************
C       GENERATE PROFIT MATRICES FOR LCLASS CLASSES TAKEN 4 AT A TIME
C***********************************************************************
      IF(LCLASS.LT.4) GO TO 1000
      LCLM3 = LCLASS - 3
      DO 400 L = 1, LCLM3
      LP1 = L + 1
      DO 400 L1 = LP1, LCLM2
      L1P1 = L1 + 1
      DO 400 L2 = L1P1, LCLM1
      L2P1 = L2 + 1
      DO 400 L3 = L2P1, LCLASS
      LL = LL + 1
      TRATE = FLAMDA(L) + FLAMDA(L1) + FLAMDA(L2) + FLAMDA(L3)
      DO 350 I = 1, NSM1
      J = I + KPHASE
      FJ = J - 1
      SUM = FLAMDA(L) * (REWARD(L) - FJ * COST(L) / U)
     1 + FLAMDA(L1) * ( REWARD(L1) - FJ * COST(L1) / U)
     2 + FLAMDA(L2) * (REWARD(L2) - FJ * COST(L2) / U)
     3 + FLAMDA(L3) * (REWARD(L3) - FJ * COST(L3) / U)

  350 B(LL, I) = SUM / TRATE
  400 CONTINUE
C***********************************************************************
C       GENERATE PROFIT MATRIX FOR 5 CLASSES TAKEN 5 AT A TIME
C       NOTE UPPER BOUND ON LCLASS IS 5.
C***********************************************************************
      IF(LCLASS.LT.5) GO TO 1000
      LL = LL + 1
      TRATE = 0.0
      DO 430 L = 1, 5
  430 TRATE = TRATE + FLAMDA(L)
      DO 500 I = 1, NSM1
      J = I + KPHASE
      FJ = J - 1
      SUM = 0.0
      DO 450 L = 1, 5
  450 SUM = SUM + FLAMDA(L) * (REWARD(L) - FJ * COST(L) / U)
  500 B(LL, I) = SUM / TRATE
 1000 RETURN
      END


      SUBROUTINE EXPECT
      COMMON LCLASS, REWARD( 50), COST( 50), NLIM( 50), U, KPHASE,
     1 FLAMDA( 50), P( 32, 50,  2), R( 32, 50), Q( 32, 50), T( 32, 50),
     2 V( 50), G, ID( 50), B( 32, 50), NSTATE
C***********************************************************************
C          R( ,I) = B( ,I) * T( ,I,1)                              N
C***********************************************************************
      LCLP1 = 2 ** LCLASS
      NSM1 = NSTATE - KPHASE
      DO 10 L = 1, LCLP1
      DO 10 I = 1, NSTATE
      T(L,I) = 0.0
      Q(L,I) = 0.0
   10 R(L,I) = 0.0
```

```
      DO 50 L = 2, LCLP1
      DO 40 I = 1, NSM1
   40 R(L,I) = P(L,I,1) * B(L,I)
   50 CONTINUE
C*****************************************************************************
C        GENERATE T FOR LCLASS CLASSES TAKEN 0 AT A TIME
C*****************************************************************************
      IOUT = 6
      LL = 1
      L = 0
      WRITE(IOUT,2000) LL, L
 2000 FORMAT(1H0,20X,'ACTION',I5,' ADMITS CLASSES', 5I5)
      DO 60 I = 2, NSTATE
   60 T(LL, I) = 1/U
C*****************************************************************************
C        GENERATE T FOR LCLASS CLASSES TAKEN 1 AT A TIME
C*****************************************************************************


