ANALYSES OF OPTIMAL POLICIES

FOR

DYNAMIC INVENTORY AND MAINTENANCE SYSTEMS

By

DANNY I. CHO

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfilment of the Requirements

for the Degree of

Doctor of Philosophy

McMaster University

DYNAMIC INVENTORY AND MAINTENANCE SYSTEMS

DOCTOR OF PHILOSOPHY (1992)
(Management Science/Systems)

McMASTER UNIVERSITY
Hamilton, Ontario

TITLE: Analyses of Optimal Policies for
Dynamic Inventory and Maintenance Systems

AUTHOR: Danny I. Cho, B.A.Sc. (University of Toronto)
M.Eng. (University of Toronto)

SUPERVISOR: Dr. Mahmut Parlar

NUMBER OF PAGES: xii, 220

ABSTRACT

This thesis represents research in the combined areas of inventory and maintenance. It analyzes two independent inventory and maintenance problems under dynamic systems: (i) a production and maintenance problem and (ii) a repairable-item inventory problem. For each problem, the thesis develops a new control model and proposes a simultaneous determination of optimal inventory and maintenance policies.

The first part of the thesis examines a production process where the process performance deteriorates over time in the absence of preventive maintenance. First, it develops a new finite-time control model for optimal production and maintenance decisions by combining a dynamic maintenance model with a production control model. Second, it derives the necessary conditions for optimal production and maintenance controls using the maximum principle. Finally, it proposes two optimization algorithms for numerically solving the necessary conditions already derived.

The second part of the thesis considers the repairable-item inventory problem, which may be faced at each period by the inventory manager responsible for determining the optimum quantities to purchase new serviceable units, to repair and to junk returned repairable units in order to satisfy random demand for serviceable units. First, it proposes an inventory model for repairables, incorporating several

important features. The model includes a periodic review policy, random demand, lost sales for unsatisfied demand, set-up costs for ordering and repair, and a dynamic return process. Second, it employs a quite different solution methodology from what the previous research has used. The approach employed here is a 'Markov decision process (MDP)'. With this approach, the inventory problem is remodelled as a discrete-time Markov decision problem with two-dimensional state and three-dimensional decision spaces and then solved for finite-time planning horizon using the backward induction algorithm and for infinite-time planning horizon using the method of successive approximations. Finally, it introduces and utilizes two acceleration techniques, the error bounds approach and State Decomposition by Dimension (SDD), for speeding up the convergence of the computational methods described above.

ACKNOWLEDGEMENTS

I am greatly indebted to my thesis advisor, Professor Mahmut Parlar for the ideas, understanding, encouragement, and guidance that he has offered me throughout this research project. I would also like to express my sincere appreciation to the other members of my thesis committee, Professors P. L. Abad and C. Kwan, for their helpful comments in preparing this dissertation.

I also wish to thank my entire family, especially my parents, brothers and sisters-in-law, whom I really love and care about.

Finally, no words may express my deepest and sincerest thanks to my wife Soo for her support, patience, prayer, and love.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

LIST OF TABLES

LIST OF TABLES
(Continued)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS
(Continued)

CHAPTER 0 INTRODUCTION

0.1 Overview

Inventory and maintenance may be the two most important areas in business. Keeping an inventory (stock of goods) by either production or purchasing for future sale is very common in today's world. Inventory policy of a firm may include when and how much it should produce or replenish. On the other hand, maintaining stability (reliability) of a system (or facility) by preventive maintenance or by controlling the performance of the system is as important as inventory theory since systems in today's dynamic environment become more complicated and require new technologies. Maintenance policy may include such decisions as time and level of preventive maintenance on the system, time and size of repair of returned units to the system, etc. This thesis represents research in the combined areas of inventory and maintenance. It analyzes two independent inventory and maintenance problems in dynamic systems: (i) a production and maintenance problem and (ii) a repairable-item inventory problem. For each problem, the thesis develops a new dynamic control model and proposes a simultaneous determination of optimal inventory and maintenance policies.

Part I of the thesis examines a production process where the process performance deteriorates over time in the absence of preventive maintenance. First, it develops a new finite continuous time control model for optimal production and maintenance decisions by combining a

dynamic maintenance model with a production control model. Second, it derives the necessary conditions for production and maintenance controls to be optimal using Pontryagin's maximum principle (Pontryagin et al. [1962]). Finally, it proposes two optimization algorithms for numerically solving the necessary conditions already derived.

Part II of the thesis considers the repairable-item inventory problem, which may be faced at each period by the inventory manager responsible for determining the optimum quantities to purchase new serviceable units, to repair, and to junk returned repairable units in order to satisfy random demand for serviceable units. First, it proposes an inventory model for repairables by incorporating several important features. The model includes a periodic review policy, random demand process, 'lost sales' for unsatisfied demand, and set-up costs for purchasing and repair. In addition, the return process is assumed to be characterized by the current and previous demand processes and Bernoulli random variables representing the return status of serviceable units currently and previously demanded. Second, it employs a quite different solution approach to the inventory model from what the previous research has used, namely, a Markov decision process (MDP). With this approach, the repairable item inventory model is converted to a discrete-time Markov decision model with two-dimensional state and three-dimensional decision spaces and then solved for finite-time planning horizons using the backward induction algorithm. An infinite-time planning horizon problem is also considered and it is solved using the method of successive approximations. Finally, this thesis introduces and utilizes two acceleration techniques, the error bounds

approach and State Decomposition by Dimension (SDD), for speeding up the convergence of the computational methods described above.

0.2 Mathematical Techniques

The production and maintenance problem, remodelled as a finite-time dynamic control problem with two constrained state and two control variables, is solved for optimal production and maintenance decisions using optimal control theory. First, the necessary conditions for production and maintenance controls to be optimal are obtained using the maximum principle in terms of the current-value functions (Sethi and Thompson [1981]). Second, optimal production and maintenance controls are determined under a specific production cost function. One of two difficulties in determining the optimal controls is that the control problem contains a pure state inequality constraint, which makes the problem more complicated. A technique called 'the indirect adjoining approach with continuous adjoint function' (Hestenes [1966], Russak [1970], Hartl et al. [1987]) is used to deal with this complication. The other difficulty arises since maintenance control contains singular arcs. This requires a somewhat complicated analysis of the model regarding a determination of optimal singular controls. Finally, two optimization techniques, centralized and decentralized approaches, are used for solving the necessary conditions numerically because an analytic solution to the control problem is quite difficult to obtain. The centralized approach is based on the idea of a centralized decision making system where the production and maintenance decisions are planned by some central authority in a company. It treats the control problem

as a whole and attempts to find the optimal trajectories to the overall problem simultaneously. A modified version of 'the initial value shooting method' (Roberts and Shipman [1972]) is used to solve the necessary conditions. The decentralized approach, which is based on the 'Interaction Prediction Principle' (Singh [1980]), takes advantage of the serial structure of the system. It adopts the idea of a distributed authority system where each of the production and maintenance decisions is planned by the corresponding department of the company. With this approach, the original problem is divided into two subproblems and then the solution to the overall problem is obtained by solving the subproblems iteratively.

The repairable-item inventory problem is converted to a discrete-time Markov decision model with two-dimensional state and three-dimensional decision spaces. First, the return process is defined from the relationship with the demand process and the transition probabilities for each and every decision are obtained using tools from probability theory. Second, the decision space is reduced by the system and decision rules defined throughout the analysis of the system so that unnecessary decisions are eliminated before being considered. A large number of states means an even larger number of decisions (i.e., the number of decisions depends upon the number of states). The decision space reduction procedure is very important (especially for a system with a large number of states) because it avoids unnecessary computation time and effort. Third, the Markov decision model is solved for both finite and infinite horizons using the successive approximation method (or the backward induction algorithm). Finally, two acceleration

techniques are introduced for speeding up the convergence of the successive approximation method: the error bounds approach (Bertsekas [1976][1987], Heyman and Sobel [1982]) for infinite-horizon problems, and State Decomposition by Dimension (SDD) for both finite and infinite horizon problems. The error bounds approach determines an optimal stationary policy and the corresponding optimal cost in a much less number of iterations, whereas the SDD accelerates the computation by reducing the number of states so the amount of work required per iteration is also reduced.

0.3 Production and Maintenance Problem

Part I is devoted to the study of optimal production and maintenance decisions. Chapter 1 presents a brief survey on production control and maintenance control models for dynamic systems and outlines contributions of this thesis to the theory of optimal control for the areas of inventory and maintenance. Chapter 2 formulates a production and maintenance model for a production system experiencing age-dependent deterioration, and then analyzes the model for optimal production and maintenance controls using optimal control theory. Chapter 3 introduces two optimization techniques and their algorithms, that solve numerically the necessary conditions developed in Chapter 2. It also considers several numerical examples to demonstrate validity of the algorithms. Chapter 4 concludes Part I with discussions and findings resulting from the analysis of the model and with possible modification to the model for future research.

0.4 Repairable-Item Inventory Problem

Part II applies a Markov decision process (MDP) to analyze inventory systems for repairables. Chapter 5 begins with a general description of repairable-item inventory systems with returns followed by an extensive literature survey of the field. It also briefly outlines contributions of this thesis to the theory of inventory control for repairables. Chapter 6 defines the proposed repairable-item inventory problem and formulates a model for the problem. Chapter 7 is concerned with the Markov decision process approach to the proposed inventory model. A brief description of a Markov decision process is followed by the analysis of the Markov decision model. The analysis includes the following: description of the system spaces (i.e., stage, state and decision), development of system and decision rules, reduction in decision space, and derivation of transition probabilities. Description of computational methods and algorithms used for finite and infinite horizon Markov decision problems and several numerical examples are also included. Chapter 8 introduces two acceleration techniques for speeding up the convergence of the computational methods described in the previous chapter and discusses efficiency and effectiveness of these techniques. Finally, Chapter 9 concludes Part II with discussions and findings resulting from the analysis of the proposed model and with possible modification to the model for future research. It also compares the proposed model with similar models studied in the literature.

BIBLIOGRAPHY

Bertsekas, D.P., Dynamic Programming and Stochastic Control, Mathematics in Science and Engineering, Vol. 125, Academic Press, (1976).

---------, Dynamic Programming: Deterministic and Stochastic Models, Prentice-Hall, (1987).

Hartl, R.F., Sethi, S.P., and Vickson, R.G., "A Survey of The Maximum Principle for Optimal Control Problems with Pure State Constraints", Research Report, (1987).

Hestenes, M.R., Calculus of Variations and Optimal Control Theory, John Wiley & Sons Publishing, (1966).

Heyman, and Sobel, M.J., Stochastic Models in Operations Research (Vol.II): Stochastic Optimization, McGraw-Hill, (1982).

Pontryagin, L.S., Boltyanskii, V.G., Gamkrelidze, R.V., and Mishchenko, E., The Mathematical Theory of Optimal Processes, John Wiley & Sons Publishing, (1962).

Roberts, S.M., and Shipman, J.S., Two Point Boundary Value Problems: Shooting Methods, New York: American Elsevier, (1972).

Russak, I.B., "On Problems with Bounded State Variables", Journal of Optimization: Theory & Applications, 5, (1970), 114-157.

Sethi, S.P., and Thompson, G.L., Optimal Control Theory: Applications to Management Science, Boston: Martinus Nijhoff Publishing, (1981).

Singh, M.G., Dynamical Hierarchical Control, North-Holland Publishing Co., (1980).

PART I

OPTIMAL PRODUCTION AND MAINTENANCE DECISIONS

CHAPTER 1 INTRODUCTION

1.1 Literature Search

Many existing dynamic control models are concerned with optimal production decisions whereas others consider machine (or process) maintenance problems. Production control models for dynamic systems were discussed by Holt et al. [1960], Sprzeuzkouski [1967], Bensoussan at al. [1974], Baker and Peterson [1979], and Sethi and Thompson [1981] under various production and holding cost functions. The problem of concern in these models is to determine an optimal production rate which minimizes the total production and inventory holding costs over a finite planning time horizon. In the case of maintenance problems for dynamic systems, the objective is to determine the optimal rate of preventive maintenance over the lifetime of an asset. The dynamic maintenance problems were considered by Näslund [1966], Thompson [1968], Arora and Lele [1970], Sarma and Alam [1975], Gaimon and Thompson [1984][1989], and Tapiero [1986]. For a brief description of each of these models, the reader is referred to the survey papers by Pierskalla and Voelker [1976] and Cho and Parlar [1989][1991].

The control models discussed above consider production decisions separately from maintenance decisions. In other words, the two types of

decisions are assumed to be independent of each other. However, we are often faced with real world situations where production and maintenance decisions may be interdependent on each other. In Part I, we consider a production process whose performance deteriorates over time in the absence of preventive maintenance. If the output of the production process is proportional to its performance, it is necessary to keep the process performance at certain levels by applying proper maintenance over the finite planning horizon to avoid unnecessary waste (e.g. defectives). Here, our interest lies in the study of possible tradeoffs between the cost (level) of preventive maintenance applied to the process and the output of the process.

1.2 Contributions

This thesis extends the theory of optimal control for the areas of inventory and maintenance to the "macro" aspects of the production process. It incorporates a dynamic maintenance problem into a production control model and proposes a simultaneous determination of optimal production and maintenance policies. In particular, the thesis

(1) develops a new dynamic control model by combining a maintenance problem with a production control model.

(2) utilizes the maximum principle to derive the necessary conditions for optimal singular/nonsingular controls for the combined model.

(3) proposes two optimization techniques which can be used to solve the necessary conditions derived in (2) numerically.

CHAPTER 2 PRODUCTION AND MAINTENANCE MODEL FOR A SYSTEM EXPERIENCING AGE-DEPENDENT DETERIORATION

2.1 Statement of Problem

Consider a production process whose performance declines over time without any maintenance. The process performance can be measured in terms of the proportion of 'good' (non-defective) units of end items produced at a certain time. Preventive maintenance may be applied to the process to slow down the rate of decline of (or improve) the process performance. Assuming that the maintenance activity is permitted to occur as a continuous stream, we can formulate the problem as an optimal control theory model with an infinite number of states between two extreme states 0 and 1. Here state-1 indicates that the process yields only "good" units while state-0 refers to the process resulting in 100% defective units. Figure I-1 shows a graphical representation of the proposed production and maintenance model.

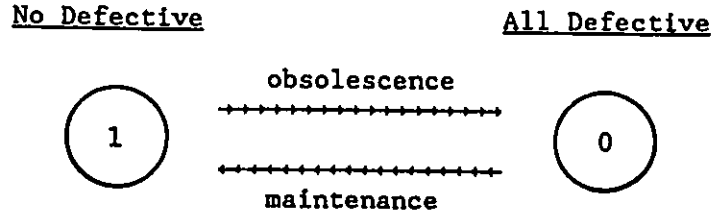


Figure I-1: A Graphical Representation of the Proposed Production and Maintenance Model

2.2 Notation and Assumptions

We define the following notation. Note that the decision variables are referred to as control variables.

x(t): Inventory level at time t; (state variable).

u(t): Scheduled production rate at time t; (control variable).

p(t): Proportion of 'good' units of the end items produced at time t representing the process performance ($0 \le p(t) \le 1$); (state variable).

$\alpha(t)$: Obsolescence rate (function) of the process performance in the absence of maintenance, where α is non-decreasing in time.

m(t): Preventive maintenance rate applied at time t to reduce the proportion of defective units produced; (control variable).

s(t): Demand rate at time t.

ρ : Constant continuous discount rate.

T : The planning horizon.

We make the following assumptions.

(1) The planning horizon is finite.

(2) There is a non-negative initial inventory (i.e., $x(0)=x_0 \ge 0$).

(3) The process yields all 'good' units at time zero (i.e., $p(0)=1$).

(4) All demand must be satisfied (i.e., $x(t) \ge 0$ for $0 < t \le T$).

(5) Negative production (i.e., disposals) is not allowed.

(6) The maintenance level is bounded by a lower limit of zero and an upper limit of M (i.e., $0 \le m(t) \le M$).

(7) Defective units have no monetary values.

(8) There is a salvage value for inventory unsold and a salvage value of the production process (e.g. facility) at time T.

Assumptions (1), (5) and (8) are self-explanatory. Assumptions (2) and (4) indicate that the system does not permit 'lost sales' due to insufficient inventory or 'backorders' of units for unsatisfied demand. It must produce in advance and keep enough inventory to meet the demand. Assumption (3) implies 'error-free' operation of the process at the beginning of the planning horizon. The upper bound of maintenance level in Assumption (6) can be explained by a budget constraint a firm faces. Assumption (7) implies that the process must keep its performance at a reasonable level to avoid unnecessary waste of production costs.

We consider the following cost and revenue functions.

(1) The production cost function $\phi[u(t)]$ is quadratic and increasing in $u(t)$. This situation occurs when some changeover costs (due to changes in work force, training, etc.) exist. The quadratic production cost may also approximate linear production cost with steeper slopes for higher levels of production.

(2) The inventory holding costs, which represent the costs associated with the storage of the inventory until it is depleted by the arrival of demand, are linear (i.e., $h[x(t)]=hx(t)$, where h is charged per unit of $x(t)$).

(3) The maintenance costs are linear in $m(t)$ (i.e., $c[m(t)]=cm(t)$, where c is charged on the level of preventive maintenance applied to the process at time t).

(4) Revenue generated at t is proportional to demand at t (i.e., $w[s(t)]=ws(t)$, where w is the revenue generated per unit sold).

(5) Salvage value of inventory unsold at T is earned (i.e., a[x(T)] =ax(T), where a is the salvage value per unit item).

(6) The salvage value of the production facility is proportional to the process performance at T (i.e., υ[p(T)]=bp(T), where b is the maximum value). The linear salvage function may not be a good assumption, especially when the process performance is very low at T, because a facility with p(T)=0.25 is usually not worth b/4. However, the model implicitly avoids such a case by keeping the process performance at quite reasonable levels to minimize waste.

Those who analyzed dynamic production control problems under quadratic production cost and linear holding cost functions include Sprzeuzkouski [1967], Bensoussan at al. [1974], and Sethi and Thompson [1981]. The preventive maintenance rate with lower and upper limits and linear maintenance costs were also discussed in Thompson [1968].

2.3 Model Formulation

To obtain state equations which serve to define the change in the system over time, we calculate the following.

The number of good units produced in $(t,t+\Delta t)$ = $p(t)u(t)\Delta t$.

The inventory level at $t+\Delta t$ can be stated as:

$x(t+\Delta t)$ = inventory at t + good units produced in $(t,t+\Delta t)$
- demand for the item in $(t,t+\Delta t)$

$= x(t) + p(t)u(t)\Delta t - s(t)\Delta t.$

From the above equation, we get

$$[x(t+\Delta t)-x(t)]/\Delta t = p(t)u(t) - s(t).$$

By taking the limit as $\Delta t \to 0$, we obtain the state equation depicting the change in the level of inventory over time as

$$\dot{x}(t) = p(t)u(t) - s(t). \qquad (2\text{-}1)$$

The proportion of good units produced at $t+\Delta t$ is:

$p(t+\Delta t)$ = proportion of good units produced at t - decrease in proportion of good units produced in $(t, t+\Delta t)$ due to obsolescence of the process + increase in proportion of good units produced in $(t, t+\Delta t)$ due to maintenance applied at t to reduce the proportion of defective units produced

$$= p(t) - \alpha(t)p(t)\Delta t + [1 - p(t)]\, m(t)\, \Delta t.$$

From the above equation, we get

$$[p(t+\Delta t)-p(t)]/\Delta t = - \alpha(t)p(t) + m(t) - m(t)p(t).$$

By taking the limit as $\Delta t \to 0$, we obtain the second state equation depicting the change in the proportion of good units over time as

$$\dot{p}(t) = - [\alpha(t) + m(t)]\, p(t) + m(t). \qquad (2\text{-}2)$$

Equation (2-2) represents the relationship between the firm's budget for preventive maintenance on its production process and process performance as measured by the percentage of units of the end items that are

acceptable. It is assumed that in the absence of preventive maintenance the performance of the production process deteriorates over time.

Our objective is to find optimal values of control variables which maximize the total discounted profit over the finite planning horizon. Thus an appropriate form of the objective functional for the problem is:

J = total discounted profit (i.e., revenue from sales - costs of production, holding and maintenance) + the discounted salvage value of inventory + the discounted salvage value of the facility

$$= \int_0^T \{w[s(t)] - h[x(t)] - \phi[u(t)] - c[m(t)]\}\, e^{-\rho t} dt + a[x(T)]\, e^{-\rho T} + b[p(T)]\, e^{-\rho T}. \qquad (2\text{-}3)$$

With the assumptions made above, the production/maintenance problem can be modelled as a dynamic control problem with two constrained state and two constrained control variables which can be stated as follows:

$$\max_{u,m} J = \int_0^T \{ws(t) - hx(t) - \phi[u(t)] - cm(t)\}\, e^{-\rho t} dt + ax(T)\, e^{-\rho T} + bp(T)\, e^{-\rho T} \qquad (2\text{-}4a)$$

subject to

$$\dot{x}(t) = p(t)u(t) - s(t), \qquad x(0)=x_0 \geq 0 \qquad (2\text{-}4b)$$

$$\dot{p}(t) = -[\alpha(t) + m(t)]\, p(t) + m(t), \qquad p(0)=1 \qquad (2\text{-}4c)$$

$$u(t) \geq 0, \qquad 0 \leq t \leq T \tag{2-4d}$$

$$0 \leq m(t) \leq M, \qquad 0 \leq t \leq T \tag{2-4e}$$

$$x(t) \geq 0, \qquad 0 \leq t \leq T \tag{2-4f}$$

2.4 Analysis of Optimal Controls

2.4.1 Solution Approach

We will use Pontryagin's maximum principle in terms of the current-value functions (Sethi and Thompson [1981]) to solve the above production and maintenance (PM) control problem. The maximum principle gives necessary conditions for controls to be optimal. What the maximum principle does is that it decouples the dynamic maximization problem into a series of static maximization problems at each instant time. The PM control problem mentioned above has two difficulties to be overcome. One difficulty is that as one may notice the problem contains a pure state inequality constraint (2-4f). Although optimal control problems with pure state inequality constraints often appear in the areas of management science and economics, these constraints would make the problems more complicated. Several techniques used to deal with such a situation have been suggested by Pontryagin et al. [1962], Hestenes [1966], Russak [1970], Bensoussan et al. [1974], and Hartl et al. [1987]. To deal with the difficulty in the PM control problem, we will use 'the indirect adjoining approach with continuous adjoint function' (Hestenes [1966], Russak [1970], Hartl et al. [1987]). With this technique, adjoint functions will still be continuous but the complementary slackness condition will not hold for the corresponding multiplier while it holds for its derivative. The other difficulty

arises in the PM control problem because the maintenance control m is partially singular (i.e., m is of the 'bang-bang' type with singular). This is an added complication, which requires a determination of singular arcs as well as of non-singular arcs.

2.4.2 Derivation of Necessary Conditions

The current-value Hamiltonian is

$$H = ws - hx - \phi[u(t)] - cm + \lambda_1(pu-s) + \lambda_2[m-(\alpha+m)p] + \eta(pu-s)$$
$$= ws - hx - \phi[u(t)] - cm + (\lambda_1+\eta)(pu-s) + \lambda_2[m-(\alpha+m)p], \qquad (2\text{-}5)$$

where λ_1 and λ_2 are the current-value adjoint variables for the constraints (2-4b) and (2-4c), respectively, and η is the current-value multiplier associated with $\dot{x}$ so that x does not become negative. Note that η is a constant when $x>0$ and is a non-increasing function in time, otherwise (Hartl et. al [1987]). The first four terms in H represent the direct contribution to J at time instant t whereas the rest of the terms in H are considered as the indirect contribution to J.

The Hamiltonian maximizing condition (Sethi and Thompson [1981]) for the optimal production rate u is:

$$H[x,p,u,m,t] \geq H[x,p,\underline{u},m,t] \qquad (2\text{-}6)$$

for all admissible controls $\underline{u}$ such that $\underline{u} \geq 0$ for $t\epsilon[0,T]$.

The Hamiltonian maximizing condition for the optimal maintenance level m is:

$$H[x,p,u,m,t] \geq H[x,p,u,\underline{m},t] \qquad (2\text{-}7)$$

for all admissible controls $\underline{m}$ such that $0 \leq \underline{m} \leq M$ for $t\epsilon[0,T]$.

To obtain adjoint equations and the multipliers associated with the constraints, we form the current-value Lagrangian:

$$\begin{aligned} L &= H + \mu u + \beta_1 m + \beta_2 (M-m) \\ &= ws - hx - \phi[u(t)] - cm + \lambda_1(pu-s) + \lambda_2[m-(\alpha+m)p] \\ &\quad + \eta(pu-s) + \mu u + \beta_1 m + \beta_2(M-m) \end{aligned} \tag{2-8}$$

where μ is the current-value multiplier for the constraint $u \geq 0$, and β_1 and β_2 are the current-value multipliers for the constraints $m \geq 0$ and $m \leq M$, respectively.

The multipliers must also satisfy the complementary slackness conditions

$$\mu \geq 0, \quad \mu u = 0 \tag{2-9a}$$

$$\beta_1 \geq 0, \quad \beta_1 m = 0 \tag{2-9b}$$

$$\beta_2 \geq 0, \quad \beta_2(M-m) = 0 \tag{2-9c}$$

$$\eta \geq 0, \quad \dot{\eta} \leq 0, \quad \dot{\eta} x = 0. \tag{2-9d}$$

Note that the complementary slackness condition does not hold for η but it holds for $\dot{\eta}$ (i.e., $\dot{\eta}x=0$ rather than $\eta x=0$).

Furthermore, the optimal trajectories must satisfy

$$L_u = \partial L/\partial u = -\phi_u + \lambda_1 p + \mu + \eta p = 0 \tag{2-10a}$$

and

$$\begin{aligned} L_m = \partial L/\partial m &= -c + \lambda_2 - \lambda_2 p + \beta_1 - \beta_2 \\ &= \lambda_2(1-p) - c + \beta_1 - \beta_2 = 0. \end{aligned} \tag{2-10b}$$

From the Lagrangian, we can get the current-value adjoint equations. The current-value adjoint equation for λ_1 is:

$$\dot{\lambda}_1 = \rho\lambda_1 - L_x = \rho\lambda_1 - [-h] = \rho\lambda_1 + h, \quad \text{and} \quad \lambda_1(T) = a.$$

or

$$\lambda_1(t) = -h/\rho + (a + h/\rho)\, e^{\rho(t-T)}. \tag{2-11}$$

Note that $\lambda_1(t)$ is the marginal benefit (or contribution to J) resulting from a small positive change in inventory level, x(t), at time t.

Also, we may get the marginal benefit resulting from a small change in p(t), the process performance at time t by solving

$$\dot{\lambda}_2 = \rho\lambda_2 - L_p = (\rho+\alpha+m)\lambda_2 - (\lambda_1+\eta)u, \quad \text{and} \quad \lambda_2(T)=b. \tag{2-12}$$

However, this ordinary linear differential equation cannot be explicitly solved for λ_2 since λ_2, u, and p are dependent on one another. Furthermore, the optimal trajectory of m requires the knowledge of λ_2 trajectory, which also depends on the trajectory of m. Thus an analytic (or closed-form) solution to this problem is difficult to obtain. We will later introduce computational methods that enable us to solve the problem numerically.

We now summarize the set of the Hamiltonian maximizing conditions and differential and algebraic equalities and inequalities previously obtained as the necessary conditions for u and m to be optimal controls.

$$\dot{x} = pu - s, \qquad x(0)=x_0 \tag{2-13a}$$

$$\dot{p} = -(\alpha+m)\, p + m, \qquad p(0) = 1 \tag{2-13b}$$

$$H[x, p, u, m, t] \geq H[x, p, \underline{u}, m, t] \tag{2-13c}$$

$$H[x, p, u, m, t] \geq H[x, p, u, \underset{\sim}{m}, t] \tag{2-13d}$$

$$\mu \geq 0, \quad \mu u = 0 \tag{2-13e}$$

$$\beta_1 \geq 0, \quad \beta_1 m = 0 \tag{2-13f}$$

$$\beta_2 \geq 0, \quad \beta_2 (M-m) = 0 \tag{2-13g}$$

$$\eta \geq 0, \quad \dot{\eta} \leq 0, \quad \dot{\eta} x = 0 \tag{2-13h}$$

$$\lambda_1 = -h/\rho + (a + h/\rho)\, e^{\rho(t-T)} \tag{2-13i}$$

$$\dot{\lambda}_2 = (\rho+\alpha+m)\lambda_2 - (\lambda_1+\eta)u, \quad \lambda_2(T) = b \tag{2-13j}$$

$$L_u = -\phi_u + (\lambda_1+\eta)p + \mu = 0 \tag{2-13k}$$

$$L_m = \lambda_2(1-p) - c + \beta_1 - \beta_2 = 0 \tag{2-13ℓ}$$

2.4.3 Determination of Optimal Controls

In Section 2.2, the production cost function $\phi[u(t)]$ was assumed to be quadratic and increasing in u (i.e., $\partial\phi/\partial u > 0$ and $\partial^2\phi/\partial u^2 > 0$). For further analysis of the problem, we now define specific production cost functions that are commonly used.

(1) $\phi[u(t)] = ru^2$, where $r>0$. This function approximates linear production cost functions with steeper slopes at high levels of production.

(2) $\phi[u(t)] = ru^2+qu+d$, where $r,q>0$. This function is more general than the one in (1). One may choose a desirable shape of the curve by selecting proper values of the parameters.

(3) $\phi[u(t)] = qu+r(u-\hat{u})^2 = ru^2+(q-2r\hat{u})u+r\hat{u}^2$, where $q,r>0$. Here, in addition to the linear production cost, a desirable level of

production $\hat{u}$ is imposed so that any deviation from $\hat{u}$ creates penalty costs.

One may use any of the above production cost functions. As far as optimization is concerned, there is little differences among them in the degree of complexity. We will use the second type of quadratic production cost function (i.e., $\phi[u(t)]=ru^2+qu+d$). We recall the current-value Hamiltonian:

$$H = ws - hx - (ru^2+qu+d) - cm + (\lambda_1+\eta)(pu-s) + \lambda_2[m-(\alpha+m)p].$$

To determine the optimal production rate u^*,

$$\max_u H: \quad H_u = \partial H/\partial u = -2ru - q + (\lambda_1+\eta)p = 0. \tag{2-14}$$

$$\text{Thus, } u^* = \begin{cases} 0 & \text{if } u' < 0 \\ u' & \text{if } u' \geq 0 \end{cases} \quad \text{whenever } x>0, \tag{2-15}$$

where $u'= [(\lambda_1+\eta)p-q]/2r$. If $x=0$, we must impose the constraint $\dot{x}\geq 0$ (i.e., $pu-s\geq 0$ or $u\geq s/p\geq 0$) so that x does not become negative. In such a case, u is constrained by a lower bound of s/p so that all demand can be satisfied. If $u'\geq s/p$, then $u^* = u'$ from $H_u = 0$; otherwise, we must produce s/p units to meet the demand.

$$\text{Thus, } u^* = \begin{cases} s/p & \text{if } u' < s/p \\ u' & \text{if } u' \geq s/p \end{cases} \quad \text{whenever } x=0, \tag{2-16}$$

where $u'= [(\lambda_1+\eta)p-q]/2r$.

To determine the optimal maintenance level m^*, we note that

$$\max_m H: \quad H_m = \partial H/\partial m = -c + \lambda_2 - \lambda_2 p = \lambda_2(1-p) - c. \tag{2-17}$$

Since the Hamiltonian is linear in m, the maintenance policy is of the 'bang-bang' type with singular control,

$$m^* = \text{bang } [0, M; \lambda_2(1-p)-c] = \begin{cases} 0 & \text{if } \lambda_2(1-p) < c \\ \text{singular} & \text{if } \lambda_2(1-p) = c. \\ M & \text{if } \lambda_2(1-p) > c \end{cases} \tag{2-18}$$

2.5 Analysis of Singular Controls

2.5.1 Existence of Singular Arcs

Singular extremals exist when the Hamiltonian H is linear in a control variable. Such arcs satisfying the derivative of H with respect to the control equals zero are called 'singular arcs'. Johnson and Gibson [1963] argue, however, that singular control will not necessarily constitute part of the optimal control. In the PM control problem, H is linear in the maintenance level m. Thus, singular arcs satisfying $H_m = \lambda_2(1-p)-c = 0$ occur on which H_{mm} is singular (i.e., $H_{mm} = 0$). We can see in the problem that in the beginning of the planning horizon the optimal control is non-singular (i.e., $m(0)=0$) because $H_m \neq 0$ (i.e., $\lambda_2(1-p) < c$, since $p(0)=1$) at $t=0$. The control function is partially singular because $H_m=0$ holds for some time intervals whose total length is less than the planning horizon. $H_m=0$ implies that coefficient of the linear term vanishes along a singular arc, and thus the control is not

determined by the state and/or adjoint variables. The control is rather determined by the requirement that the coefficient of the linear term remains zero on the singular arc (and thus the time derivatives of H_m must be zero). Therefore, along the singular arcs the following conditions must be satisfied.

$$H_m = H_m^{(1)} = H_m^{(2)} = H_m^{(3)} = \ldots = 0, \tag{2-19}$$

where $H_m^{(k)}$ is the k^{th} time derivative of H_m.

2.5.2 Determination of Singular Controls

In this section, we will characterize the optimal values of singular controls, which satisfy (2-19).

On singular arcs,

$$H_m = H_m^{(1)} = H_m^{(2)} = \ldots = 0.$$

Recall from (2-13b) and (2-13j),

$$\dot{p} = -(\alpha+m)p + m = (1-p)m - \alpha p$$

and

$$\dot{\lambda}_2 = (\rho+\alpha+m)\lambda_2 - (\lambda_1+\eta)u.$$

We have

$$H_m = \lambda_2(1-p) - c \tag{2-20}$$

and

$$H_m^{(1)} = \dot{\lambda}_2(1-p) - \lambda_2\dot{p} = [\alpha+\rho(1-p)]\lambda_2 - (\lambda_1+\eta)u(1-p). \tag{2-21}$$

Let $\psi = (\lambda_1+\eta)u.$ (2-22)

Then, $H_m^{(1)} = [\alpha+\rho(1-p)]\lambda_2 - \psi(1-p).$ (2-23)

Since $H_m = H_m^{(1)} = 0$ on the singular arc, we have from (2-20) and (2-23)

$$\lambda_2 = \frac{c}{(1-p)} = \frac{\psi(1-p)}{\alpha+\rho(1-p)}. \tag{2-24}$$

After solving (2-24) for p, we obtain

$$p = 1 - \frac{1}{2\psi}\left(\rho c + \sqrt{\rho^2c^2+4\alpha c\psi}\right) \qquad \text{where} \quad \psi>0. \tag{2-25}$$

By differentiating $H_m^{(1)}$ with respec · to time once more, we get

$$H_m^{(2)} = (\dot{\alpha}-\rho\dot{p})\lambda_2 + [\alpha+\rho(1-p)]\dot{\lambda}_2 - \dot{\psi}(1-p) + \psi\dot{p} \tag{2-26}$$

$$= [\alpha\lambda_2+\psi(1-p)]m + [\dot{\alpha}+\alpha^2+2\alpha\rho+\rho^2(1-p)]\lambda_2 - \alpha\psi(1+p) - (1-p)(\dot{\psi}+\rho\psi).$$

On singular arcs, $H_m^{(2)} = 0$. After simplifying the above equation and solving it for m, we get

$$m_s = \frac{1}{\alpha\lambda_2+\psi(1-p)}\{\alpha\psi(1+p)+(1-p)(\dot{\psi}+\rho\psi)-[\dot{\alpha}+\alpha^2+2\alpha\rho+\rho^2(1-p)]\lambda_2\}, \tag{2-27}$$

where $\dot{\psi}$ may contain m. Note that ψ is defined as follows.

$$\psi = (\lambda_1+\eta)u = \begin{cases} 0 & \text{if} \quad u = 0 \\ (\lambda_1+\eta)[(\lambda_1+\eta)p-q]/2r & \text{if} \quad u = [(\lambda_1+\eta)p-q)]/2r. \\ (\lambda_1+\eta)s/p & \text{if} \quad u = s/p \end{cases}$$

Thus, the singular trajectories are characterized by Equations (2-24), (2-25), and (2-27). We now determine singular controls for each of the three different values of u. Note that for all three cases the generalized Legendre Clebsch condition (Kelley et al. [1967]), i.e., the necessary condition for singular subarcs,

$$(-1)^n \frac{\partial}{\partial m}[H_m^{(2n)}] \le 0,$$

is satisfied, where n=1 is the order of the singular problem.

<Case 1>: $u = 0$.

$$\psi = 0, \quad \dot{\psi} = 0.$$

$$m_s = -\frac{1}{\alpha}\left[\dot{\alpha} + \alpha^2 + 2\rho\alpha + \rho^2(1-p)\right]. \tag{2-28}$$

In this case, m_s is always negative since α, ρ and 1-p are nonnegative and α is non-decreasing in time. Thus, $m_s^* = 0$ because the control m is restricted by a lower bound of zero.

<Case 2>: $u = [(\lambda_1+\eta)p-q]/2r$.

$\dot{\eta}=0$ (thus η is constant),

$$\psi = (\lambda_1+\eta)[(\lambda_1+\eta)p-q]/2r,$$

$$\dot{\psi} = \frac{1}{2r}(\lambda_1+\eta)^2(1-p)m + \frac{1}{2r}[2(\lambda_1+\eta)\dot{\lambda}_1 p - \dot{\lambda}_1 q - \alpha(\lambda_1+\eta)^2 p].$$

Thus, $$m_s^* = \left(\frac{S2}{2r} - S3\right) / S1 \tag{2-29}$$

where

$$S1 = [2\alpha+\rho(1-p)]\lambda_2 - \frac{1}{2r}(\lambda_1+\eta)^2(1-p)^2$$

$$S2 = \dot{\lambda}_1(1-p)[2(\lambda_1+\eta)p-q]+(\lambda_1+\eta)\{(\lambda_1+\eta)p[\rho(1-p)+2\alpha p]-q[\rho(1-p)+\alpha(1+p)]\}$$

$$S3 = [\dot{\alpha}+\alpha^2+2\rho\alpha+\rho^2(1-p)]\lambda_2.$$

<Case 3>: $u = s/p$.

$$\eta = \frac{2rs+qp}{p^2} - \lambda_1.$$

$$\psi = \frac{2rs^2+qsp}{p^3}.$$

$$\dot{\psi} = \frac{1}{p^3}[(4rs+qp)\dot{s}+2\alpha(qsp+3rs^2)] - \frac{2}{p^4}(1-p)(qsp+3rs^2)m.$$

Thus, $$m_s^* = \left(\frac{S5}{p^3} - S3 \right) / S4 \qquad (2\text{-}30)$$

where

$S3$ = defined above

$$S4 = \alpha\lambda_2 + \frac{1}{p^4}(1-p)[qsp(2-p)+2rs^2(3-2p)]$$

$$S5 = \alpha[qsp(3-p)+4rs^2(2-p)] + \rho(1-p)(2rs^2+qsp) + (1-p)(4rs+qp)\dot{s}.$$

CHAPTER 3 OPTIMIZATION TECHNIQUES

In the previous chapter, we have obtained necessary conditions for controls u and m to be optimal. Because an analytic solution to the proposed control problem is difficult to obtain, we illustrate two types of optimization techniques, the centralized and decentralized approaches, which can be used to solve the necessary conditions numerically. The centralized approach is based on the idea of a centralized decision making system where production and maintenance decisions are planned by some central authority in a company. These decisions are made simultaneously and each of the production and maintenance department has no control over the decisions. On the other hand, the decentralized approach adopts the idea of a distributed authority system where the production and maintenance decisions are planned in a decentralized mode. With this approach, each of the two departments makes its own decision regardless of the decision by the other department. These decisions are later modified gradually in order to cope with the best interest of the company.

3.1 Centralized Approach and Its Algorithm

The centralized approach considers the dynamic maximizing problem as a whole and attempts to find the optimal production and maintenance trajectories simultaneously. This situation occurs in a real life where production and maintenance decisions are planned by some

central authority in a company. Table I-1 shows the necessary conditions for optimal singular and non-singular controls which were obtained previously.

One should notice that η is constant whenever x>0 (i.e, $\dot{\eta}x = 0$). In such a case, the value of η (denoted by η_{const}) must be determined from the global characteristics of the problem. A modified version of the initial value shooting method (Roberts and Shipman [1972]) is used to solve the necessary conditions for the two-point boundary-value problem and then to obtain the optimal trajectories. With this iterative method, $x(0)=x_0$, $p(0)=1$, a guessed value for η_{const}, and a guessed value for the unknown adjoint variable $\lambda_2(0)$ are used as initial conditions to solve the two state differential equations and the adjoint differential equation $\dot{\lambda}_2$ forward in time, determining the controls u and m, and η from the other necessary conditions. If the computed terminal value for the unknown adjoint variable is close to the given terminal value and the guessed value for η_{const} yields the maximum J, the problem has been solved; otherwise, the guessed value for the adjoint variable must be revised (in such a way that the difference between the known and unknown terminal values is minimized) and/or η_{const} must be updated. The following iterative steps are used in the centralized approach.

Algorithm I-1: Centralized Approach

(a) Guess η_{const}.

(b) Guess $\lambda_2(0)$.

(c) Let $x(0)=x_0$, $p(0)=1$, η_{const}, and $\lambda_2(0)$ be initial conditions.

(d) Check the singularity of m (i.e., $H_m = \lambda_2(1-p)-c = 0$).

Non-Singular Arcs	Singular Arcs
$\dot{x} = pu - s$	$\dot{x} = pu - s$
$\dot{p} = m - (\alpha+m)p$	$p = 1 - \frac{\rho c + \sqrt{\rho^2 c^2 + 4\alpha c\psi}}{2\psi}$
$u = \begin{cases} 0 & \text{if } u'<0 \ \& \ x>0 \\ u' & \text{if } u'\geq 0 \ \& \ x>0 \\ s/p & \text{if } u'<s/p \ \& \ x=0 \\ u' & \text{if } u'\geq s/p \ \& \ x=0 \end{cases}$ where $u' = [(\lambda_1+\eta)p - q]/2r$	SAME
$m = \begin{cases} 0 & \text{if } \lambda_2(1-p) < c \\ m_s & \text{if } \lambda_2(1-p) = c, \\ M & \text{if } \lambda_2(1-p) > c \end{cases}$ where $m_s = \frac{1}{\alpha\lambda_2+\psi(1-p)} \{\alpha\psi(1+p)+(1-p)(\dot{\psi}+\rho\psi) - [\dot{\alpha}+\alpha^2+2\rho\alpha+\rho^2(1-p)]\lambda_2\}$	
$\mu \geq 0, \quad \mu u = 0$	
$\beta_1 \geq 0, \quad \beta_1 m = 0$	
$\beta_2 \geq 0, \quad \beta_2(M-m) = 0$	SAME
$\eta \geq 0, \quad \dot{\eta}\leq 0, \quad \dot{\eta}x = 0$	
$\lambda_1 = -h/\rho + (a+h/\rho)e^{\rho(t-T)}$	
$L_u = -2ru + (\lambda_1+\eta)p - q + \mu = 0$	SAME
$L_m = -c + \lambda_2(1-p) + \beta_1 - \beta_2 = 0$	
$\dot{\lambda}_2 = (\rho+\alpha+m)\lambda_2 - (\lambda_1+\eta)u, \quad \lambda_2(T) = b$	$\lambda_2 = \frac{c}{1-p} = \frac{\psi(1-p)}{\alpha+\rho(1-p)}$

Table I-1: Necessary Conditions for Optimal Singular & Nonsingular Arcs

(e) Determine the controls u and m using (2-15) & (2-16), and (2-18), (2-28), (2-29) & (2-30), respectively. Note that p, λ_2 and m on singular arcs are different than those on non-singular arcs.

(f) Determine the multiplier η. If x(t)>0, $\eta = \eta_{const}$. Otherwise, η is obtained from (2-13k).

(g) Solve the two state equations and $\dot{\lambda}_2$ forward in time using (2-13a), (2-13b), and (2-13j). If t≠T, let t=t+Δt and go to (d).

(h) If $\lambda_2(T)$=b at the final time T, go to (i). Otherwise, modify the initial guesses for $\lambda_2(0)$ using a simple search technique and go to Step (c).

(i) If η_{const} yields the maximum J, we have solved the problem. Otherwise, update η_{const} using a search technique and go to (b).

3.2 Decentralized Approach and Its Algorithm

In the decentralized approach, the original problem is divided into two independent subproblems (i.e., maintenance and production subproblems) via another control variable, and then solution for the overall problem is obtained by solving the subproblems iteratively. The new control variable decouples the connection between the two subproblems so that each subproblem can be solved separately with some measure of independence. This approach can be used in a real life situation where production and maintenance decisions are planned in a decentralized mode (i.e., a serial system where each decision is made by the corresponding department of a company).

The production and maintenance system being studied is a serial system, where the process performance (maintenance decision) is an input to the level of scheduled production (production decision). Since the interdependencies between the production and maintenance subproblems are embodied in the term pu in the state equation $\dot{x} = pu - s$, we introduce another control variable z.

Let $p = z$. Then, the state equation becomes $\dot{x} = zu - s$.
The new current-value Hamiltonian is then

$$H = ws - hx - (ru^2+qu+d) - cm + (\lambda_1+\eta)(zu-s) + \lambda_2[m-(\alpha+m)p] - \theta z + \theta p$$
$$= \{ ws - hx - (ru^2+qu+d) - \theta z + (\lambda_1+\eta)(zu-s) \}$$
$$+ \{ \theta p - cm + \lambda_2[(1-p)m-\alpha p] \}. \qquad (3\text{-}1)$$

Also, the new current-value Lagrangian is

$$L = ws - hx - (ru^2+qu+d) - cm + (\lambda_1+\eta)(zu-s)$$
$$+ \lambda_2[m-(\alpha+m)p] + \mu u + \beta_1 m + \beta_2(M-m) + \theta(p-z), \qquad (3\text{-}2)$$

where θ is the Lagrangian multiplier associated with p=z.

Then, we have the following additional necessary conditions.

$$\partial L/\partial z = \lambda_1 u + \eta u - \theta = (\lambda_1+\eta)u - \theta = 0 \qquad (3\text{-}3a)$$

$$\partial L/\partial u = -2ru - q + (\lambda_1+\eta)z + \mu = 0 \qquad (3\text{-}3b)$$

and $$\partial L/\partial \theta = p - z = 0. \qquad (3\text{-}3c)$$

From the above equations, we have

$$u = [(\lambda_1+\eta)z-q]/2r \doteq [(\lambda_1+\eta)p-q]/2c \qquad (3\text{-}4)$$

and $$\theta = (\lambda_1+\eta)u = (\lambda_1+\eta)[(\lambda_1+\eta)z-q]/2r. \qquad (3\text{-}5)$$

We introduce the two subproblems.

The maintenance (MAIN) subproblem is:

$$\max_{m} \; J_1 = \int_0^T (\theta p - cm)\, e^{-\rho t}\, dt + bp(T)e^{-\rho T} \tag{3-6a}$$

s.t.

$$\dot{p} = (1-p)m - \alpha p, \qquad p(0)=1 \tag{3-6b}$$

$$0 \le m \le M \tag{3-6c}$$

The Hamiltonian for the (MAIN) subproblem is

$$H_1 = \theta p - cm + \lambda_2[m-(\alpha+m)p]. \tag{3-7}$$

$$\max_{m} H_1 : \; H_{1_m} = \partial H_1/\partial m = \lambda_2(1-p) - c = 0.$$

$$\therefore \; m^* = \text{bang}\,[0, M; \lambda_2(1-p)-c] = \begin{cases} 0 & \text{if } \lambda_2(1-p) < c \\ m_s & \text{if } \lambda_2(1-p) = c \\ M & \text{if } \lambda_2(1-p) > c \end{cases} \tag{3-8}$$

where $m_s = (S7-S8)/S6$,

$$\begin{cases} S6 = \alpha\lambda_2+\theta(1-p) \\ S7 = \alpha\theta(1+p)+\rho(1-p)(\dot{\theta}+\rho\theta), \\ S8 = [\dot{\alpha}+\alpha^2+2\rho\alpha+\rho^2(1-p)]\lambda_2 \end{cases}$$

$$p_s = 1 - \frac{\rho c + \sqrt{\rho^2c^2+4\alpha c\theta}}{2\theta},$$

and $$\lambda_2^s = \frac{c}{1-p} = \frac{\theta(1-p)}{\alpha+\rho(1-p)}.$$

The production (PROD) subproblem is:

$$\max_{u,z} \; J_2 = \int_0^T [ws - hx - (ru^2+qu+d) - \theta z]\, e^{-\rho t} dt + ax(T)e^{-\rho T} \qquad (3\text{-}9a)$$

s.t.

$$\dot{x} = zu - s, \qquad x(0)=x_0 \geq 0 \qquad (3\text{-}9b)$$

$$u \geq 0 \qquad (3\text{-}9c)$$

$$x \geq 0. \qquad (3\text{-}9d)$$

The Hamiltonian for the (PROD) subproblem is

$$H_2 = ws - hx - (ru^2+qu+d) - \theta z + (\lambda_1+\eta)(zu-s). \qquad (3\text{-}10)$$

$$\max_u H_2 \; : \; H_{2_u} = \partial H_2/\partial u = -2ru - q + (\lambda_1+\eta)z = 0.$$

After solving for u, we get

$$u^* = \begin{cases} 0 & \text{if } u''<0 \;\&\; x>0 \\ u'' & \text{if } u''\geq 0 \;\&\; x>0 \\ s/z & \text{if } u''<s/z \;\&\; x=0 \\ u'' & \text{if } u''\geq s/z \;\&\; x=0 \end{cases} \qquad (3\text{-}11)$$

where $u'' = [(\lambda_1+\eta)z-q]/2r.$ (3-12)

We now present a decentralized algorithm, which is similar to the approach proposed by Abad [1982, 1985]. The algorithm is based upon 'Interaction Prediction Principle' (Singh [1980]) and it exploits the serial structure of the system. With this algorithm, each subproblem is solved independently while information are exchanged between the two

subproblems. Note that 'the initial value shooting method' is used to solve the necessary conditions for the (MAIN)-subproblem and that η_{const} must also be determined from the global characteristics of the (PROD)-subproblem. The following iterative steps describe the algorithm.

Algorithm I-2: Decentralized Approach

(a) Assume trajectories $_i=\theta_0$ for i=0.

(b) Solve (MAIN) subproblem to obtain p_i.

(c) Pass p_i to (PROD) subproblem by letting $z_{i+1}= p_i$, and solve (PROD) subproblem.

(d) Check the convergence of θ by $e = \int_0^T |\theta_i - \theta| \, dt$.
If $e<\epsilon$, where ϵ is a very small number, we have solved the problem.
Otherwise, update θ by $\theta_{i+1} = (1-\gamma)\theta_i + \gamma\theta = \theta_i + \gamma(\theta-\theta_i)$, where $0\le\gamma\le1$ and $\theta = (\lambda_1+\eta)u$.

(e) Repeat Steps (b), (c) and (d).

3.3 Numerical Examples

Computer programmes (Appendices I-A and I-B) have been written in QuickBasic to demonstrate the two algorithms discussed in the previous section. These programmes were run on a microcomputer and then optimal trajectories were plotted using Lotus 123.

As the first example, let us consider a case with T=1, a=0, b=10, h=1, r=2, q=0, d=0, w=8, c=2.5, α=1, s(t)=4, M=5, ρ=.1, x(0)=3, and p(0)=1. This problem was solved by using the centralized approach as well as the decentralized approach. As we had expected, both

approaches gave us similar results. With the centralized approach, J^*=30.80 which includes 23.72 (profit from operation) and 7.08 (salvage value of the facility). With the decentralized approach, J^*=30.63 which includes 6.2 (profit from (MAIN)-operation), 17.49 (profit from (PROD)-operation), and 6.94 (salvage value of the facility). Figures I-2 to I-4 show the optimal trajectories for x, u, p, m, λ_1 and λ_2. We can see that the inventory level remains positive until the final time. Note that production starts even at t=0 (although there is enough inventory to satisfy the demand) and then continues until the final time. The major reason is that by producing a small quantity continuously over the entire planning horizon high production costs could be avoided. The similar solution structure always exists for cases with a quadratic production cost function and a positive initial inventory. No maintenance (i.e., m^*=0) is applied to the process until t=.71 and the maximum maintenance (i.e., m^*=M) is applied after t=.83 until the final time. Note that singular controls exist in the middle of the planning horizon (i.e., $t\epsilon(.71,.83]$), where $\lambda_2(1-p)=c$. As seen in Figure I-5, $\eta=\eta_{const}$=9.0 for the entire planning horizon since x>0 for $t\epsilon(0,T)$ (i.e., $\dot{\eta}x$=0). Other optimal trajectories are also shown in Figures I-6 to I-7.

The second example uses the same data as the first one does, except for M and x(0). We consider two cases: (i) M=2 and x(0)=3, and (ii) M=2 and x(0)=0. The optimal solution structure (as shown in Figures I-8 to I-11) for case (i) is very similar to that for the first example. Production begins in the beginning and continues until the final time T with x>0 and η_{const}=8.81 for $t\epsilon(0,T)$, and singular controls

exist in the middle of the planning horizon (i.e., $t\epsilon[.60,.61]$). For case (ii), because $x(0)=0$ production must start in the beginning of the planning horizon to satisfy the demand. The optimal trajectories (as shown in Figures I-12 to I-15) show that more units (than just enough to meet the demand) must be produced in the beginning so that inventory builds up and is kept for the future demand. This implies that $x>0$ and $\eta=\eta_{const}=27.2$ for $t\epsilon(0,T)$. Note that inventory is increasing for $t\epsilon[0,.37]$ and is then decreasing, thereafter. One interesting result may be the fact that no singular arcs exist as a part of optimal maintenance control. Thus, the optimal maintenance policy is of the 'bang-bang' type, where $m^*=0$ for $t\epsilon[0,.05]$ and $m^*=M$ for $t\epsilon(.05,T]$.

The above examples restrict their attention to the constant demand function, that is, $s(t)=4$. The third example uses the same data as those in the first case of the second example; however, it considers the following time-dependent demand functions: (i) $s(t)=8-8t$ for $t\epsilon[0,T]$, decreasing in time; (ii) $s(t)=8t$ for $t\epsilon[0,T]$, increasing in time; and (iii) $s(t)=16t$ for $t\epsilon[0,T/2]$ and $s(t)=16-16t$ for $t\epsilon[T/2,T]$, increasing and then decreasing in time. Note that $\int_0^T s(t)dt = 4$ for all cases. For case (i), the highest demand occurs at $t=0$ which requires high production rate in the beginning; however, production rate decreases at steeper rates in the end (see Figures I-16 to I-19). Note that u is discontinuous at $t=.87$ where inventory reaches to zero, and is then decreasing at steeper rates to zero. x is positive and concave for $t\epsilon[0,.87]$ and $x=0$ thereafter, implying that $\eta=\eta_{const}=9.45$ for $t\epsilon[0,.87]$ and is decreasing to zero thereafter. The optimal solution structure for case (ii) (shown in Figures I-20 to I-23) is similar to the case

with the constant demand (i.e., s(t)=4). More units (than just enough to satisfy the demand) are produced in the beginning and the excessive units are kept for the future demand. Note that x is positive for $t\epsilon[0,T)$ (thus $\eta=\eta_{const}=8.54$), and it is strictly concave because of the low demand in the beginning of the planning horizon. For case (iii), the optimal solution structure (shown in Figures I-24 to I-27) consists of those features of both case (i) and case (ii). u is discontinuous at t=.94 where inventory reaches to zero, and is decreasing at steeper rates to zero. x is positive and concave for $t\epsilon[0,.94]$, resulting in $\eta=\eta_{const}=8.94$, and x=0 thereafter, implying that η is decreasing to zero for $t\epsilon[.94,T]$.

The procedure used in Algorithms I-1 and I-2 for determining the junction points between nonsingular and singular subarcs was not tested for validity (i.e., it may result in a sub-optimal solution), and thus further study may be required. To find out how effective the algorithms are, however, the results obtained by them have been compared with those by GINO (General INteractive Optimization) after discretizing the problem. The comparison shows that both approaches yield the same optimal solution structures, and that the corresponding objective functional values are very close to each other (see Table I-2). Algorithms for calculating the exact junction points between nonsingular and singular subarcs of an optimal trajectory for boundary-value problems, given that the structure of the optimal path is known, are discussed in Maurer [1976]. A study for synthesizing the optimal maintenance control for the (PM) control problem in a decentralized mode is in progress (Cho et al. [1992]).

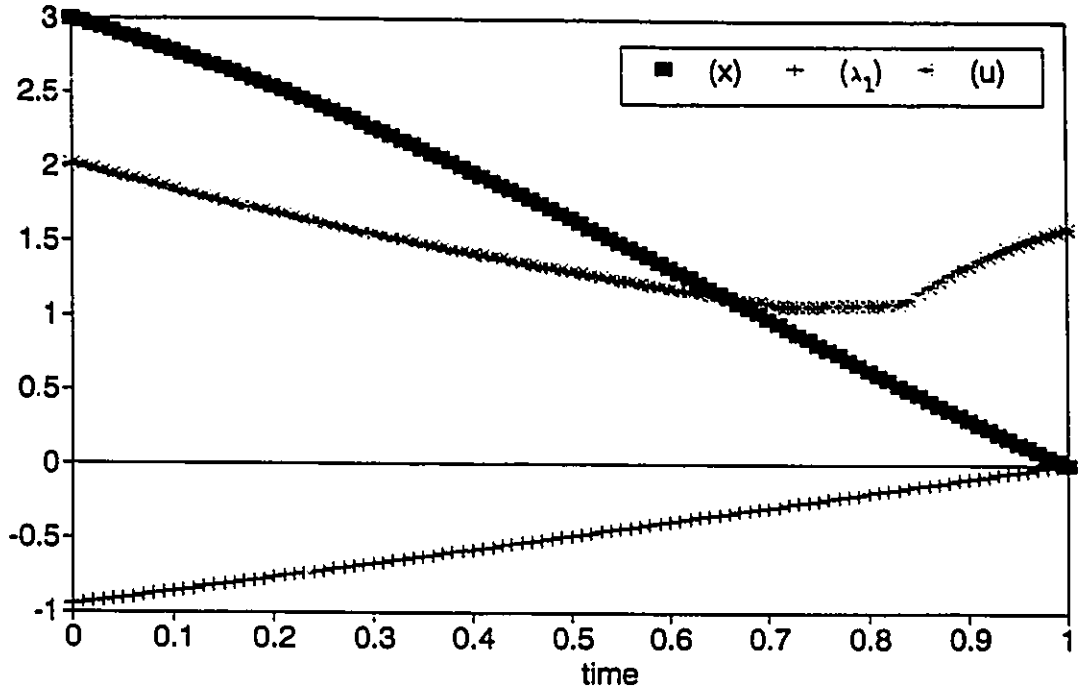


Figure I-2: Optimal x, λ_1 and u trajectories for Example 1.

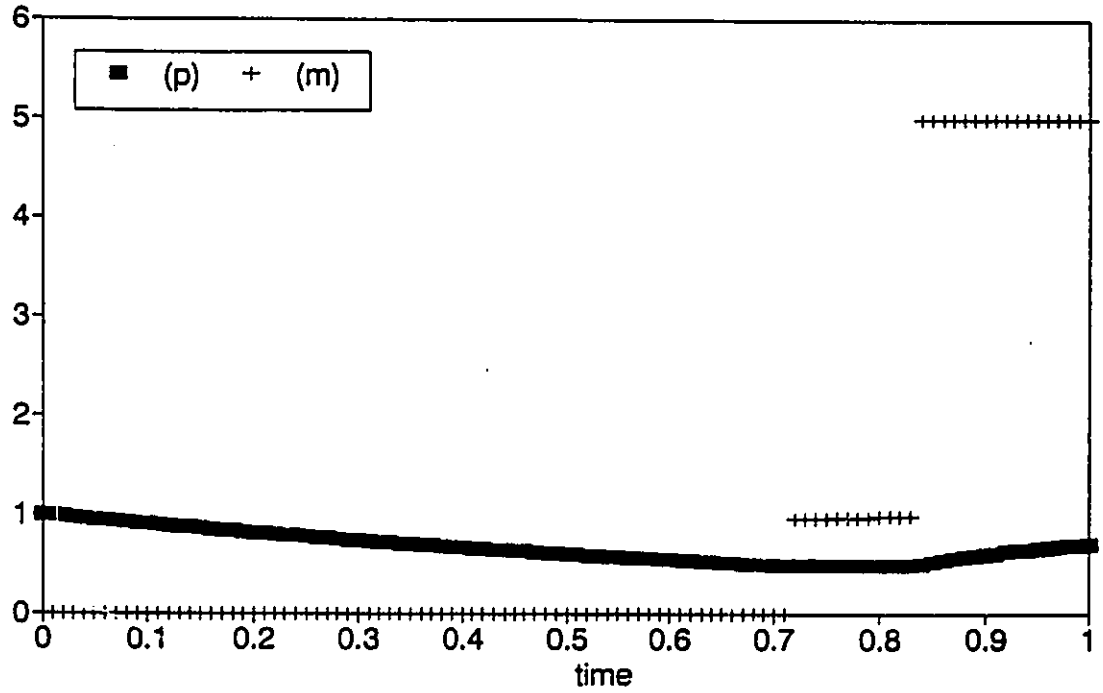


Figure I-3: Optimal p and m trajectories for Example 1.

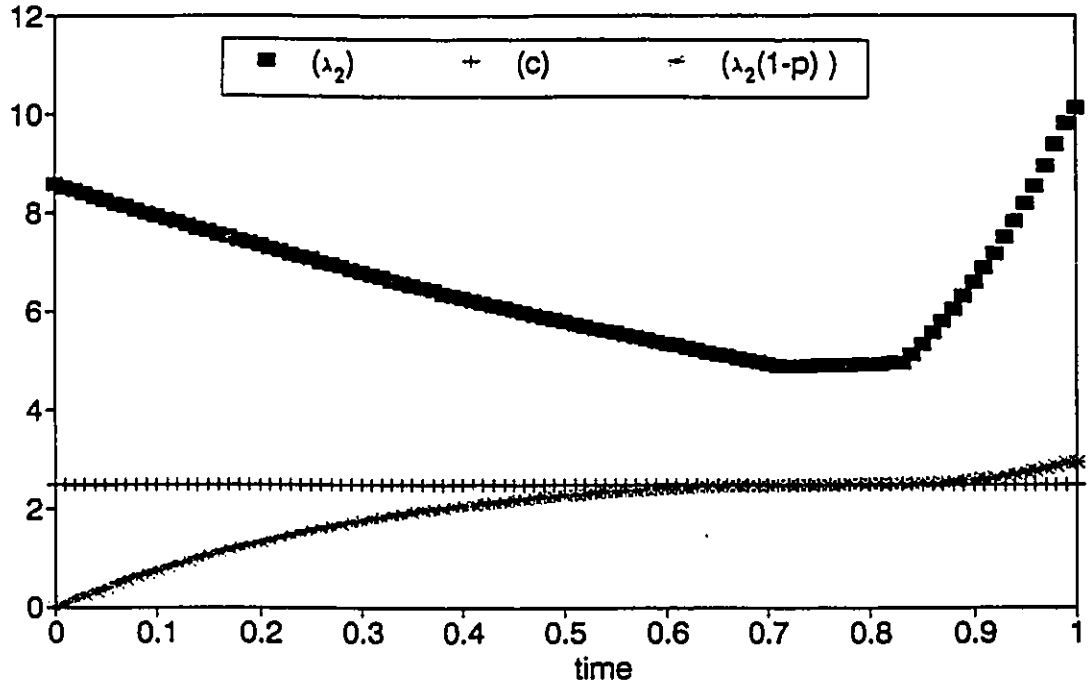


Figure I-4: Optimal λ_2, c and $\lambda_2(1-p)$ trajectories for Example 1.

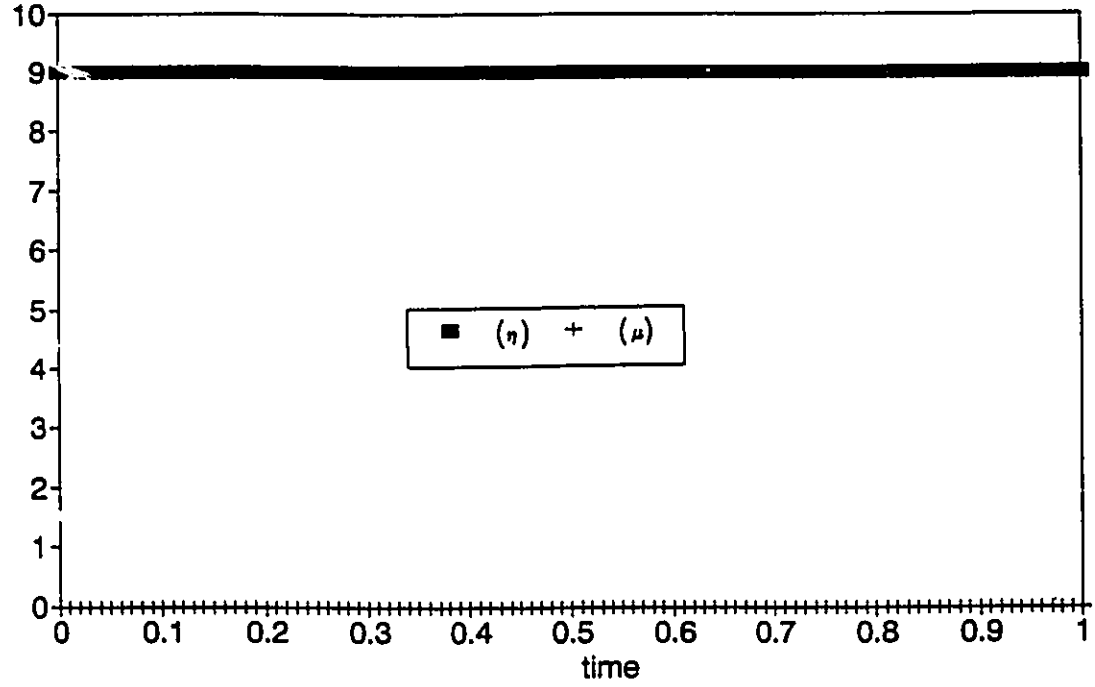


Figure I-5: Optimal η and μ trajectories for Example 1.

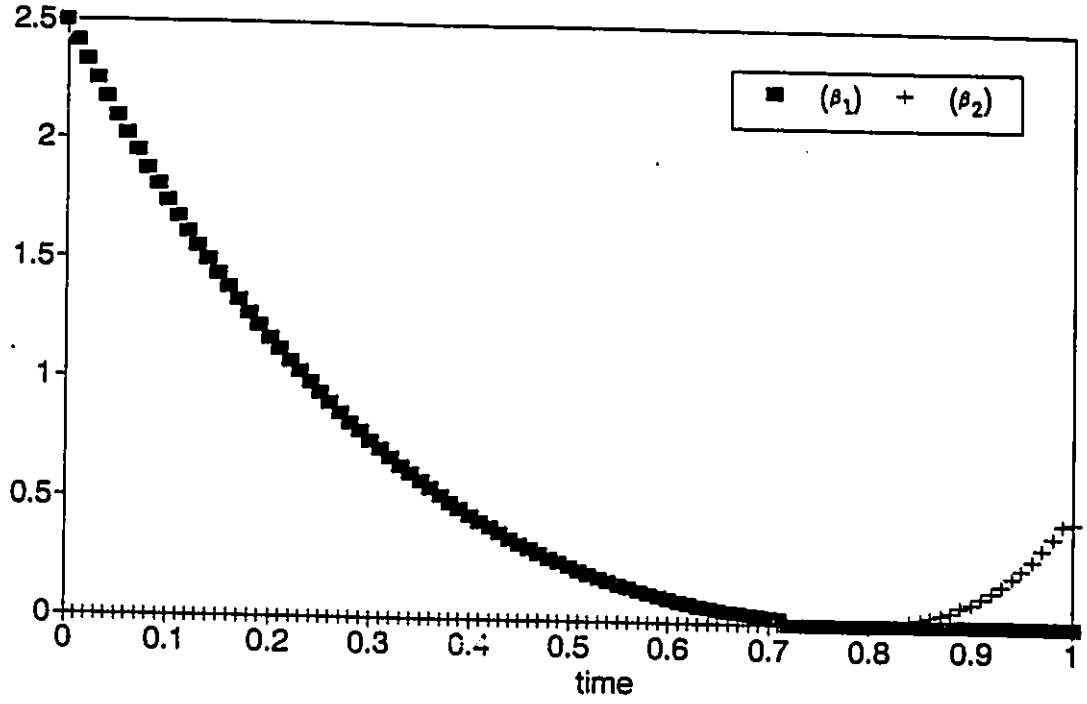


Figure I-6: Optimal β_1 and β_2 trajectories for Example 1.

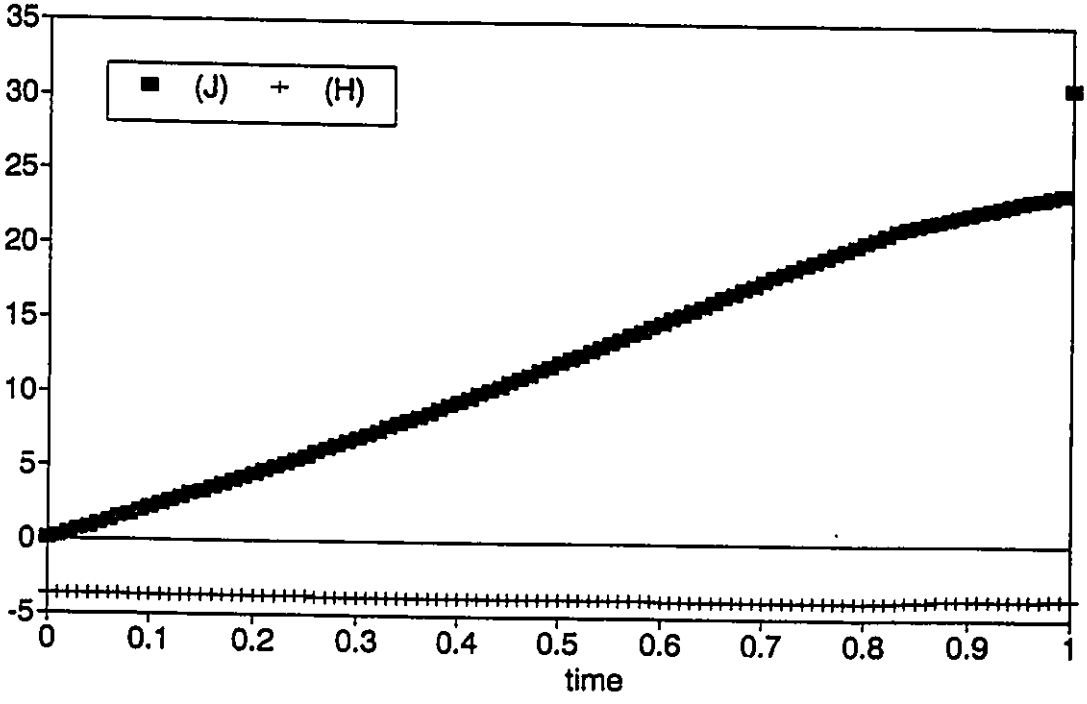


Figure I-7: Optimal J and H trajectories for Example 1.

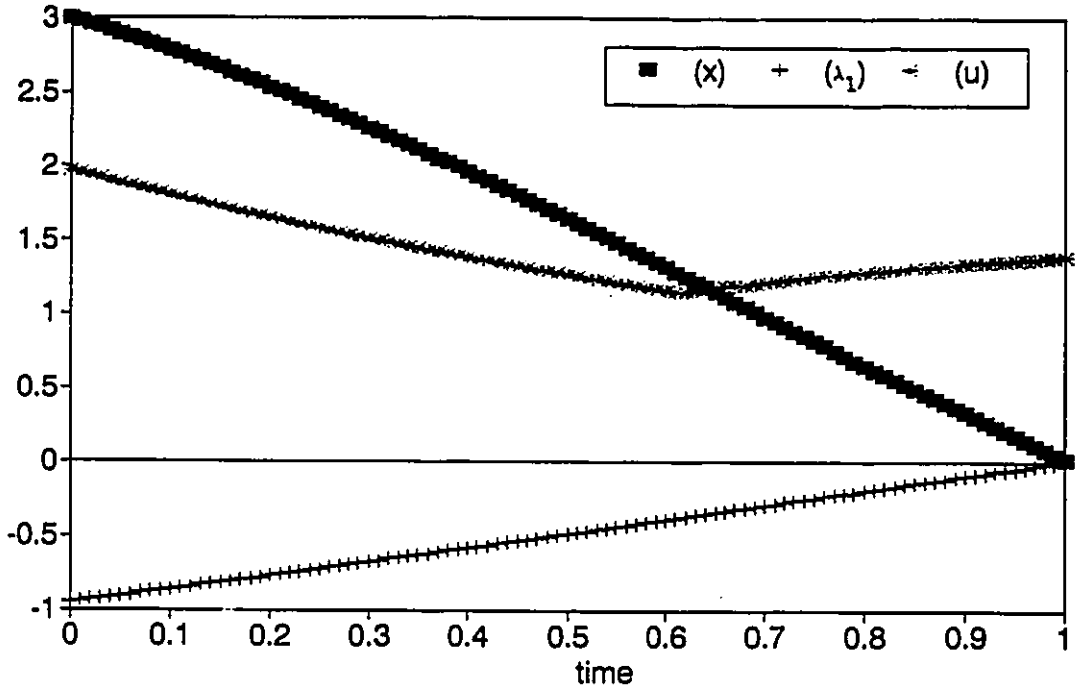


Figure I-8: Optimal x, λ_1 and u trajectories for Example 2(i).

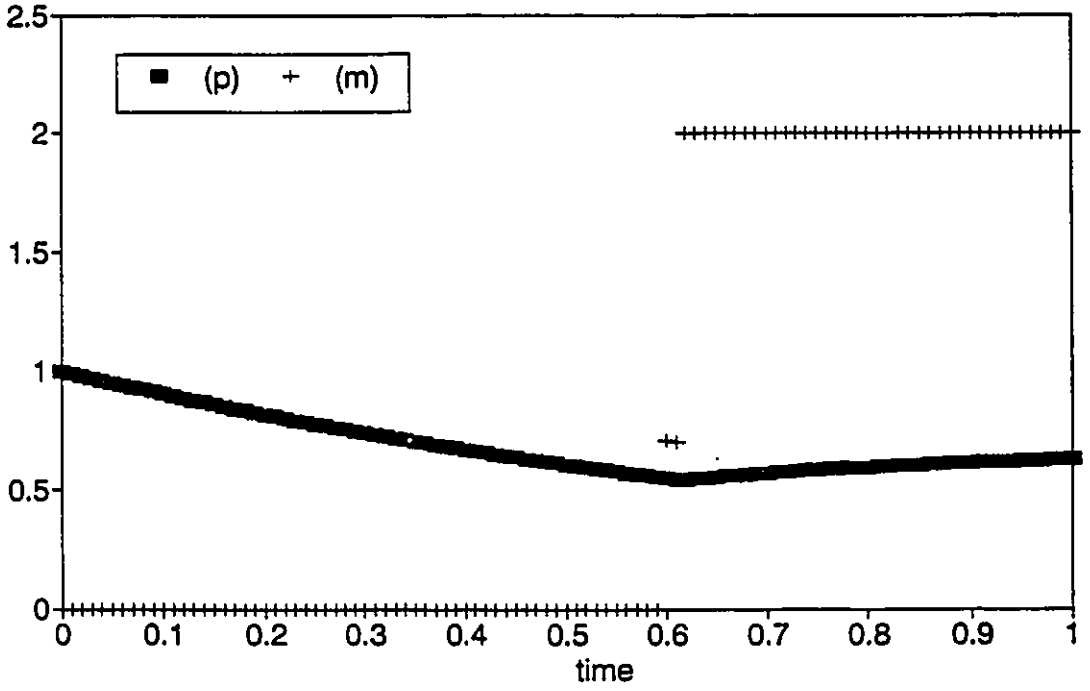


Figure I-9: Optimal p and m trajectories for Example 2(i).

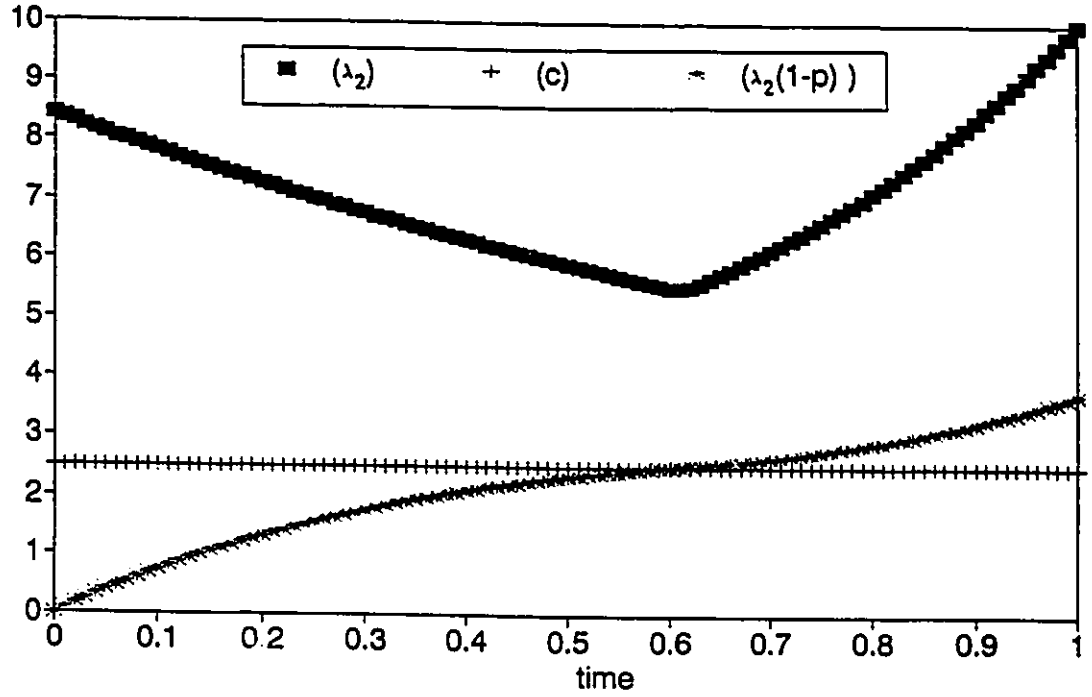


Figure I-10: Optimal λ_2, c and $\lambda_2(1-p)$ trajectories for Example 2(i).

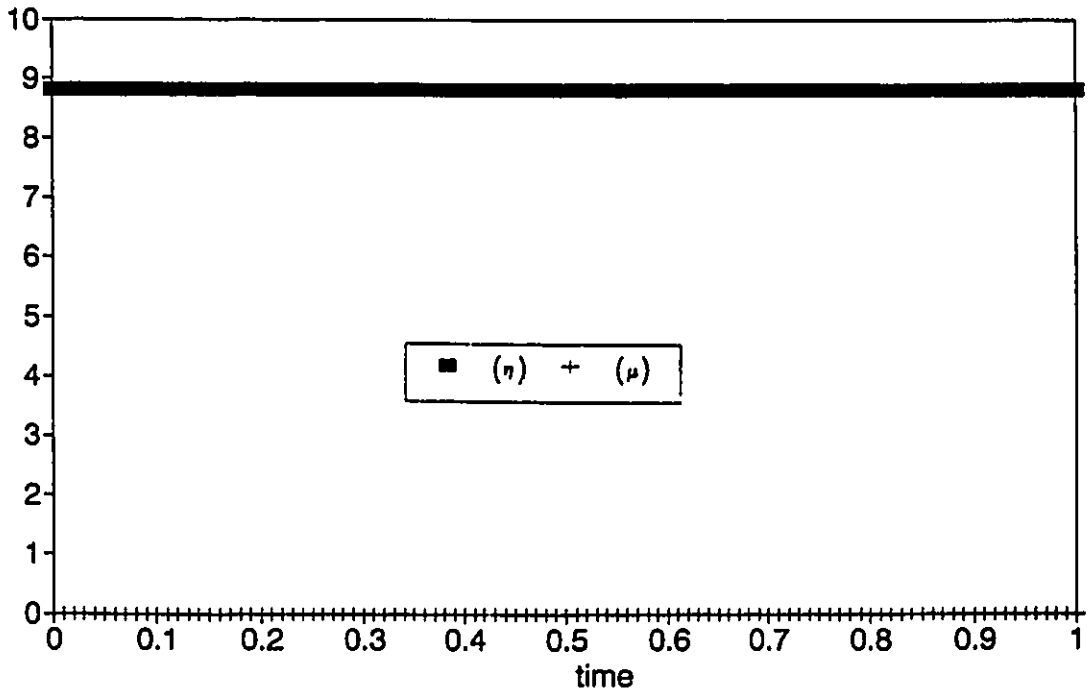


Figure I-11: Optimal η and μ trajectories for Example 2(i).

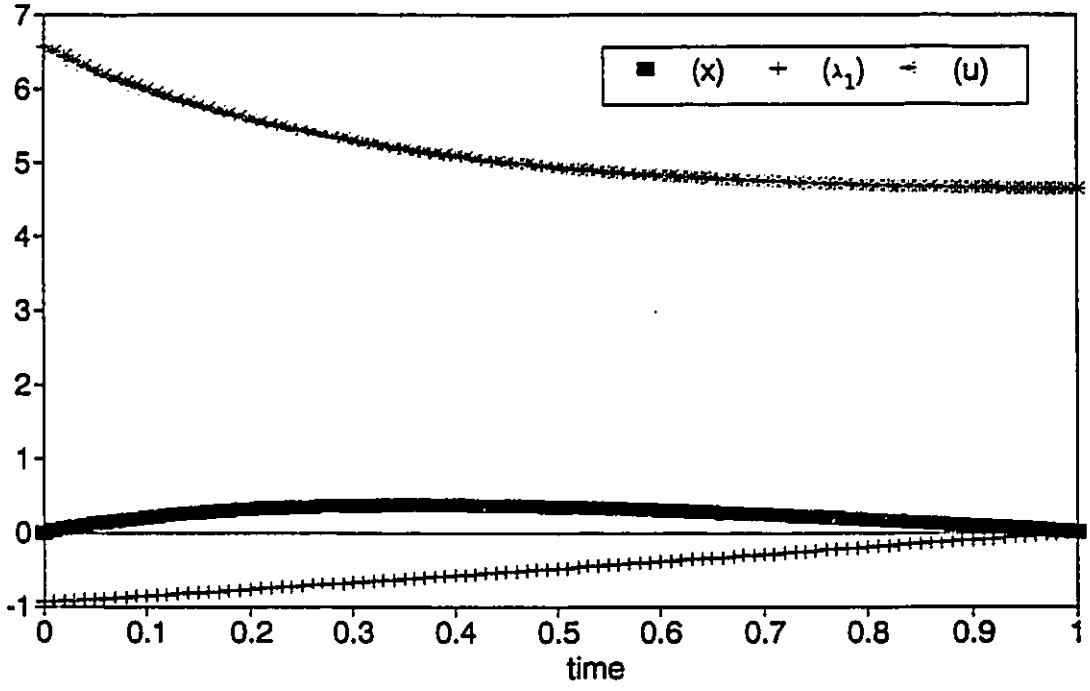


Figure I-12: Optimal x, λ_1 and u trajectories for Example 2(ii).