      DO 100 L = 1, LCLASS
      LL = LL + 1
      TRATE = FLAMDA(L)
      WRITE(IOUT, 2000) LL, L
      T(LL, 1) = 1/TRATE
      TRATE = TRATE + U
      DO 70 I = 2, NSM1
   70 T(LL, I) = 1/TRATE
  100 CONTINUE
C*****************************************************************************
C        GENERATE T FOR LCLASS CLASSES TAKEN 2 AT A TIME
C*****************************************************************************
      IF(LCLASS.LT.2) GO TO 1000
      LCLM1 = LCLASS - 1
      DO 200 L = 1, LCLM1
      LP1 = L + 1
      DO 200 L1 = LP1, LCLASS
      LL = LL + 1
      TRATE = FLAMDA(L) + FLAMDA(L1)
      WRITE(IOUT, 2000) LL, L, L1
      T(LL, 1) = 1/ TRATE
      TRATE = TRATE + U
      DO 170 I = 2, NSM1
  170 T(LL,I) = 1/ TRATE
  200 CONTINUE
C*****************************************************************************
C        GENERATE T FOR LCLASS CLASSES TAKEN 3 AT A TIME
C*****************************************************************************
      IF(LCLASS.LT.3) GO TO 1000
      LCLM2 = LCLASS - 2
      DO 300 L = 1, LCLM2
      LP1 = L + 1
      DO 300 L1 = LP1, LCLM1
      L1P1 = L1 + 1
      DO 300 L2 = L1P1, LCLASS
      LL = LL + 1
      TRATE = FLAMDA(L) + FLAMDA(L1) + FLAMDA(L2)
      WRITE(IOUT, 2000) LL, L, L1, L2
      T(LL,1) = 1/ TRATE
      TRATE = TRATE + U
      DO 270 I = 2, NSM1
  270 T(LL, I) = 1/ TRATE
  300 CONTINUE
C*****************************************************************************
C        GENERATE T FOR LCLASS CLASSES TAKEN 4 AT A TIME
C*****************************************************************************
      IF(LCLASS.LT.4) GO TO 1000
```

```
      LCLM3 = LCLASS - 3
      DO 400 L = 1, LCLM3
      LP1 = L + 1
      DO 400 L1 = LP1, LCLM2
      L1P1 = L1 + 1
      DO 400 L2 = L1P1, LCLM1
      L2P1 = L2 + 1
      DO 400 L3 = L2P1, LCLASS
      LL = LL + 1
      TRATE = FLAMDA(L) + FLAMDA(L1) + FLAMDA(L2) + FLAMDA(L3)
      WRITE(IOUT, 2000) LL, L, L1, L2, L3
      T(LL,1) = 1/TRATE
      TRATE = TRATE + U

      DO 370 I = 2, NSM1
  370 T(LL, I) = 1/ TRATE
  400 CONTINUE
C*********************************************************************
C         GENERATE T FOR 5 CLASSES TAKEN 5 AT A TIME
C         NOTE LCLASS IS BOUNDED BY 5
C*********************************************************************
      IF(LCLASS.LT.5) GO TO 1000
      LL = LL + 1
      TRATE = 0.0
      DO 430 L = 1, 5
  430 TRATE = TRATE + FLAMDA(L)
      WRITE(IOUT, 2000) LL, (I, L = 1, 5)
      T(LL, 1) = 1/ TRATE
      TRATE = TRATE + U
      DO 470 I = 2, NSM1
  470 T(LL, I) = 1/ TRATE
 1000 CONTINUE
C*********************************************************************
C         GENERATE Q FOR ALL POLICIES
C*********************************************************************
      DO 1090 I = 2, NSTATE
 1090 Q(1,I) = P(1,I) / T(1,I)
      DO 1200 L = 2, LCLP1
      DO 1100 I = 1, NSM1
 1100 Q(L,I) = P(L,I) / T(L,I)
 1200 CONTINUE
      RETURN
      END