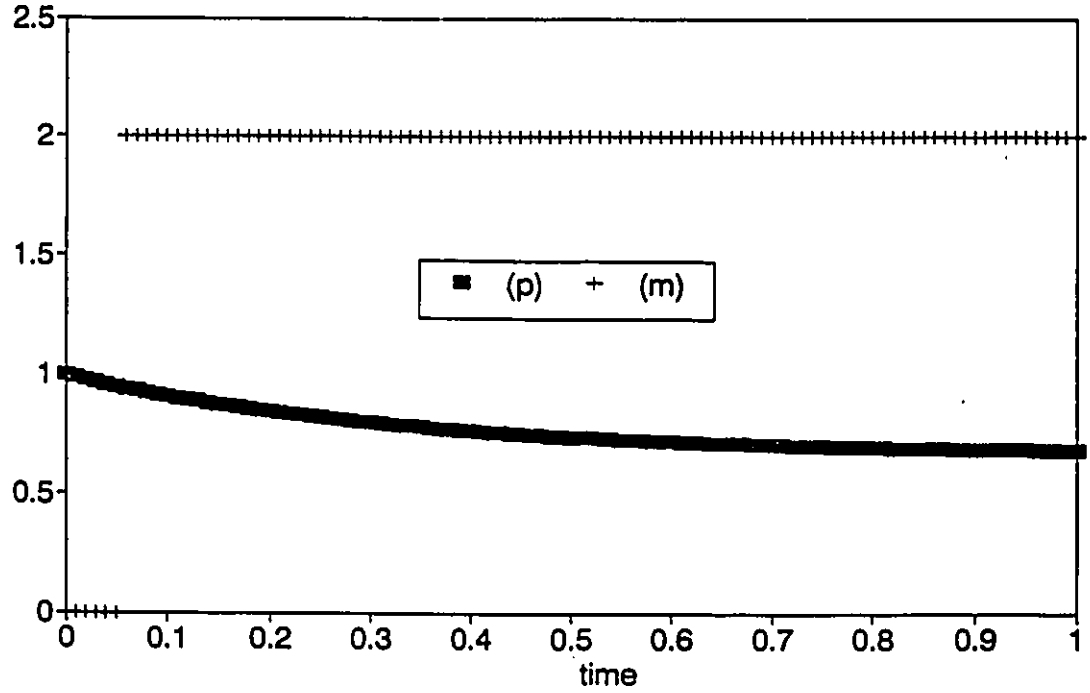


Figure I-13: Optimal p and m trajectories for Example 2(ii).

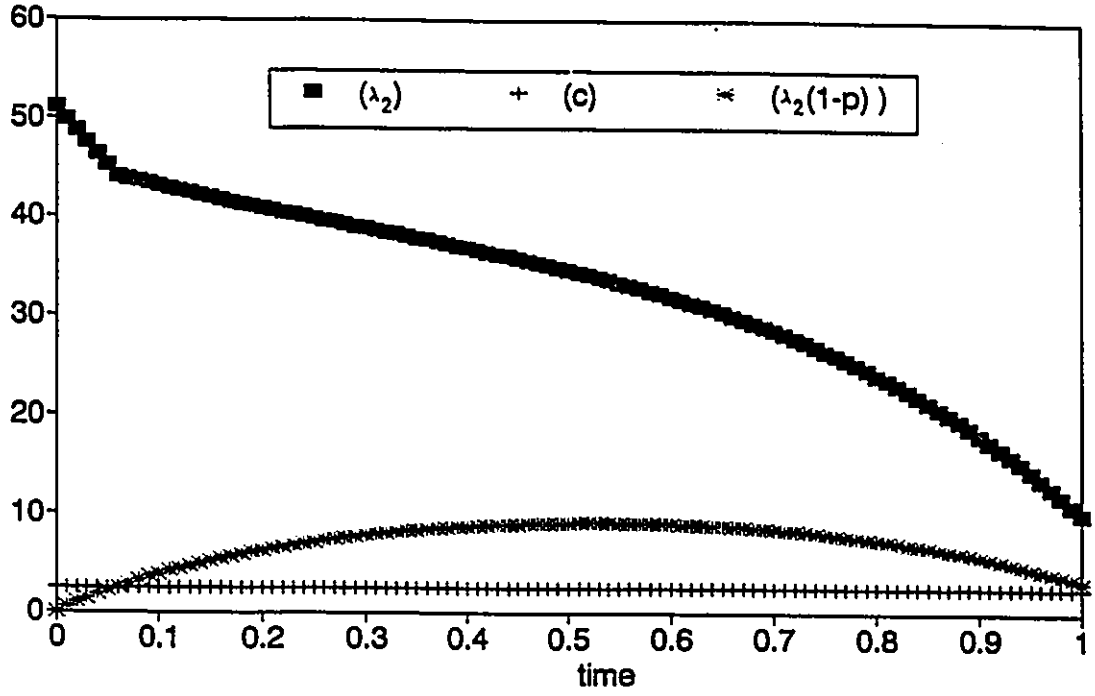


Figure I-14: Optimal λ_2, c and $\lambda_2(1-p)$ trajectories for Example 2(ii).

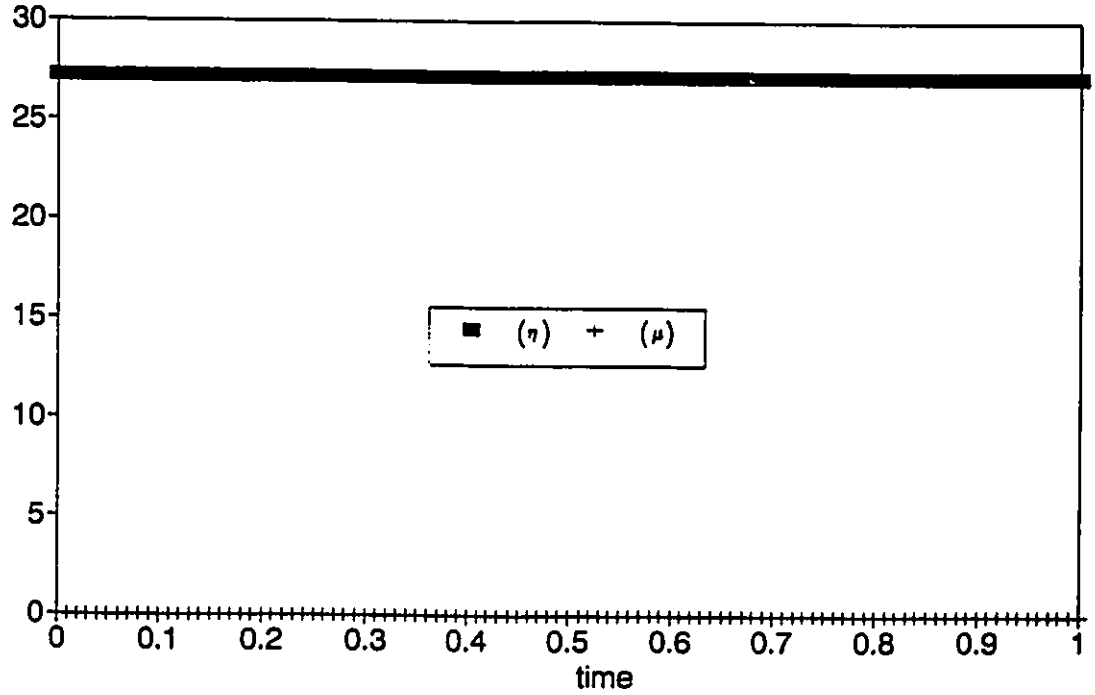


Figure I-15: Optimal η and μ trajectories for Example 2(ii).

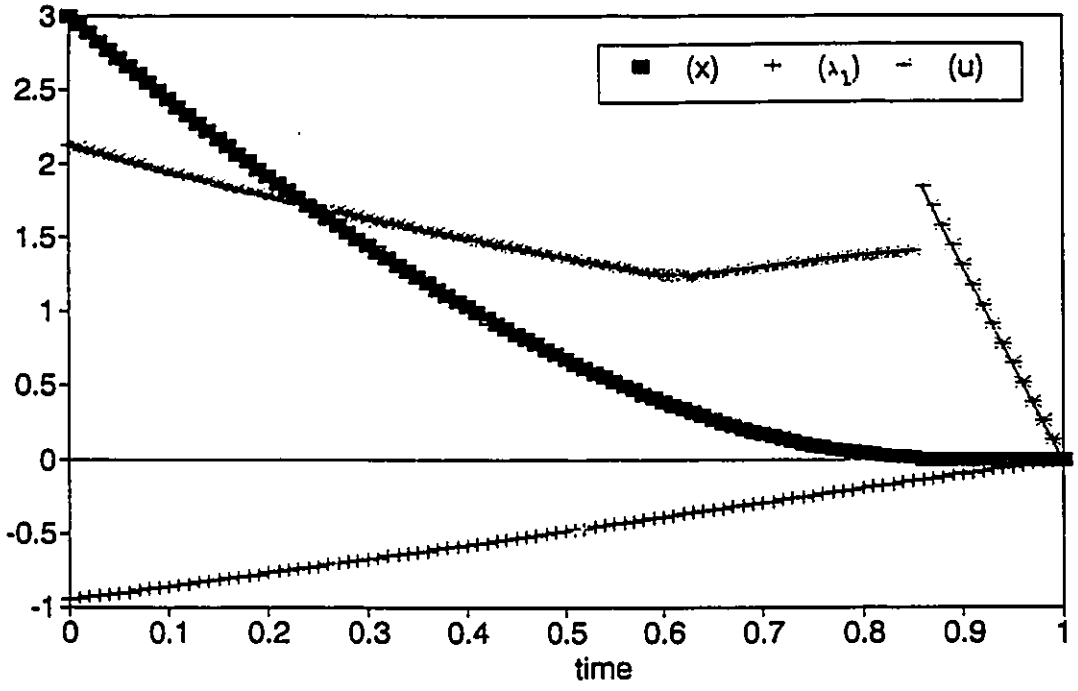


Figure I-16: Optimal x, λ_1 and u trajectories for Example 3(i).

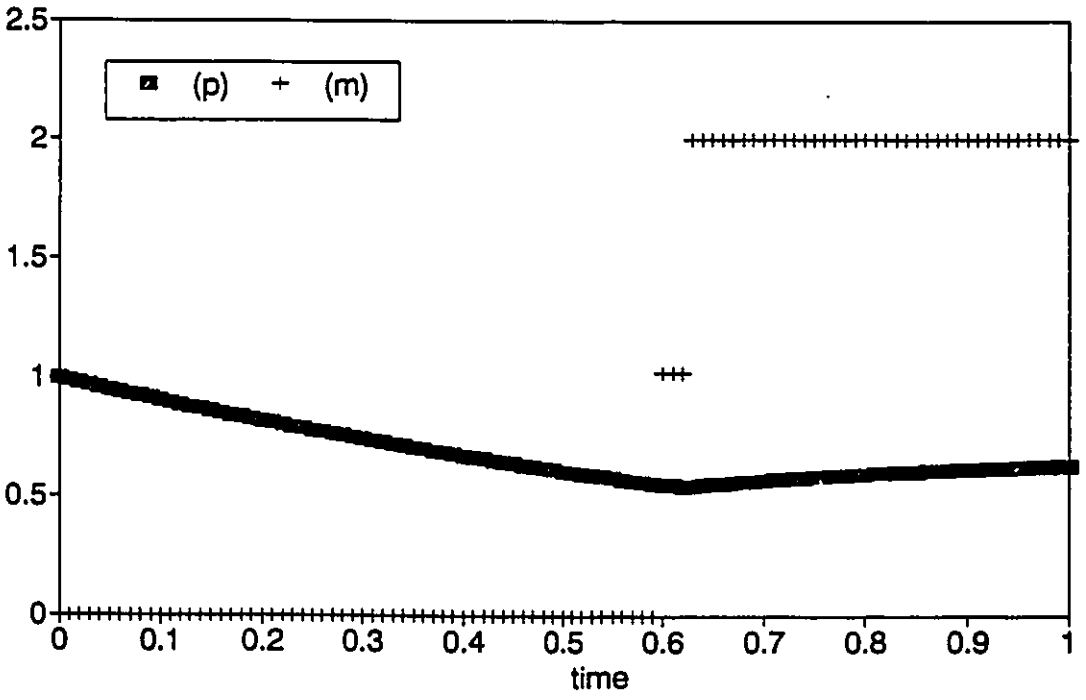


Figure I-17: Optimal p and m trajectories for Example 3(i).

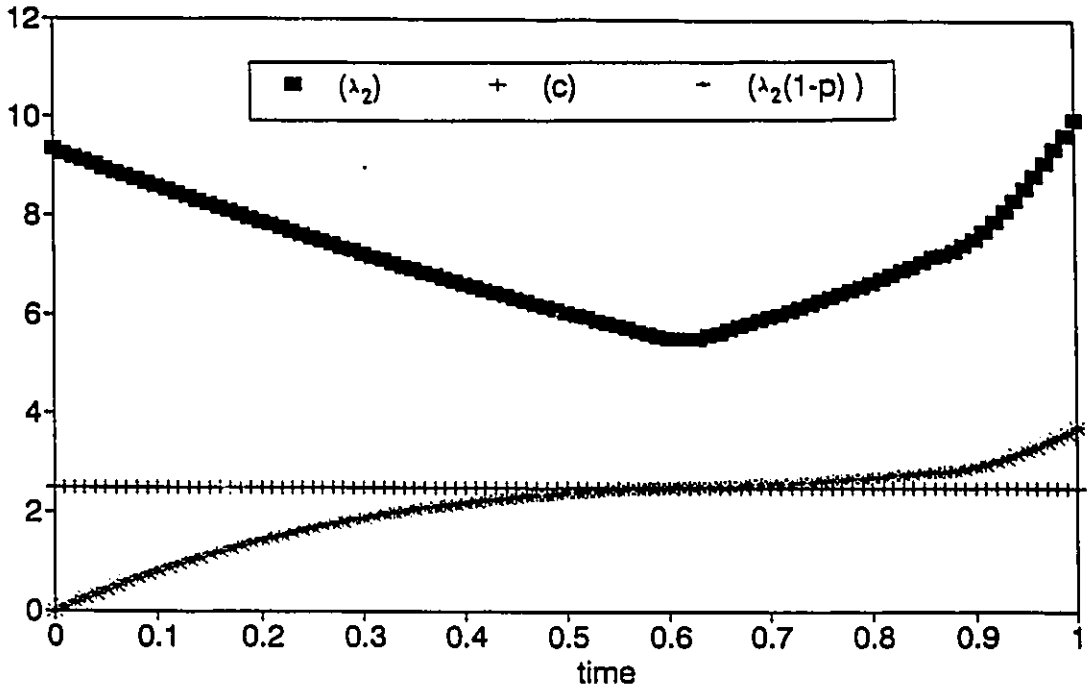


Figure I-18: Optimal λ_2, c and $\lambda_2(1-p)$ trajectories for Example 3(i).

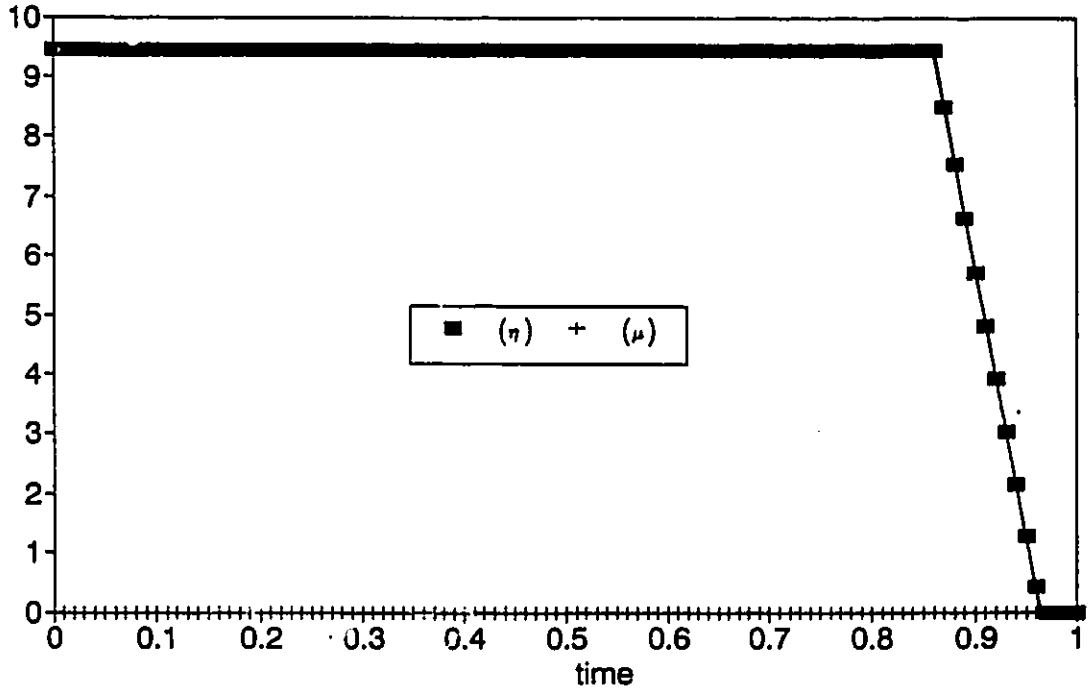


Figure I-19: Optimal η and μ trajectories for Example 3(i).

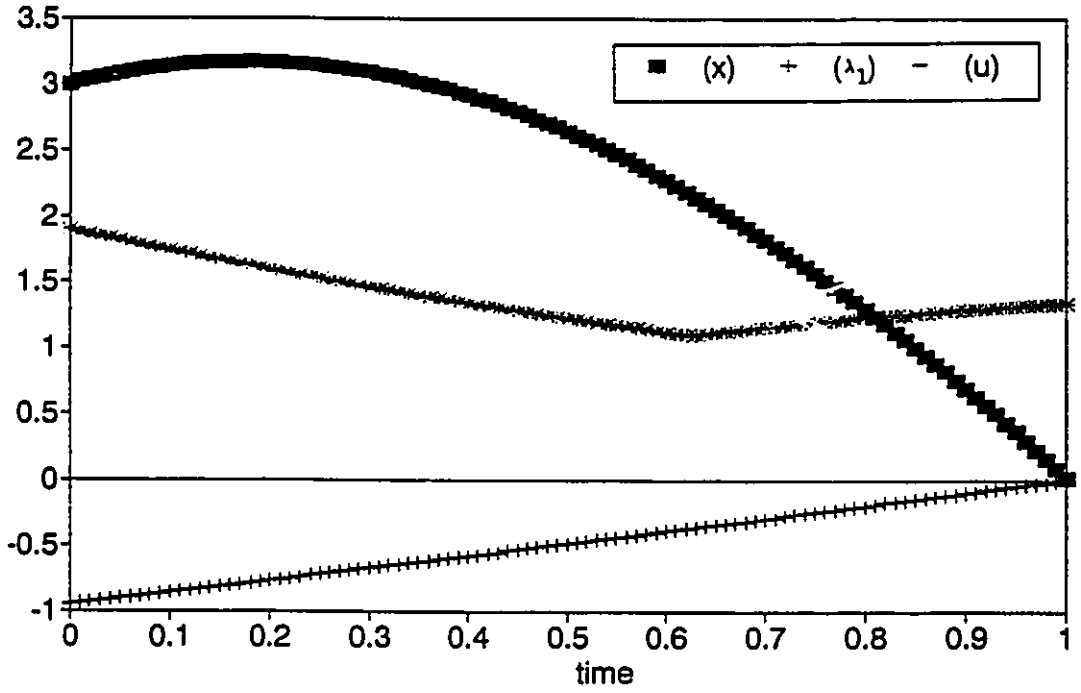


Figure I-20: Optimal x, λ_1 and u trajectories for Example 3(ii).

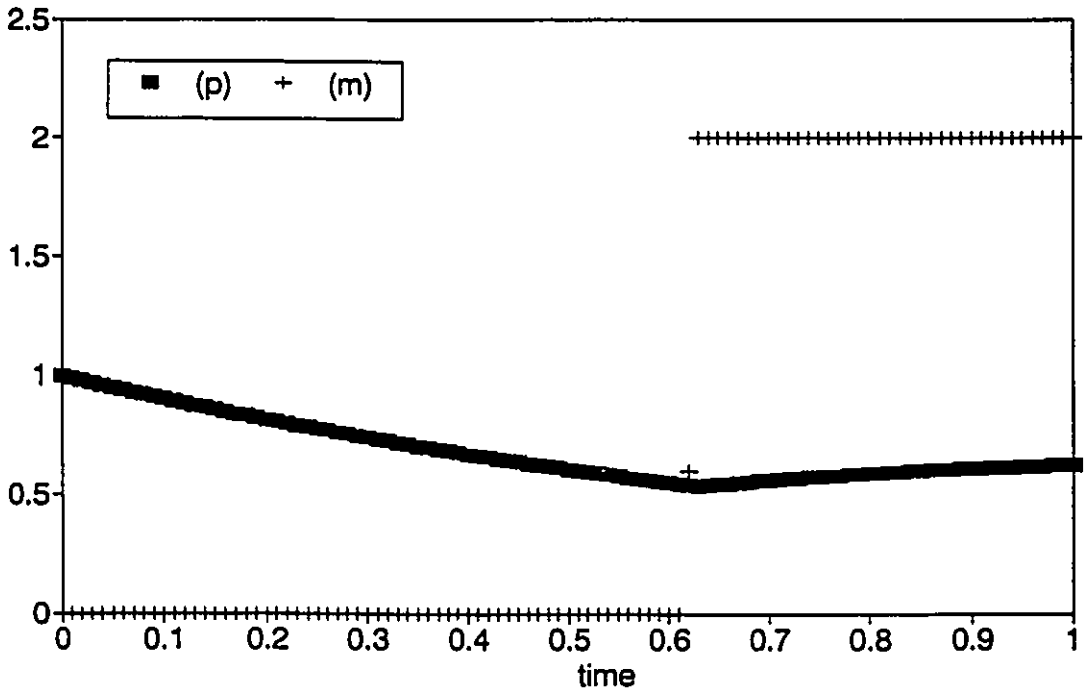


Figure I-21: Optimal p and m trajectories for Example 3(ii).

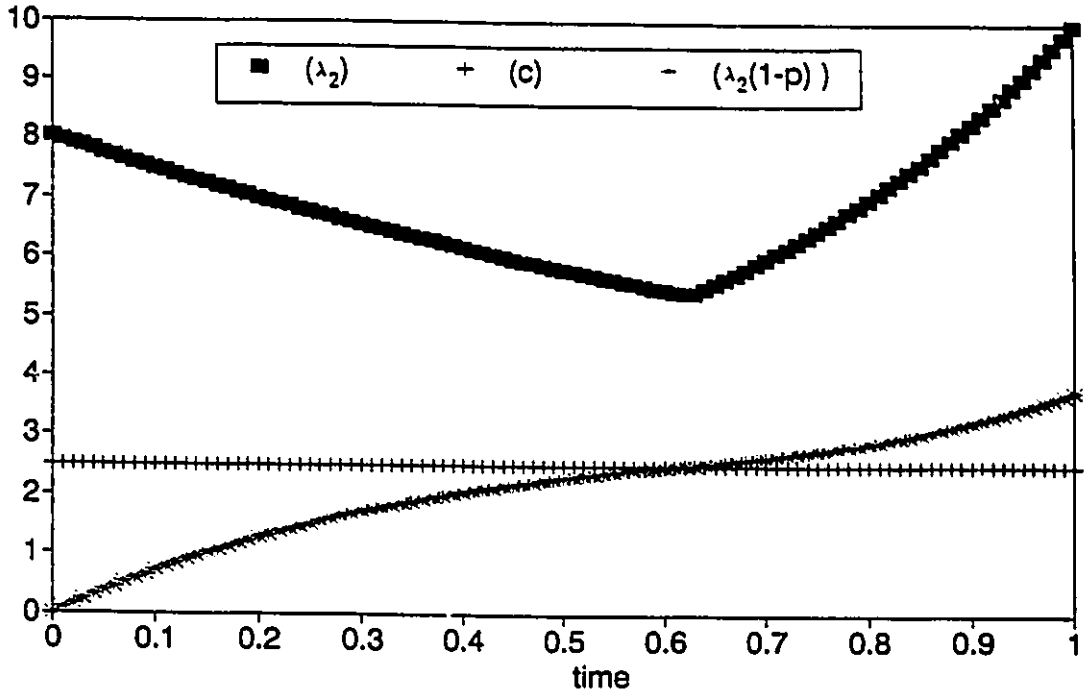


Figure I-22: Optimal λ_2, c and $\lambda_2(1-p)$ trajectories for Example 3(ii).

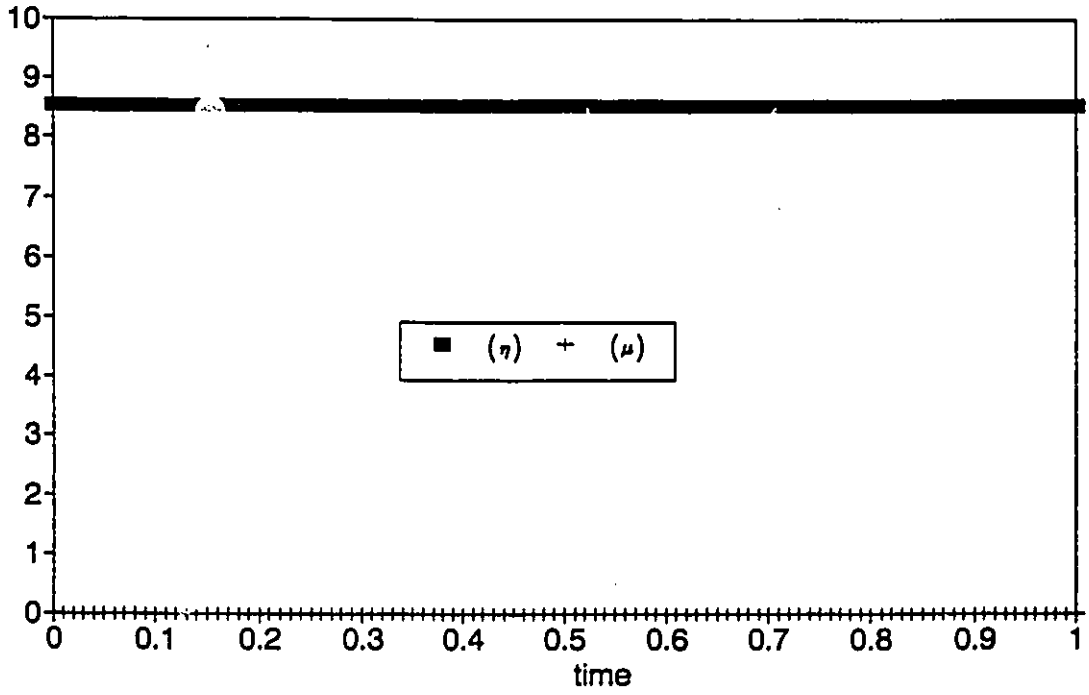


Figure I-23: Optimal η and μ trajectories for Example 3(ii).

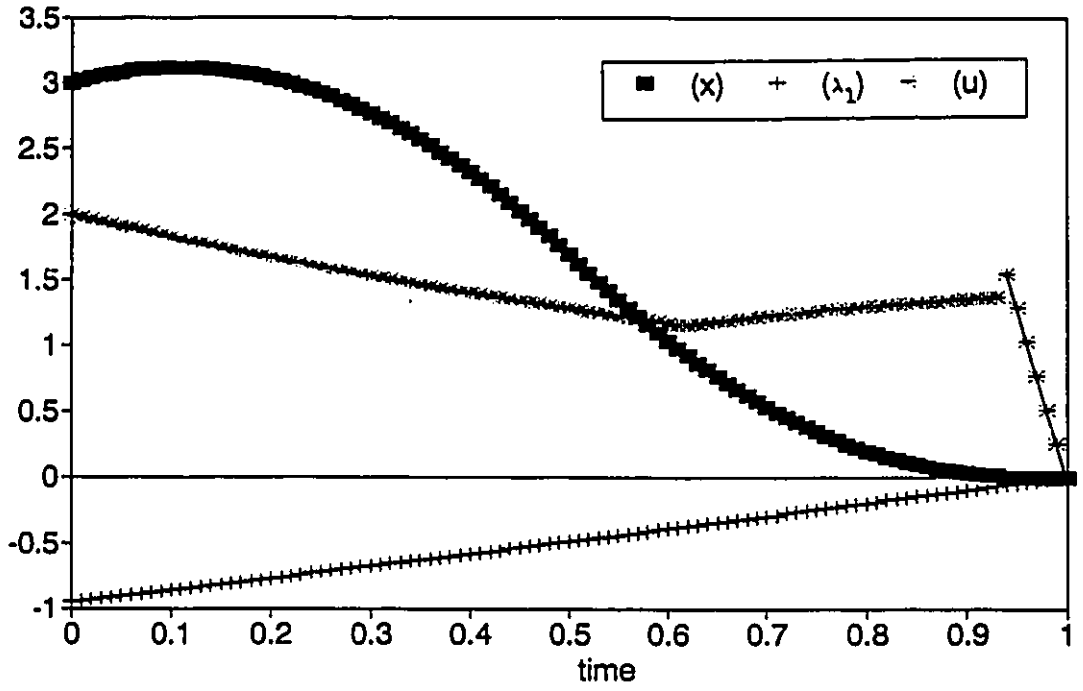


Figure I-24: Optimal x, λ_1 and u trajectories for Example 3(iii).

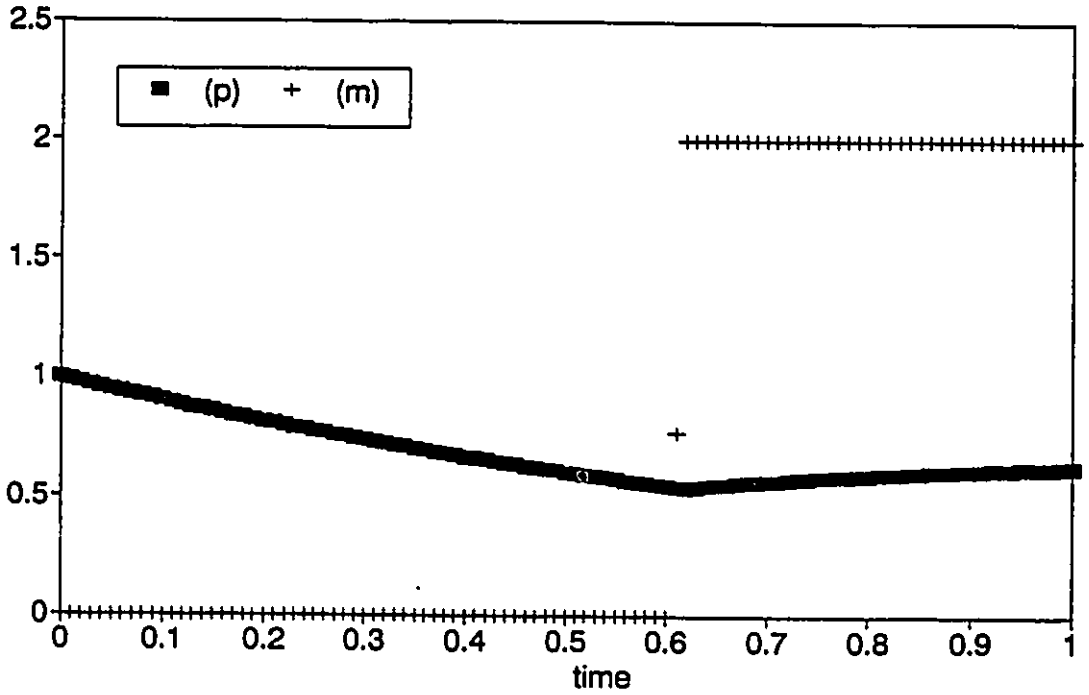


Figure I-25: Optimal p and m trajectories for Example 3(iii).

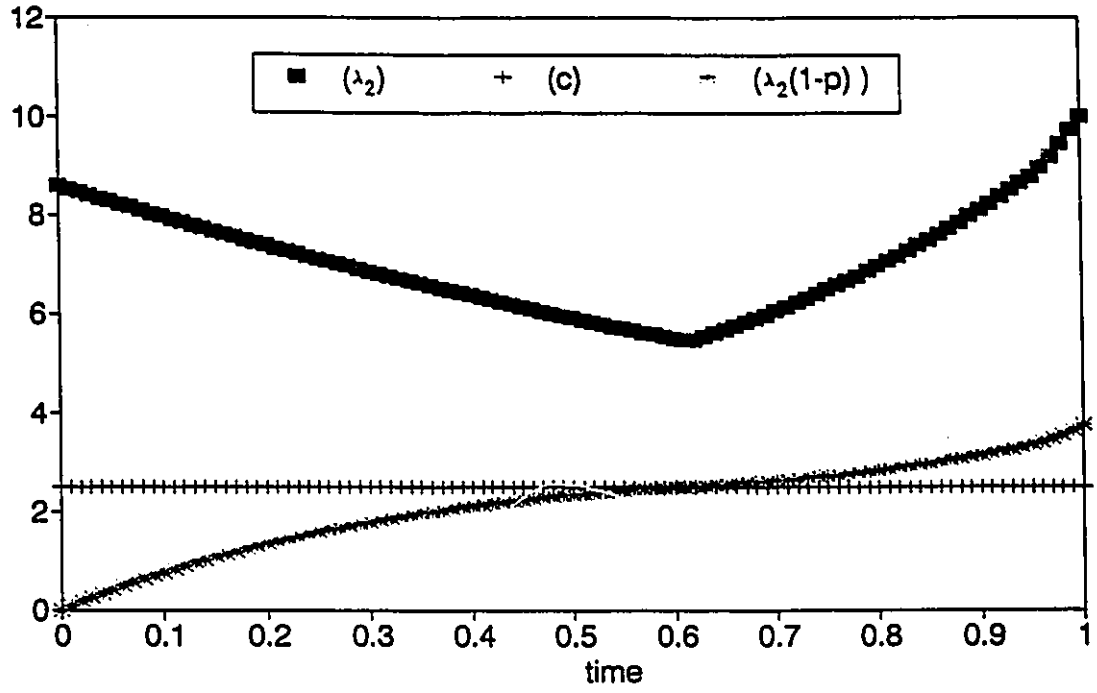


Figure I-26: Optimal λ_2, c and $\lambda_2(1-p)$ trajectories for Example 3(iii).

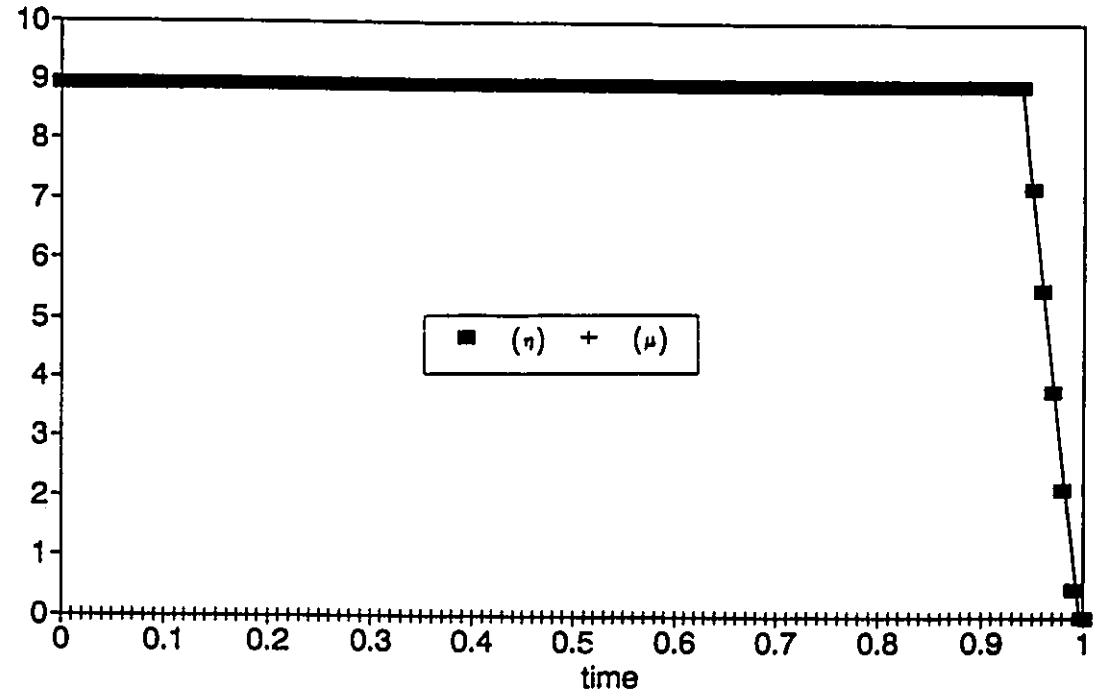


Figure I-27: Optimal η and μ trajectories for Example 3(iii).

Demand Rate Functions: (a) $s(t)=4$ (b) $s(t) = 8-8t$

(c) $s(t)=8t$ (d) $s(t) = \begin{cases} 16t & \text{for } 0 \le t \le T \\ 16-16t & \text{for } T/2 \le t \le T \end{cases}$

Examples	O.C.T. vs. GINO	Demand Rate Functions			
		(a)	(b)	(c)	(d)
x(0)=3, M=5	O.C.T.	30.80	31.02	30.40	30.51
	GINO	31.07	31.33	30.12	30.92
x(0)=3, M=2	O.C.T.	30.81	31.07	30.48	30.49
	GINO	30.66	31.07	29.84	30.58
x(0)=0, M=2	O.C.T.	(19.73)*	(27.76)	(19.27)	(22.57)
	GINO	(18.92)	(29.75)	(23.42)	(23.40)

* O.F. values in parentheses indicate negative profits.

Table I-2: Comparison of Objective Functional Values
(Optimal Control Theory (O.C.T.) vs. GINO)

CHAPTER 4 CONCLUSIONS

4.1 Discussions

In Part I, we incorporated a maintenance problem into a production control model and developed a combined decision model for a simultaneous determination of optimal production and maintenance policies. We utilized Pontryagin's maximum principle to obtain the necessary conditions for the controls to be optimal. We also introduced two types of optimization techniques, the centralized and decentralized approaches, for solving the necessary conditions numerically. With the centralized approach, all necessary conditions are simultaneously solved and optimal controls are obtained after several iterations. With the decentralized approach, on the other hand, exchanges of information between the two subproblems continue until a convergence of θ occurs. The decentralized approach may require a larger number of computations; however, it provides not only the global characteristics of the main problem (e.g. net profit, net salvage value, etc.), but also additional information regarding local characteristics of each subproblem (e.g. profit and salvage value of individual departments, transfer prices).

We now summarize other findings which result from our study. (1) For the case where there is no initial inventory, it is always optimal to produce more units (than just enough to satisfy the current demand) in the beginning of the planning horizon and keep the excess for the future demand unless the demand function is decreasing in time. In

this way, high production costs (resulting from high production rates required later) can be avoided although some holding costs due to the excessive inventory may be incurred. For a decreasing demand function, it is optimal to produce units exactly equal to the current demand.

(2) The salvage value of the production facility plays an important role in determining an optimal level of maintenance. For a case where there is no salvage value for the facility, we have found that the optimal policies usually include zero level of maintenance. However, the zero maintenance policy may not always be optimal, especially for a process with a very high obsolescence rate.

(3) Singular arcs, if exist, seem to occur in the middle of the planning horizon. It is difficult to prove this due to the complexity of our model; however, we have shown in the previous section that no singular arcs exist at the start of the planning horizon. Sethi [1973] and Abad [1982] discussed an existence of singular arcs in the entire middle portion of a long-term planning horizon.

(4) For a linear production cost function, it is always optimal not to produce units until on-hand inventory is completely depleted. This statement is justified by the following obvious reason. With a linear production cost function, a fixed unit production cost is incurred no matter how many units are produced. Thus, by producing units when they are needed one can avoid unnecessary inventory holding costs.

4.2 Modification to Model

For future research, we may consider the following modification.

(1) Sometimes, the process performance deteriorates on the basis of the

number of units produced rather than of its age. Here, the process performance is assumed to decline as the total number of units that have been produced by the process increases. In such a case, the state equation for the process performance may be written as

$$p(t+\Delta t) = p(t) - \sigma(\xi)p(t)\,\Delta t + (1 - p(t))\,m(t)\,\Delta t,$$

where $\sigma(\xi)$ is the obsolescence function of the process, which is a non-decreasing function of the total number of units produced by the process up to t (i.e., $\xi(t)=\int_0^t u(\tau)d\tau$). Note that the proposed state equation now contains u(t), implying that a higher degree of dependency between the two state equations (for x(t) and p(t)) exists. A similar analysis may be used to deal with this problem; however, the decentralized approach seems to be inappropriate because of the high degree of dependency between the two subproblems.

(2) The linear salvage value of the facility (i.e., bp(T)) can be replaced by a quadratic salvage function (i.e., $bp^2(T)$). The quadratic salvage function may be a more realistic assumption since it puts different weights based on the process performance at the final time (i.e., more weights on the process with higher quality than on that with lower quality). With the quadratic salvage function, the problem would become more complicated because we would have the ending adjoint value $\lambda_2(T)=2bp(T)$, which is unknown.

(3) Models of preventive maintenance decisions often include the possibility of system (machine) breakdown. In other words, preventive maintenance is applied to the system to change the probability of system (machine) failure. Such a consideration requires an introduction of randomness in the model. However, it will change the model entirely.

APPENDIX I-A: COMPUTER PROGRAM - CENTRALIZED APPROACH

```
REM ** THIS PROGRAM SOLVES THE CONTINUOUS-TIME PRODUCTION/MAINTENANCE **
REM ** PROBLEM USING THE INITIAL-VALUE SHOOTING METHOD                  **
REM **                ( The Centralized Approach )                     **

' Program TS3-1  23/3/89   Revised  24/6/89  4/8/89  27/9/89  2/10/89

REM ** MAIN PROGRAM **

' Open a file to write trajectories so that LOTUS-123 can use them
' to draw graphs

OPEN "a:ts3-1-3.prn" FOR OUTPUT AS #1

' Input data

tf = 1                    'planning time horizon
del = .01                 'delta
a = 0                     'salvage value for each unit unsold
b = 10                    'salvage value of the facility
h = 1                     'holding cost
r = 2                     'coefficient of quadratic production cost term
q = 0                     'coefficient of linear production cost term
d = 0                     'constant production cost term
w = 8                     'revenue form selling a unit item
c = 2.5                   'maintenance cost
alpha = 1                 'obsolescence rate of process performance
s = 4                     'demand rate
mbar = 2                  'maximum level of maintenance
xzero = 3                 'initial inventory level
pzero = 1                 'initial process performance
rho = .1                  'constant continuous discount rate
etaconst = 9

' Calculate initial value of lambda1

IF rho = 0 THEN
   lam1zero = a + h * (-tf)
ELSE
   lam1zero = -h / rho + (a + h / rho) * EXP(rho * (-tf))
END IF
```

```
' Guess initial value of lambda2

lam2zero = 0


' Print input data on screen

PRINT "tf="; tf, "final time"
PRINT "del="; del, "delta"
PRINT "a="; a, "salvage value for each unit unsold"
PRINT "b="; b, "salvage value of the facility"
PRINT "h="; h, "inventory holding cost"
PRINT "r="; r, "coefficient of quadradic production cost term"
PRINT "q="; q, "coefficient of linear production cost term"
PRINT "d="; d, "constant production cost term"
PRINT "w="; w, "revenue from selling a unit item"
PRINT "c="; c, "maintenance cost"
PRINT "alpha="; alpha, "obsolescence rate of process performance"
PRINT "s(t)="; s, "demand rate"
PRINT "MBAR="; mbar, "maximum level of maintenance"
PRINT "xzero="; xzero, "initial inventory level"
PRINT "pzero="; pzero, "initial process performance"
PRINT "rho="; rho, "constant continuous discount rate"
PRINT "lam1(0)="; lam1zero, "initial (known) value of lambda1"
PRINT "lam2(0)="; lam2zero, "initial (guessed) value of lambda2"
PRINT "etaconst="; etaconst
PRINT "      "
PRINT "      "


' Initialization

lam1old = lam1zero:      lam2old = lam2zero
lam2low = 0:     lam2high = 50          'low and high values of lam2zero
xold = xzero:     pold = pzero
inc = 0:     profit = 0

count = 0
flag = 0


DO WHILE flag <= 1

   IF count > 0 THEN
      IF flag = 1 THEN
         lam2zero = lam2zero
      ELSE
         IF lam2tf > b THEN
            lam2high = lam2zero
         ELSE
            lam2low = lam2zero
         END IF
```

```
      lam2zero = (lam2high + lam2low) / 2
   END IF
ELSE
   lam2zero = lam2zero
END IF

lam1old = lam1zero
lam2old = lam2zero
xold = xzero
pold = pzero
inc = 0
profit = 0

IF flag > 0 THEN
   PRINT "    "
   PRINT SPC(5); "t"; SPC(5); "x"; SPC(5); "p"; SPC(5); "lam1";
         SPC(3); "lam2"; SPC(5); "u"; SPC(6); "m"; SPC(5); "eta";
         SPC(5); "mu"; SPC(3); "beta1"; SPC(2); "beta2"
ELSE
   PRINT "    "
END IF


 ' Determination of optimal production controls

FOR t = 0 TO tf STEP del

   s = 4
   sdot = 0

   IF rho = 0 THEN
      lamlold = a - h * (tf - t)
   ELSE
      value1 = EXP(rho * (t - tf))
      lamlold = -h / rho + (a + h / rho) * value1
   END IF

   SELECT CASE ABS(xold)
      CASE IS > .01
         etaold = etaconst
         utest = ((lamlold + etaold) * pold - q) / (2 * r)
         IF utest > 0 THEN
            uold = utest
            muold = 0
         ELSE
            uold = 0
            muold = q - (lamlold + etaold) * pold
         END IF
      CASE ELSE
         muold = 0
         etaold = etaconst
         utest = ((lamlold + etaold) * pold - q) / (2 * r)
```

```
      IF utest > s / pold THEN
         uold = utest
      ELSE
         uold = s / pold
         etaold = (2 * r * uold + q) / pold - lam1old
      END IF

END SELECT


' Obtain non-singular and singular optimal maintenance controls

lam1dot = (a * rho + h) * value1
sinarc = lam2old * (1 - pold) - c
sintest = ABS(sinarc)

SELECT CASE sintest

   CASE IS > .02
      IF sinarc < 0 THEN
         mold = 0
      ELSE
         mold = mbar
      END IF

   CASE ELSE

      IF uold = 0 THEN
         mold = -(alpha ^ 2 + 2 * rho * alpha + rho ^ 2
                  * (1 - pold)) / alpha
      ELSEIF uold = utest THEN
         mainval1 = (2 * alpha + rho * (1 - pold)) * lam2old -
                    ((lam1old + etaold) * (1 - pold)) ^ 2 / (2*r)
         subval1 = lam1dot * (1 - pold) * (2 * (lam1old + etaold)
                   * pold - q)
         subval2 = (lam1old + etaold) * pold * (rho * (1 - pold)
                   + 2 * alpha * pold)
         subval3 = q * (rho * (1 - pold) + alpha * (1 + pold))
         mainval2 = subval1 + (lam1old + etaold) * (subval2
                    - subval3)
         mainval3 = (alpha ^ 2 + 2 * rho * alpha + rho ^ 2 *
                    (1 - pold)) * lam2old
         mold = (mainval2 / (2 * r) - mainval3) / mainval1
      ELSE
         mainval4 = alpha * lam2old + (1 - pold) * (q * s * pold
                    * (2 - pold) + 2 * r * s ^ 2 * (3 - 2 * pold))
                    / pold ^ 4
         subval4 = alpha * (q * s * pold * (3 - pold) + 4 * r
                   * s ^ 2 * (2 - pold))
         subval5 = rho * (1 - pold) * (2 * r * s ^ 2 + q * s
                   * pold) + (1 - pold) * (4 * r * s + q * pold)
                   * sdot
```

```
         mainval5 = subval4 + subval5
         mainval6 = (alpha ^ 2 + 2 * rho * alpha + rho ^ 2
                    * (1 - pold)) * lam2old
         mold = (mainval5 / pold ^ 3 - mainval6) / mainval4
      END IF

   IF mold < 0 THEN
      mold = 0
   ELSEIF mold > mbar THEN
      mold = mbar
   ELSE
      mold = mold
   END IF

END SELECT

pnew = pold + del * (mold - (alpha + mold) * pold)
lam2new = lam2old + del * ((rho + alpha + mold) * lam2old
          - (lam1old + etaold) * uold)


' Otain other trajectories

bet1bet2 = -sinarc

IF sintest < .02 THEN
   beta1old = 0
   beta2old = 0
ELSEIF bet1bet2 > 0 THEN
   beta1old = bet1bet2
   beta2old = 0
ELSE
   beta1old = 0
   beta2old = -bet1bet2
END IF

profit = profit + del * (w * s - h * xold - (r * uold ^ 2
         + q * uold + d) - c * mold)

hamilt = w * s - h * xold - (r * uold ^ 2 + q * uold + d)
         - c * mold + (lam1old + etaold) * (pold * uold - s)
         + lam2old * (mold - (alpha + mold) * pold)

xnew = xold + del * (pold * uold - s)

IF xnew < 0 THEN
   xnew = 0
ELSE
   xnew = xnew
END IF
```

```
      value2 = ABS(t - inc)
      IF value2 > .0005 OR flag < 1 THEN
         xold = xnew
         pold = pnew
         lam2final = lam2old
         lam2old = lam2new
      ELSE
         inc = inc + del
         lam2def = lam2old * (1 - pold)
         PRINT USING "###.###"; t, xold; pold; lam1old; lam2old; uold;
               mold; etaold; muold; beta1old; beta2old
         PRINT #1, USING "###.####"; t, xold; pold; lam1old; lam2old;
                   uold; mold; etaold; muold; beta1old; beta2old;
                   lam2def; c; profit; hamilt
         xold = xnew
         pold = pnew
         lam2final = lam2old
         lam2old = lam2new
      END IF

   NEXT t

   IF flag = 1 THEN
      EXIT DO
   ELSE
      lam2tf = lam2final
      PRINT "lambda2(0)="; lam2zero
      PRINT "lambda2(tf)="; lam2tf
      test = ABS(lam2tf - b)
      IF test < .1 THEN
         flag = flag + 1
      ELSE
         flag = flag
      END IF
   END IF
   count = count + 1

LOOP

salvage = a * xold + b * pold
PRINT "       "
PRINT "Number of iterations required before convergence ="; count + 1
PRINT "Profit (from operation) ="; profit
PRINT "Salvage value (of inventory/facility) ="; salvage
PRINT "TOTAL PROFIT ="; profit + salvage
PRINT "       "
CLOSE #1
```

APPENDIX I-B: COMPUTER PROGRAM - DECENTRALIZED APPROACH

```
REM ** THIS PROGRAM SOLVES THE CONTINUOUS-TIME PRODUCTION/MAINTENANCE **
REM ** PROBLEM USING THE GOAL-CORDINATION APPROACH                     **
REM **              ( The Decentralized Approach )                    **

' Program TS3D-1    Developed on 19/6/89   Revised on 23/6/89

REM ** MAIN PROGRAM **

' Dimension arrays as STATIC

DIM x(200), p(200), u(200), m(200), lam1(200), lam2(200), z(200)
DIM theta(200), mu(200), beta1(200), beta2(200), eta(200), lam2def(200)
DIM profitsub1(200), profitsub2(200), hamiltsub1(200), hamiltsub2(200),
DIM record(2), valuetheta(200)

' Open a file to write trajectories so that LOTUS-123 can use them
' to draw graphs

OPEN "a:ts3d-1.prn" FOR OUTPUT AS #1

' Input data

tf = 1                      'planning time horizon
del = .01                   'delta
a = 0                       'salvage value for each unit unsold
b = 10                      'salvage value of the facility
h = 1                       'holding cost
r = 2                       'coefficient of quadratic production cost term
q = 0                       'coefficient of linear production cost term
d = 0                       'constant production cost term
w = 8                       'revenue form selling a unit item
c = 2.5                     'maintenance cost
alpha = 1                   'obsolescence rate of process performance
s = 4                       'demand rate
mbar = 2                    'maximum level of maintenance
x(0) = 3                    'initial inventory level
p(0) = 1                    'initial process performance
rho = .1                    'constant continuous discount rate
thetaint = 5
lam2(0) = 0

' Print input data on screen

PRINT "     "
PRINT "tf="; tf, "final time"
PRINT "del="; del, "delta"
PRINT "a="; a, "salvage value for each unit unsold"
PRINT "b="; b, "salvage value of the facility"
```

```
PRINT "h="; h, "inventory holding cost"
PRINT "r="; r, "coefficient of quadradic production cost term"
PRINT "q="; q, "coefficient of linear production cost term"
PRINT "d="; d, "constant production cost term"
PRINT "w="; w, "revenue from selling a unit item"
PRINT "c="; c, "maintenance cost"
PRINT "alpha="; alpha, "obsolescence rate of process performance"
PRINT "s="; s, "demand rate"
PRINT "MBAR="; mbar, "maximum level of maintenance"
PRINT "x(0)="; x(0), "initial inventory level"
PRINT "p(0)="; p(0), "initial process performance"
PRINT "rho="; rho, "constant continuous discount rate"
PRINT "theta="; thetaint
PRINT "     "

' Initialization

kf = tf / del

FOR k = 0 TO kf STEP 1
   theta(k) = thetaint
NEXT k

' Loop for the main problem

DO

   ' Solve (MAIN) Subproblem

   lam2low = -10:   lam2high = 20      'low and high values of lam2zero
   flagmain = 0

   DO WHILE flagmain < 1

      ' Obtain non-singular and singular maintenance controls

      FOR k = 0 TO kf STEP 1
         sinarc = lam2(k) * (1 - p(k)) - c
         sintest = ABS(sinarc)

         SELECT CASE sintest
            CASE IS > .01
               IF sinarc < 0 THEN
                  m(k) = 0
               ELSE
                  m(k) = mbar
               END IF
            CASE ELSE
               mainval1 = alpha * lam2(k) + theta(k) * (1 - p(k))
               mainval2 = theta(k) * (alpha * (1 + p(k)) + rho
                          * (1 - p(k)))
```

```
            mainval3 = (alpha ^ 2 + 2 * rho * alpha + rho ^ 2
                       * (1 - p(k))) * lam2(k)
            m(k) = (mainval2 - mainval3) / mainval1

         IF m(k) < 0 THEN
            m(k) = 0
         ELSEIF m(k) > mbar THEN
            m(k) = mbar
         ELSE
            m(k) = m(k)
         END IF

      END SELECT

      ' Obtain other maintenance trajectories

      lam2def(k) = lam2(k) * (1 - p(k))
      betlbet2 = c - lam2def(k)

      IF betlbet2 > 0 THEN
         beta1(k) = betlbet2
         beta2(k) = 0
      ELSE
         beta1(k) = 0
         beta2(k) = -betlbet2
      END IF

      IF k = 0 THEN
         profitsub1(k) = 0
      ELSE
         profitsub1(k) = profitsub1(k - 1) + del * (theta(k) * p(k)
                         - c * m(k))
      END IF
      hamiltsub1(k) = theta(k) * p(k) - c * m(k) + lam2(k) * (m(k)
                      - (alpha + m(k)) * p(k))
      p(k + 1) = p(k) + del * (m(k) - (alpha + m(k)) * p(k))
      lam2(k + 1) = lam2(k) + del * ((rho + alpha + m(k)) * lam2(k)
                    - theta(k))

NEXT k

PRINT "     "
PRINT "lambda2(0)="; lam2(0)
PRINT "lambda2(tf)="; lam2(kf)

test = ABS(lam2(kf) - b)
IF test < .1 THEN
   flagmain = flagmain + 1
ELSE
   IF lam2(kf) > b THEN
      lam2high = lam2(0)
```

```
      ELSE
         lam2low = lam2(0)
      END IF
      lam2(0) = (lam2high + lam2low) / 2
   END IF

LOOP

salvagemain = b * p(kf)
profitmain = profitsubl(kf) + salvagemain

PRINT "     "
PRINT "! (MAIN)-subproblem has been solved."
PRINT "         p(0),p(.5),p(tf):"; p(0), p(50), p(kf)
PRINT "     "

' Transfer (MAIN)-solution to (PROD)-Subproblem by z(t)=p(t) and then
' solve (PROD)-Subproblem

etamin = 0: etamax = 20
flagprod = 0

DO WHILE flagprod < 1

   cycle = 0

   etalow = etamin + (etamax - etamin) / 4
   etamiddle = etamin + (etamax - etamin) / 2
   etaupper = etamin + 3 * (etamax - etamin) / 4

   DO WHILE cycle < 3

      FOR k = 0 TO kf STEP 1

         z(k) = p(k)
         laml(k) = -h / rho + (a + h / rho) * EXP(rho * (k * del
                   - tf))

         SELECT CASE x(k)
            CASE IS > 0
               IF cycle = 0 THEN
                  eta(k) = etalow
               ELSEIF cycle = 1 THEN
                  eta(k) = etamiddle
               ELSE
                  eta(k) = etaupper
               END IF

               utest = ((laml(k) + eta(k)) * z(k) - q) / (2 * r)
               IF utest > 0 THEN
                  u(k) = utest
                  mu(k) = 0
```

```
            ELSE
               u(k) = 0
               mu(k) = q - (laml(k) + eta(k)) * z(k)
            END IF
         CASE ELSE
            mu(k) = 0
            IF utest < s / z(k) THEN
               u(k) = s / z(k)
            ELSE
               u(k) = utest
            END IF
            eta(k) = (2 * r * u(k) + q) / z(k) - laml(k)

      END SELECT

      IF k = 0 THEN
         profitsub2(k) = 0
      ELSE
         profitsub2(k) = profitsub2(k - 1) + del * (w * s
                         - h * x(k) - (r * u(k) ^ 2 + q * u(k)
                         + d) - theta(k) * z(k))
      END IF
      hamiltsub2(k) = w * s - h * x(k) - (r * u(k) ^ 2 + q * u(k)
                      + d) - theta(k) * z(k) + (laml(k) + eta(k))
                      * (z(k) * u(k) - s)
      x(k + 1) = x(k) + del * (z(k) * u(k) - s)

   NEXT k

   salvageprod = a * x(kf)
   profitprod = profitsub2(kf) + salvageprod
   record(cycle) = profitprod
   cycle = cycle + 1

LOOP

PRINT "     "
PRINT "rec(L),rec(M),rec(U):"; record(0), record(1),record(2)
PRINT "etalow, etamiddle, etaupper: "; etalow, etamiddle, etaupper

diff1 = ABS(record(0) - record(1))
diff2 = ABS(record(1) - record(2))
diff3 = ABS(record(0) - record(2))
totdiff = diff1 + diff2 + diff3

IF totdiff < .5 THEN
   flagprod = flagprod + 1
   EXIT DO
ELSE
   flagprod = flagprod
END IF
```

```
      IF record(1) >= record(0) THEN
         IF record(1) >= record(2) THEN
            etamin = etalow
            etamax = etaupper
         ELSE
            etamin = etamiddle
            etamax = etamax
         END IF
      ELSE
         etamin = etamin
         etamax = etamiddle
      END IF

   LOOP

   PRINT "      "
   PRINT "! (PROD) has been solved."
   PRINT "      "

   ' Check convergence of theta

   epsilon = .2
   deviation = 0
   FOR k = 0 TO kf STEP 1
      valuetheta(k) = (laml(k) + eta(k)) * u(k)
      deviation = deviation + ABS(theta(k) - valuetheta(k))
   NEXT k

   avedeviation = deviation * del
   PRINT "Check convergency of theta:"
   PRINT "theta(0),theta(.5),theta(tf):"; theta(0), theta(50), theta(kf)
   PRINT "avedeviation, epsilon:       "; avedeviation, epsilon

   ' Update theta if necessary

   IF avedeviation > epsilon THEN
      FOR k = 0 TO kf STEP 1
         theta(k) = theta(k) + .5 * (valuetheta(k) - theta(k))
      NEXT k
   ELSE
      EXIT DO
   END IF

LOOP

' Print out optimal solution on screen and on disk

PRINT "      "
PRINT "      "
PRINT "! Congratulation! "
PRINT "! The whole problem has been solved."
```

```
PRINT "! Printing begins now."
PRINT "     "
PRINT "     "
PRINT SPC(5); "t"; SPC(5); "x"; SPC(5); "p"; SPC(5); "lam1"; SPC(3);
      "lam2"; SPC(5); "u"; SPC(6); "m"; SPC(5); "eta"; SPC(5); "mu";
      SPC(3); "beta1"; SPC(2); "beta2"
PRINT "     "

inc = 1
FOR k = 0 TO kf STEP inc
   t = k * del
   profit = profitsub1(k) + profitsub2(k)
   hamiltonian = hamiltsub1(k) + hamiltsub2(k)
   PRINT USING "###.###"; t; x(k); p(k); lam1(k); lam2(k); u(k); m(k);
         eta(k); mu(k); beta1(k); beta2(k)
   PRINT #1, USING "###.####"; t; x(k); p(k); lam1(k); lam2(k); u(k);
         m(k); eta(k); mu(k); beta1(k); beta2(k); theta(k); lam2def(k);
         c; profit; hamiltonian
NEXT k

profitoperate = profitsub1(kf) + profitsub2(kf)
salvagetotal = salvagemain + salvageprod

PRINT "    "
PRINT "Profit (from operation) ="; profitoperate
PRINT "Salvage value ="; salvagetotal
PRINT "Profit from (MAIN)-subproblem ="; profitmain
PRINT "Profit from (PROD)-subproblem ="; profitprod
PRINT "TOTAL PROFIT ="; profitoperate + salvagetotal
PRINT "    "

CLOSE #1
```

BIBLIOGRAPHY

Abad, P.L., "Approach to Decentralized Marketing-Production Planning", International Journal of Systems Science, 13, (1982), 227-235.

----, "Two-level Algorithm for Decentralized Control of a Serially Connected Dynamic System", International Journal of Systems Science, 16, (1985), 619-624.

Arora, S.R., and Lele, P.T., "A Note on Optimal Maintenance Policy and Sale Date of a Machine", Management Science, 17, (1970), 170-173.

Baker, K., and Peterson, D., "An Analytic Framework for Evaluating Rolling Schedules", Management Science, 25, (1979), 341-351.

Bensoussan, A., Hurst, E., and Näslund, B., Management Applications of Modern Control Theory, American Elsevier: New York, (1974).

Cho, D.I., and Parlar, M., "A Survey of Maintenance Models", Working Paper 329, Faculty of Business, McMaster University, Hamilton, Ont., Canada, (1989).

-----, "A Survey of Maintenance Models for Multi-Unit Systems", European Journal of Operational Research, 51, (1991), 1-23.

-----, Abad, P., and Parlar, M., "Optimal Production and Maintenance Decisions for a System Experiencing Age-dependent Deterioration", (1992)

Gaimon, C.M., and Thompson, G.L., "Optimal Preventive and Repair Maintenance of A Machine Subject to Failure", Optimal Control Application and Methods, 5, (1984), 57-67.

------, "A Real-Time Solution for Preventive and Repair Maintenance", Optimal Control Applications & Methods, 10, 1989.

Hartl, R.F., Sethi, S.P., and Vickson, R.G., "A Survey of The Maximum Principle for Optimal Control Problems with Pure State Constraints", Research Report, (1987).

Hestenes, M.R., Calculus of Variations and Optimal Control Theory, John Wiley & Sons Publishing, (1966).

Holt, C.F., Modigliani, F., Muth, J., and Simon, H., Planning Production, Inventories and Work Force, Englewood Cliffs: Prentice Hall, (1960).

Johnson, C.D., and Gibson, J.E., "Singular Solutions in Problems of Optimal Control," IEEE Transactions on Automatic Control, 8, 4-15, (1963).

Kelley, H.J., Kopp, R.E., and Moyer, H.G., "Singular Extremals", Topics in Optimization, G. Leitmann, Ed., Academic Press, N.Y., (1967).

Maurer, H., "Numerical Solution of Singular Control Problems Using Multiple Shooting Techniques", J. of Optimization Theory and Applications, 18, 2, 235-257, (1976).

Näslund, B., "Simultaneous Determination of Optimal Repair Policy and Service Life", The Swedish Journal of Economics, 68, (1966), 63-73.

Pierskalla, W.P., and Voelker, J.A., "A Survey of Maintenance Models: The Control and Surveillance of Deteriorating Systems", Naval Research Logistics Quarterly, 23, (1976), 353-388.

Pontryagin, L.S., Boltyanskii, V.G., Gamkrelidze, R.V., and Mishchenko, E., The Mathematical Theory of Optimal Processes, John Wiley & Sons Publishing, (1962).

Roberts, S.M., and Shipman, J.S., Two Point Boundary Value Problems: Shooting Methods, New York: American Elsevier, (1972)..

Russak, I.B., "On Problems with Bounded State Variables", Journal of Optimization: Theory & Applications, 5, (1970), 114-157.

Sarma, V.V.S., and Alam, M., "Optimal Maintenance Policies for Machines Subject to Deterioration and Intermittent Breakdowns, IEEE Transactions on System, Man, and Cybernetics, SMC-5, (1975), 396-398.

Sethi, S. P., "Optimal Control of Vidale-Wolfe Advertising Model, Operations Research, 21, (1973), 998-1013.

-----, and Thompson, G.L., Optimal Control Theory: Applications to Management Science, Boston: Martinus Nijhoff Publishing, (1981).

Singh, M.G., Dynamical Hierarchical Control, North-Holland Publishing Co., (1980).

Sprzeuzkouski, A.Y., "A Problem in Optimal Stock Management", Journal of Optimization: Theory & Applications, 1, (1967), 232-241.

Tapiero, C.S., "Continuous Quality Production and Machine Maintenance", Naval Research Logistics Quarterly, 33, (1986), 489-499.

Thompson, G.L., "Optimal Maintenance Policy and Sale Date of a Machine", Management Science, 14, (1968), 543-550.

PART II

INVENTORY SYSTEMS FOR REPAIRABLES

CHAPTER 5 INTRODUCTION

5.1 Description of Systems

Importance of the study in the field of inventory theory for repairables has been rapidly realized by many researchers in the recent years. In many supply systems involving spare parts, repairing failed items (if possible) certainly has cost advantages over purchasing new ones. Especially in the military, repairable items usually account for more than the half of the inventory investment (Schrady [1967], Sherbrooke [1968], and Muckstadt [1973]).

In repairable-item inventory systems with returns, two types of inventories (the serviceable inventory and the repairable inventory) are carried. The serviceable inventory is depleted by demand for serviceable units kept in the storage facility and it is replenished by repairing repairable units available in the repair facility and/or purchasing new serviceable units from outside. The repairable inventory, on the other hand, is depleted by repairing and/or junking repairable units and is replenished by returns of previously demanded units by customers to the repair facility. The decision maker must determine the optimum quantities of purchasing, repairing, and junking based on such information available to him as the serviceable and

repairable inventory levels, the size of demand, and the size of returns. Figure II-1 shows a schematic representation of the typical repairable-item inventory system with returns.

The inventory theory for repairables has the potential for greater complexity and challenge than the inventory theory for consumables because the former has a higher degree of dimensions. These added dimensions, however, require more complex analysis of the system and restrict use of alternative solution methods.

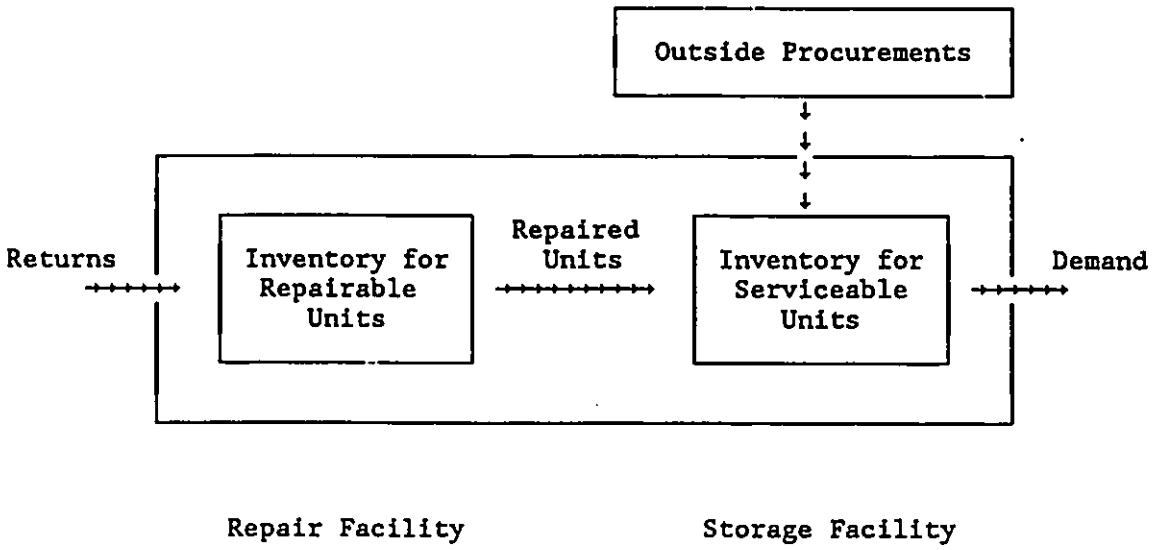


Figure II-1: A Schematic Representation of A Repairable-Item Inventory System with Returns.

5.2 Literature Search

The field of repairable-item inventory systems with returns has been gradually recognized by a number of researchers and practitioners.

Based on assumptions made and objectives of systems considered, the literature related to repairable-item inventory systems with returns can be classified into many possible ways. Various ways of classifying the literature in that field and the categorization of each of the surveyed articles are shown in Appendix II-A.

Phelps [1962] and Veinott [1966] considered the situation in which the demand process for serviceable units and the return process of repairable units are perfectly correlated. This means that in any given period a known proportion of the demand for serviceables are returned to the repair facility for repair. They both utilized a periodic review policy where the inventory levels are reviewed periodically. Their models are characterized by two state variables, the levels of serviceables and repairables, at each period. In Phelps' model, four decisions (purchase serviceables, repair repairables, junk serviceables, and junk repairables) are made at the beginning of each period. Allowing for the case of lost sales, Phelps has examined the steady-state solution to the two-dimensional problem and shown that the optimal policy is described by seven regions in the state plane. Although a determination of the region boundaries is quite complex, Phelps' model provides some useful and interesting properties of an optimal solution. Veinott [1966] modified Phelps' model by considering the case of backorders for unsatisfied demands and by deleting the junking decisions. He has shown the structure of an optimal policy, which is specified by three regions in the state plane, without determining the boundaries of the regions.

Schrady [1967] considered a deterministic inventory system in which the demand and return rates are fixed constants. Treating the serviceable and repairable inventories as two independent parts, he developed closed-form formulae (i.e., modified EOQs) for optimal procurement and repair batch quantities.

Allen and D'Esopo [1968] and Simpson [1970] treated the repairable returns as additional random uncontrollable replenishments to the serviceable inventory so that inventory theory for consumables can be used to calculate the optimum procurement quantity. In the model of Allen and D'Esopo, a unit demanded is returned for repair with probability of p while it is non-repairable (so is discarded) with probability of 1-p. The model assumes a fixed repair time, a fixed replenishment lead time, and a possibility of backorders. A continuous ordering policy of the (r,Q) type (i.e., reorder point - order quantity) is followed for the analysis of the model. Simpson [1970], on the other hand, used a fixed periodic review in which the net inventory is brought up to a maximum level whenever an order for new stock is placed (i.e., order up to level). He allowed stochastic demand, repair output and lead time, but restricted his analysis to the situation where the mean demand rate is greater than the mean repair rate. The optimum order quantity was determined by successive approximations which was then verified by simulation methods.

Brown et al. [1971] considered a discrete time inventory system with a positive delivery lag in which demands in successive periods are dependent on one another. They formulated the problem as a single state inventory model and showed a way to calculate the optimal order-up-to

level. An important contribution of their paper is a possible application of the model to a special case of repairable-item inventory models, especially, to the one mentioned in Allen and D'Esopo [1968], in which a portion of demands generated are returned for repairs.

Simpson [1972][1978] examined a repairable-item inventory problem similar to the one in Phelps [1962] and in Veinott [1966]. He formulated a two-dimensional dynamic programming model under a periodic review policy. Simpson assumed, as Veinott did, backlogging of the unsatisfied demand for serviceable units, instantaneous delivery of purchased units, and instantaneous repair of repairable units. However, Simpson's model differs in the following sense: first, the complete dependency between demand and returns was removed, but a joint probability density function between demand and return processes, which is known for each period, was allowed; second, a more definitive solution structure was developed and then proved to be optimal. Simpson used a backward dynamic programming technique with the Kuhn-Tucker saddle point theorems to describe the n-period solution, which is completely determined by three time-dependent constants: the repair-up-to-level, the purchase-up-to-level, and the scrap-down-to-level. The solution structure consists of seven regions describing the optimal number of units to purchase, repair and scrap. The methodology employed by Simpson is not applicable to the 'lost sales' case.

A similar repairable-item inventory problem has been considered by Heyman [1977]. In his model, returned units can either be repaired or be disposed of at a certain net receipt per unit. In addition, a maximum level (N) of the serviceable inventory is chosen, and returns

that would raise the serviceable inventory level above N are disposed of. The objective here is to determine the optimum N^* which minimizes the total of repair, disposal, purchasing and holding costs. The model also assumes a continuous review policy, negligible repair and procurement lead times, and independent demand and return processes. Heyman showed that the above problem is exactly the same as the problem of a single-server queue with a maximum system capacity of N, and then reformulated it as a queueing design model with a finite system capacity. He has obtained an exact solution for the Markovian demand and return processes (i.e., M/M/1/N) while he has used diffusion methods to obtain an approximate solution for more general cases of demand and return processes (i.e., G/G/1/N).

As an extension of the typical repairable-item inventory models with returns, Heyman [1978] considered an inventory system with both positive and negative demands. Here, because of the positive and negative stock fluctuations, items can not only be purchased from a central warehouse but also be sent back at an additional fixed cost to the central warehouse when inventory levels for serviceable units become large. Assuming a fixed holding cost per item per unit time, instantaneous shipments and returns, and Poisson demand and return processes, Heyman showed that an optimal two-critical-number (a,b) policy which minimizes the asymptotic cost rate of the system exists, where b is the "trigger level" for sending a quantity of b-a units back to the central warehouse ($0<a<b$).

As Heyman [1977][1978] did, Isaac [1979] and Muckstadt and Isaac [1981] assumed independent Poisson processes between demand for

serviceables and returns of repairables. A significant distinction of their models from the others, however, is that they allowed for a constant procurement lead time and stochastic repair times. They first considered a single location repairable inventory model in which the stationary return rate is less than the stationary demand rate. It was further assumed in the models that unsatisfied demands are backordered and that the ordering rule follows a continuous review (r,Q) procurement policy, in which an order of Q units is placed when the inventory position for serviceable units drops to r. The models develop a normal approximation to the stationary distribution of net inventory which is then used, under the optimality criterion of total expected cost of procurements, holding and backorders, to determine optimal values of r and Q. Later, they studied a two-echelon system in which a central warehouse having both repair and storage facilities supports a number of retailers only having storage facilities. Assuming that the retailers follow (s-1,s) continuous review ordering policies while the central warehouse maintains the (r,Q) procurement policy, they proposed an algorithm for determining the policy parameter values at each retailer and the central warehouse. Finally, Isaac [1979] considered a case where the stationary return rate is greater than the stationary demand rate. In this situation, outside procurement of new units is no longer necessary since in the long-run repair of returned units alone is sufficient to meet demand for serviceable units. This implies that the repairable inventory system now becomes a single-server repair system with exponential service times (i.e., no procurement decision), which can be easily analyzed by queueing theory.

Recently, Albright and Soni [1988] considered a somewhat complicated repairable-item inventory model with returns. In their model, the system consists of a storage facility and a repair facility with a finite number of identical repairmen having exponential repair times. Demand for serviceable units occurs whenever any item being leased fails or new customers are generated. An item is returned when a customer brings a failed item for repair or a lease expires. Any of the above events is assumed to occur independently according to Poisson processes with different mean rates. The system is assumed to have as many as M items while the total number of customers in the system is limited to of size N. A fixed proportion of failed items is assumed to be irreparable, and thus new items must be brought from an outside source. In such a case, a continuous ordering policy of the (s,S) type is followed. They formulated the problem as a continuous-time Markov process with multidimensional states and presented an approximate procedure for calculating the stationary distribution of the process. The approximation method was shown to be simple, non-iterative, easy to understand, and reasonably accurate. Appendix II-B provides a brief description of each of the four models from the literature, directly related to the proposed research, and a comparison among them in various categories.

5.3 Contributions

Part II of the thesis contributes to the theory of inventory control for repairables the following:

(1) The proposed model incorporates several important features

mentioned in the previous literature. These include a periodic review policy, a two-dimensional discrete state space, stochastic demand and return processes, 'lost sales' for unsatisfied demand, negligible procurement and repair times, both finite-time and infinite-time planning horizons, and three types of decisions (purchase, repair and/or junk) at each period. In addition, it has the following distinct features. First, the return process is dynamic in terms of the demands in the current and previous periods. This means that random proportions of the serviceable units demanded in the current and previous periods are returned to the system during the current period. Second, a more realistic cost function, i.e., set-up costs for procurement and repair, is included. The set-up costs include a fixed cost to place a purchasing order for serviceable units, and a fixed cost to set up the repair facility to repair repairable units.

(2) A completely different solution approach is employed for the analysis of the proposed problem. The methodology employed in this paper is use of stochastic dynamic programming in Markov Chains, sometimes referred as a Markov Decision Process (MDP), with a decision space reduction procedure developed throughout the analysis of the problem to eliminate unnecessary decisions.

(3) While the method of successive approximations is used for solving the problem, two acceleration techniques, the error bounds approach and SDD (State Decomposition by Dimension), are introduced for speeding up the convergence of successive approximations. The decomposition technique approximates the optimal solution to repairable-item inventory problems even with a large number of states.

CHAPTER 6 REPAIRABLE-ITEM INVENTORY MODEL WITH RANDOM RETURNS

6.1 Statement of Problem

Consider an inventory system with a single type of repairable item. As seen in Figure II-2, the system consists of a storage facility for serviceables and a repair facility for repairables. Random demand for serviceable units occurs independently while a return occurs when a customer brings a unit previously demanded to the repair facility (e.g. customer service) for a refund. Random proportions of serviceable units demanded in the current and previous periods are assumed to be returned during the current period. Note that units demanded a long time ago may be returned now, and that the return of a unit to the system does not necessarily generate a demand for a serviceable unit. Thus the return process in the current period is dependent upon the demand processes in the current and previous periods. We call such a relation a 'dynamic' (period-to-period) correlation between the demand and return processes. Outside replenishments (i.e., purchase) of serviceable units are necessary from time to time because the number of returned units may be less than that of serviceable units demanded. No other assumptions are made about the return process. At the beginning of each period, the serviceable and repairable inventories are reviewed (i.e., periodic review) and then proper decisions (purchase, repair and/or junk) are made. Only returned units waiting for repair in the repair facility can be junked. Junking is necessary in the situation where holding and

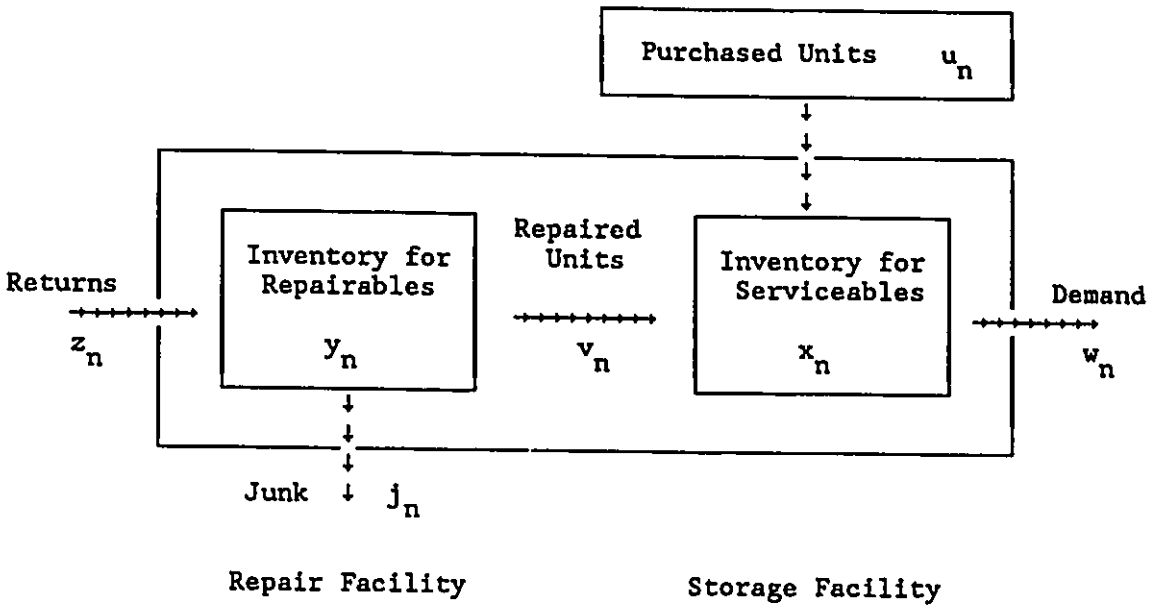


Figure II-2: A Discrete-Time Repairable-Item Inventory System with Dynamic Returns.

repairing a returned unit is more costly than junking it and purchasing a new one. Returned units will stay in the repair facility until they are to be either junked or repaired to become additional replenishments to the serviceable inventory. The situation illustrated above can be seen in real life. For an example, we may consider a distributing store having its own storage and repair facilities. Demand for an item may occur at any time (e.g. any day) within a period (e.g. a week) when a customer enters into the store. If the item which the customer wants to buy is not available in the store, he or she usually goes to another store, indicating a situation of lost sales. Upon the sale of a unit item to the customer, the store provides a warranty so that the customer

can return the purchased item (if he is not satisfied with the item or he has found it defective) to the store within a certain time interval (e.g. 2 weeks) for a full refund of money. The customer may or may not purchase a new unit, implying that the return of a unit does not necessarily generate a demand for a serviceable unit. The returned item is sent to the repair facility for repair. At each periodic review point (e.g. the beginning of each week), the store manager must make appropriate decisions to purchase, to repair, and to junk so that the total of purchase, repair, storage and shortage (or penalty) costs is minimized.

6.2 Assumptions and Notation

We make the following assumptions.

(1) Time is treated as a discrete variable ($n = 0,1,2,\ldots$).

(2) A periodic review policy is followed such that the system is reviewed at each and every period.

(3) Demands for serviceables are independent, identically distributed (i.i.d.) nonnegative random variables.

(4) All demand not immediately satisfied is lost (i.e., lost sales).

(5) In each period, random fractions of serviceable units (currently and previously demanded) are returned.

(6) The return of a repairable unit may not necessarily generate a demand for a serviceable unit.

(7) The return status of each serviceable unit demanded is represented by Bernoulli random variable.

(8) Purchase and repair times are negligible.

(9) Each of the storage and repair facilities has a limited storage capacity, implying that there is an upper limit on the number of units that can be handled by each facility.

Assumptions (1) to (6) have already been discussed. Assumption (7) will be discussed later in detail. Assumption (8) may not be realistic in general; however, it can be justified by some real life situations. If distributing stores having their own storage and repair facilities are controlled by a regional headquarter with a central warehouse (e.g. Consumers Distributing), order deliveries from the central warehouse to the stores can be made as quickly as possible (which results in short replenishment times). Also, a distributing store usually performs only minor repairs (which take very short times) such as reorganizing contents of a box and sealing the box. If returned items require major repairs, then they are usually sent to the central warehouse. Assumption (9) can be easily justified without any questions because it explains a real life resource constraint. But, what would happen if the storage facility or the repair facility reached its maximum storage capacity? If the storage facility for serviceables reached its maximum capacity, then neither purchasing nor repairing could be done until the serviceable inventory is depleted by some demand. If the repair facility reached its maximum capacity, however, then units that are returned to the facility cannot be entered and are sent to another store.

We define the following notation.

x_n : Serviceable inventory at the beginning of period n.

y_n : Repairable inventory at the beginning of period n.

$\bar{X}$: Maximum capacity of the storage facility.

$\bar{Y}$: Maximum capacity of the repair facility.

u_n : Number of serviceables purchased at the beginning of period n.

v_n : Number of repairables repaired at the beginning of period n.

j_n : Number of repairables junked at the beginning of period n.

w_n : Demand for serviceable units in period n, where $\omega_n=E(w_n)$.

z_n : Repairable units returned in period n.

θ_i : Bernoulli random variables with parameters δ_i representing the return status of each serviceable unit demanded i periods ago, where $\delta_i=E(\theta_i)$ $(0\leq\delta_i\leq 1;\ i=0,1,2,\ldots)$.
i.e., $P(\text{No return})=P(\theta_i=0)=1-\delta_i$ and $P(\text{Return})=P(\theta_i=1)=\delta_i$.

α : Discount factor $(0\leq\alpha<1)$.

We consider the following cost factors.

(1) The costs of purchasing serviceable units are linear in the number of units purchased (i.e., a constant cost of c(>0) per unit). Also, a fixed ordering charge C(≥0) per replenishment is assumed. Thus, the total cost of purchasing u units of serviceables is: C+cu for u>0 and 0 for u=0.