      SUBROUTINE POLIT
      COMMON LCLASS, REWARD( 50), COST( 50), NLIM( 50), U, KPHASE,
     1 FLAMDA( 50), P( 32, 50, 2), R( 32, 50), Q( 32, 50), T( 32, 50),
     2 V( 50), G, ID( 50), P( 32, 50), NSTATE
C*********************************************************************
C         USE POLICY ITERATION TO FIND OPTIMAL POLICY - UNDISCOUNTED,
C         INFINITE HORIZON.
C         AA WILL CONTAIN COEFFICIENTS OF SYSTEM OF LINEAR EQUATIONS IN
C         V(1),     ,V(NSTATE-1),G. NOTE V(NSTATE) = 0.0 BUT WILL CONTAIN
C         G FOR CONVENIENCE.
C         BB WILL BE RHS OF SYSTEM.
C*********************************************************************
      DIMENSION AA(2500), BB(50), IDOLD( 50)
C*********************************************************************
C         DIMENSIONS OF AA AND BB MUST BE EXACTLY N BY N AND N FOR SIMQ
C         THESE ARE AUTOMATICALLY TAKEN CARE OF BELOW
C*********************************************************************
      ITER = 0
      DO 10 I = 1, NSTATE
      V(I) = 0.0
```

```
   10 IDOLD(I) = 0
      LCLP1 = 2 ** LCLASS
      NSM1 = NSTATE - 1
      IOUT = 6
   15 ITER = ITER + 1
      DO 1000 I = 1, NSTATE
      FMAX = 0.0
      LCHOIC  = 0
      DO 100 L = 1, LCLP1
      IM1 = I - 1
      IP1 = I + KPHASE
      IF(I.EQ.1) SUM = P(L,I,1) * V(IP1)
      IF(I.GT.(NSTATE-KPHASE)) GO TO 18
      IF(I.NE.1) SUM = P(L,I,1) * V(IP1) + P(L,I,2) * V(IM1)
      GO TO 19
   18 SUM = P(L,I,2) * V(IM1)
   19 IF(T(L,I).EQ.0.0) GO TO 100
      TOTAL = Q(L,I) + (1/T(L,I))* (SUM - V(I))
C*****************************************************************
C     ADMIT AS MANY CLASSES AS POSSIBLE IF A TIE
C*****************************************************************
      IF(TOTAL.GE.FMAX) GO TO 60
      GO TO 100
   60 FMAX = TOTAL
      LCHOIC  = L
  100 CONTINUE
      IF(LCHOIC.NE.0) GO TO 110
      WRITE(IOUT, 2000)
 2000 FORMAT(1H0,20X, 'LCHOIC  = 0')
      STOP
  110 ID(I) = LCHOIC
C*****************************************************************
C        SET UP AA AND BB
C*****************************************************************
      DO 120 J = 1,NSM1
      IXJ = I + (J-1) * NSTATE
      AA(IXJ) = 0.0
      IF(J.EQ.(I-1)) AA(IXJ) = -P(LCHOIC, I, 2)
  120 IF(J.EQ.(I+KPHASE)) AA(IXJ) = -P(LCHOIC, I, 1)
      IXI = I + (I - 1) * NSTATE


      AA(IXI) = 1.0
      IXN = I + (NSTATE - 1) * NSTATE
      AA(IXN) = T(LCHOIC,I)
      BB(I) = Q(LCHOIC, I)
 1000 CONTINUE
      CALL SIMQ(AA,BB,NSTATE,KS)
      IF(KS.EQ.0) GO TO 200
      WRITE(IOUT, 2100)
 2100 FORMAT(1H0, 20X, 'SINGULAR SYSTEM')
      STOP
  200 WRITE(IOUT, 2200) ITER
 2200 FORMAT(1H0, 20X, 'RESULTS FOR ITERATION ', I5, ' FOLLOW')
      WRITE(IOUT, 2300)
 2300 FORMAT(1H0, 20X, 'DECISION VECTOR')
      WRITE(IOUT, 2400) (ID(I), I = 1, NSTATE)
 2400 FORMAT(16I5)
      DO 1100 I = 1, NSTATE
 1100 V(I) = BB(I)
      G = V(NSTATE)
      V(NSTATE) = 0.0
      WRITE(IOUT, 2500) G
 2500 FORMAT(1H0, 20X, 'GAIN = ', F10.3)
      WRITE(IOUT, 2600)
 2600 FORMAT(1H0, 20X, 'V VECTOR')
```