(2) The costs of repairing repairables are linear in the number of units repaired (i.e., a constant cost of r(>0) per unit). However, there is a fixed cost R(≥0) to set up the repair facility. Thus, the total cost of repairing v units is: R+rv for v>0 and 0 for v=0.

(3) A cost (or revenue) of p per unit from junking repairables is incurred. If $p>0$, a cost occurs. If $p<0$, a salvage value occurs.

(4) The storage (or holding) cost is linear in the number of units either in the serviceable inventory or in the repair facility waiting for repair. Constant charges of f and h per unit per period for repairable units in the repair facility and for serviceable units in the storage facility, respectively, are incurred, where $f \leq h$.

(5) A fixed shortage (or penalty) cost of b for each unit of demand unsatisfied is incurred (i.e., the case of lost sales).

(6) The cost of repairing a repairable unit is assumed to be less than the net cost of junking the unit and purchasing a new serviceable unit (i.e., $r<c+p$). Otherwise, repairing is never optimal.

6.3 Model Formulation

The proposed model contains two state variables and three control (decision) variables.

State variables : Serviceable inventory, x_n
Repairable inventory, y_n

Control variables: Number of serviceable units purchased, u_n
Number of repairable units repaired, v_n
Number of repairable units junked, j_n

The proposed repairable-item inventory problem consists of two state equations, an equation representing the return process and a system constraint as described below.

Number of serviceable units at beginning of period n+1

= max [0, # of serviceables at n + # of serviceables purchased in n + # of units repaired in n - demand in n]

or

$$x_{n+1} = \max [\, 0,\ x_n + u_n + v_n - w_n \,]. \tag{6-1}$$

Number of repairable units at beginning of period n+1

= # of repairables at n - # of units repaired in n - # of units junked in n + # of units returned in n

or

$$y_{n+1} = y_n - v_n - j_n + z_n. \tag{6-2}$$

Note that Equacion (6-2) does not satisfy the requirement that the repairable inventory be always nonnegative (i.e., $y_n \geq 0$ for n=0,1,2,...). However, this requirement will be met by the system constraint (6-6).

The current return process is characterized by Bernoulli random variables θ_i (with parameters δ_i) representing the return status of each serviceable unit demanded i periods ago and the demands in the current and previous periods (which are known with certainty).

Number of units returned in period n

= # of serviceables demanded in period n and returned in period n

+ # of serviceables demanded in period n-1 and returned in period n

+ # of serviceables demanded in period n-2 and returned in period n

+ . . . . .

+ # of serviceables demanded in period n-i and returned in period n

+ . . . . .

Let $\psi_{i,n}$ be the number of serviceables demanded in period n-i (or i periods ago) and returned in period n. Thus it can be written as

$$\psi_{i,n} = \sum_{k=1}^{w_{n-i}} \theta_i, \qquad (6\text{-}3)$$

where θ_i are random variables each having value 0 with probability of $1-\delta_i$ or 1 with probability of δ_i, and w_{n-i} is the demand for serviceable units in period n-i. Note that $\psi_{i,n}$ are independent binomial random variables with parameters (w_{n-i},δ_i) since θ_i are Bernoulli random variables with parameters δ_i.

Using the notation defined above, we may write

$$z_n = \psi_{0,n} + \psi_{1,n} + \psi_{2,n} + \ldots\ldots = \sum_{i=0}^{\infty} \psi_{i,n}. \qquad (6\text{-}4)$$

Realistically, however, there is a limited refund period (M) in which the customer can bring his or her unit to the repair facility for a refund. In other words, the customer can return the unit only within

M periods from the time of purchasing that unit. Thus, the number of repairable units returned in period n is the sum of M+1 terms, each of which represents the number of serviceables demanded i periods ago and returned in period n.

$$z_n = \psi_{0,n} + \psi_{1,n} + \psi_{2,n} + \cdots + \psi_{M,n} = \sum_{i=0}^{M} \psi_{i,n}. \qquad (6\text{-}5)$$

Different types of return behaviour of returnable units and the corresponding forecasting methods, including one similar to (6-5), are discussed in Kelle and Silver [1989].

Finally, the system constraint implies that the total of units repaired in period n and units junked in period n must not exceed the number of repairable units in the repair facility at the beginning of period n.

$$v_n + j_n \leq y_n. \qquad (6\text{-}6)$$

CHAPTER 7 MARKOV DECISION PROCESS (MDP) APPROACH

7.1 Description of MDP

Consider a process that is observed at discrete time points $n=0,1,2,\ldots,N$ ($N\to\infty$ for the infinite-horizon problem) and that is classified into one of a number of possible states. The set of possible states is assumed to be countable and thus can be labeled by positive integers $(1,2,\ldots)$. After each observation, an action from a set of all possible alternatives (actions) labeled $1,2,\ldots,K$ (which is assumed to be finite) is chosen. If action a is chosen when the process is in state i at time n, the following two things will happen. First, an immediate expected cost $C(i,a)$ will be incurred. Second, the state of the system at the next observed time will be determined according to the transition probabilities $P_{ij}^{n}(a)$, i.e., the probabilities that the system will be in state j at time n+1 given that action a is chosen in state i at time n. Such a stochastic sequential decision process (with a sequence of observed states and sequence of decisions made) is called a Markov decision process (MDP). The objective is to choose a policy which optimizes the performance of the system over the planning horizon.

A Markov decision process and the corresponding policy are said to be non-stationary if the transition probabilities depend upon time (i.e., $P_{ij}^{n}(a)\neq P_{ij}^{n+1}(a)$ for some i,j,n). A Markov decision process is stationary if the transition probabilities are independent of the time in which the decision is made (i.e., $P_{ij}^{n}(a)=P_{ij}^{n+1}(a)=P_{ij}(a)$), and the

corresponding policy for an infinite-horizon problem is said to be stationary since it uses an identical decision rule in each period. For finite-horizon problems, the transition probabilities can be either stationary or non-stationary for existence of optimality. For infinite-horizon problems, however, the transition probabilities must be stationary for stationary policies to be optimal (Puterman [1990]).

MDP theory including existence of optimality, characterization, and computational results is based mainly on Bellman's equation referred as the principle of optimality (Bellman [1957]). In other words, many of the concepts in dynamic programming (DP) have very similar interpretations with those in MDPs. Thus some computational methods used in MDP are analogous to the DP algorithm. An analysis of Markov decision models usually does not result in an analytic (or closed-form) solution; however, it provides the following: (a) an optimality equation which characterizes the performance of the objective function, (b) efficient computational algorithms for determining optimal or near-optimal policies, and (c) the form of an optimal policy if it exists.

Importance and potential applicability of Markov decision processes have been realized by a number of authors. They include Bellman [1957], Bellman and Dreyfus [1962], Howard [1960], Manne [1960], Derman [1962], Bertsekas [1976][1987][1989], Puterman and Shin [1979], Porteus [1980], Heyman and Sobel [1982], Puterman [1990].

7.2 Mathematical Statement

In this section, we formulate the dynamic control problem defined in Chapter 6 as a Markov decision problem. We recall the

control problem as follows. A system with two types of inventories, the serviceables and the repairables, faces random demand for serviceables whereas random proportions of the serviceables demanded in the current and previous periods are returned in the current period. At the beginning of each discrete time period, the serviceable and repairable inventories are reviewed and then proper decisions to purchase, to repair, and/or to junk are made. Here, we are interested in analyzing two issues simultaneously: (a) a tradeoff between the costs associated with having serviceables on hand to meet random demand and penalty costs associated with 'lost sales' due to inability to satisfy the demand, and (b) a tradeoff between the costs associated with having repairables on hand (that are waiting for repair) and the costs associated with junking repairables and then purchasing new serviceables later from outside.

7.2.1 Description of the System Spaces

Time is treated as a discrete variable.

Stage (or decision epochs): n $\quad n=0,1,\ldots,N-1$, where $N\leq\infty$.

The serviceable and repairable inventories (x,y), referred as "discrete state space" have upper capacity limits of $\tilde{X}$ and $\tilde{Y}$, respectively.

State : (x,y) $\quad x=0,1,2,\ldots,\tilde{X}$; $y=0,1,2,\ldots,\tilde{Y}$.

At each period, proper decisions on the number of units to purchase (u), to repair (v), and to junk (j) are made.

Decision : (u,v,j) $\quad u=0,1,2,\ldots$; $v=0,1,2,\ldots$; $j=0,1,2,\ldots$

The state (x_n,y_n) is an element of a state space S, the control (u_n,v_n,j_n) is an element of a control space U, and the random demand w_n is an element of a disturbance space D. If decision (u,v,j) is made when the system is in state (x,y), the following cost per stage will be incurred.

State	Decision	Purchase	Repair	Junk
(x,y)	(u,v,j)	$\begin{cases} C+cu & \text{if } u>0 \\ 0 & \text{if } u=0 \end{cases}$	$\begin{cases} R+rv & \text{if } v>0 \\ 0 & \text{if } v=0 \end{cases}$	pj

Hold (serviceables)	Hold (repairables)	Lost sales
$\max\{0,h(x+u+v-w)\}$	$\max\{0,f(y-v-j)\}$	$\max\{0,b(w-x-u-v)\}$

Thus the expected cost incurred during the next transition if the system is in state (x,y) and decision (u,v,j) is made will be denoted by:

$$\begin{aligned}\phi_{(x,y)}(u,v,j) &= \phi(x,y,u,v,j,w) \\ &= \sigma_u C + cu + \sigma_v R + rv + pj + f\max\{0,y-v-j\} \\ &\quad + h\max\{E[0,x+u+v-w]\} + b\max\{E[0,w-x-u-v]\}, \end{aligned} \tag{7-1}$$

where $\sigma_u = \begin{cases} 1 & \text{if } u>0 \\ 0 & \text{if } u=0 \end{cases}$ and $\sigma_v = \begin{cases} 1 & \text{if } v>0 \\ 0 & \text{if } v=0. \end{cases}$

Given an initial state (x_0,y_0), we want to determine a policy $\pi=(\mu_0,\mu_1,\ldots,\mu_{N-1})$ where $\mu_n: S\to U$, for all $(x_n,y_n)\in S$, which minimizes the expected total discounted cost

$$J_\pi(x_0,y_0) = \underset{w_n}{E}\left\{ \alpha^N \phi_N(x_N,y_N) + \sum_{n=0}^{N-1} \alpha^n \phi(x_n,y_n,u_n,v_n,j_n,w_n) \right\} \tag{7-2}$$

subject to the constraints

$$x_{n+1} = \max [0, x_n + u_n + v_n - w_n] \tag{7-3a}$$

$$y_{n+1} = y_n - v_n - j_n + z_n \tag{7-3b}$$

$$z_n = \sum_{i=0}^{M} \psi_{i,n} \tag{7-3c}$$

$$v_n + j_n \le y_n. \tag{7-3d}$$

Note that ϕ:SxUxD→R is a real-valued function which is given and $\phi_N(x_N, y_N)$ is a terminal cost incurred at the end of the process. The discount factor ($0 \le \alpha < 1$) can be interpreted as equal to $1/(1+i)$, where i is the interest rate. With a discount factor α, the present value of one unit of cost n periods in the future is equal to α^n. Initial values of the state variables, x_0 and y_0, are not included in (7-3a) and (7-3b) because the MDP yields an optimal decision for each and every possible state (i.e., each pair of x_0 and y_0 values). Also, y_{n+1} in (7-3b) always takes a nonnegative value as long as (7-3d) is satisfied.

7.2.2 System Rules and Decision Rules

The system is invalid (i.e., decisions not permitted) if one or more of the following conditions hold.

$$\text{(a) } v_n+j_n>y_n, \quad \text{(b) } x_n+u_n+v_n>\bar{X}, \quad \text{(c) } \Delta x_n>\bar{X}, \quad \text{(d) } \Delta y_n>\bar{Y}, \tag{7-4}$$

where $\Delta x_n = u_n+v_n$ and $\Delta y_n = v_n+j_n$. We call these 'system rules'. Note that (a) results from the system constraint, that is, the sum of the repaired and junked units must not exceed the repairables. The remaining inequalities indicate possible situations of an invalid system due to the limited capacity of either the serviceable facility (i.e., (b) and (c)) or the repair facility (i.e., (d)). Inequalities (c) and (d) are

more relaxed than (b) and (a), respectively. The former will be used to reduce the number of decisions to be considered in the MDP while the later will be used to eliminate some decisions at certain states of the system.

Also, the following two types of decisions will never be optimal: (1) (u>0,v>0,j>0) and (2) (u>0,v=0,j>0), that is the case when both positive purchasing and positive junking actions occur. The proof is given in Lemma 7.2. We call these 'decision rules', which will be used to reduce the total number of decisions considered in the MDP.

7.2.3 Reduction in Decision Space

The Markov decision process (MDP) enables us to determine an optimal policy in a dynamic decision environment. In the proposed model, however, we deal with multi-dimensional state and decision spaces, which usually results in a large number of states and actions. Thus any reduction in the number of states and/or the number of decisions to be considered in the MDP will relieve a burden of excessive work on computation. The decision and system rules described in Section 7.2.2 do not reduce the number of states, but they may reduce the number of decisions. In this section, we will determine the actual number of decisions (after elimination) to be considered in the MDP and find out how much reduction in the decision space is to be achieved.

Proposition 7.1: For the two-dimensional state MDP with upper bounds $\bar{X}$ and $\bar{Y}$ for the serviceable and repairable inventories, respectively, the total number of possible states is $(\bar{X}+1)(\bar{Y}+1)$.

Proof: The state variables x and y of state (x,y) may take any one of the following integers, $(0,1,\ldots,\bar{X})$ and $(0,1,\ldots,\bar{Y})$, respectively. Since x may take any one of $(\bar{X}+1)$ different integers and y may take any one of $(\bar{Y}+1)$ different integers, the total number of states is $(\bar{X}+1)(\bar{Y}+1)$.

Proposition 7.2: For the two-dimensional state, three-dimensional decision MDP with upper bounds $\bar{X}$ and $\bar{Y}$ for the state variables x and y, respectively, the upper bounds of the decision variables u, v, and j can be set to be $\bar{X}$, $\min(\bar{X},\bar{Y})$, and $\bar{Y}$, respectively.

Proof: From the system rules (c) and (d) in Section 7.2.2, u may take any one of nonnegative integers $(0,1,2,\ldots,\bar{X})$, v may take any one of the following nonnegative integers $(0,1,2,\ldots,K)$, and j may take any one of the following nonnegative integers $(0,1,2,\ldots,\bar{Y})$, where $K=\min(\bar{X},\bar{Y})$. Thus, $\bar{U}=\bar{X}$, $\bar{V}=\min(\bar{X},\bar{Y})$, and $\bar{J}=\bar{Y}$.

Proposition 7.3: For the two-dimensional state, three-dimensional decision MDP with upper bounds $\bar{X}$ and $\bar{Y}$ for the state variables x and y, respectively, and with upper bounds $\bar{U}$, $\bar{V}$ and $\bar{J}$ for the decision variables u, v and j, respectively, the total number of decisions remaining after some decisions have been eliminated by the decision and system rules is $(K+1)(L+1)$, where $K=\min(\bar{X},\bar{Y})$ and $L=\max(\bar{X},\bar{Y})$, which is indeed the same as the total number of states $(\bar{X}+1)(\bar{Y}+1)$.

Proposition 7.3 is proved by proving the following Lemmas. The proofs of the Lemmas are given in Appendix II-C.

Lemma 7.1: The total number of decisions before some decisions are eliminated by the decision and system rules is $(K+1)(\bar{X}+1)(\bar{Y}+1)$.

Lemma 7.2: Two types of decisions, $(u>0, v>0, j>0)$ and $(u>0, v=0, j>0)$, would never be optimal.

Lemma 7.3: The number of decisions eliminated by the decision rules, $(u>0, v>0, j>0)$ or $(u>0, v=0, j>0)$, is $(K+1)\bar{X}\bar{Y}$.

Lemma 7.4: The number of decisions eliminated by the system rules, $(u+v>\bar{X})$ or $(v+j>\bar{Y})$, is $\frac{1}{6} K(K+1)(3L+K+5)$.

Lemma 7.5: The number of decisions eliminated by both the decision rules, $(u>0, v>0, j>0)$ or $(u>0, v=0, j>0)$, and the system rules, $(u+v>\bar{X})$ or $(v+j>\bar{Y})$, is $\frac{1}{6} K(K+1)(3L+K-1)$.

Corollary 7.3: For the two-dimensional state, three-dimensional decision MDP with upper bounds $\bar{X}$ and $\bar{Y}$ for the state variables x and y, respectively, and with upper bounds $\bar{U}$, $\bar{V}$ and $\bar{J}$ for the decision variables u, v and j, respectively, the proportion (of the total number of decisions) eliminated by the decision and system rules is $\frac{K}{K+1}$.

Proof: By Proposition 7.3,

Total number of decisions before elimination = $(K+1)(\bar{X}+1)(\bar{Y}+1)$.

Number decisions remaining after elimination = $(K+1)(L+1)$.

Thus,

Proportion (of the total number of decisions) eliminated

= 1 - (number of decisions remaining / total number of decisions)

$$= 1 - \frac{(K+1)(L+1)}{(K+1)(\bar{X}+1)(\bar{Y}+1)} = 1 - \frac{(\bar{X}+1)(\bar{Y}+1)}{(K+1)(\bar{X}+1)(\bar{Y}+1)} = 1 - \frac{1}{K+1} = \frac{K}{K+1}.$$

Corollary 7.3 shows how much reduction in the decision space is achieved by the decision and system rules. It implies that most of

potential decisions are eliminated and then tremendous computation time savings will follow. For example, an inventory system with $\bar{X}$=5 and $\bar{Y}$=5 has 36 states (by Proposition 7.1). The decision space of the system without the decision and system rules consists of 216 decisions (by Lemma 7.1). This means that we must consider for each and every one of the 216 decisions and must have a transient probability matrix for each and every decision. If we exercise the decision and system rules, however, the decision space will be reduced by 83% (by Corollary 7.3) and the number of decisions remaining after elimination will be 36 (by Proposition 7.3). Table II-1 and Figures II-3 and II-4 represent the number of decisions remaining before/after being eliminated by the decision and system rules and the percentage reduction in decision space for each pair of $\bar{X}$ and $\bar{Y}$.

$(\bar{X},\bar{Y})$	# of Decisions Before Elimination	# of Decisions After Elimination	% Reduction in Decision Space
(1,1)	8	4	50.0
(3,3)	64	16	75.0
(5,5)	216	36	83.3
(7,7)	512	64	87.5
(10,10)	1331	121	90.9
(15,15)	4096	256	93.8
(20,20)	9261	441	95.2
(30,30)	29791	961	96.8

Table II-1: Percentage Reduction in Decision Space

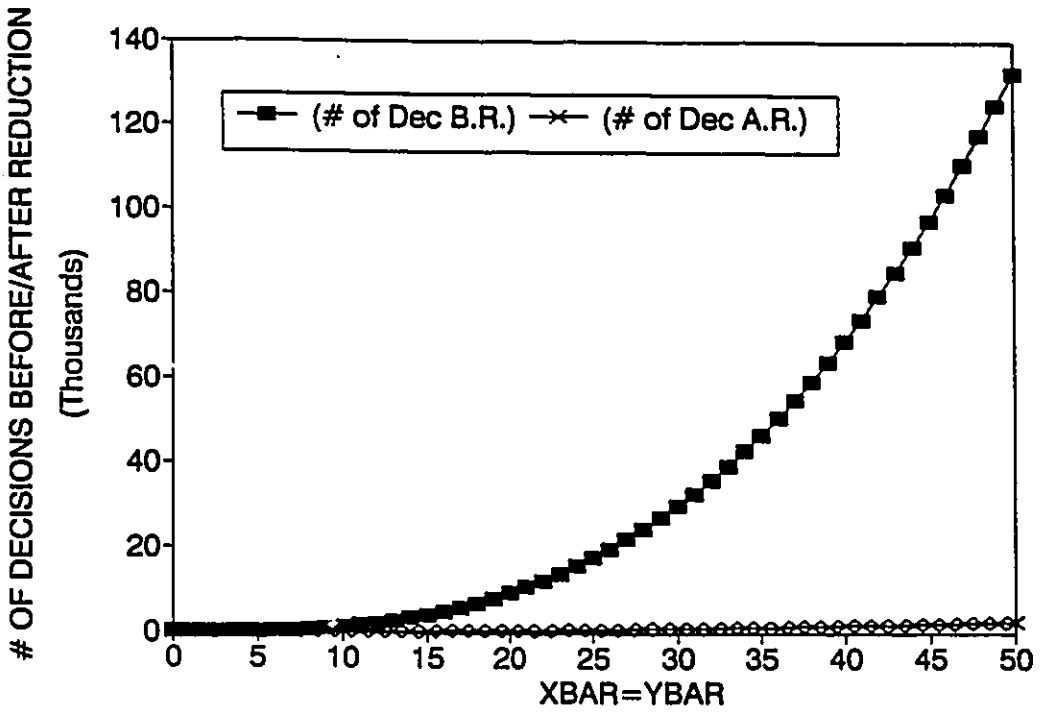
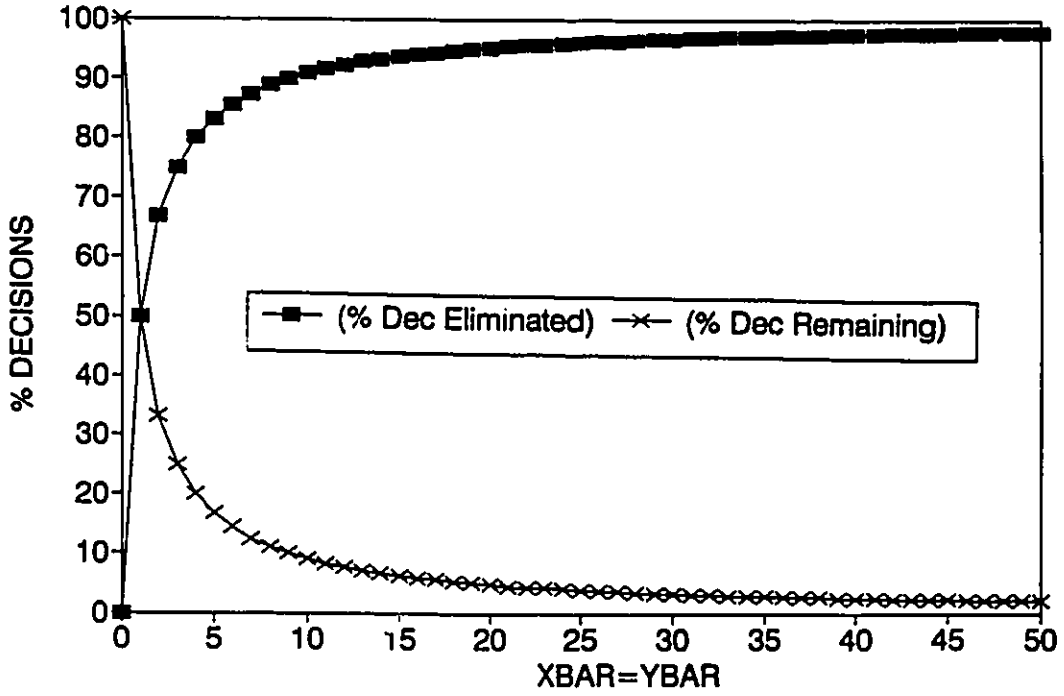


Figure II-3: Number of Decisions before and after Elimination



Figures II-4: Percentage Reduction in Decision Space

7.2.4 Derivation of Transition Probabilities

As discussed in Chapter 6, the return process is characterized by the sum of M+1 independent binomial random variables, that is,

$$z_n = \psi_{0,n} + \psi_{1,n} + \psi_{2,n} + \ldots + \psi_{M,n} = \sum_{i=0}^{M} \psi_{i,n} \qquad (7\text{-}5)$$

for $i \le n$ where $i=0,1,\ldots,M$ and $n=0,1,\ldots,N-1$. The following propositions and corollaries are used as the basis for deriving the transition probabilities of the system.

Proposition 7.4: If demands in successive periods $w_0, w_1, \ldots, w_{N-1}$ are independent, identically distributed (i.i.d.) nonnegative random variables and the return process z_n is characterized by (7-5), then the joint probability mass function of w_n and z_n is defined as follows.

$$P(w_n=\tau \text{ and } z_n=\gamma) = P\left(\sum_{i=0}^{M} \psi_{i,n}=\gamma \mid w_n=\tau\right) P(w_n=\tau),$$

where $\psi_{i,n} = \sum_{k=1}^{w_{n-i}} \theta_i$ for $i \le n$ $(i=0,1,\ldots,M;\ n=0,1,\ldots,N-1)$ are independent binomial random variables with parameters (w_{n-i}, δ_i) and θ_i are Bernoulli random variables with parameters δ_i, where $\delta_i = E(\theta_i)$.

Proof: $P(w_n=\tau \text{ and } z_n=\gamma)$

$$= P(z_n=\gamma \mid w_n=\tau)\, P(w_n=\tau) = P\left(\sum_{i=0}^{M} \psi_{i,n}=\gamma \mid w_n=\tau\right) P(w_n=\tau).$$

Corollary 7.4a: $P(w_n=\tau \text{ and } z_n \ge \gamma) = P\left(\sum_{i=0}^{M} \psi_{i,n} \ge \gamma \mid w_n=\tau\right) P(w_n=\tau).$

Proof: $P(w_n=\tau \text{ and } z_n \ge \gamma)$

$$= P(z_n \ge \gamma \mid w_n=\tau)\, P(w_n=\tau) = P\left(\sum_{i=0}^{M} \psi_{i,n} \ge \gamma \mid w_n=\tau\right) P(w_n=\tau).$$

Corollary 7.4b: $P(w_n \geq r \text{ and } z_n = \gamma) = \sum_{k=r}^{\infty} [\, P(\sum_{i=0}^{M} \psi_{i,n} = \gamma | w_n = k)\, P(w_n = k)\,]$.

Proof: $P(z_n = \gamma | w_n \geq r)$

$= P(z_n = \gamma \text{ and } w_n \geq r) \,/\, P(w_n \geq r)$

$= \sum_{k=r}^{\infty} [\, P(z_n = \gamma | w_n = k)\, P(w_n = k)\,] \,/\, P(w_n \geq r)$

$= \sum_{k=r}^{\infty} [\, P(\sum_{i=0}^{M} \psi_{i,n} = \gamma | w_n = k)\, P(w_n = k)\,] \,/\, P(w_n \geq r)$.

Thus, $P(w_n \geq r \text{ and } z_n = \gamma)$

$= P(z_n = \gamma | w_n \geq r)\, P(w_n \geq r) = \sum_{k=r}^{\infty} [\, P(\sum_{i=0}^{M} \psi_{i,n} = \gamma | w_n = k)\, P(w_n = k)\,]$.

Corollary 7.4c: $P(w_n \geq r \text{ and } z_n \geq \gamma) = \sum_{k=r}^{\infty} [\, P(\sum_{i=0}^{M} \psi_{i,n} \geq \gamma | w_n = k)\, P(w_n = k)\,]$.

Proof: $P(w_n \geq r \text{ and } z_n \geq \gamma)$

$= P(z_n \geq \gamma | w_n \geq r)\, P(w_n \geq r) = \sum_{k=r}^{\infty} [\, P(\sum_{i=0}^{M} \psi_{i,n} \geq \gamma | w_n = k)\, P(w_n = k)\,]$.

Proposition 7.5: Let $\psi_{i,n}$ $(i \leq n,\ i=0,1,..,M;\ n=0,1,..,N-1)$ be independent binomial random variables with parameters (w_{n-i}, δ_i). If $\delta_i = \delta_j = \delta$ for all $i,j (\leq n)$, $\sum_{i=0}^{M} \psi_{i,n}$ is binomial with parameters (Ω_n, δ), where $\Omega_n = \sum_{i=0}^{M} w_{n-i}$.

Proof: Since $\psi_{i,n}$ are binomial with parameters (w_{n-i}, δ),

$$P(\psi_{i,n} = \gamma) = \binom{w_{n-i}}{\gamma} \delta^{\gamma} (1-\delta)^{w_{n-i}-\gamma}.$$

The moment generating function of $\psi_{i,n}$, $\phi_{\psi_{i,n}}(t)$, is given by

$$\phi_{\psi_{i,n}}(t) = E[e^{t\psi_{i,n}}]$$

$$= \sum_{k=0}^{w_{n-i}} e^{tk} \binom{w_{n-i}}{k} \delta^k (1-\delta)^{w_{n-i}-k}$$

$$= \sum_{k=0}^{w_{n-i}} \binom{w_{n-i}}{k} (\delta e^t)^k (1-\delta)^{w_{n-i}-k}$$

$$= (\delta e^t + 1 - \delta)^{w_{n-i}}.$$

Let $\Gamma = \sum_{i=0}^{M} \psi_{i,n}$. The moment generating function of Γ, $\phi_\Gamma(t)$, is given by

$$\phi_\Gamma(t) = E[e^{t\Gamma}] = \prod_{i=0}^{M} \phi_{\psi_{i,n}}(t)$$

$$= \prod_{i=0}^{M} (\delta e^t + 1 - \delta)^{w_{n-i}}$$

$$= (\delta e^t + 1 - \delta)^{\Omega_n}, \quad \text{where } \Omega_n = \sum_{i=0}^{M} w_{n-i}.$$

Thus,

$$P(\Gamma=\gamma) = P\{ \sum_{i=0}^{M} \psi_{i,n} = \gamma \} = \binom{\Omega_n}{\gamma} \delta^\gamma (1-\delta)^{\Omega_n-\gamma}.$$

Corollary 7.5: Let $\psi_{i,n}$ ($i \le n$, $i=0,1,\ldots,M$; $n=0,1,\ldots,N-1$) be independent binomial random variables with parameters (w_{n-i}, δ_i). If $\delta_i \neq \delta_j$ for some $i \neq j (\le n)$, $\sum_{i=0}^{M} \psi_{i,n}$ has the following probability mass function.

$$P(\Gamma=\gamma) = P\{ \sum_{i=0}^{M} \psi_{i,n} = \gamma \}$$

$$= \sum_{k_1=0}^{\gamma} \left[\prod_{i=0}^{M-2} \left(\sum_{k_{i+1}=0}^{T} \right) \right] \left[\prod_{i=0}^{M-1} \binom{w_{n-i}}{k_i} \delta_i^{k_i} (1-\delta_i)^{w_{n-i}-k_i} \right]$$

$$\binom{w_{n-M}}{T} \delta_M^T (1-\delta_M)^{w_{n-M}-T} \quad \text{where } T = \gamma - \sum_{i=0}^{M-2} k_i.$$

Proof: Corollary 7.5 is proved as follows.

Given $P\{\Gamma=\gamma\} = P\{\sum_{i=0}^{M} \psi_{i,n}=\gamma\}$ and $P\{\psi_{i,n}=\gamma\} = \binom{w_{n-i}}{\gamma} \delta_i^{\gamma} (1-\delta_i)^{w_{n-i}-\gamma}$.

For M=1: $P\{\Gamma=\gamma\} = \sum_{k=0}^{\gamma} \binom{w_n}{k} \delta_0^k (1-\delta_0)^{w_n-k} \binom{w_{n-1}}{\gamma-k} \delta_1^{\gamma-k} (1-\delta_1)^{w_{n-1}-(\gamma-k)}$.

For M=2: $P\{\Gamma=\gamma\} = \sum_{k_1=0}^{\gamma} \sum_{k_2=0}^{\gamma-k_1} \binom{w_n}{k_1} \delta_0^{k_1} (1-\delta_0)^{w_n-k_1} \binom{w_{n-1}}{k_2} \delta_1^{k_2} (1-\delta_1)^{w_{n-1}-k_2}$

$$\binom{w_{n-2}}{\gamma-k_1-k_2} \delta_2^{\gamma-k_1-k_2} (1-\delta_2)^{w_{n-2}-(\gamma-k_1-k_2)}.$$

Proceeding sequentially, we get $P\{\Gamma=\gamma\}$ for M as shown in Corollary 7.5.

From now on, we assume that $\delta_i=\delta_j=\delta$ for all $i,j(\leq n)$. Proposition 7.6 restates (using Proposition 7.5) the joint probability mass function of w_n and z_n defined in Proposition 7.4.

Proposition 7.6: If $\delta_i=\delta_j=\delta$ for all $i,j(\leq n)$, then the joint probability mass function of w_n and z_n is stated as follows.

$$P\{w_n=\tau \text{ and } z_n=\gamma\} = \left[\binom{S_{M,n}+\tau}{\gamma} \delta^{\gamma}(1-\delta)^{S_{M,n}+\tau-\gamma} \right] P\{w_n=\tau\},$$

where $S_{M,n}$ is the total demand in the M previous periods observed at the beginning of period n (i.e., $S_{M,n} = \sum_{i=1}^{M} w_{n-i}$).

Proof: Let Ω'_n be the sum of the total demand in the M previous periods plus the demand in period n, given that $w_n=\tau$. Then, $\Omega'_n=S_{M,n}+\tau$.

Thus, $P\{w_n=\tau \text{ and } z_n=\gamma\}$

$$= P\{\sum_{i=0}^{M} \psi_{i,n}=\gamma | w_n=\tau\}\ P\{w_n=\tau\} = \left[\binom{\Omega'_n}{\gamma} \delta^{\gamma}(1-\delta)^{\Omega'_n-\gamma} \right] P\{w_n=\tau\}.$$

Corollary 7.6a:

$$P\{w_n=\tau \text{ and } z_n\geq\gamma\} = \left[\sum_{\ell=\gamma}^{S_{M,n}+\tau} \binom{S_{M,n}+\tau}{\ell} \delta^{\ell}(1-\delta)^{S_{M,n}+\tau-\ell} \right] P\{w_n=\tau\}.$$

Corollary 7.6b:

$$P\{w_n\geq\tau \text{ and } z_n=\gamma\} = \sum_{k=\tau}^{\infty} \left[\left[\binom{S_{M,n}+k}{\gamma} \delta^{\gamma}(1-\delta)^{S_{M,n}+k-\gamma} \right] P\{w_n=k\} \right].$$

Corollary 7.6c:

$$P\{w_n\geq\tau \text{ and } z_n\geq\gamma\} = \sum_{k=\tau}^{\infty} \left[\left[\sum_{\ell=\gamma}^{S_{M,n}+k} \binom{S_{M,n}+k}{\ell} \delta^{\ell}(1-\delta)^{S_{M,n}+k-\ell} \right] P\{w_n=k\} \right].$$

Transition probabilities, $P^{(n,S_{M,n})}_{(x,y)(x',y')}(u,v,j)$, i.e., the probabilities that the system will be in state (x',y') at period $n+1$ given that it is in state (x,y) at period n and the total demand in the last M periods observed at period n is $S_{M,n}$, and action (u,v,j) is chosen, are functions of the current state and the subsequent action. Thus, they can also be written as functions of the demand and return behaviors (i.e., in terms of the joint probabilities of the demand and returns). Let $\tau=x+u+v-x'$ and $\gamma=y'-(y-v-j)$. Then, the transition probabilities associated with decision (u,v,j) for a given $S_{M,n}$ at period n are defined as follows:

$$P^{(n,S_{M,n})}_{(x,y)(x',y')}(u,v,j) = \begin{cases} 0 & \text{if } \tau<0 \text{ or } \gamma<0 \\ P\{w_n \geq \tau \text{ and } z_n \geq \gamma\} & \text{if } x'=0 \ \& \ y'=\hat{Y} \\ P\{w_n \geq \tau \text{ and } z_n = \gamma\} & \text{if } x'=0 \ \& \ y'<\hat{Y} \\ P\{w_n = \tau \text{ and } z_n \geq \gamma\} & \text{if } x'>0 \ \& \ y'=\hat{Y} \\ P\{w_n = \tau \text{ and } z_n = \gamma\} & \text{if } x'>0 \ \& \ y'<\hat{Y}, \end{cases} \tag{7-6}$$

where the joint probabilities, $P\{w_n \text{ and } z_n\}$, are defined in Proposition 7.6 and Corollaries 7.6a to 7.6c.

A brief explanation of the derivation of the above transition probabilities follows. If the serviceable inventory on hand at the beginning of period n is x and decision (u,v,j) is made, the serviceable inventory available to satisfy the demand in period n must be greater than or equal to the ending serviceable inventory for period n (or the beginning inventory for period n+1) (i.e., $x+u+v \geq x'$ or $\tau \geq 0$). By the same token, the ending repairable inventory for period n (or the beginning inventory for period n+1) must be greater then or equal to the beginning repairable inventory for period n less the number of repairables repaired plus junked in period n (i.e., $y' \geq y-v-j$ or $\gamma \geq 0$). Otherwise (i.e., $\tau<0$ or $\gamma<0$), the transition probability becomes zero.

Considering the cases when both $\tau \geq 0$ and $\gamma \geq 0$, we first look at the serviceable inventory. For the ending serviceable inventory for period n to be zero (i.e., $x'=0$), the demand in period n must have been greater than or equal to the serviceable inventory available prior to the demand (i.e., $w_n \geq \tau (=x+u+v)$). On the other hand, for the ending serviceable inventory for period n to be greater than zero the demand in period n must have been equal to the serviceable inventory available

prior to the demand less the ending serviceable inventory (i.e., $w_n=r(=x+u+v-x')$). Let's examine the repairable inventory. For the ending repairable inventory for period n to be the maximum storage capacity of the repair facility (i.e., $y'=\bar{Y}$), the number of repairable units returned in period n must have been greater than or equal to the storage space available at period n (i.e., $z_n\geq\gamma(=\bar{Y}-(y-v-j))$). On the other hand, for the ending repairable inventory for period n to be less than the maximum capacity (i.e., $y'<\bar{Y}$) the number of returns in period n must have been equal to the difference between the beginning repairable inventory (less the number of units repaired plus junked) and the ending repairable inventory (i.e., $z_n=\gamma(=y'-(y-v-j))$). Note that the transition probabilities are the joint probabilities between the demand and returns resulting from the following (x',y') pairs: $(x'=0,y'=\bar{Y})$, $(x'=0,y'<\bar{Y})$, $(x'>0,y'=\bar{Y})$, and $(x'>0,y'<\bar{Y})$.

7.3 Finite-horizon Markov Decision Problems

This section presents a computational method for the two-dimensional state, three-dimensional decision finite-horizon Markov decision problem with the initial state (x_0,y_0). The method referred as 'the backward induction (or Dynamic Programming) algorithm' finds an optimal policy $\pi=\{\mu_0,\mu_1,\ldots,\mu_{N-1}\}$ where $\mu_n:S\rightarrow U$, for all $(x_n,y_n)\in S$, which minimizes the expected total discounted cost

$$J_\pi(x_0,y_0) = \underset{w_n}{E}\left\{ \alpha^N \phi_N(x_N,y_N) + \sum_{n=0}^{N-1} \alpha^n \phi(x_n,y_n,u_n,v_n,j_n,w_n) \right\} \qquad (7\text{-}7)$$

subject to the state equations and constraints (7-3a) to (7-3d). The computational method rests mainly on the principle of optimality.

Theorem 7.1. Principle of Optimality: (Bellman [1957])
Let $\pi^*=(\mu_0^*,\mu_1^*,\ldots,\mu_{N-1}^*)$ be an optimal control law for the basic N-period problem with the initial state $\chi(0)=\chi_0$ which minimizes

$$J_\pi(\chi_0) = \underset{\xi_n}{E}\left(\alpha^N \phi_N(\chi_N) + \sum_{n=0}^{N-1} \alpha^n \phi(\chi_n,\mu_n,\xi_n) \right), \tag{7-8}$$

where χ_n is the state of the system at time n and ξ_n is the disturbance in time n. Then the truncated control law $(\mu_i^*,\mu_{i+1}^*,\ldots,\mu_{N-1}^*)$ is optimal for the (N-i)-period subproblem with the initial state $\chi(i)=\chi_i$ which minimizes

$$J_\pi(\chi_i) = \underset{\xi_n}{E}\left(\alpha^N \phi_N(\chi_N) + \sum_{n=i}^{N-1} \alpha^n \phi(\chi_n,\mu_n,\xi_n) \right). \tag{7-9}$$

Proofs of the principle of optimality appear in Heyman and Sobel [1984], Derman [1970], and Hinderer [1970].

7.3.1 Backward Induction Algorithm

The backward induction algorithm for the proposed inventory problem determines the optimal (or a near-optimal) policy for the N-period problem by finding the optimal (or a near-optimal) policy for a given period model and then moving backward period by period using the following recursive relationship.

$$J_{(x,y)}^{(N,S_{M,N})} = \phi_N(x,y), \qquad \text{for } (x,y)\in S, \tag{7-10a}$$

and

$$J_{(x,y)}^{(n,S_{M,n})} = \min_{(u,v,j)} \Big(\phi_{(x,y)}(u,v,j) + \alpha \sum_{x'=0}^{\tilde{X}} \sum_{y'=0}^{\tilde{Y}} P_{(x,y)(x',y')}^{(n,S_{M,n})}(u,v,j)\, J_{(x',y')}^{(n+1,.)} \Big) \tag{7-10b}$$

$$\text{for } x=0,1,..,\tilde{X};\quad y=0,1,..,\tilde{Y};\quad S_{M,n}=0,1,..,\tilde{S}_M;\quad n=0,1,..,N-1,$$

where $J^{(n+1,.)}_{(x',y')} = \frac{1}{\tilde{S}_M+1}\left[\sum_{S_{M,n+1}=0}^{\tilde{S}_M} J^{(n+1,S_{M,n+1})}_{(x',y')}\right]$, $J^{(n,S_{M,n})}_{(x,y)}$ is a near-optimal (i.e., close-to-minimum expected total discounted) cost for the (N-n)-stage Markov Decision problem starting at state (x,y) at time n and ending at time N, and $\tilde{S}_M$ is the upper bound of $S_{M,n}$. We use the term "near-optimal" in the analysis of the finite-horizon problem because we do not consider the full transition probabilities for the system at a certain state at present being at the same or another state at the next observed time. Denote by $(x,y,S_{M,n})\rightarrow(x',y',S'_{M,n})$, the full transition of the system over the next time period. Then, the recursive relationship (7-10b) implies a partial transition of the system over the next time period, that is, $(x,y,S_{M,n})\rightarrow(x',y')$.