```
      WRITE(IOUT, 2700) (V(I), I = 1, NSTATE)
 2700 FORMAT(8F10.4)
      DO 1200 I = 1, NSTATE
 1200 IF(ID(I).NE.IDOLD(I)) GO TO 1500
      GO TO 1700
 1500 CONTINUE
      DO 1600 I = 1, NSTATE
 1600 IDOLD(I) = ID(I)
      GO TO 15
 1700 WRITE(IOUT, 2300)
 2300 FORMAT(1H0, 20X, 'POLICY ITERATION TERMINATED')
C*******************************************************************
C          COMPUTE STEADY STATE PROBABILITIES
C*******************************************************************
      DO 3200 I = 1, NSTATE
      LCHOIC = ID(I)
      DO 3120 J = 1, NSM1
      IXJ = (I-1)*NSTATE + J
      AA(IXJ) = 0.0
      IF(J.EQ.(I-1)) AA(IXJ) =  P(LCHOIC, I, 2)
 3120 IF(J.EQ.(I+KPHASE)) AA(IXJ) =  P(LCHOIC, I, 1)
      IXI = I + (I-1) * NSTATE
      AA(IXI) = AA(IXI) - 1.0
      IXN = I * NSTATE
      AA(IXN) = 1.0
      BB(I) = 0.0
 3200 CONTINUE
      BB(NSTATE) = 1.0
      CALL SIMQ(AA,BB,NSTATE,KS)
      IF(KS.EQ.0) GO TO 3300
      WRITE(IOUT,2100)
      STOP
 3300 XDIV = 0.0
      DO 3340 I = 1, NSTATE
      LCHOIC = ID(I)
      V(I) = Q(LCHOIC,I)

      BB(I) = BB(I) * T(LCHOIC,I)
 3340 XDIV = XDIV + BB(I)
      XMULT = 0.0
      DO 3360 I = 1, NSTATE
      BB(I) = BB(I) / XDIV
 3360 XMULT = XMULT + BB(I) * V(I)
      WRITE(IOUT, 3400)
 3400 FORMAT(1H0, 20X, 'STEADY STATE PROBABILITIES')
      WRITE(IOUT,2700) (BB(I), I = 1, NSTATE)
      WRITE(IOUT,3600)
 3600 FORMAT(1H0,20X,'OPTIMAL Q VALUES')
      WRITE(IOUT,2700) (V(I), I = 1, NSTATE)
      WRITE(IOUT,3700)
 3700 FORMAT(1H 20X,' GAIN RATE CHECK')
      WRITE(IOUT,2700) XMULT
      RETURN
      END

      SUBROUTINE SIMQ(A,B,N,KS)
      DIMENSION A(1), B(1)
C*******************************************************************
C          PURPOSE
C          OBTAIN SOLUTION OF A SET OF SIMULTANEOUS LINEAR EQUATIONS,
C          AX=B
C          USAGE
C          CALL SIMQ(A,B,N,KS)
C          DESCRIPTION OF PARAMETERS
```

```
C        A - MATRIX OF COEFF, STORED COLUMNWISE. THESE ARE
C        DESTROYED IN COMPUTATION.  SIZE OF A MUST BE EXACTLY N BY N
C        B- VECTOR OF ORIGINAL RHS VALUES. THESE ARE REPLACED BY SOLN.
C        LENGTH OF B MUST BE EXACTLY N.
C        N - NUMBER OF EQUATIONS AND VARIABLES. N.GT.ONE.
C        KS - OUTPUT DIGIT
C        0 FOR NORMAL SOLUTION
C        1 FOR SINGULAR SET OF EQUATIONS
C        METHOD
C        METHOD OF SOLUTION IS BY ELIMINATION USING LARGEST PIVOTAL
C        DIVISOR. EACH STAGE OF ELIMINATION CONSISTS OF INTERCHANGING
C        ROWS WHEN NECESSARY TO AVOID DIVISION BY ZERO OR SMALL
C        ELEMENTS.
C        THE FORWARD SOLUTION TO OBTAIN VARIABLE N IS DONE IN
C        N STAGES.  THE BACK SOLUTION FOR THE OTHER VARIABLES IS
C        CALCULATED BY SUCCESSIVE SUBSTITUTIONS. FINAL SOLUTION
C        VALUES ARE DEVELOPED IN VECTOR B, WITH VARIABLE 1 IN B(1), ETC.
C        IF NO PIVOT CAN BE FOUND EXCEEDING A TOLERANCE OF 0.0,
C        THE MATRIX IS CONSIDERED SINGULAR AND KS IS SET TO 1. THIS
C        TOLERANCE CAN BE MODIFIED BY REPLACING THE FIRST STATEMENT.
C*************************************************************************
C