Algorithm 7.1: Backward Induction

Step 1: Define a state space S and a decision space U.
(i.e., Identify all possible states $(x_n,y_n)\epsilon S$ and all possible decisions $(u_n,v_n,j_n)\epsilon U$.) (Proposition 7.3)

Step 2: Compute the probability mass function (p.m.f.) and the cumulative distribution function (c.d.f.) of w_n, where w_n are i.i.d. nonnegative random variables.

Step 3: For each possible value of $S_{M,n}$, compute the conditional p.m.f. and the conditional c.d.f. of z_n using expressions in Proposition 7-6, where z_n are nonnegative random variables characterized by (7-5).

Step 4: Set n=N and $J^{(N,S_{M,N})}_{(x,y)} = \phi_N(x,y)$ for $(x,y)\epsilon S$ and for each $S_{M,n}$, where $\phi_N(x,y)$ is a terminal cost associated with the ending

inventories.

Step 5: Set n=n-1. Compute transition probabilities using (7-6) for each and every decision, given $(x,y)\epsilon S$ and $S_{M,n}$. These probabilities are non-stationary.

Step 6: Using the cost function (7-1) and expression (7-10b), compute $J_{(x,y)}^{(n,S_{M,n})}$ for each $(x,y)\epsilon S$ and for each $S_{M,n}$, and denote by μ_n^*, the set of all optimal decisions for $(x,y)\epsilon S$ and $S_{M,n}$, satisfying $\mu_n^* = \arg J_{(x,y)}^{(n,S_{M,n})}$.

Step 7: If n=0, stop. Otherwise, return to Step 5.

7.3.2 Numerical Example

A computer program (Appendix II-F) was written in QuickBasic to demonstrate the backward induction algorithm discussed in the previous section. The program runs on a microcomputer with 286 microprocessor. Although systems with more than 900 states (i.e., $(\bar{X},\bar{Y})=(30,30)$) are very time consuming, the advanced computer technology (i.e., main frames, 486 microprocessors, etc.) can take care of such a problem. Let us consider a finite-horizon problem with N=10, α=.9, C=4, c=6, R=4, r=4, f=1, h=2, p=0, b=15, $\bar{X}$=3, $\bar{Y}$=2, M=2, and δ=.2. We assume that the demand for serviceables in each period can be one of the following values, 0, 1, 2 and 3, and that each demand level has an equal probability. The algorithm starts with the last period of the planning horizon and moves backward period by period-each time finding the optimal (or a near-optimal) policy for that period-until it finds the optimal (or a near-optimal) policy for the whole problem. For each

state of each period, the algorithm finds the optimal (or a near-optimal) decision and the corresponding cost for each possible value of $S_{M,n}$. We chose $(\bar{X},\bar{Y})=(3,2)$ because it is large enough to characterize the solution to the finite-horizon Markov decision problem while it is small enough so that the complete solution tables can be included in this thesis.

A near-optimal policy to the 10-period Markov decision problem, $\pi_{10}=(\mu_0^*,\ldots,\mu_9^*)$, where μ_n^* (n=0,...,9) is the set of near-optimal decisions taken in period n, is shown in Tables II-2(i) to II-2(x). In each table, n representing decision epochs (periods) is shown in Column (a) followed by state of the system shown in Column (b). Possible values that $S_{M,n}$ can take are listed in Column (c). The set of near-optimal decisions taken at period n determined by the backward induction algorithm and the corresponding expected total discounted cost (E.T.D.C.) for the (10-n)-period problem are represented by Columns (d) and (e), respectively.

A brief explanation regarding use of the solution tables follows. At the beginning of each period starting n=0, the inventory manager looks at the serviceable and repairable inventories (i.e., the system state) and then observes the total demand ($S_{M,n}$) occurred during the last M periods. Based on these two observations, he makes his decision to purchase, to repair, and to junk. In reality, $S_{2,0}=w_{-2}+w_{-1}=0$ since $w_{-2}=w_{-1}=0$. Also $S_{2,1}=w_0$ since $w_{-1}=0$. However, the inventory manager can also consider other possible situations at n=0,1 since Tables II-2(i) and II-2(ii) show his best decision for each of all possible values of $S_{M,n}$.

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)	TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
9	(0 0)	0	(2 0 0)	21.25	9	(2 0)	0	(0 0 0)	5.25
		1	(2 0 0)	21.25			1	(0 0 0)	5.25
		2	(2 0 0)	21.25			2	(0 0 0)	5.25
		3	(2 0 0)	21.25			3	(0 0 0)	5.25
		4	(2 0 0)	21.25			4	(0 0 0)	5.25
		5	(2 0 0)	21.25			5	(0 0 0)	5.25
		6	(2 0 0)	21.25			6	(0 0 0)	5.25
9	(0 1)	0	(0 1 0)	19.75	9	(2 1)	0	(0 0 1)	5.25
		1	(0 1 0)	19.75			1	(0 0 1)	5.25
		2	(0 1 0)	19.75			2	(0 0 1)	5.25
		3	(0 1 0)	19.75			3	(0 0 1)	5.25
		4	(0 1 0)	19.75			4	(0 0 1)	5.25
		5	(0 1 0)	19.75			5	(0 0 1)	5.25
		6	(0 1 0)	19.75			6	(0 0 1)	5.25
9	(0 2)	0	(0 2 0)	17.25	9	(2 2)	0	(0 0 2)	5.25
		1	(0 2 0)	17.25			1	(0 0 2)	5.25
		2	(0 2 0)	17.25			2	(0 0 2)	5.25
		3	(0 2 0)	17.25			3	(0 0 2)	5.25
		4	(0 2 0)	17.25			4	(0 0 2)	5.25
		5	(0 2 0)	17.25			5	(0 0 2)	5.25
		6	(0 2 0)	17.25			6	(0 0 2)	5.25
9	(1 0)	0	(0 0 0)	11.75	9	(3 0)	0	(0 0 0)	3.00
		1	(0 0 0)	11.75			1	(0 0 0)	3.00
		2	(0 0 0)	11.75			2	(0 0 0)	3.00
		3	(0 0 0)	11.75			3	(0 0 0)	3.00
		4	(0 0 0)	11.75			4	(0 0 0)	3.00
		5	(0 0 0)	11.75			5	(0 0 0)	3.00
		6	(0 0 0)	11.75			6	(0 0 0)	3.00
9	(1 1)	0	(0 0 1)	11.75	9	(3 1)	0	(0 0 1)	3.00
		1	(0 0 1)	11.75			1	(0 0 1)	3.00
		2	(0 0 1)	11.75			2	(0 0 1)	3.00
		3	(0 0 1)	11.75			3	(0 0 1)	3.00
		4	(0 0 1)	11.75			4	(0 0 1)	3.00
		5	(0 0 1)	11.75			5	(0 0 1)	3.00
		6	(0 0 1)	11.75			6	(0 0 1)	3.00
9	(1 2)	0	(0 0 2)	11.75	9	(3 2)	0	(0 0 2)	3.00
		1	(0 0 2)	11.75			1	(0 0 2)	3.00
		2	(0 0 2)	11.75			2	(0 0 2)	3.00
		3	(0 0 2)	11.75			3	(0 0 2)	3.00
		4	(0 0 2)	11.75			4	(0 0 2)	3.00
		5	(0 0 2)	11.75			5	(0 0 2)	3.00
		6	(0 0 2)	11.75			6	(0 0 2)	3.00

Table II-2(i): Solution to 10-Period Finite Horizon Problem
(n=9; Last period of the planning horizon)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)	TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
8	(0 0)	0	(3 0 0)	34.06	8	(2 0)	0	(0 0 0)	18.27
		1	(3 0 0)	33.98			1	(0 0 0)	18.11
		2	(3 0 0)	33.91			2	(0 0 0)	17.96
		3	(2 0 0)	33.82			3	(0 0 0)	17.82
		4	(2 0 0)	33.69			4	(0 0 0)	17.69
		5	(2 0 0)	33.57			5	(0 0 0)	17.57
		6	(2 0 0)	33.47			6	(0 0 0)	17.47
8	(0 1)	0	(2 0 0)	34.49	8	(2 1)	0	(0 0 1)	18.27
		1	(2 0 0)	34.36			1	(0 0 1)	18.11
		2	(2 0 0)	34.25			2	(0 0 1)	17.96
		3	(2 0 0)	34.17			3	(0 0 1)	17.82
		4	(2 0 0)	34.10			4	(0 0 1)	17.69
		5	(2 0 0)	34.05			5	(0 0 1)	17.57
		6	(2 0 0)	34.01			6	(0 0 1)	17.47
8	(0 2)	0	(0 2 0)	30.27	8	(2 2)	0	(0 0 2)	18.27
		1	(0 2 0)	30.11			1	(0 0 2)	18.11
		2	(0 2 0)	29.96			2	(0 0 2)	17.96
		3	(0 2 0)	29.82			3	(0 0 2)	17.82
		4	(0 2 0)	29.69			4	(0 0 2)	17.69
		5	(0 2 0)	29.57			5	(0 0 2)	17.57
		6	(0 2 0)	29.47			6	(0 0 2)	17.47
8	(1 0)	0	(2 0 0)	28.06	8	(3 0)	0	(0 0 0)	12.06
		1	(2 0 0)	27.98			1	(0 0 0)	11.98
		2	(0 0 0)	27.84			2	(0 0 0)	11.91
		3	(0 0 0)	27.62			3	(0 0 0)	11.84
		4	(0 0 0)	27.42			4	(0 0 0)	11.78
		5	(0 0 0)	27.23			5	(0 0 0)	11.72
		6	(0 0 0)	27.06			6	(0 0 0)	11.67
8	(1 1)	0	(0 1 0)	26.27	8	(3 1)	0	(0 0 1)	12.06
		1	(0 1 0)	26.11			1	(0 0 1)	11.98
		2	(0 1 0)	25.96			2	(0 0 1)	11.91
		3	(0 1 0)	25.82			3	(0 0 1)	11.84
		4	(0 1 0)	25.69			4	(0 0 1)	11.78
		5	(0 1 0)	25.57			5	(0 0 1)	11.72
		6	(0 1 0)	25.47			6	(0 0 1)	11.67
8	(1 2)	0	(0 2 0)	24.06	8	(3 2)	0	(0 0 2)	12.06
		1	(0 2 0)	23.98			1	(0 0 2)	11.98
		2	(0 2 0)	23.91			2	(0 0 2)	11.91
		3	(0 2 0)	23.84			3	(0 0 2)	11.84
		4	(0 2 0)	23.78			4	(0 0 2)	11.78
		5	(0 2 0)	23.72			5	(0 0 2)	11.72
		6	(0 2 0)	23.67			6	(0 0 2)	11.67

Table II-2(ii): Solution to 10-Period Finite Horizon Problem (n=8)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)	TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
7	(0 0)	0	(3 0 0)	45.28	7	(2 0)	0	(0 0 0)	30.54
		1	(3 0 0)	45.13			1	(0 0 0)	30.34
		2	(3 0 0)	44.98			2	(0 0 0)	30.12
		3	(3 0 0)	44.84			3	(0 0 0)	29.90
		4	(3 0 0)	44.71			4	(0 0 0)	29.68
		5	(3 0 0)	44.58			5	(0 0 0)	29.48
		6	(3 0 0)	44.47			6	(0 0 0)	29.29
7	(0 1)	0	(3 0 0)	45.55	7	(2 1)	0	(0 0 1)	30.54
		1	(3 0 0)	45.39			1	(0 0 0)	30.25
		2	(3 0 0)	45.27			2	(0 0 0)	30.01
		3	(3 0 0)	45.17			3	(0 0 0)	29.82
		4	(3 0 0)	45.08			4	(0 0 0)	29.67
		5	(3 0 0)	45.02			5	(0 0 1)	29.48
		6	(3 0 0)	44.97			6	(0 0 1)	29.29
7	(0 2)	0	(0 2 0)	42.54	7	(2 2)	0	(0 0 0)	30.06
		1	(0 2 0)	42.34			1	(0 0 0)	30.06
		2	(0 2 0)	42.12			2	(0 0 1)	30.01
		3	(0 2 0)	41.90			3	(0 0 1)	29.82
		4	(0 2 0)	41.68			4	(0 0 1)	29.67
		5	(0 2 0)	41.48			5	(0 0 2)	29.48
		6	(0 2 0)	41.29			6	(0 0 2)	29.29
7	(1 0)	0	(2 0 0)	39.28	7	(3 0)	0	(0 0 0)	23.28
		1	(2 0 0)	39.13			1	(0 0 0)	23.13
		2	(2 0 0)	38.98			2	(0 0 0)	22.98
		3	(2 0 0)	38.84			3	(0 0 0)	22.84
		4	(2 0 0)	38.71			4	(0 0 0)	22.71
		5	(2 0 0)	38.58			5	(0 0 0)	22.58
		6	(2 0 0)	38.47			6	(0 0 0)	22.47
7	(1 1)	0	(0 1 0)	38.54	7	(3 1)	0	(0 0 1)	23.28
		1	(0 1 0)	38.34			1	(0 0 1)	23.13
		2	(0 1 0)	38.12			2	(0 0 1)	22.98
		3	(0 1 0)	37.90			3	(0 0 1)	22.84
		4	(0 1 0)	37.68			4	(0 0 1)	22.71
		5	(0 1 0)	37.48			5	(0 0 1)	22.58
		6	(0 1 0)	37.29			6	(0 0 1)	22.47
7	(1 2)	0	(0 2 0)	35.28	7	(3 2)	0	(0 0 2)	23.28
		1	(0 2 0)	35.13			1	(0 0 2)	23.13
		2	(0 2 0)	34.98			2	(0 0 2)	22.98
		3	(0 2 0)	34.84			3	(0 0 2)	22.84
		4	(0 2 0)	34.71			4	(0 0 2)	22.71
		5	(0 2 0)	34.58			5	(0 0 2)	22.58
		6	(0 2 0)	34.47			6	(0 0 2)	22.47

Table II-2(iii): Solution to 10-Period Finite Horizon Problem (n=7)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
6	(0 0)	0	(3 0 0)	55.56
		1	(3 0 0)	55.44
		2	(3 0 0)	55.31
		3	(3 0 0)	55.18
		4	(3 0 0)	55.05
		5	(3 0 0)	54.93
		6	(3 0 0)	54.82
6	(0 1)	0	(3 0 0)	55.95
		1	(3 0 0)	55.78
		2	(3 0 0)	55.65
		3	(3 0 0)	55.54
		4	(3 0 0)	55.46
		5	(3 0 0)	55.39
		6	(3 0 0)	55.33
6	(0 2)	0	(0 2 0)	52.82
		1	(0 2 0)	52.67
		2	(0 2 0)	52.50
		3	(0 2 0)	52.31
		4	(0 2 0)	52.12
		5	(0 2 0)	51.95
		6	(0 2 0)	51.78
6	(1 0)	0	(2 0 0)	49.56
		1	(2 0 0)	49.44
		2	(2 0 0)	49.31
		3	(2 0 0)	49.18
		4	(2 0 0)	49.05
		5	(2 0 0)	48.93
		6	(2 0 0)	48.82
6	(1 1)	0	(0 1 0)	48.82
		1	(0 1 0)	48.67
		2	(0 1 0)	48.50
		3	(0 1 0)	48.31
		4	(0 1 0)	48.12
		5	(0 1 0)	47.95
		6	(0 1 0)	47.78
6	(1 2)	0	(0 2 0)	45.56
		1	(0 2 0)	45.44
		2	(0 2 0)	45.31
		3	(0 2 0)	45.18
		4	(0 2 0)	45.05
		5	(0 2 0)	44.93
		6	(0 2 0)	44.82
6	(2 0)	0	(0 0 0)	40.82
		1	(0 0 0)	40.67
		2	(0 0 0)	40.50
		3	(0 0 0)	40.31
		4	(0 0 0)	40.12
		5	(0 0 0)	39.95
		6	(0 0 0)	39.78
6	(2 1)	0	(0 0 1)	40.82
		1	(0 0 1)	40.67
		2	(0 0 1)	40.50
		3	(0 0 1)	40.31
		4	(0 0 1)	40.12
		5	(0 0 1)	39.95
		6	(0 0 1)	39.78
6	(2 2)	0	(0 0 0)	40.82
		1	(0 0 0)	40.67
		2	(0 0 2)	40.50
		3	(0 0 2)	40.31
		4	(0 0 2)	40.12
		5	(0 0 2)	39.95
		6	(0 0 2)	39.78
6	(3 0)	0	(0 0 0)	33.56
		1	(0 0 0)	33.44
		2	(0 0 0)	33.31
		3	(0 0 0)	33.18
		4	(0 0 0)	33.05
		5	(0 0 0)	32.93
		6	(0 0 0)	32.82
6	(3 1)	0	(0 0 1)	33.56
		1	(0 0 1)	33.44
		2	(0 0 1)	33.31
		3	(0 0 1)	33.18
		4	(0 0 1)	33.05
		5	(0 0 1)	32.93
		6	(0 0 1)	32.82
6	(3 2)	0	(0 0 2)	33.56
		1	(0 0 2)	33.44
		2	(0 0 2)	33.31
		3	(0 0 2)	33.18
		4	(0 0 2)	33.05
		5	(0 0 2)	32.93
		6	(0 0 2)	32.82

Table II-2(iv): Solution to 10-Period Finite Horizon Problem (n=6)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
5	(0 0)	0	(3 0 0)	64.89
		1	(3 0 0)	64.77
		2	(3 0 0)	64.64
		3	(3 0 0)	64.51
		4	(3 0 0)	64.39
		5	(3 0 0)	64.27
		6	(3 0 0)	64.17
5	(0 1)	0	(3 0 0)	65.30
		1	(3 0 0)	65.13
		2	(3 0 0)	65.00
		3	(3 0 0)	64.89
		4	(3 0 0)	64.81
		5	(3 0 0)	64.74
		6	(3 0 0)	64.68
5	(0 2)	0	(0 2 0)	62.15
		1	(0 2 0)	62.01
		2	(0 2 0)	61.84
		3	(0 2 0)	61.65
		4	(0 2 0)	61.47
		5	(0 2 0)	61.30
		6	(0 2 0)	61.13
5	(1 0)	0	(2 0 0)	58.89
		1	(2 0 0)	58.77
		2	(2 0 0)	58.64
		3	(2 0 0)	58.51
		4	(2 0 0)	58.39
		5	(2 0 0)	58.27
		6	(2 0 0)	58.17
5	(1 1)	0	(0 1 0)	58.15
		1	(0 1 0)	58.01
		2	(0 1 0)	57.84
		3	(0 1 0)	57.65
		4	(0 1 0)	57.47
		5	(0 1 0)	57.30
		6	(0 1 0)	57.13
5	(1 2)	0	(0 2 0)	54.89
		1	(0 2 0)	54.77
		2	(0 2 0)	54.64
		3	(0 2 0)	54.51
		4	(0 2 0)	54.39
		5	(0 2 0)	54.27
		6	(0 2 0)	54.17
5	(2 0)	0	(0 0 0)	50.15
		1	(0 0 0)	50.01
		2	(0 0 0)	49.84
		3	(0 0 0)	49.65
		4	(0 0 0)	49.47
		5	(0 0 0)	49.30
		6	(0 0 0)	49.13
5	(2 1)	0	(0 0 1)	50.15
		1	(0 0 1)	50.01
		2	(0 0 1)	49.84
		3	(0 0 1)	49.65
		4	(0 0 1)	49.47
		5	(0 0 1)	49.30
		6	(0 0 1)	49.13
5	(2 2)	0	(0 0 0)	50.15
		1	(0 0 2)	50.01
		2	(0 0 2)	49.84
		3	(0 0 2)	49.65
		4	(0 0 2)	49.47
		5	(0 0 2)	49.30
		6	(0 0 2)	49.13
5	(3 0)	0	(0 0 0)	42.89
		1	(0 0 0)	42.77
		2	(0 0 0)	42.64
		3	(0 0 0)	42.51
		4	(0 0 0)	42.39
		5	(0 0 0)	42.27
		6	(0 0 0)	42.17
5	(3 1)	0	(0 0 1)	42.89
		1	(0 0 1)	42.77
		2	(0 0 1)	42.64
		3	(0 0 1)	42.51
		4	(0 0 1)	42.39
		5	(0 0 1)	42.27
		6	(0 0 1)	42.17
5	(3 2)	0	(0 0 2)	42.89
		1	(0 0 2)	42.77
		2	(0 0 2)	42.64
		3	(0 0 2)	42.51
		4	(0 0 2)	42.39
		5	(0 0 2)	42.27
		6	(0 0 2)	42.17

Table II-2(v): Solution to 10-Period Finite Horizon Problem (n=5)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)	TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
4	(0 0)	0	(3 0 0)	73.29	4	(2 0)	0	(0 0 0)	58.56
		1	(3 0 0)	73.17			1	(0 0 0)	58.41
		2	(3 0 0)	73.05			2	(0 0 0)	58.24
		3	(3 0 0)	72.92			3	(0 0 0)	58.06
		4	(3 0 0)	72.79			4	(0 0 0)	57.88
		5	(3 0 0)	72.68			5	(0 0 0)	57.71
		6	(3 0 0)	72.57			6	(0 0 0)	57.54
4	(0 1)	0	(3 0 0)	73.70	4	(2 1)	0	(0 0 1)	58.56
		1	(3 0 0)	73.54			1	(0 0 1)	58.41
		2	(3 0 0)	73.40			2	(0 0 1)	58.24
		3	(3 0 0)	73.30			3	(0 0 1)	58.06
		4	(3 0 0)	73.21			4	(0 0 1)	57.88
		5	(3 0 0)	73.14			5	(0 0 1)	57.71
		6	(3 0 0)	73.09			6	(0 0 1)	57.54
4	(0 2)	0	(0 2 0)	70.56	4	(2 2)	0	(0 0 0)	58.43
		1	(0 2 0)	70.41			1	(0 0 2)	58.41
		2	(0 2 0)	70.24			2	(0 0 2)	58.24
		3	(0 2 0)	70.06			3	(0 0 2)	58.06
		4	(0 2 0)	69.88			4	(0 0 2)	57.88
		5	(0 2 0)	69.71			5	(0 0 2)	57.71
		6	(0 2 0)	69.54			6	(0 0 2)	57.54
4	(1 0)	0	(2 0 0)	67.29	4	(3 0)	0	(0 0 0)	51.29
		1	(2 0 0)	67.17			1	(0 0 0)	51.17
		2	(2 0 0)	67.05			2	(0 0 0)	51.05
		3	(2 0 0)	66.92			3	(0 0 0)	50.92
		4	(2 0 0)	66.79			4	(0 0 0)	50.79
		5	(2 0 0)	66.68			5	(0 0 0)	50.68
		6	(2 0 0)	66.57			6	(0 0 0)	50.57
4	(1 1)	0	(0 1 0)	66.56	4	(3 1)	0	(0 0 1)	51.29
		1	(0 1 0)	66.41			1	(0 0 1)	51.17
		2	(0 1 0)	66.24			2	(0 0 1)	51.05
		3	(0 1 0)	66.06			3	(0 0 1)	50.92
		4	(0 1 0)	65.88			4	(0 0 1)	50.79
		5	(0 1 0)	65.71			5	(0 0 1)	50.68
		6	(0 1 0)	65.54			6	(0 0 1)	50.57
4	(1 2)	0	(0 2 0)	63.29	4	(3 2)	0	(0 0 2)	51.29
		1	(0 2 0)	63.17			1	(0 0 2)	51.17
		2	(0 2 0)	63.05			2	(0 0 2)	51.05
		3	(0 2 0)	62.92			3	(0 0 2)	50.92
		4	(0 2 0)	62.79			4	(0 0 2)	50.79
		5	(0 2 0)	62.68			5	(0 0 2)	50.68
		6	(0 2 0)	62.57			6	(0 0 2)	50.57

Table II-2(vi): Solution to 10-Period Finite Horizon Problem (n=4)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)	TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
3	(0 0)	0	(3 0 0)	80.86	3	(2 0)	0	(0 0 0)	66.12
		1	(3 0 0)	80.74			1	(0 0 0)	65.98
		2	(3 0 0)	80.61			2	(0 0 0)	65.81
		3	(3 0 0)	80.48			3	(0 0 0)	65.63
		4	(3 0 0)	80.36			4	(0 0 0)	65.45
		5	(3 0 0)	80.24			5	(0 0 0)	65.27
		6	(3 0 0)	80.14			6	(0 0 0)	65.11
3	(0 1)	0	(3 0 0)	81.27	3	(2 1)	0	(0 0 1)	66.12
		1	(3 0 0)	81.10			1	(0 0 1)	65.98
		2	(3 0 0)	80.97			2	(0 0 1)	65.81
		3	(3 0 0)	80.86			3	(0 0 1)	65.63
		4	(3 0 0)	80.78			4	(0 0 1)	65.45
		5	(3 0 0)	80.71			5	(0 0 1)	65.27
		6	(3 0 0)	80.66			6	(0 0 1)	65.11
3	(0 2)	0	(0 2 0)	78.12	3	(2 2)	0	(0 0 0)	65.99
		1	(0 2 0)	77.98			1	(0 0 2)	65.98
		2	(0 2 0)	77.81			2	(0 0 2)	65.81
		3	(0 2 0)	77.63			3	(0 0 2)	65.63
		4	(0 2 0)	77.45			4	(0 0 2)	65.45
		5	(0 2 0)	77.27			5	(0 0 2)	65.27
		6	(0 2 0)	77.11			6	(0 0 2)	65.11
3	(1 0)	0	(2 0 0)	74.86	3	(3 0)	0	(0 0 0)	58.86
		1	(2 0 0)	74.74			1	(0 0 0)	58.74
		2	(2 0 0)	74.61			2	(0 0 0)	58.61
		3	(2 0 0)	74.48			3	(0 0 0)	58.48
		4	(2 0 0)	74.36			4	(0 0 0)	58.36
		5	(2 0 0)	74.24			5	(0 0 0)	58.24
		6	(2 0 0)	74.14			6	(0 0 0)	58.14
3	(1 1)	0	(0 1 0)	74.12	3	(3 1)	0	(0 0 1)	58.86
		1	(0 1 0)	73.98			1	(0 0 1)	58.74
		2	(0 1 0)	73.81			2	(0 0 1)	58.61
		3	(0 1 0)	73.63			3	(0 0 1)	58.48
		4	(0 1 0)	73.45			4	(0 0 1)	58.36
		5	(0 1 0)	73.27			5	(0 0 1)	58.24
		6	(0 1 0)	73.11			6	(0 0 1)	58.14
3	(1 2)	0	(0 2 0)	70.86	3	(3 2)	0	(0 0 2)	58.86
		1	(0 2 0)	70.74			1	(0 0 2)	58.74
		2	(0 2 0)	70.61			2	(0 0 2)	58.61
		3	(0 2 0)	70.48			3	(0 0 2)	58.48
		4	(0 2 0)	70.36			4	(0 0 2)	58.36
		5	(0 2 0)	70.24			5	(0 0 2)	58.24
		6	(0 2 0)	70.14			6	(0 0 2)	58.14

Table II-2(vii): Solution to 10-Period Finite Horizon Problem (n=3)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)	TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
2	(0 0)	0	(3 0 0)	87.67	2	(2 0)	0	(0 0 0)	72.93
		1	(3 0 0)	87.55			1	(0 0 0)	72.79
		2	(3 0 0)	87.42			2	(0 0 0)	72.62
		3	(3 0 0)	87.29			3	(0 0 0)	72.44
		4	(3 0 0)	87.17			4	(0 0 0)	72.26
		5	(3 0 0)	87.05			5	(0 0 0)	72.08
		6	(3 0 0)	86.95			6	(0 0 0)	71.92
2	(0 1)	0	(3 0 0)	88.08	2	(2 1)	0	(0 0 1)	72.93
		1	(3 0 0)	87.91			1	(0 0 1)	72.79
		2	(3 0 0)	87.78			2	(0 0 1)	72.62
		3	(3 0 0)	87.67			3	(0 0 1)	72.44
		4	(3 0 0)	87.59			4	(0 0 1)	72.26
		5	(3 0 0)	87.52			5	(0 0 1)	72.08
		6	(3 0 0)	87.47			6	(0 0 1)	71.92
2	(0 2)	0	(0 2 0)	84.93	2	(2 2)	0	(0 0 0)	72.80
		1	(0 2 0)	84.79			1	(0 0 2)	72.79
		2	(0 2 0)	84.62			2	(0 0 2)	72.62
		3	(0 2 0)	84.44			3	(0 0 2)	72.44
		4	(0 2 0)	84.26			4	(0 0 2)	72.26
		5	(0 2 0)	84.08			5	(0 0 2)	72.08
		6	(0 2 0)	83.92			6	(0 0 2)	71.92
2	(1 0)	0	(2 0 0)	81.67	2	(3 0)	0	(0 0 0)	65.67
		1	(2 0 0)	81.55			1	(0 0 0)	65.55
		2	(2 0 0)	81.42			2	(0 0 0)	65.42
		3	(2 0 0)	81.29			3	(0 0 0)	65.29
		4	(2 0 0)	81.17			4	(0 0 0)	65.17
		5	(2 0 0)	81.05			5	(0 0 0)	65.05
		6	(2 0 0)	80.95			6	(0 0 0)	64.95
2	(1 1)	0	(0 1 0)	80.93	2	(3 1)	0	(0 0 1)	65.67
		1	(0 1 0)	80.79			1	(0 0 1)	65.55
		2	(0 1 0)	80.62			2	(0 0 1)	65.42
		3	(0 1 0)	80.44			3	(0 0 1)	65.29
		4	(0 1 0)	80.26			4	(0 0 1)	65.17
		5	(0 1 0)	80.08			5	(0 0 1)	65.05
		6	(0 1 0)	79.92			6	(0 0 1)	64.95
2	(1 2)	0	(0 2 0)	77.67	2	(3 2)	0	(0 0 2)	65.67
		1	(0 2 0)	77.55			1	(0 0 2)	65.55
		2	(0 2 0)	77.42			2	(0 0 2)	65.42
		3	(0 2 0)	77.29			3	(0 0 2)	65.29
		4	(0 2 0)	77.17			4	(0 0 2)	65.17
		5	(0 2 0)	77.05			5	(0 0 2)	65.05
		6	(0 2 0)	76.95			6	(0 0 2)	64.95

Table II-2(viii): Solution to 10-Period Finite Horizon Problem (n=2)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)	TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
1	(0 0)	0	(3 0 0)	93.80	1	(2 0)	0	(0 0 0)	79.06
		1	(3 0 0)	93.68			1	(0 0 0)	78.92
		2	(3 0 0)	93.55			2	(0 0 0)	78.75
		3	(3 0 0)	93.42			3	(0 0 0)	78.57
		4	(3 0 0)	93.30			4	(0 0 0)	78.38
		5	(3 0 0)	93.18			5	(0 0 0)	78.21
		6	(3 0 0)	93.08			6	(0 0 0)	78.05
1	(0 1)	0	(3 0 0)	94.21	1	(2 1)	0	(0 0 1)	79.06
		1	(3 0 0)	94.04			1	(0 0 1)	78.92
		2	(3 0 0)	93.91			2	(0 0 1)	78.75
		3	(3 0 0)	93.80			3	(0 0 1)	78.57
		4	(3 0 0)	93.72			4	(0 0 1)	78.38
		5	(3 0 0)	93.65			5	(0 0 1)	78.21
		6	(3 0 0)	93.59			6	(0 0 1)	78.05
1	(0 2)	0	(0 2 0)	91.06	1	(2 2)	0	(0 0 0)	78.93
		1	(0 2 0)	90.92			1	(0 0 2)	78.92
		2	(0 2 0)	90.75			2	(0 0 2)	78.75
		3	(0 2 0)	90.57			3	(0 0 2)	78.57
		4	(0 2 0)	90.38			4	(0 0 2)	78.38
		5	(0 2 0)	90.21			5	(0 0 2)	78.21
		6	(0 2 0)	90.05			6	(0 0 2)	78.05
1	(1 0)	0	(2 0 0)	87.80	1	(3 0)	0	(0 0 0)	71.80
		1	(2 0 0)	87.68			1	(0 0 0)	71.68
		2	(2 0 0)	87.55			2	(0 0 0)	71.55
		3	(2 0 0)	87.42			3	(0 0 0)	71.42
		4	(2 0 0)	87.30			4	(0 0 0)	71.30
		5	(2 0 0)	87.18			5	(0 0 0)	71.18
		6	(2 0 0)	87.08			6	(0 0 0)	71.08
1	(1 1)	0	(0 1 0)	87.06	1	(3 1)	0	(0 0 1)	71.80
		1	(0 1 0)	86.92			1	(0 0 1)	71.68
		2	(0 1 0)	86.75			2	(0 0 1)	71.55
		3	(0 1 0)	86 57			3	(0 0 1)	71.42
		4	(0 1 0)	86.38			4	(0 0 1)	71.30
		5	(0 1 0)	86.21			5	(0 0 1)	71.18
		6	(0 1 0)	86.05			6	(0 0 1)	71.08
1	(1 2)	0	(0 2 0)	83.80	1	(3 2)	0	(0 0 2)	71.80
		1	(0 2 0)	83.68			1	(0 0 2)	71.68
		2	(0 2 0)	83.55			2	(0 0 2)	71.55
		3	(0 2 0)	83.42			3	(0 0 2)	71.42
		4	(0 2 0)	83.30			4	(0 0 2)	71.30
		5	(0 2 0)	83.18			5	(0 0 2)	71.18
		6	(0 2 0)	83.08			6	(0 0 2)	71.08

Table II-2(ix): Solution to 10-Period Finite Horizon Problem (n=1)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)	TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
0	(0 0)	0	(3 0 0)	99.31	0	(2 0)	0	(0 0 0)	84.58
		1	(3 0 0)	99.19			1	(0 0 0)	84.43
		2	(3 0 0)	99.07			2	(0 0 0)	84.26
		3	(3 0 0)	98.94			3	(0 0 0)	84.08
		4	(3 0 0)	98.81			4	(0 0 0)	83.90
		5	(3 0 0)	98.70			5	(0 0 0)	83.73
		6	(3 0 0)	98.59			6	(0 0 0)	83.56
0	(0 1)	0	(3 0 0)	99.72	0	(2 1)	0	(0 0 1)	84.58
		1	(3 0 0)	99.56			1	(0 0 1)	84.43
		2	(3 0 0)	99.42			2	(0 0 1)	84.26
		3	(3 0 0)	99.32			3	(0 0 1)	84.08
		4	(3 0 0)	99.23			4	(0 0 1)	83.90
		5	(3 0 0)	99.16			5	(0 0 1)	83.73
		6	(3 0 0)	99.11			6	(0 0 1)	83.56
0	(0 2)	0	(0 2 0)	96.58	0	(2 2)	0	(0 0 0)	84.45
		1	(0 2 0)	96.43			1	(0 0 2)	84.43
		2	(0 2 0)	96.26			2	(0 0 2)	84.26
		3	(0 2 0)	96.08			3	(0 0 2)	84.08
		4	(0 2 0)	95.90			4	(0 0 2)	83.90
		5	(0 2 0)	95.73			5	(0 0 2)	83.73
		6	(0 2 0)	95.56			6	(0 0 2)	83.56
0	(1 0)	0	(2 0 0)	93.31	0	(3 0)	0	(0 0 0)	77.31
		1	(2 0 0)	93.19			1	(0 0 0)	77.19
		2	(2 0 0)	93.07			2	(0 0 0)	77.07
		3	(2 0 0)	92.94			3	(0 0 0)	76.94
		4	(2 0 0)	92.81			4	(0 0 0)	76.81
		5	(2 0 0)	92.70			5	(0 0 0)	76.70
		6	(2 0 0)	92.59			6	(0 0 0)	76.59
0	(1 1)	0	(0 1 0)	92.58	0	(3 1)	0	(0 0 1)	77.31
		1	(0 1 0)	92.43			1	(0 0 1)	77.19
		2	(0 1 0)	92.26			2	(0 0 1)	77.07
		3	(0 1 0)	92.08			3	(0 0 1)	76.94
		4	(0 1 0)	91.90			4	(0 0 1)	76.81
		5	(0 1 0)	91.73			5	(0 0 1)	76.70
		6	(0 1 0)	91.56			6	(0 0 1)	76.59
0	(1 2)	0	(0 2 0)	89.31	0	(3 2)	0	(0 0 2)	77.31
		1	(0 2 0)	89.19			1	(0 0 2)	77.19
		2	(0 2 0)	89.07			2	(0 0 2)	77.07
		3	(0 2 0)	88.94			3	(0 0 2)	76.94
		4	(0 2 0)	88.81			4	(0 0 2)	76.81
		5	(0 2 0)	88.70			5	(0 0 2)	76.70
		6	(0 2 0)	88.59			6	(0 0 2)	76.59

Table II-2(x): Solution to 10-Period Finite Horizon Problem
(n=0; Beginning of the planning horizon)

7.4 Infinite-horizon Problems

Consider an infinite-horizon Markov decision problem with the initial state (x_0,y_0). The objective is to find a steady-state policy $\pi_\infty=(\mu,\mu,\ldots,\mu)$ for all $(x_n,y_n)\epsilon S$, $n=0,1,2,\ldots$, which minimizes the expected long-run total discounted cost

$$J_\pi(x_0,y_0) = \lim_{N\to\infty} E_{w_n} \{ \sum_{n=0}^{N-1} \alpha^n \phi[x_n,y_n,u_n,v_n,j_n,w_n] \} \qquad (7\text{-}11)$$

subject to the system equation constraints (7-3a) to (7-3d).

For the infinite-horizon problem, an optimal stationary policy will not be obtained if the transition probabilities are not stationary. In other words, we must have stationary transition probabilities for a stationary policy to be optimal. This requires additional assumptions on the return process (7-5). For the infinite-horizon Markov decision problem, we assume that the total (Ω_n) of the demands in the current period and the last M warranty periods is approximated by the number of warranty periods times the mean of the demand distribution (i.e., $\Omega_n=(M+1)E\{w_{n-i}\}=(M+1)\omega$). Then, the distribution of the return process is binomial with parameters (Ω,δ), where $\Omega=\Omega_n$, and the transition probabilities become independent of time (stationary). Let $\tau=x+u+v-x'$ and $\gamma=y'-(y-v-j)$. The transition probabilities for decision (u,v,j) chosen at period n are defined as follows:

$$P_{(x,y)(x',y')}(u,v,j) = \begin{cases} 0 & \text{if } \tau<0 \text{ or } \gamma<0 \\ P\{w_n\geq\tau \text{ and } z_n\geq\gamma\} & \text{if } x'=0 \ \&\ y'=\tilde{Y} \\ P\{w_n\geq\tau \text{ and } z_n=\gamma\} & \text{if } x'=0 \ \&\ y'<\tilde{Y} \\ P\{w_n=\tau \text{ and } z_n\geq\gamma\} & \text{if } x'>0 \ \&\ y'=\tilde{Y} \\ P\{w_n=\tau \text{ and } z_n=\gamma\} & \text{if } x'>0 \ \&\ y'<\tilde{Y}, \end{cases} \qquad (7\text{-}12)$$

where $P\{w_n=r \text{ and } z_n=\gamma\} = [\binom{\Omega}{\gamma} \delta^{\gamma}(1-\delta)^{\Omega}]\, P\{w_n=r\}$, (7-13a)

$P\{w_n=r \text{ and } z_n\geq\gamma\} = [\sum_{\ell=\gamma}^{\Omega} \binom{\Omega}{\ell} \delta^{\ell}(1-\delta)^{\Omega-\ell}]\, P\{w_n=r\}$, (7-13b)

$P\{w_n\geq r \text{ and } z_n=\gamma\} = \sum_{k=r}^{\infty} [\binom{\Omega}{\gamma} \delta^{\gamma}(1-\delta)^{\Omega}]\, P\{w_n=k\}$, (7-13c)

$P\{w_n\geq r \text{ and } z_n\geq\gamma\} = \sum_{k=r}^{\infty} [\sum_{\ell=\gamma}^{\Omega} \binom{\Omega}{\ell} \delta^{\ell}(1-\delta)^{\Omega-\ell}]\, P\{w_n=k\}$. (7-13d)

Alternative approaches for solving the infinite-horizon Markov decision problem include linear programming (LP), policy iteration, and the method of successive approximations. Appendix II-D compares these three computational methods. Linear programming and policy iteration terminate in a finite number of iterations and yield an optimal policy, but in both approaches each iteration takes a long time because it requires solution of a linear system of simultaneous equations. On the other hand, the method of successive approximation (or stochastic DP algorithm) yields in the limit the optimal cost function and an optimal stationary policy after a finite number of iterations. We will use the successive approximation method to solve the two-dimensional state, three-dimensional decision infinite-horizon Markov decision problem.

7.4.1 Method of Successive Approximation

The following two propositions are necessary for the method to ensure that costs are discounted so that a convergence is achieved.

Proposition 7.7: In the two-dimensional state, three-dimensional decision infinite-time Markov decision problem, the cost per stage

function ϕ in the objective cost functional (7-11) satisfies $0 \le \phi(x,y,u,v,j,w) \le A$ for all $(x,y,u,v,j,w) \in SxUxD$, where A is some scalar.

Proof: Recall the cost per stage function in (7-1)

$$\phi(x,y,u,v,j,w) = \sigma_u C + cu + \sigma_v R + rv + pj + f \max\{0,y-v-j\} + h \max\{E[0,x+u+v-w]\} + b \max\{E[0,w-x-u-v]\},$$

where $\sigma_u = \begin{cases} 1 & \text{if } u>0 \\ 0 & \text{if } u=0 \end{cases}$ and $\sigma_v = \begin{cases} 1 & \text{if } v>0 \\ 0 & \text{if } v=0. \end{cases}$

The state variables x and y of state (x,y) are nonnegative, and have upper bounds $\bar{X}$ and $\bar{Y}$, respectively. Also, the decision variables u, v and j are nonnegative, and have upper bounds $\bar{U}$, $\bar{V}$ and $\bar{J}$, respectively. Furthermore, C, c, R, r, p, f, h and b are constants and w is an element of the disturbance space D, which is a countable set. Since each term of ϕ is bounded by some scalar, ϕ is bounded by some scalar A.

Proposition 7.8: Let Π be the set of all admissible policies π, that is, $\pi=(\mu_0,\mu_1,\ldots)$. Then the real-valued optimal cost function J^* of the two-dimensional state and three-dimensional decision infinite-horizon Markov decision problem with a discount factor α, $J^*(x,y) = \min_{\pi\in\Pi} J_\pi(x,y)$ for $(x,y)\in S$, is bounded by some scalar.

Proof: Recall the cost functional of the Markov decision problem as

$$J_\pi(x,y) = \lim_{N\to\infty} E_{w_n}\left(\sum_{n=0}^{N-1} \alpha^n \phi(x_n,y_n,u_n,v_n,j_n,w_n) \right), \quad n=0,1,\ldots,N-1.$$

Since ϕ is bounded by 0 and A (by Proposition 7.7), for every ϕ bounded by A at each time period n=0,1,,...,N-1

$$\lim_{N\to\infty} \; E_{w_n} \left\{ \sum_{n=0}^{N-1} \alpha^n \phi(x_n, y_n, u_n, v_n, j_n, w_n) \right\} \le \sum_{n=0}^{\infty} \alpha^n A = A/(1-\alpha).$$

$$\therefore \; 0 \le J^*(x,y) \le J_\pi(x,y) \le A/(1-\alpha).$$

The successive approximation method is somewhat similar to the backward induction algorithm described for the finite-horizon problem in Section 7.3. The notation $J^n_{(x,y)}$ (for a fixed value of $S_{M,n}$) in (7-10b) involves moving backward period by period (i.e., the stage is measured by the system being in period n). To treat the infinite-horizon problem, a certain change of the stage definition is required. For the infinite-horizon problem, the stage is measured by the system evolving for t time periods and the notation $J^t_{(x,y)}$ is now being used for the expected total discounted cost of the system starting in state (x,y) and evolving for t periods. Assume that the terminal cost is zero for the infinite-horizon problem, i.e., $J^N_{(x,y)} = \phi_N(x,y) = 0$ $(x,y) \in S$ for $N \to \infty$. After transforming $J^n_{(x,y)}$ into $J^t_{(x,y)}$ (see Appendix II-E), we obtain the expressions for the terminal cost and the recursive relationship of successive approximation method.

$$J^0_{(x,y)} = 0, \qquad \text{for } (x,y) \in S, \tag{7-14a}$$

and

$$J^{t+1}_{(x,y)} = \min_{(u,v,j)} \left\{ \phi_{(x,y)}(u,v,j) + \alpha \sum_{x'=0}^{\bar{X}} \sum_{y'=0}^{\bar{Y}} P_{(x,y)(x',y')}(u,v,j) \, J^t_{(x',y')} \right\}$$

$$\text{for } (x,y) \in S, \quad t=0,1,\ldots,N-1, \tag{7-14b}$$

where $J^t_{(x,y)}$ is the expected total discounted cost of the system starting in state (x,y) and evolving for t time periods. As $t\to\infty$, $J^t_{(x,y)}$ will converge to $J^*(x,y)$, where $J^*(x,y)$ is the minimum expected (long-run) total discounted cost of the system starting at state (x,y) and evolving indefinitely. Proofs of the convergence are given in Heyman and Sobel [1984] and Bertsekas [1987].

Algorithm 7.2: Successive Approximation

Step 1: Define a state space S and a decision space U (Proposition 7.3).

Step 2: Compute the p.m.f. and c.d.f. of w_n, and compute the p.m.f. and c.d.f. of z_n using expressions in (7-13a) to (7-13d).

Step 3: Set t=0 and $J^t_{(x,y)} = 0$ for all $(x,y)\epsilon S$.

Step 4: For each $(x,y)\epsilon S$, compute $J^{t+1}_{(x,y)}$ using the cost function (7-1), the stationary transition probabilities (7-12), and expression (7-14b).

Step 5: If $|J^{t+1}_{(x,y)} - J^t_{(x,y)}| < \epsilon$ for all $(x,y)\epsilon S$, where ϵ is the stopping constant, go to step 6. Otherwise, increment t by 1 and return to Step 4.

Step 6: For each $(x,y)\epsilon S$, set $\mu^* = \arg J^{t+1}_{(x,y)}$ and stop.

Some literature (Puterman [1990]) use a stopping constant β (instead of ϵ) and call the optimal policy determined by the stopping criterion in Step 5 "β-optimal", where $\epsilon = \frac{1-\alpha}{2\alpha}\beta$. It states that the algorithm will terminate with a stationary policy having expected total discounted cost within β of optimal.

7.4.2 Numerical Example

A computer program (Appendix II-G) was written to demonstrate the successive approximation method for the infinite-horizon Markov decision problem. Let us consider the following data: $N \to \infty$, $\alpha = .9$, $C = 0$, $c = 6$, $R = 0$, $r = 4$, $f = 1$, $h = 2$, $p = 0$, $b = 15$, $\tilde{X} = 5$, $\tilde{Y} = 5$, $M = 2$, and $\delta = .2$. We will later consider the case of non-zero C and R. We assume that the demand for serviceables follows a Poisson distribution with mean rate of 3 (i.e., $\omega = 3$). Thus, $\Omega = 3(2+1) = 9$. Note that the transition probabilities for the infinite-horizon problem are stationary.

Letting $N \to \infty$, the program starts with $t = 1$ and runs indefinitely until a convergence of the expected total discounted cost for each state is achieved. The speed of the convergence seems to be quite slow especially when $\alpha \to 1$. We use the stopping constant $\epsilon = .001$ (or $\beta = .018$), meaning that the algorithm will terminate with a stationary policy having expected long-run total discounted cost within .018 of optimal.

The successive approximation method yields an optimal solution in a finite number of iterations. The solution to the infinite-horizon Markov decision problem is characterized by Table II-3. In the table, period t is shown in Column (a) followed by state of the system shown in Column (b). A set of optimal decisions taken in period t and the corresponding expected total discounted cost over the past t periods determined by the successive approximation method are represented by Columns (c) and (d), respectively.

The problem had taken 113 iterations (i.e., $t = 113$) before a convergence occurred (i.e., no changes in expected total discounted cost more than .001 after $t = 113$). Table II-3 indeed represents the optimal

stationary policy, $\pi_\infty^* = (\mu^*, \ldots, \mu^*)$, which minimizes the expected long-run total discounted cost. Note that the optimal stationary policy had been achieved after two iterations (t=2) although the convergence of the total expected discounted cost was not achieved until t=113.

From the optimal solution obtained numerically, we conjecture that the optimal policy structure for this problem is similar to the one in Simpson [1972]. When the system is in state (x_n, y_n) at period n, an action (u_n, v_n, j_n) is chosen based on the following three time-dependent parameters: repair-up-to serviceable inventory point (β_n), purchase-up-to serviceable inventory point (η_n), and junk-down-to repairable inventory point (ζ_n). A brief explanation follows. At period n, the inventory manager looks at the serviceable inventory x_n. If the serviceable inventory is less than β_n, he repairs $(\beta_n - x_n)$ units of repairables if they are available so that the number of serviceables after the repair decision is equal to β_n. If there are not enough repairables to raise the serviceable inventory to β_n, he repairs all available repairables and then purchases some serviceables so that the number of serviceables after the repair and purchase decisions is equal to η_n. After the repair and purchase decisions, the manager looks at the total of serviceables and repairables (including those to be replenished). If it is greater than ζ_n, he junks repairables down to ζ_n so that the total is either equal to or close to ζ_n. To clarify the policy structure described above, we may characterize policy in terms of decisions u_n, v_n and j_n as functions of β_n, η_n and ζ_n. The following formulae have been derived.

$$v_n = \max\,[0, \min\{y_n, \beta_n - x_n\}], \quad u_n = \max\,[0, \eta_n - (x_n + y_n)],$$

and $j_n = \min [\max(0, x_n+y_n-\beta_n), \max(0, \max(x_n+y_n, \eta_n)-\zeta_n)]$.

Denote the set of solution parameters for period n by $\Phi_n=(\beta_n, \eta_n, \zeta_n)$, where $\eta_n \leq \beta_n \leq \zeta_n$. We can determine the set of solution parameters for the infinite-horizon problem, $\Phi_\infty=(\beta_\infty, \eta_\infty, \zeta_\infty)$, by examining the state/decision pairs (Columns (b) and (c) of Table II-3). Whenever the serviceable inventory is less than 5 units, it is required to repair up to $5-x_n$ units if repairables are available for repair (i.e., $\beta_\infty=5$). If repairables are not available or they are not enough to satisfy the requirement, the serviceable inventory is replenished first by repairing all repairables and then by purchasing additional serviceables so that the number of serviceables is equal to 4 (i.e., $\eta_\infty=4$). Finally, if the sum of the serviceable and repairable inventories (after the repair and purchase decisions) is greater than 5 some or all of the remaining repairables are scrapped so that the total is either equal to or close to 5 (i.e., $\zeta_\infty=5$). Thus we have obtained $\Phi_\infty=(\beta_\infty, \eta_\infty, \zeta_\infty)=(5,4,5)$. Based on the set of solution parameters just obtained, we can determine the optimal decision (u_n, v_n, j_n) using the formulae derived above. For an example, if the system is in state (2,5) at period n, $u_n=\max[0,4-7]=0$, $v_n=\max[0,\min\{5,5-2\}]=3$, and $j_n=\min[\max\{0,7-5\},\max\{0,\max(7,4)-5\}]=2$, that is, $(u_n^*, v_n^*, j_n^*)=(0,3,2)$.

To analyze the infinite-horizon problem with non-zero C and R, we consider the case of C=R=4. The algorithm terminated with an optimal stationary policy after 115 iterations (i.e., t=115) using a stopping constant $\epsilon=.001$; however, no fixed policy structures (i.e., a set of solution parameters) was found.

PERIOD t (a)	STATE (x,y) (b)	DECISION (u,v,j) (c)	EXPECTED TOTAL DISCOUNTED COST (d)*
113	(0 0)	(4 0 0)	214.90
113	(0 1)	(3 1 0)	212.90
113	(0 2)	(2 2 0)	210.90
113	(0 3)	(1 3 0)	208.90
113	(0 4)	(0 4 0)	206.90
113	(0 5)	(0 5 0)	205.89
113	(1 0)	(3 0 0)	208.90
113	(1 1)	(2 1 0)	206.90
113	(1 2)	(1 2 0)	204.90
113	(1 3)	(0 3 0)	202.90
113	(1 4)	(0 4 0)	201.89
113	(1 5)	(0 4 1)	201.89
113	(2 0)	(2 0 0)	202.90
113	(2 1)	(1 1 0)	200.90
113	(2 2)	(0 2 0)	198.90
113	(2 3)	(0 3 0)	197.89
113	(2 4)	(0 3 1)	197.89
113	(2 5)	(0 3 2)	197.89
113	(3 0)	(1 0 0)	196.90
113	(3 1)	(0 1 0)	194.90
113	(3 2)	(0 2 0)	193.89
113	(3 3)	(0 2 1)	193.89
113	(3 4)	(0 2 2)	193.89
113	(3 5)	(0 2 3)	193.89
113	(4 0)	(0 0 0)	190.90
113	(4 1)	(0 1 0)	189.89
113	(4 2)	(0 1 1)	189.89
113	(4 3)	(0 1 2)	189.89
113	(4 4)	(0 1 3)	189.89
113	(4 5)	(0 1 4)	189.89
113	(5 0)	(0 0 0)	185.89
113	(5 1)	(0 0 1)	185.89
113	(5 2)	(0 0 2)	185.89
113	(5 3)	(0 0 3)	185.89
113	(5 4)	(0 0 4)	185.89
113	(5 5)	(0 0 5)	185.89

* Minimum expected long-run total discounted cost

Table II-3: Solution to Infinite-horizon Problem (for t=113; N=∞).

CHAPTER 8 ACCELERATION OF COMPUTATIONS

8.1 Error Bounds Approach for Infinite-horizon MDPs

Although the successive approximation algorithm for an infinite-horizon Markov decision problem presented in Section 7.3.2 yields in the limit the optimal cost function and an optimal stationary policy for $0<\alpha<1$, its convergence is quite slow when α approaches one and it often requires a large number of iterations. In this section, we present an improved successive approximation algorithm which utilizes error bounds. With the error bounds approach, an optimal stationary policy and the corresponding optimal cost are found in a much less number of iterations.

8.1.1 Description of Error Bounds Approach

With this improved successive approximation approach, the lower and upper bounds for the optimal cost J^* determine whether an optimal stationary policy and the corresponding optimal cost have been reached. In other words, the optimal stationary policy and the corresponding optimal cost are achieved when the lower and upper bounds converge to each other for all elements of the state space. References on the topic include Bertsekas [1976][1987] and Puterman [1990]. Let $J:S\rightarrow R$. We have

$$\lim_{t\rightarrow\infty} T^t_{(x,y)}(J) = J^*(x,y) \quad \text{for all } (x,y)\epsilon S, \tag{8-1}$$

where $J^*(x,y)$ is the cost function corresponding to a stationary policy and $T^t_{(x,y)}(J)$ is the expected total discounted cost over t time periods

computed by the successive approximation method. Note that $T^{t}_{(x,y)}(J)$ is obtained by computing $J^{t}_{(x,y)}$ using (7-12b).

Proposition 8.1: Denote $B^{t+1}_{(x,y)} = T^{t+1}_{(x,y)}(J) - T^{t}_{(x,y)}(J)$. (8-2)

For all $(x,y)\epsilon S$ and t=0,1,2,...,

$$T^{t+1}_{(x,y)}(J) + e_{t+1} \leq T^{t+2}_{(x,y)}(J) + e_{t+2}$$

$$\leq J^{*}_{(x,y)} \leq T^{t+2}_{(x,y)}(J) + e'_{t+2} \leq T^{t+1}_{(x,y)}(J) + e'_{t+1}, \quad (8\text{-}3)$$

where $e_{t+1} = \frac{\alpha}{1-\alpha} \min_{(x,y)\epsilon S} B^{t+1}_{(x,y)}$ and $e'_{t+1} = \frac{\alpha}{1-\alpha} \max_{(x,y)\epsilon S} B^{t+1}_{(x,y)}$. (8-4)

Proof: Refer to Bertsekas [1976].

Algorithm 8.1: Successive Approximation with Error Bounds

Step 1: Define a state space S and a decision space U (Proposition 7.3).

Step 2: Compute the p.m.f. and c.d.f. of w_n, and compute the p.m.f. and c.d.f. of z_n using expressions in (7-13a) to (7-13d).

Step 3: Set t=0 and $T^{t}_{(x,y)}(J) = 0$ for all $(x,y)\epsilon S$.

Step 4: For each $(x,y)\epsilon S$, compute $T^{t+1}_{(x,y)}(J)$ using expression (7-14b), the cost function (7-1), the stationary transition probabilities (7-12), and compute e_{t+1} and e'_{t+1} using (8-2) and (8-4).

Step 5: Let $G^{t+1}_{(x,y)} = T^{t+1}_{(x,y)}(J) + e_{t+1}$ and $G'^{t+1}_{(x,y)} = T^{t+1}_{(x,y)}(J) + e'_{t+1}$. If $G^{t+1}_{(x,y)} = G'^{t+1}_{(x,y)}$ for all $(x,y)\epsilon S$, go to step 6. Otherwise, increment t by 1 and return to Step 4.

Step 6: For each $(x,y)\epsilon S$, set $\mu^{*} = \arg T^{t+1}_{(x,y)}(J)$ and $J^{*}(x,y) = G^{t+1}_{(x,y)}$. Stop.

8.1.2 Numerical Examples

The computer program previously written to demonstrate the successive approximation method was extended to include the error bounds approach discussed in Section 8.1.1 (listed in Appendix II-G). The program runs indefinitely and generates lower and upper bounds for each and every state of the system and compare them for a convergence. When the lower and upper bounds converge to each other in a given period, the algorithm terminates with an optimal stationary policy, $\pi_{\infty}^{*}=(\mu^{*},\ldots,\mu^{*})$, which minimizes the expected long-run total discounted cost.

Let us consider the same data used in Section 7.4.2. Table II-4 shows how an optimal stationary policy and the corresponding optimal cost are obtained using the error bounds approach. The algorithm terminated after seven iterations (i.e., t=7) when the upper and lower bounds have exactly the same value for each and every state (see Columns (e) and (f)). Note that the cost in Column (d) is not optimal (i.e., without the error bounds approach, the optimal cost is not achieved until t=113 as demonstrated in Section 7.4.2). The convergence of the upper and lower bounds for each and every state, however, indicates that not only the optimal stationary policy but the optimal cost per each state have been obtained after seven iterations. Column (c) represents the optimal stationary policy, $\pi_{\infty}^{*}=(\mu,\ldots,\mu^{*})$, and Columns (e) and (f) indicate the minimum expected long-run total discounted costs. For an example, when the system is in state (1,2) at in a given period it is optimal to purchase one serviceable and to repair all repairables (i.e., $(u^{*},v^{*},j^{*})=(1,2,0)$). From the optimal stationary policy, the set of solution parameters was also found as $\Phi_{\infty}=(\beta_{\infty},\ \eta_{\infty},\zeta_{\infty})=(5,4,5)$, which is

PERIOD t (a)	STATE (x,y) (b)	DECISION (u,v,j) (c)	EXPECTED TOTAL DISCOUNTED COST (d)	LOWER BOUND (e)*	UPPER BOUND (f)*
7	(0 0)	(4 0 0)	116.96	214.90	214.90
7	(0 1)	(3 1 0)	114.96	212.90	212.90
7	(0 2)	(2 2 0)	112.96	210.90	210.90
7	(0 3)	(1 3 0)	110.96	208.90	208.90
7	(0 4)	(0 4 0)	108.96	206.90	206.90
7	(0 5)	(0 5 0)	107.94	205.89	205.89
7	(1 0)	(3 0 0)	110.96	208.90	208.90
7	(1 1)	(2 1 0)	108.96	206.90	206.90
7	(1 2)	(1 2 0)	106.96	204.90	204.90
7	(1 3)	(0 3 0)	104.96	202.90	202.90
7	(1 4)	(0 4 0)	103.94	201.89	201.89
7	(1 5)	(0 4 1)	103.94	201.89	201.89
7	(2 0)	(2 0 0)	104.96	202.90	202.90
7	(2 1)	(1 1 0)	102.96	200.90	200.90
7	(2 2)	(0 2 0)	100.96	198.90	198.90
7	(2 3)	(0 3 0)	99.94	197.89	197.89
7	(2 4)	(0 3 1)	99.94	197.89	197.89
7	(2 5)	(0 3 2)	99.94	197.89	197.89
7	(3 0)	(1 0 0)	98.96	196.90	196.90
7	(3 1)	(0 1 0)	96.96	194.90	194.90
7	(3 2)	(0 2 0)	95.94	193.89	193.89
7	(3 3)	(0 2 1)	95.94	193.89	193.89
7	(3 4)	(0 2 2)	95.94	193.89	193.89
7	(3 5)	(0 2 3)	95.94	193.89	193.89
7	(4 0)	(0 0 0)	92.96	190.90	190.90
7	(4 1)	(0 1 0)	91.94	189.89	189.89
7	(4 2)	(0 1 1)	91.94	189.89	189.89
7	(4 3)	(0 1 2)	91.94	189.89	189.89
7	(4 4)	(0 1 3)	91.94	189.89	189.89
7	(4 5)	(0 1 4)	91.94	189.89	189.89
7	(5 0)	(0 0 0)	87.94	185.89	185.89
7	(5 1)	(0 0 1)	87.94	185.89	185.89
7	(5 2)	(0 0 2)	87.94	185.89	185.89
7	(5 3)	(0 0 3)	87.94	185.89	185.89
7	(5 4)	(0 0 4)	87.94	185.89	185.89
7	(5 5)	(0 0 5)	87.94	185.89	185.89

* LOWER BOUND = UPPER BOUND

Table II-4: Solution to Infinite-horizon Problem
(Error bounds approach; for t=7)

identical to the one obtained in Section 7.4.2.

For the case of C=R=4, the algorithm terminated with an optimal stationary policy and the corresponding expected total discounted cost after eleven iterations (i.e., t=11).

8.2 State Decomposition by Dimension (SDD)

In Section 8.1, we were concerned with an acceleration technique utilizing error bounds for speeding up the convergence of the successive approximation method for infinite-horizon problems so that the total number of iterations required before a convergence is reduced. In this section, we introduce another acceleration technique called 'State Decomposition by Dimension (SDD)', which speeds up the convergence of the DP algorithms via reduction in state space. The technique can be used to approximate the optimal solution to both finite and infinite horizon repairable-item inventory problems.

8.2.1 Description of The SDD

The SDD decomposes a multi-dimensional state Markov decision process into several lower-dimensional state MDPs so that the total number of states considered in the system is reduced. With the SDD, a two-dimensional state problem is divided into two one-dimensional state subproblems. The SDD is based upon the following unique features of the repairable-item inventory model. First, as shown in Figure II-5 the main problem is divided into two subproblems (i.e., repair/junk and purchase/repair subproblems) which can be treated somewhat independently as the repair decision v_n is the only link between the two subproblems.

In other words, the SDD decouples the connection between the two subproblems so that each subproblem can be solved separately with some measure of independence. Second, the transient probability matrix for each decision in the main problem can also be decomposed into two lower-dimensional transient probability matrices. In this way, the demand and return processes are also treated somewhat independently as $S_{M,n}$ is the only link between the two transient probability matrices. An additional cost due to the loss of interdependency between the two subproblems will be incurred and will be included in the expected total discounted cost.

The SDD may be applied to infinite-horizon problems as well as finite-horizon problems. For each of the two subproblems, the reduced state and decision spaces, the cost per stage function, the transition probabilities, and the recursive relationships for finite and infinite planning horizons can be stated as follows.

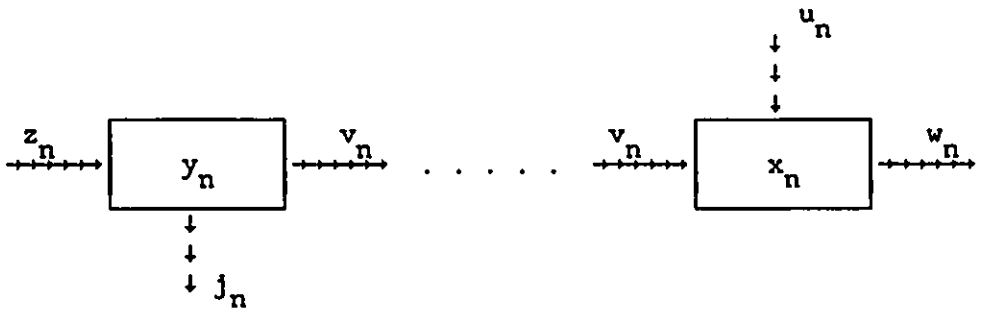


Figure II-5: The Decoupled Subproblems.

Repair/junk Subproblem:

The state y_n is an element of a state space S_y, and the control (v_n, j_n) is an element of a control space U_1.

The cost per stage function is

$$\phi_y(v,j) = pj + f \max\{0, y-v-j\}. \tag{8-5}$$

Note that the repair cost terms $\sigma_v R + rv$ are not included in (8-5) because they are cost factors associated with replenishment (i.e., the number of units repaired) to the serviceable inventory.

The transition probabilities that the system will be in state y' at period n given that it is in state y at period n and the total demand in the last M periods observed at the beginning of period n is $S_{M,n}$, and action (v,j) is chosen are

$$P_{yy'}^{(n,S_{M,n})}(v,j) = \begin{cases} 0 & \text{if } \gamma<0 \\ P\{z_n \geq \gamma\} & \text{if } y'=\tilde{Y} \quad \text{where } \gamma = y' - (y-v-j), \\ P\{z_n = \gamma\} & \text{if } y'<\tilde{Y} \end{cases} \tag{8-6}$$

where z_n are mixed binomial random variables since Ω_n contains random variables w_n (i.e., $\Omega_n = S_{M,n} + w_n$). The return distribution is defined as:

$$P\{z_n=\gamma\} = P\{\sum_{i=0}^{M} \psi_{i,n} = \gamma\} = \sum_{k=0}^{\tilde{W}} \left[\binom{S_{M,n}+k}{\gamma} \delta^{\gamma}(1-\delta)^{S_{M,n}+k-\gamma} P\{w_n=k\} \right],$$

where $\tilde{W}$ is the upper bound of w_n determined by the distribution of w_n.

For the finite-horizon repair/junk subproblem,

$$J_y^{(N,S_{M,n})} = \phi_N(y) \qquad y\epsilon S_y \tag{8-7a}$$

and

$$J_y^{(n,S_{M,n})}(v,j) = \phi_y(v,j) + \alpha \sum_{y'=0}^{\bar{Y}} P_{yy'}^{(n,S_{M,n})}(v,j)\, J_{y'}^{(n+1,.)} \tag{8-7b}$$

$$\text{for } y\epsilon S_y;\quad S_{M,n}=0,1,..,\bar{S}_M;\quad n=0,1,..,N-1,$$

where $J_{y'}^{(n+1,.)} = \frac{1}{\bar{S}_M+1}\left[\sum_{S_{M,n+1}=0}^{\bar{S}_M} J_{y'}^{(n+1,S_{M,n+1})}\right]$

and $\bar{S}_M$ is the upper bound of $S_{M,n}$.

For the infinite-horizon repair/junk subproblem,

$$J_y^0 = 0 \qquad y\epsilon S_y \tag{8-8a}$$

and

$$J_y^{t+1}(v,j) = \phi_y(v,j) + \alpha \sum_{y'=0}^{\bar{Y}} P_{yy'}(v,j)\, J_{y'}^t \tag{8-8b}$$

$$\text{where } y\epsilon S_y \text{ and } t=1,\ldots,N.$$

Purchase/Repair Subproblem:

The state x_n is an element of a state space S_x and the control (u_n,v_n) is an element of a control space U_2.

The cost per stage function is

$$\phi_x(u,v) = \sigma_u C + cu + \sigma_v R + rv + h \max\{E[0,x+u+v-w]\} + b \max\{E[0,w-x-u-v]\}, \quad (8\text{-}9)$$

$$\text{where } \sigma_u = \begin{cases} 1 & \text{if } u>0 \\ 0 & \text{if } u=0 \end{cases} \quad \text{and} \quad \sigma_v = \begin{cases} 1 & \text{if } v>0 \\ 0 & \text{if } v=0 \end{cases}$$

The transition probabilities that the system will be in state x' at the next observed time period given that it is in state x at present and action (u,v) is chosen are

$$P_{xx'}^{(n)}(u,v) = \begin{cases} 0 & \text{if } r<0 \\ P(w_n \geq r) & \text{if } x'=0 \\ P(w_n = r) & \text{if } x'>0, \end{cases} \quad \text{where } r=x+u+v-x'. \quad (8\text{-}10)$$

For the finite-horizon purchase/repair subproblem,

$$J_x^N = \phi_N(x) \qquad x \in S_x \quad (8\text{-}11a)$$

and

$$J_x^n(u,v) = \phi_x(u,v) + \alpha \sum_{x'=0}^{\bar{X}} P_{xx'}^{(n)}(u,v)\, J_{x'}^{n+1} \quad (8\text{-}11b)$$

where $x \in S_x$ and $n=0,1,\ldots,N-1$.

For the infinite-horizon purchase/repair subproblem,

$$J_x^0 = 0 \qquad x \in S_x \quad (8\text{-}12a)$$

and

$$J_x^{t+1}(u,v) = \phi_x(u,v) + \alpha \sum_{x'=0}^{\bar{X}} P_{xx'}(u,v)\, J_{x'}^t, \quad (8\text{-}12b)$$

where $x \in S_x$ and $t=1,\ldots,N$.

After solving the two subproblems separately for each $x \epsilon S_x$ and $y \epsilon S_y$, we optimize the expected total discounted cost as follows.

For the finite-horizon problem:

For each $(x,y) \epsilon S$ and each $S_{M,n}$,

$$\bar{J}_{(x,y)}^{(n,S_{M,n})} = \min_{(u,v,j)} \bar{J}_{(x,y)}^{(n,S_{M,n})}(u,v,j) \quad \text{and} \quad \tilde{\mu}_n = \arg \bar{J}_{(x,y)}^{(n,S_{M,n})},$$

where $\bar{J}_{(x,y)}^{(n,S_{M,n})}(u,v,j) = J_x^n(u,v) + J_y^{(n,S_{M,n})}(v,j)$, (i.e., the expected total discounted cost obtained using the decomposed cost functions).

For the infinite-horizon problem:

For each $(x,y) \epsilon S$,

$$\bar{J}_{(x,y)}^{t+1} = \min_{(u,v,j)} \bar{J}_{(x,y)}^{t+1}(u,v,j),$$

where $\bar{J}_{(x,y)}^{t+1}(u,v,j) = J_x^{t+1}(u,v) + J_y^{t+1}(v,j)$, (i.e., the expected total discounted cost obtained using the decomposed cost functions).

8.2.2 Algorithms

We present SDD algorithms for both finite and infinite horizon problems. These algorithms utilize the backward induction algorithm (or the successive approximation method) with the SDD approach.

Algorithm 8.2: The SDD for Finite-horizon

Step 1: Define a state space S and a decision space U (Proposition 7.3).

Step 2: Compute the p.m.f. and c.d.f. of w_n.

Step 3: For each possible value of $S_{M,n}$, compute the conditional p.m.f. and the conditional c.d.f. of z_n using expressions in Proposition 7-6. Also, for each possible value of $S_{M,n}$, compute the conditional p.m.f. and the conditional c.d.f. of z_n, given $w_n=r$. These conditional functions will be used in Step 7.

Step 4: Set n=N. Also, set $J_y^{(N,S_{M,N})}=\phi_N(y)$ for $y\epsilon S_y$ and $S_{M,N}$, and $J_x^N=\phi_N(x)$ for $x\epsilon S_x$, where $\phi_N(y)$ and $\phi_N(x)$ are terminal costs associated with the ending repairable and serviceable inventories, respectively.

Step 5: Set n=n-1. Compute transition probabilities (8-6) for each decision. For each $y\epsilon S_y$ and each $S_{M,n}$, compute $J_y^{(n,S_{M,n})}(v,j)$ using the cost function (8-5) and expression (8-7b). Similarly, compute transition probabilities (8-10) for each decision. For each $x\epsilon S_x$, compute $J_x^n(u,v)$ using the cost function (8-9) and expression (8-11b).

Step 6: Let $\bar{J}_{(x,y)}^{(n,S_{M,n})}(u,v,j) = J_x^n(u,v) + J_y^{(n,S_{M,n})}(v,j)$, where $\bar{J}_{(x,y)}^{(n,S_{M,n})}(u,v,j)$ is the expected total discounted cost obtained using the decomposed cost functions.

For each $(x,y)\epsilon S$,

$$\text{set } \bar{J}_{(x,y)}^{(n,S_{M,n})} = \min_{(u,v,j)} \bar{J}_{(x,y)}^{(n,S_{M,n})}(u,v,j) \quad \text{and} \quad \bar{\mu}_n = \arg \bar{J}_{(x,y)}^{(n,S_{M,n})}.$$

Step 7: Compute $\bar{J}_{(x,y)}^{(n,S_{M,n})}(\bar{\mu}_n)$ using the original cost function (7-1) and the original transition probabilities (7-6) given in Chapter 7.

Step 8: If n=0, stop. Otherwise, return to step 5.

Algorithm 8.3: The SDD for Infinite-horizon

Step 1: Define a state space S and a decision space U (Proposition 7.3).

Step 2: Compute the p.m.f. and c.d.f. of w_n, and compute the p.m.f. and c.d.f. of z_n using expressions in (7-13a) to (7-13d).

Step 3: Set t=0. Also, set $J_y^t = 0$ for all $y \epsilon S_y$ and $J_x^t = 0$ for all $x \epsilon S_x$.

Step 4: For each $y \epsilon S_y$, compute $J_y^{t+1}(v,j)$ using expression (8-8b) and the cost function (8-5), and the stationary transition probabilities of (8-6). For each $x \epsilon S_x$, compute $J_x^{t+1}(u,v)$ using expression (8-12b), the cost function (8-9), and the stationary transition probabilities of (8-10).

Step 5: Set $\tilde{J}_{(x,y)}^{t+1}(u,v,j) = J_x^{t+1}(u,v) + J_y^{t+1}(v,j)$, where $\tilde{J}_{(x,y)}^{t+1}(u,v,j)$ the expected total discounted cost obtained using the decomposed cost functions.

For each $(x,y) \epsilon S$, set $\tilde{J}_{(x,y)}^{t+1} = \min_{(u,v,j)} \tilde{J}_{(x,y)}^{t+1}(u,v,j)$.

Step 6: Compute $J_{(x,y)}^{t+1}$ using the original cost function (7-1) and the stationary transition probabilities of (7-12).

Step 7: If $|J_{(x,y)}^{t+1} - J_{(x,y)}^{t}| < \epsilon$ for all $(x,y) \epsilon S$, where ϵ is a stopping constant, go to Step 8. Otherwise, increment t by 1 and return to Step 4.

Step 8: $\tilde{\mu}_\infty = \arg J_{(x,y)}^{t+1}$ and stop.

8.2.3 The SDD with Error Bounds Approach

The error bounds approach discussed in Section 8.1 accelerates the convergence of the successive approximation method for infinite-horizon Markov decision problems. Although the approach usually yields an optimal stationary policy and the corresponding cost in a much less

number of iterations it does not reduce the amount of work required per iteration. The SDD reduces the amount of work required per iteration because it reduces the number of states (as well as the number of decisions) by converting a two-dimensional state problem into two one-dimensional subproblems. Thus, the SDD utilizing the error bounds approach will accelerate the convergence of the successive approximation method even faster for infinite-horizon problems.

8.2.4 Performance of The SDD

The SDD approximates the optimal solution to the repairable-item inventory problem: thus, it may not yield an optimal policy and/or the corresponding optimal cost. Performance of the SDD can be measured by the following factors: efficiency and effectiveness. Efficiency measures how much time savings the SDD achieves whereas effectiveness refers to the deviation from the optimal cost resulting from use of the SDD. Efficiency of the SDD rests on the following proposition.

Proposition 8.2: Consider the following two MDPs: (a) a two-dimensional state MDP with upper bounds $\bar{X}$ and $\bar{Y}$, and (b) two single-dimensional MDPs, each with upper bound $\bar{X}$ and $\bar{Y}$, respectively. Assume that the number of decisions to be considered is the same (K) in both problems. Using the computational algorithm 7.1 or 7.2, problem (b) requires less computational time and effort than problem (a) when $\bar{X}>2$ and $\bar{Y}>2$.

Proof: We compare efficiencies of the two problems based on the number of states and the size of the transition probability matrices required for computation.

# of states in problem (a) = $(\bar{X}+1)(\bar{Y}+1) = \bar{X}\bar{Y}+\bar{X}+\bar{Y}+1$.

# of states in problem (b) = $(\bar{X}+1)+(\bar{Y}+1) = \bar{X}+\bar{Y}+2$.

Thus, # of states in (a) > # of states in (b) for $\bar{X}>1$ and $\bar{Y}>1$.

Transition probabilities must be calculated for each and every decision and at each and every stage. Each transition probability matrix in problem (a) has $(\bar{X}+1)^2(\bar{Y}+1)^2$ elements, while each matrix in problem (b) has $(\bar{X}+\bar{Y}+2)^2$ elements. Thus, for the N-period MDP,

# of elements in (a) > # of elements in (b)

$N(\bar{X}+1)^2(\bar{Y}+1)^2 > N(\bar{X}+\bar{Y}+2)^2$ for for $\bar{X}>1$ and $\bar{Y}>1$.

Table II-5 measures performance of the SDD for a 5-period finite-horizon Markov decision problem (with different upper limits for the serviceable and repairable inventories). It shows the computation time required to find a solution based on a fixed sequence of demand over time, the average total expected discounted cost incurred per state with/without the SDD, the percentage savings in computation time (efficiency), and the percentage increase in average total expected discounted cost per state (effectiveness) due to use of the SDD. As seen in the table, the SDD results in tremendous savings in computation time (especially for the serviceable and repairable inventories with high upper limits) with little increase in cost. For a Markov decision problem with $(\bar{X},\bar{Y})=(20,20)$, the SDD reduces the computation time by approximately 86% while it increases total cost by only 0.86%. Figures II-6 to II-8 are graphical representations of its performance.

$(\bar{X},\bar{Y})$	# of STATES	TIME		TOTAL COST/STATE		% CHANGE	
		WITHOUT SDD	WITH SDD	WITHOUT SDD	WITH SDD	% ↓ TIME	% ↑ COST
(2,2)	9	6.70 s	6.20 s	62.93	62.93	7.5	0
(3,3)	16	20.30 s	17.40 s	55.66	56.34	14.3	1.22
(4,4)	25	58.30 s	42.80 s	78.63	78.63	26.6	0
(5,5)	36	2.36 m	1.45 m	75.17	75.25	38.6	0.11
(6,6)	49	5.20 m	2.77 m	95.65	95.65	46.8	0
(7,7)	64	10.36 m	4.72 m	93.21	93.21	54.5	0
(8,8)	81	19.43 m	7.85 m	113.10	113.24	59.6	0.12
(9,9)	100	34.17 m	12.06 m	110.80	110.94	64.7	0.13
(10,10)	121	57.73 m	18.37 m	130.93	131.40	68.2	0.36
(15,15)	256	7.84 h	1.59 h	206.60	208.42	79.7	0.88
(20,20)	441	38.62 h	5.42 h	300.71	303.29	86.0	0.86

Table II-5: Efficiency and Effectiveness of the SDD for a 5-period Finite-horizon MDP.

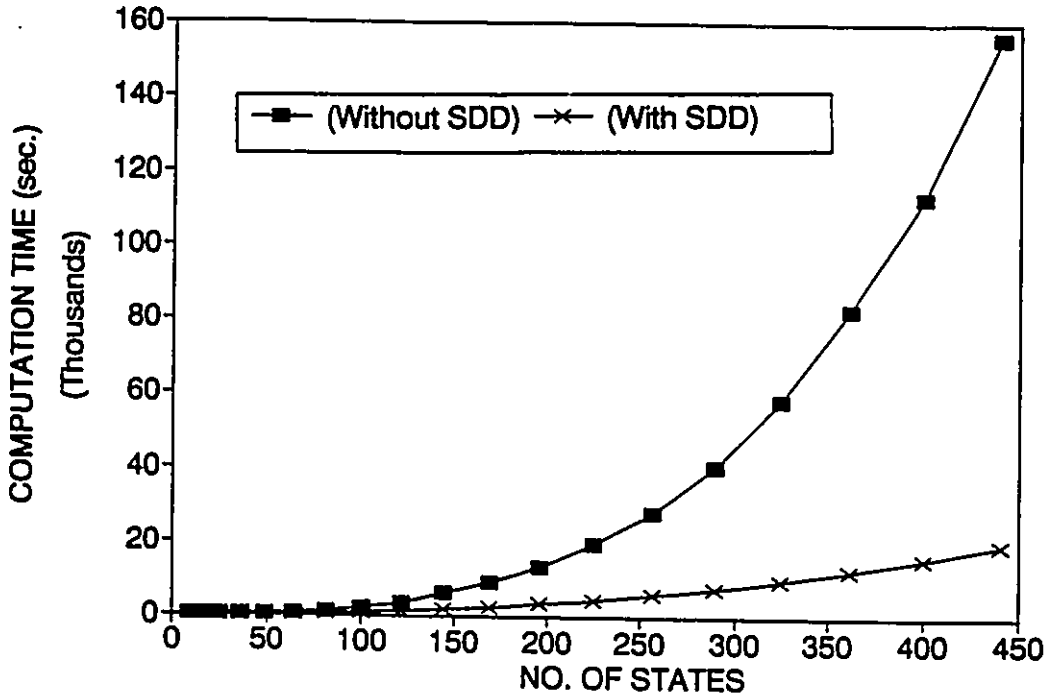


Figure II-6: Computation Time Required (without/with the SDD)

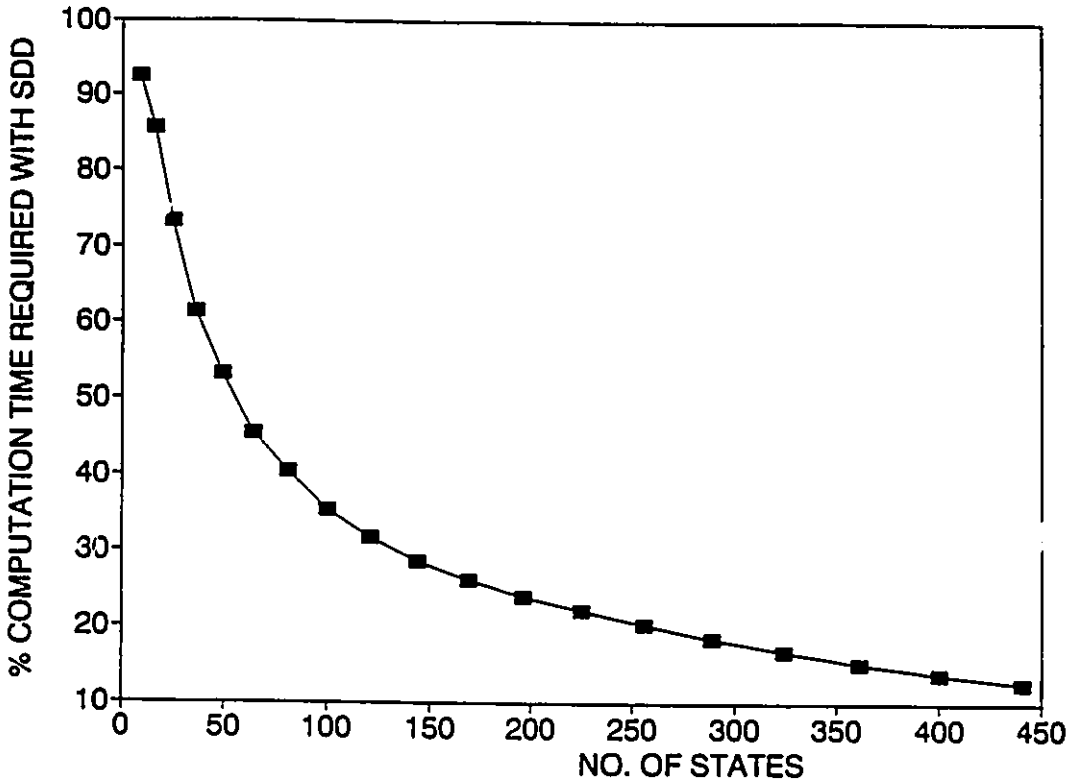


Figure II-7: Percentage of Computation Time Required with the SDD (vs. without the SDD)

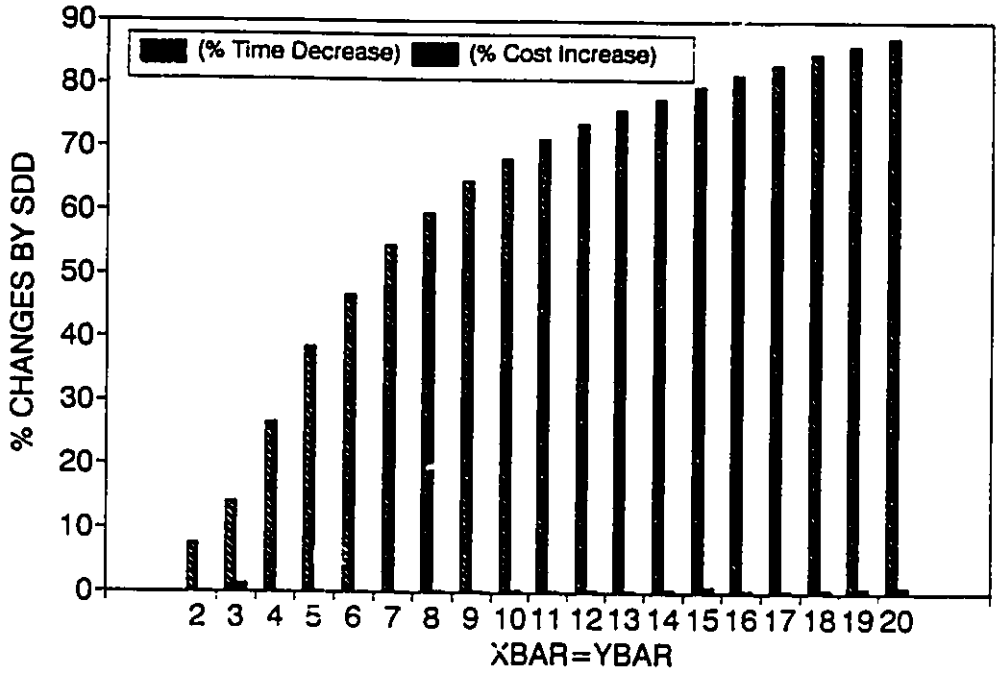


Figure II-8: Percentage Changes in Time and Cost by the SDD

The SDD also accelerates the convergence of the successive approximation method (SAM) for infinite-horizon Markov decision problems. Table II-6 measures the efficiencies of the two acceleration techniques, the error bounds approach and the SDD approach, for the infinite-horizon Markov decision problem using the example considered in Section 7.4.2. The error bounds approach and the SDD reduce the total computation time required prior to a convergence by approximately 93% and 39%, respectively. In fact, the SAM utilizing both acceleration techniques may reduce the computation time by almost 96%.