C*************************************************************************
C        FORWARD SOLUTION
C*************************************************************************
      TOL = 0.0
      KS = 0
      JJ = -N
      DO 65 J = 1, N
      JY = J + 1
      JJ = JJ + N + 1
      BIGA = 0
      IT = JJ - J
      DO 30 I = J, N
C*************************************************************************
C        SEARCH FOR MAXIMUM COEFFICIENT IN COLUMN
C*************************************************************************
      IJ = IT+I
      IF(ABS(BIGA)-ABS(A(IJ))) 20,30,30
   20 BIGA= A(IJ)
      IMAX = I
   30 CONTINUE
C*************************************************************************
C        TEST FOR PIVOT LESS THAN TOLERANCE (SINGULAR MATRIX)
C*************************************************************************
      IF(ABS(BIGA)-TOL) 35, 35, 40
   35 KS = 1
      RETURN
C*************************************************************************
C        INTERCHANGE ROWS IF NECESSARY
C*************************************************************************
   40 I1 = J + N * (J - 2)
      IT = IMAX - J
      DO 50 K = J, N
      I1= I1 + N
      I2 = I1 + IT
      SAVE = A(I1)
      A(I1) = A(I2)
      A(I2) = SAVE
C*************************************************************************
C        DIVIDE EQUATION BY LEADING COEFFICIENT
C*************************************************************************
   50 A(I1) = A(I1) / BIGA
      SAVE = B(IMAX)
```

```
              B(IMAX) = B(J)
              B(J) = SAVE/BIGA
        C*****************************************************************
        C          ELIMINATE NEXT VARIABLE
        C*****************************************************************
              IF (J-N) 55, 70, 55
         55   IQS = N * (J-1)
              DO 65 IX = JY, N
              IXJ = IQS + IX
              IT = J - IX
              DO 60 JX = JY, N
              IXJX = N *(JX-1) + IX
              JJX = IXJX + IT
         60   A(IXJX) = A(IXJX) - (A(IXJ)*A(JJX))
         65   B(IX) = B(IX) - (B(J) * A(IXJ))
        C*****************************************************************
        C          BACK SOLUTION
        C*****************************************************************


         70   NY = N - 1
              IT = N * N
              DO 80 J = 1, NY
              IA = IT-J
              IB = N - J
              IC = N
              DO 80 K = 1, J
              B(IB) = B(IB) - A(IA) * B(IC)
              IA = IA - N
         80   IC = IC - 1
              RETURN
              END
```

```
STATEMENTS EXECUTED=    6782

CORE USAGE       OBJECT CODE=   29088 BYTES,ARRAY AREA=   50020 BYTES,TOTAL AREA AVAILABLE=

DIAGNOSTICS          NUMBER OF ERRORS=       0, NUMBER OF WARNINGS=       0, NUMBER OF EXTENSIC

,COMPILE TIME=     0.37 SEC,EXECUTION TIME=     0.09 SEC,
```

```
        THIS QUEUE CONTROL PROBLEM HAS    1 CLASSES OF CUSTOMERS. THE MEAN SERVICE RATE OF

                  THE ERLANG    1 SERVER   IS       3.00

        INPUT VALUES OF THE PARAMETERS FOR EACH CLASS ARE GIVEN IN THE FOLLOWING TABLE

        CLASS            REWARD               COST               NLIM                FLAMDA

          1                5.00               2.00                 7                   1.00

                    P AND R MATRICES FOLLOW FOR ACTION    1

                                 TRANSITION PROBABILITIES
 0.0       0.0
 0.0       1.000000
 0.0       1.000000
 0.0       1.000000
 0.0       1.000000
 0.0       1.000000
 0.0       1.000000
 0.0       1.000000
```

```
                          EXPECTED REWARDS FOR TRANSITION
0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0

                   P AND R MATRICES FOLLOW FOR ACTION    2

                          TRANSITION PROBABILITIES
1.000000  0.0
0.250000  0.750000
0.250000  0.750000
0.250000  0.750000
0.250000  0.750000
0.250000  0.750000
0.250000  0.750000
0.0       0.0

                          EXPECTED REWARDS FOR TRANSITION
4.333333  3.666667  3.000000  2.333334  1.666667  1.000000  0.333334  0.0

                   ACTION    1 ADMITS CLASSES    0

                   ACTION    2 ADMITS CLASSES    1

                   R AND Q VECTORS FOLLOW FOR ACTION    1

                          R(I)
   0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0

                          Q(I)
   0.0       0.0       0.0       0.0       0.0       0.0       0.0       0.0

                   R AND Q VECTORS FOLLOW FOR ACTION    2

                          R(I)
   4.333     0.917     0.750     0.583     0.417     0.250     0.083     0.0

                          Q(I)

   4.333     3.667     3.000     2.333     1.667     1.000     0.333     0.0

                   RESULTS FOR ITERATION     1 FOLLOW

                   DECISION VECTOR
  2    2    2    2    2    2    2    1

                   GAIN =      4.001

                   V VECTOR
  7.9370    7.5046    6.9416    5.8535    4.5567    3.0007    1.3336    0.0

                   RESULTS FOR ITERATION     2 FOLLOW

                   DECISION VECTOR
  2    2    2    2    2    1    1    1

                   GAIN =      4.003

                   V VECTOR
  7.1767    6.9461    6.1905    5.2262    4.0027    2.6685    1.3342    0.0

                   RESULTS FOR ITERATION     3 FOLLOW

                   DECISION VECTOR
  2    2    2    2    2    1    1    1
```

```
                         GAIN =      4.003

                         V VECTOR
7.1767      6.8461      6.1905      5.2262      4.0027      2.6685      1.3342      0.0

                         POLICY ITERATION TERMINATED

                         STEADY STATE PROBABILITIES
0.6676      0.2225      0.0742      0.0247      0.0082      0.0027     -0.0000     -0.0000

                         OPTIMAL Q VALUES
4.3333      3.6667      3.0000      2.3333      1.6667      0.0         0.0         0.0

                           GAIN RATE CHECK
4.0027
```


```
ACCOUNT: 03764               MAXIMUM TIME (SEC):   20  NET CPU (SEC):          1
DATE: 08/29/79 IDENT:        ACTUAL TIME, INCLUDING 2.0 SEC SYSTEM TIME:     [illegible] $.17
USER: RUE 8                  LINES PRINTED:       780  CARDS PUNCHED:        0 [illegible] $.35
DESTINATION: AA              MAXIMUM RECORDS:    5000  TOTAL RECORDS:      780 [illegible] $.12
OS-21.8 HASP-2.T5G 370/3033  CARDS READ:          712                  ***** TOTAL
```

VITA

The author was born in Grafton, North Dakota, on April 5, 1947. He attended primary school in Cavalier, North Dakota, and Menomonie, Wisconsin, where he also attended secondary school. He entered the United States Air Force Academy in 1965 and received the B. S. degree in Mathematics in June 1969.

The author continued his education at the University of Florida where he received the M. S. degree in Systems Engineering in March 1975.

He is a member of Alpha Pi Mu, Phi Kappa Phi, Pi Mu Epsilon, and Sigma Xi.