	WITHOUT ERROR BOUNDS	WITH ERROR BOUNDS	TIME SAVINGS
WITHOUT SDD	49.121 min.	3.313 min.	93.26 %
WITH SDD	30.098 min.	2.022 min.	93.28 %
TIME SAVINGS	38.73 %	38.97 %	95.88 %

Table II-6: Efficiency of the SDD and of the Error Bounds Approach for Infinite-horizon MDP with $(\bar{X},\bar{Y})=(5,5)$.

8.2.5 Numerical Examples

Two additional computer programs (Appendices II-H and II-I) were written to demonstrate the backward induction algorithm utilizing the SDD approach for finite-horizon problems and the method of successive

approximations utilizing the SDD approach for infinite-horizon problems. They terminate in a finite number of iterations with a near-optimal policy and the corresponding expected discounted cost. These programs require less storage and memory space than those introduced in Chapter 7 because the two-dimensional state space has been reduced to two one-dimensional state spaces. The program for infinite-horizon problems may also utilize the error bounds approach discussed in the previous section.

Let us recall the data previously used in Section 7.3.2, that is, N=10, α=.9, C=4, c=6, R=4, r=4, f=1, h=2, p=0, b=15, $\bar{X}$=3, $\bar{Y}$=2, M=2, and δ=.2. Tables II-7(i) to II-7(x) represent a near-optimal policy to the 10-period finite-horizon problem, $\tilde{\pi}_{10}=(\tilde{\mu}_0,\tilde{\mu}_1,\ldots,\tilde{\mu}_9)$, determined by the backward induction algorithm utilizing the SDD approach, where $\tilde{\mu}_n$ is the set of actions taken at the beginning of period n. As seen in the tables (and also in Tables II-2(i) to II-2(x)), the costs obtained by using the SDD approach are very close to the optimal costs although $\tilde{\mu}_n=\mu_n^*$ for n=9 and $\tilde{\mu}_n\neq\mu_n^*$ for n=0,1,..,8.

In Section 7.4.2, we have demonstrated that the successive approximation method (SAM) yields an optimal stationary policy, i.e., $\pi_\infty^*=(\mu^*,\ldots,\mu^*)$, for the infinite-horizon problem after 113 iterations. Table II-8 shows that the SAM utilizing the SDD technique also yields a stationary policy, $\tilde{\pi}_\infty=(\tilde{\mu},\ldots,\tilde{\mu})$, in the same number of iterations. We can see that this policy is indeed the optimal stationary policy, that is, $\tilde{\pi}_\infty=\pi_\infty^*$, where $\tilde{\mu}=\mu^*$, which minimizes the expected long-run total discounted cost. From the stationary policy, the set of solution parameters can also be determined as $\Phi_\infty=(\beta_\infty,\eta_\infty,\zeta_\infty)=(5,4,5)$. Although

both the SAM utilizing the SDD technique and the SAM without the SDD technique yield the optimal stationary policy in the same number of iterations, the former finds the policy much faster than the latter because it requires less computation time per iteration (as discussed in Section 8.2.3). Numerical examples show that the SAM utilizing the SDD technique reduces the computation time by approximately 39% (see Table II-6).

In Section 8.2.3, we also discussed a synergy of accelerating the convergence of the successive approximation method for infinite-horizon Markov decision problems which results from combining the SDD technique with the error bounds approach. Table II-9 shows how an optimal stationary policy and the corresponding optimal cost have been obtained using both the SDD technique and the error bounds approach. The SAM utilizing both the SDD technique and the error bounds approach yields the optimal stationary policy after eight iterations (i.e., t=8). Column (c) represents the optimal stationary policy, $\pi_{\infty}^{*}=(\mu^{*},\ldots,\mu^{*})$, and Columns (e) and (f) indicate the values of the lower and upper bounds representing the minimum expected long-run total discounted costs. The set of solution parameters is determined as $\Phi_{\infty}=(\beta_{\infty},\eta_{\infty},\zeta_{\infty})=(5,4,5)$, which is identical to the one obtained in Section 8.1.2. Although the SAM utilizing both the SDD technique and the error bounds approach requires one more iteration than the SAM utilizing only the error bounds approach before the convergence of the upper and lower bounds, the former finds the optimal policy much faster than the latter because it requires less computation time per iteration (as discussed in Section 8.2.4). Numerical examples show that the SAM utilizing both the SDD technique

and the error bounds approach reduces the total computation time by almost 94% (see Table II-6).

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
9	(0 0)	0	(2 0 0)	21.25
		1	(2 0 0)	21.25
		2	(2 0 0)	21.25
		3	(2 0 0)	21.25
		4	(2 0 0)	21.25
		5	(2 0 0)	21.25
		6	(2 0 0)	21.25
9	(0 1)	0	(0 1 0)	19.75
		1	(0 1 0)	19.75
		2	(0 1 0)	19.75
		3	(0 1 0)	19.75
		4	(0 1 0)	19.75
		5	(0 1 0)	19.75
		6	(0 1 0)	19.75
9	(0 2)	0	(0 2 0)	17.25
		1	(0 2 0)	17.25
		2	(0 2 0)	17.25
		3	(0 2 0)	17.25
		4	(0 2 0)	17.25
		5	(0 2 0)	17.25
		6	(0 2 0)	17.25
9	(1 0)	0	(0 0 0)	11.75
		1	(0 0 0)	11.75
		2	(0 0 0)	11.75
		3	(0 0 0)	11.75
		4	(0 0 0)	11.75
		5	(0 0 0)	11.75
		6	(0 0 0)	11.75
9	(1 1)	0	(0 0 1)	11.75
		1	(0 0 1)	11.75
		2	(0 0 1)	11.75
		3	(0 0 1)	11.75
		4	(0 0 1)	11.75
		5	(0 0 1)	11.75
		6	(0 0 1)	11.75
9	(1 2)	0	(0 0 2)	11.75
		1	(0 0 2)	11.75
		2	(0 0 2)	11.75
		3	(0 0 2)	11.75
		4	(0 0 2)	11.75
		5	(0 0 2)	11.75
		6	(0 0 2)	11.75
9	(2 0)	0	(0 0 0)	5.25
		1	(0 0 0)	5.25
		2	(0 0 0)	5.25
		3	(0 0 0)	5.25
		4	(0 0 0)	5.25
		5	(0 0 0)	5.25
		6	(0 0 0)	5.25
9	(2 1)	0	(0 0 1)	5.25
		1	(0 0 1)	5.25
		2	(0 0 1)	5.25
		3	(0 0 1)	5.25
		4	(0 0 1)	5.25
		5	(0 0 1)	5.25
		6	(0 0 1)	5.25
9	(2 2)	0	(0 0 2)	5.25
		1	(0 0 2)	5.25
		2	(0 0 2)	5.25
		3	(0 0 2)	5.25
		4	(0 0 2)	5.25
		5	(0 0 2)	5.25
		6	(0 0 2)	5.25
9	(3 0)	0	(0 0 0)	3.00
		1	(0 0 0)	3.00
		2	(0 0 0)	3.00
		3	(0 0 0)	3.00
		4	(0 0 0)	3.00
		5	(0 0 0)	3.00
		6	(0 0 0)	3.00
9	(3 1)	0	(0 0 1)	3.00
		1	(0 0 1)	3.00
		2	(0 0 1)	3.00
		3	(0 0 1)	3.00
		4	(0 0 1)	3.00
		5	(0 0 1)	3.00
		6	(0 0 1)	3.00
9	(3 2)	0	(0 0 2)	3.00
		1	(0 0 2)	3.00
		2	(0 0 2)	3.00
		3	(0 0 2)	3.00
		4	(0 0 2)	3.00
		5	(0 0 2)	3.00
		6	(0 0 2)	3.00

Table II-7(i): Solution to 10-Period Problem by SDD
(n=9; Last period of the planning horizon)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)	TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
8	(0 0)	0	**(2 0 0)***	34.27	8	(2 0)	0	(0 0 0)	18.27
		1	**(2 0 0)**	34.11			1	(0 0 0)	18.11
		2	**(2 0 0)**	33.96			2	(0 0 0)	17.96
		3	(2 0 0)	33.82			3	(0 0 0)	17.82
		4	(2 0 0)	33.69			4	(0 0 0)	17.69
		5	(2 0 0)	33.57			5	(0 0 0)	17.57
		6	(2 0 0)	33.47			6	(0 0 0)	17.47
8	(0 1)	0	(2 0 0)	34.49	8	(2 1)	0	(0 0 1)	18.27
		1	(2 0 0)	34.36			1	(0 0 1)	18.11
		2	(2 0 0)	34.25			2	(0 0 1)	17.96
		3	(2 0 0)	34.17			3	(0 0 1)	17.82
		4	(2 0 0)	34.10			4	(0 0 1)	17.69
		5	(2 0 0)	34.05			5	(0 0 1)	17.57
		6	(2 0 0)	34.01			6	(0 0 1)	17.47
8	(0 2)	0	(0 2 0)	30.27	8	(2 2)	0	(0 0 2)	18.27
		1	(0 2 0)	30.11			1	(0 0 2)	18.11
		2	(0 2 0)	29.96			2	(0 0 2)	17.96
		3	(0 2 0)	29.82			3	(0 0 2)	17.82
		4	(0 2 0)	29.69			4	(0 0 2)	17.69
		5	(0 2 0)	29.57			5	(0 0 2)	17.57
		6	(0 2 0)	29.47			6	(0 0 2)	17.47
8	(1 0)	0	**(0 0 0)**	28.30	8	(3 0)	0	(0 0 0)	12.06
		1	**(0 0 0)**	28.07			1	(0 0 0)	11.98
		2	(0 0 0)	27.84			2	(0 0 0)	11.91
		3	(0 0 0)	27.62			3	(0 0 0)	11.84
		4	(0 0 0)	27.42			4	(0 0 0)	11.78
		5	(0 0 0)	27.23			5	(0 0 0)	11.72
		6	(0 0 0)	27.06			6	(0 0 0)	11.67
8	(1 1)	0	(0 1 0)	26.27	8	(3 1)	0	(0 0 1)	12.06
		1	(0 1 0)	26.11			1	(0 0 1)	11.98
		2	(0 1 0)	25.96			2	(0 0 1)	11.91
		3	(0 1 0)	25.82			3	(0 0 1)	11.84
		4	(0 1 0)	25.69			4	(0 0 1)	11.78
		5	(0 1 0)	25.57			5	(0 0 1)	11.72
		6	(0 1 0)	25.47			6	(0 0 1)	11.67
8	(1 2)	0	(0 2 0)	24.06	8	(3 2)	0	(0 0 2)	12.06
		1	(0 2 0)	23.98			1	(0 0 2)	11.98
		2	(0 2 0)	23.91			2	(0 0 2)	11.91
		3	(0 2 0)	23.84			3	(0 0 2)	11.84
		4	(0 2 0)	23.78			4	(0 0 2)	11.78
		5	(0 2 0)	23.72			5	(0 0 2)	11.72
		6	(0 2 0)	23.67			6	(0 0 2)	11.67

* Decisions in **bold** are different than those in Table II-2(ii).

Table II-7(ii): Solution to 10-Period Problem by SDD (n=8)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
7	(0 0)	0	(3 0 0)	45.29
		1	(3 0 0)	45.14
		2	(3 0 0)	44.99
		3	(3 0 0)	44.85
		4	(3 0 0)	44.71
		5	(3 0 0)	44.59
		6	(3 0 0)	44.47
7	(0 1)	0	(3 0 0)	45.55
		1	(3 0 0)	45.39
		2	(3 0 0)	45.27
		3	(3 0 0)	45.17
		4	(3 0 0)	45.08
		5	(3 0 0)	45.02
		6	(3 0 0)	44.97
7	(0 2)	0	(0 2 0)	42.56
		1	(0 2 0)	42.36
		2	(0 2 0)	42.14
		3	(0 2 0)	41.91
		4	(0 2 0)	41.69
		5	(0 2 0)	41.49
		6	(0 2 0)	41.30
7	(1 0)	0	**(0 0 0)**	40.76
		1	**(0 0 0)**	40.53
		2	**(0 0 0)**	40.26
		3	**(0 0 0)**	39.96
		4	**(0 0 0)**	39.66
		5	**(0 0 0)**	39.37
		6	**(0 0 0)**	39.11
7	(1 1)	0	(0 1 0)	38.56
		1	(0 1 0)	38.36
		2	(0 1 0)	38.14
		3	(0 1 0)	37.91
		4	(0 1 0)	37.69
		5	(0 1 0)	37.49
		6	(0 1 0)	37.30
7	(1 2)	0	(0 2 0)	35.29
		1	(0 2 0)	35.14
		2	(0 2 0)	34.99
		3	(0 2 0)	34.85
		4	(0 2 0)	34.71
		5	(0 2 0)	34.59
		6	(0 2 0)	34.47
7	(2 0)	0	(0 0 0)	30.56
		1	(0 0 0)	30.36
		2	(0 0 0)	30.14
		3	(0 0 0)	29.91
		4	(0 0 0)	29.69
		5	(0 0 0)	29.49
		6	(0 0 0)	29.30
7	(2 1)	0	(0 0 1)	30.56
		1	**(0 0 1)**	30.36
		2	**(0 0 1)**	30.14
		3	**(0 0 1)**	29.91
		4	**(0 0 1)**	29.69
		5	(0 0 1)	29.49
		6	(0 0 1)	29.30
7	(2 2)	0	**(0 0 2)**	30.56
		1	**(0 0 2)**	30.36
		2	**(0 0 2)**	30.14
		3	**(0 0 2)**	29.91
		4	**(0 0 2)**	29.69
		5	(0 0 2)	29.49
		6	(0 0 2)	29.30
7	(3 0)	0	(0 0 0)	23.29
		1	(0 0 0)	23.14
		2	(0 0 0)	22.99
		3	(0 0 0)	22.85
		4	(0 0 0)	22.71
		5	(0 0 0)	22.59
		6	(0 0 0)	22.47
7	(3 1)	0	(0 0 1)	23.29
		1	(0 0 1)	23.14
		2	(0 0 1)	22.99
		3	(0 0 1)	22.85
		4	(0 0 1)	22.71
		5	(0 0 1)	22.59
		6	(0 0 1)	22.47
7	(3 2)	0	(0 0 2)	23.29
		1	(0 0 2)	23.14
		2	(0 0 2)	22.99
		3	(0 0 2)	22.85
		4	(0 0 2)	22.71
		5	(0 0 2)	22.59
		6	(0 0 2)	22.47

Table II-7(iii): Solution to 10-Period Problem by SDD (n=7)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
6	(0 0)	0	(3 0 0)	55.73
		1	(3 0 0)	55.58
		2	(3 0 0)	55.42
		3	(3 0 0)	55.27
		4	(3 0 0)	55.13
		5	(3 0 0)	55.00
		6	(3 0 0)	54.89
6	(0 1)	0	(3 0 0)	55.97
		1	(3 0 0)	55.81
		2	(3 0 0)	55.68
		3	(3 0 0)	55.57
		4	(3 0 0)	55.49
		5	(3 0 0)	55.42
		6	(3 0 0)	55.37
6	(0 2)	0	(0 2 0)	53.03
		1	(0 2 0)	52.84
		2	(0 2 0)	52.63
		3	(0 2 0)	52.43
		4	(0 2 0)	52.22
		5	(0 2 0)	52.03
		6	(0 2 0)	51.85
6	(1 0)	0	**(0 0 0)**	51.00
		1	**(0 0 0)**	50.80
		2	**(0 0 0)**	50.56
		3	**(0 0 0)**	50.30
		4	**(0 0 0)**	50.04
		5	**(0 0 0)**	49.78
		6	**(0 0 0)**	49.54
6	(1 1)	0	(0 1 0)	49.03
		1	(0 1 0)	48.84
		2	(0 1 0)	48.63
		3	(0 1 0)	48.43
		4	(0 1 0)	48.22
		5	(0 1 0)	48.03
		6	(0 1 0)	47.85
6	(1 2)	0	(0 2 0)	45.73
		1	(0 2 0)	45.58
		2	(0 2 0)	45.42
		3	(0 2 0)	45.27
		4	(0 2 0)	45.13
		5	(0 2 0)	45.00
		6	(0 2 0)	44.89
6	(2 0)	0	(0 0 0)	41.03
		1	(0 0 0)	40.84
		2	(0 0 0)	40.63
		3	(0 0 0)	40.43
		4	(0 0 0)	40.22
		5	(0 0 0)	40.03
		6	(0 0 0)	39.85
6	(2 1)	0	(0 0 1)	41.03
		1	(0 0 1)	40.84
		2	(0 0 1)	40.63
		3	(0 0 1)	40.43
		4	(0 0 1)	40.22
		5	(0 0 1)	40.03
		6	(0 0 1)	39.85
6	(2 2)	0	**(0 0 2)**	41.03
		1	**(0 0 2)**	40.84
		2	(0 0 2)	40.63
		3	(0 0 2)	40.43
		4	(0 0 2)	40.22
		5	(0 0 2)	40.03
		6	(0 0 2)	39.85
6	(3 0)	0	(0 0 0)	33.73
		1	(0 0 0)	33.58
		2	(0 0 0)	33.42
		3	(0 0 0)	33.27
		4	(0 0 0)	33.13
		5	(0 0 0)	33.00
		6	(0 0 0)	32.89
6	(3 1)	0	(0 0 1)	33.73
		1	(0 0 1)	33.58
		2	(0 0 1)	33.42
		3	(0 0 1)	33.27
		4	(0 0 1)	33.13
		5	(0 0 1)	33.00
		6	(0 0 1)	32.89
6	(3 2)	0	(0 0 2)	33.73
		1	(0 0 2)	33.58
		2	(0 0 2)	33.42
		3	(0 0 2)	33.27
		4	(0 0 2)	33.13
		5	(0 0 2)	33.00
		6	(0 0 2)	32.89

Table II-7(iv): Solution to 10-Period Problem by SDD (n=6)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)	TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
5	(0 0)	0	(3 0 0)	65.13	5	(2 0)	0	(0 0 0)	50.42
		1	(3 0 0)	64.98			1	(0 0 0)	50.24
		2	(3 0 0)	64.83			2	(0 0 0)	50.04
		3	(3 0 0)	64.68			3	(0 0 0)	49.84
		4	(3 0 0)	64.55			4	(0 0 0)	49.64
		5	(3 0 0)	64.42			5	(0 0 0)	49.45
		6	(3 0 0)	64.30			6	(0 0 0)	49.28
5	(0 1)	0	(3 0 0)	65.39	5	(2 1)	0	(0 0 1)	50.42
		1	(3 0 0)	65.23			1	(0 0 1)	50.24
		2	(3 0 0)	65.10			2	(0 0 1)	50.04
		3	(3 0 0)	65.00			3	(0 0 1)	49.84
		4	(3 0 0)	64.91			4	(0 0 1)	49.64
		5	(3 0 0)	64.84			5	(0 0 1)	49.45
		6	(3 0 0)	64.79			6	(0 0 1)	49.28
5	(0 2)	0	(0 2 0)	62.42	5	(2 2)	0	**(0 0 2)**	50.42
		1	(0 2 0)	62.24			1	(0 0 2)	50.24
		2	(0 2 0)	62.04			2	(0 0 2)	50.04
		3	(0 2 0)	61.84			3	(0 0 2)	49.84
		4	(0 2 0)	61.64			4	(0 0 2)	49.64
		5	(0 2 0)	61.45			5	(0 0 2)	49.45
		6	(0 2 0)	61.28			6	(0 0 2)	49.28
5	(1 0)	0	**(0 0 0)**	60.36	5	(3 0)	0	(0 0 0)	43.13
		1	**(0 0 0)**	60.17			1	(0 0 0)	42.98
		2	**(0 0 0)**	59.94			2	(0 0 0)	42.83
		3	**(0 0 0)**	59.69			3	(0 0 0)	42.68
		4	**(0 0 0)**	59.43			4	(0 0 0)	42.55
		5	**(0 0 0)**	59.19			5	(0 0 0)	42.42
		6	**(0 0 0)**	58.95			6	(0 0 0)	42.30
5	(1 1)	0	(0 1 0)	58.42	5	(3 1)	0	(0 0 1)	43.13
		1	(0 1 0)	58.24			1	(0 0 1)	42.98
		2	(0 1 0)	58.04			2	(0 0 1)	42.83
		3	(0 1 0)	57.84			3	(0 0 1)	42.68
		4	(0 1 0)	57.64			4	(0 0 1)	42.55
		5	(0 1 0)	57.45			5	(0 0 1)	42.42
		6	(0 1 0)	57.28			6	(0 0 1)	42.30
5	(1 2)	0	(0 2 0)	55.13	5	(3 2)	0	(0 0 2)	43.13
		1	(0 2 0)	54.98			1	(0 0 2)	42.98
		2	(0 2 0)	54.83			2	(0 0 2)	42.83
		3	(0 2 0)	54.68			3	(0 0 2)	42.68
		4	(0 2 0)	54.55			4	(0 0 2)	42.55
		5	(0 2 0)	54.42			5	(0 0 2)	42.42
		6	(0 2 0)	54.30			6	(0 0 2)	42.30

:ble II-7(v): Solution to 10-Period Problem by SDD (n=5)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
4	(0 0)	0	(3 0 0)	73.59
		1	(3 0 0)	73.45
		2	(3 0 0)	73.30
		3	(3 0 0)	73.15
		4	(3 0 0)	73.01
		5	(3 0 0)	72.89
		6	(3 0 0)	72.77
4	(0 1)	0	(3 0 0)	73.86
		1	(3 0 0)	73.70
		2	(3 0 0)	73.57
		3	(3 0 0)	73.46
		4	(3 0 0)	73.38
		5	(3 0 0)	73.31
		6	(3 0 0)	73.26
4	(0 2)	0	(0 2 0)	70.89
		1	(0 2 0)	70.71
		2	(0 2 0)	70.51
		3	(0 2 0)	70.31
		4	(0 2 0)	70.11
		5	(0 2 0)	69.92
		6	(0 2 0)	69.75
4	(1 0)	0	**(0 0 0)**	68.83
		1	**(0 0 0)**	68.64
		2	**(0 0 0)**	68.41
		3	**(0 0 0)**	68.16
		4	**(0 0 0)**	67.90
		5	**(0 0 0)**	67.66
		6	**(0 0 0)**	67.42
4	(1 1)	0	(0 1 0)	66.89
		1	(0 1 0)	66.71
		2	(0 1 0)	66.51
		3	(0 1 0)	66.31
		4	(0 1 0)	66.11
		5	(0 1 0)	65.92
		6	(0 1 0)	65.75
4	(1 2)	0	(0 2 0)	63.59
		1	(0 2 0)	63.45
		2	(0 2 0)	63.30
		3	(0 2 0)	63.15
		4	(0 2 0)	63.01
		5	(0 2 0)	62.89
		6	(0 2 0)	62.77
4	(2 0)	0	(0 0 0)	58.89
		1	(0 0 0)	58.71
		2	(0 0 0)	58.51
		3	(0 0 0)	58.31
		4	(0 0 0)	58.11
		5	(0 0 0)	57.92
		6	(0 0 0)	57.75
4	(2 1)	0	(0 0 1)	58.89
		1	(0 0 1)	58.71
		2	(0 0 1)	58.51
		3	(0 0 1)	58.31
		4	(0 0 1)	58.11
		5	(0 0 1)	57.92
		6	(0 0 1)	57.75
4	(2 2)	0	**(0 0 2)**	58.89
		1	(0 0 2)	58.71
		2	(0 0 2)	58.51
		3	(0 0 2)	58.31
		4	(0 0 2)	58.11
		5	(0 0 2)	57.92
		6	(0 0 2)	57.75
4	(3 0)	0	(0 0 0)	51.59
		1	(0 0 0)	51.45
		2	(0 0 0)	51.30
		3	(0 0 0)	51.15
		4	(0 0 0)	51.01
		5	(0 0 0)	50.89
		6	(0 0 0)	50.77
4	(3 1)	0	(0 0 1)	51.59
		1	(0 0 1)	51.45
		2	(0 0 1)	51.30
		3	(0 0 1)	51.15
		4	(0 0 1)	51.01
		5	(0 0 1)	50.89
		6	(0 0 1)	50.77
4	(3 2)	0	(0 0 2)	51.59
		1	(0 0 2)	51.45
		2	(0 0 2)	51.30
		3	(0 0 2)	51.15
		4	(0 0 2)	51.01
		5	(0 0 2)	50.89
		6	(0 0 2)	50.77

Table II-7(vi): Solution to 10-Period Problem by SDD (n=4)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
3	(0 0)	0	(3 0 0)	81.21
		1	(3 0 0)	81.07
		2	(3 0 0)	80.92
		3	(3 0 0)	80.77
		4	(3 0 0)	80.64
		5	(3 0 0)	80.51
		6	(3 0 0)	80.39
3	(0 1)	0	(3 0 0)	81.49
		1	(3 0 0)	81.32
		2	(3 0 0)	81.19
		3	(3 0 0)	81.09
		4	(3 0 0)	81.00
		5	(3 0 0)	80.93
		6	(3 0 0)	80.88
3	(0 2)	0	(0 2 0)	78.51
		1	(0 2 0)	78.33
		2	(0 2 0)	78.13
		3	(0 2 0)	77.93
		4	(0 2 0)	77.73
		5	(0 2 0)	77.54
		6	(0 2 0)	77.37
3	(1 0)	0	**(0 0 0)**	76.45
		1	**(0 0 0)**	76.26
		2	**(0 0 0)**	76.03
		3	**(0 0 0)**	75.78
		4	**(0 0 0)**	75.53
		5	**(0 0 0)**	75.28
		6	**(0 0 0)**	75.05
3	(1 1)	0	(0 1 0)	74.51
		1	(0 1 0)	74.33
		2	(0 1 0)	74.13
		3	(0 1 0)	73.93
		4	(0 1 0)	73.73
		5	(0 1 0)	73.54
		6	(0 1 0)	73.37
3	(1 2)	0	(0 2 0)	71.21
		1	(0 2 0)	71.07
		2	(0 2 0)	70.92
		3	(0 2 0)	70.77
		4	(0 2 0)	70.64
		5	(0 2 0)	70.51
		6	(0 2 0)	70.39
3	(2 0)	0	(0 0 0)	66.51
		1	(0 0 0)	66.33
		2	(0 0 0)	66.13
		3	(0 0 0)	65.93
		4	(0 0 0)	65.73
		5	(0 0 0)	65.54
		6	(0 0 0)	65.37
3	(2 1)	0	(0 0 1)	66.51
		1	(0 0 1)	66.33
		2	(0 0 1)	66.13
		3	(0 0 1)	65.93
		4	(0 0 1)	65.73
		5	(0 0 1)	65.54
		6	(0 0 1)	65.37
3	(2 2)	0	**(0 0 2)**	66.51
		1	(0 0 2)	66.33
		2	(0 0 2)	66.13
		3	(0 0 2)	65.93
		4	(0 0 2)	65.73
		5	(0 0 2)	65.54
		6	(0 0 2)	65.37
3	(3 0)	0	(0 0 0)	59.21
		1	(0 0 0)	59.07
		2	(0 0 0)	58.92
		3	(0 0 0)	58.77
		4	(0 0 0)	58.64
		5	(0 0 0)	58.51
		6	(0 0 0)	58.39
3	(3 1)	0	(0 0 1)	59.21
		1	(0 0 1)	59.07
		2	(0 0 1)	58.92
		3	(0 0 1)	58.77
		4	(0 0 1)	58.64
		5	(0 0 1)	58.51
		6	(0 0 1)	58.39
3	(3 2)	0	(0 0 2)	59.21
		1	(0 0 2)	59.07
		2	(0 0 2)	58.92
		3	(0 0 2)	58.77
		4	(0 0 2)	58.64
		5	(0 0 2)	58.51
		6	(0 0 2)	53.39

Table II-7(vii): Solution to 10-Period Problem by SDD (n=3)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
2	(0 0)	0	(3 0 0)	88.07
		1	(3 0 0)	87.93
		2	(3 0 0)	87.78
		3	(3 0 0)	87.63
		4	(3 0 0)	87.50
		5	(3 0 0)	87.37
		6	(3 0 0)	87.25
2	(0 1)	0	(3 0 0)	88.35
		1	(3 0 0)	88.18
		2	(3 0 0)	88.05
		3	(3 0 0)	87.95
		4	(3 0 0)	87.86
		5	(3 0 0)	87.79
		6	(3 0 0)	87.74
2	(0 2)	0	(0 2 0)	85.37
		1	(0 2 0)	85.19
		2	(0 2 0)	84.99
		3	(0 2 0)	84.79
		4	(0 2 0)	84.59
		5	(0 2 0)	84.40
		6	(0 2 0)	84.23
2	(1 0)	0	**(0 0 0)**	83.31
		1	**(0 0 0)**	83.12
		2	**(0 0 0)**	82.89
		3	**(0 0 0)**	82.64
		4	**(0 0 0)**	82.39
		5	**(0 0 0)**	82.14
		6	**(0 0 0)**	81.91
2	(1 1)	0	(0 1 0)	81.37
		1	(0 1 0)	81.19
		2	(0 1 0)	80.99
		3	(0 1 0)	80.79
		4	(0 1 0)	80.59
		5	(0 1 0)	80.40
		6	(0 1 0)	80.23
2	(1 2)	0	(0 2 0)	78.07
		1	(0 2 0)	77.93
		2	(0 2 0)	77.78
		3	(0 2 0)	77.63
		4	(0 2 0)	77.50
		5	(0 2 0)	77.37
		6	(0 2 0)	77.25
2	(2 0)	0	(0 0 0)	73.37
		1	(0 0 0)	73.19
		2	(0 0 0)	72.99
		3	(0 0 0)	72.79
		4	(0 0 0)	72.59
		5	(0 0 0)	72.40
		6	(0 0 0)	72.23
2	(2 1)	0	(0 0 1)	73.37
		1	(0 0 1)	73.19
		2	(0 0 1)	72.99
		3	(0 0 1)	72.79
		4	(0 0 1)	72.59
		5	(0 0 1)	72.40
		6	(0 0 1)	72.23
2	(2 2)	0	**(0 0 2)**	73.37
		1	(0 0 2)	73.19
		2	(0 0 2)	72.99
		3	(0 0 2)	72.79
		4	(0 0 2)	72.59
		5	(0 0 2)	72.40
		6	(0 0 2)	72.23
2	(3 0)	0	(0 0 0)	66.07
		1	(0 0 0)	65.93
		2	(0 0 0)	65.78
		3	(0 0 0)	65.63
		4	(0 0 0)	65.50
		5	(0 0 0)	65.37
		6	(0 0 0)	65.25
2	(3 1)	0	(0 0 1)	66.07
		1	(0 0 1)	65.93
		2	(0 0 1)	65.78
		3	(0 0 1)	65.63
		4	(0 0 1)	65.50
		5	(0 0 1)	65.37
		6	(0 0 1)	65.25
2	(3 2)	0	(0 0 2)	66.07
		1	(0 0 2)	65.93
		2	(0 0 2)	65.78
		3	(0 0 2)	65.63
		4	(0 0 2)	65.50
		5	(0 0 2)	65 37
		6	(0 0 2)	65.25

Table II-7(viii): Solution to 10-Period Problem by SDD (n=2)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
1	(0 0)	0	(3 0 0)	94.25
		1	(3 0 0)	94.10
		2	(3 0 0)	93.95
		3	(3 0 0)	93.81
		4	(3 0 0)	93.67
		5	(3 0 0)	93.54
		6	(3 0 0)	93.43
1	(0 1)	0	(3 0 0)	94.52
		1	(3 0 0)	94.36
		2	(3 0 0)	94.22
		3	(3 0 0)	94.12
		4	(3 0 0)	94.03
		5	(3 0 0)	93.97
		6	(3 0 0)	93.91
1	(0 2)	0	(0 2 0)	91.54
		1	(0 2 0)	91.36
		2	(0 2 0)	91.16
		3	(0 2 0)	90.96
		4	(0 2 0)	90.76
		5	(0 2 0)	90.58
		6	(0 2 0)	90.40
1	(1 0)	0	**(0 0 0)**	89.48
		1	**(0 0 0)**	89.30
		2	**(0 0 0)**	89.07
		3	**(0 0 0)**	88.82
		4	**(0 0 0)**	88.56
		5	**(0 0 0)**	88.31
		6	**(0 0 0)**	88.08
1	(1 1)	0	(0 1 0)	87.54
		1	(0 1 0)	87.36
		2	(0 1 0)	87.16
		3	(0 1 0)	86.96
		4	(0 1 0)	86.76
		5	(0 1 0)	86.58
		6	(0 1 0)	86.40
1	(1 2)	0	(0 2 0)	84.25
		1	(0 2 0)	84.10
		2	(0 2 0)	83.95
		3	(0 2 0)	83.81
		4	(0 2 0)	83.67
		5	(0 2 0)	83.54
		6	(0 2 0)	83.43
1	(2 0)	0	(0 0 0)	79.54
		1	(0 0 0)	79.36
		2	(0 0 0)	79.16
		3	(0 0 0)	78.96
		4	(0 0 0)	78.76
		5	(0 0 0)	78.58
		6	(0 0 0)	78.40
1	(2 1)	0	(0 0 1)	79.54
		1	(0 0 1)	79.36
		2	(0 0 1)	79.16
		3	(0 0 1)	78.96
		4	(0 0 1)	78.76
		5	(0 0 1)	78.58
		6	(0 0 1)	78.40
1	(2 2)	0	**(0 0 2)**	79.54
		1	(0 0 2)	79.36
		2	(0 0 2)	79.16
		3	(0 0 2)	78.96
		4	(0 0 2)	78.76
		5	(0 0 2)	78.58
		6	(0 0 2)	78.40
1	(3 0)	0	(0 0 0)	72.25
		1	(0 0 0)	72.10
		2	(0 0 0)	71.95
		3	(0 0 0)	71.81
		4	(0 0 0)	71.67
		5	(0 0 0)	71.54
		6	(0 0 0)	71.43
1	(3 1)	0	(0 0 1)	72.25
		1	(0 0 1)	72.10
		2	(0 0 1)	71.95
		3	(0 0 1)	71.81
		4	(0 0 1)	71.67
		5	(0 0 1)	71.54
		6	(0 0 1)	71.43
1	(3 2)	0	(0 0 2)	72.25
		1	(0 0 2)	72.10
		2	(0 0 2)	71.95
		3	(0 0 2)	71.81
		4	(0 0 2)	71.67
		5	(0 0 2)	71.54
		6	(0 0 2)	71.43

Table II-7(ix): Solution to 10-Period Problem by SDD (n=1)

TIME n (a)	STATE (x,y) (b)	$S_{M,n}$ (c)	DECISION (u,v,j) (d)	E.T. D.C. (e)
0	(0 0)	0	(3 0 0)	99.80
		1	(3 0 0)	99.66
		2	(3 0 0)	99.51
		3	(3 0 0)	99.36
		4	(3 0 0)	99.23
		5	(3 0 0)	99.10
		6	(3 0 0)	98.98
0	(0 1)	0	(3 0 0)	100.08
		1	(3 0 0)	99.91
		2	(3 0 0)	99.78
		3	(3 0 0)	99.67
		4	(3 0 0)	99.59
		5	(3 0 0)	99.52
		6	(3 0 0)	99.47
0	(0 2)	0	(0 2 0)	97.10
		1	(0 2 0)	96.92
		2	(0 2 0)	96.72
		3	(0 2 0)	96.52
		4	(0 2 0)	96.32
		5	(0 2 0)	96.13
		6	(0 2 0)	95.96
0	(1 0)	0	**(0 0 0)**	95.04
		1	**(0 0 0)**	94.85
		2	**(0 0 0)**	94.62
		3	**(0 0 0)**	94.37
		4	**(0 0 0)**	94.12
		5	**(0 0 0)**	93.87
		6	**(0 0 0)**	93.64
0	(1 1)	0	(0 1 0)	93.10
		1	(0 1 0)	92.92
		2	(0 1 0)	92.72
		3	(0 1 0)	92.52
		4	(0 1 0)	92.32
		5	(0 1 0)	92.13
		6	(0 1 0)	91.96
0	(1 2)	0	(0 2 0)	89.80
		1	(0 2 0)	89.66
		2	(0 2 0)	89.51
		3	(0 2 0)	89.36
		4	(0 2 0)	89.23
		5	(0 2 0)	89.10
		6	(0 2 0)	88.98
0	(2 0)	0	(0 0 0)	85.10
		1	(0 0 0)	84.92
		2	(0 0 0)	84.72
		3	(0 0 0)	84.52
		4	(0 0 0)	84.32
		5	(0 0 0)	84.13
		6	(0 0 0)	83.96
0	(2 1)	0	(0 0 1)	85.10
		1	(0 0 1)	84.92
		2	(0 0 1)	84.72
		3	(0 0 1)	84.52
		4	(0 0 1)	84.32
		5	(0 0 1)	84.13
		6	(0 0 1)	83.96
0	(2 2)	0	**(0 0 2)**	85.10
		1	(0 0 2)	84.92
		2	(0 0 2)	84.72
		3	(0 0 2)	84.52
		4	(0 0 2)	84.32
		5	(0 0 2)	84.13
		6	(0 0 2)	83.96
0	(3 0)	0	(0 0 0)	77.80
		1	(0 0 0)	77.66
		2	(0 0 0)	77.51
		3	(0 0 0)	77.36
		4	(0 0 0)	77.23
		5	(0 0 0)	77.10
		6	(0 0 0)	76.98
0	(3 1)	0	(0 0 1)	77.80
		1	(0 0 1)	77.66
		2	(0 0 1)	77.51
		3	(0 0 1)	77.36
		4	(0 0 1)	77.23
		5	(0 0 1)	77.10
		6	(0 0 1)	76.98
0	(3 2)	0	(0 0 2)	77.80
		1	(0 0 2)	77.66
		2	(0 0 2)	77.51
		3	(0 0 2)	77.36
		4	(0 0 2)	77.23
		5	(0 0 2)	77.10
		6	(0 0 2)	76.98

Table II-7(x): Solution to 10-Period Problem by SDD
(n=0; Beginning of the planning horizon)

PERIOD t (a)	STATE (x,y) (b)	DECISION (u,v,j) (c)	EXPECTED TOTAL DISCOUNTED COST (d)*
113	(0 0)	(4 0 0)	214.90
113	(0 1)	(3 1 0)	212.90
113	(0 2)	(2 2 0)	210.90
113	(0 3)	(1 3 0)	208.90
113	(0 4)	(0 4 0)	206.90
113	(0 5)	(0 5 0)	205.89
113	(1 0)	(3 0 0)	208.90
113	(1 1)	(2 1 0)	206.90
113	(1 2)	(1 2 0)	204.90
113	(1 3)	(0 3 0)	202.90
113	(1 4)	(0 4 0)	201.89
113	(1 5)	(0 4 1)	201.89
113	(2 0)	(2 0 0)	202.90
113	(2 1)	(1 1 0)	200.90
113	(2 2)	(0 2 0)	198.90
113	(2 3)	(0 3 0)	197.89
113	(2 4)	(0 3 1)	197.89
113	(2 5)	(0 3 2)	197.89
113	(3 0)	(1 0 0)	196.90
113	(3 1)	(0 1 0)	194.90
113	(3 2)	(0 2 0)	193.89
113	(3 3)	(0 2 1)	193.89
113	(3 4)	(0 2 2)	193.89
113	(3 5)	(0 2 3)	193.89
113	(4 0)	(0 0 0)	190.90
113	(4 1)	(0 1 0)	189.89
113	(4 2)	(0 1 1)	189.89
113	(4 3)	(0 1 2)	189.89
113	(4 4)	(0 1 3)	189.89
113	(4 5)	(0 1 4)	189.89
113	(5 0)	(0 0 0)	185.89
113	(5 1)	(0 0 1)	185.89
113	(5 2)	(0 0 2)	185.89
113	(5 3)	(0 0 3)	185.89
113	(5 4)	(0 0 4)	185.89
113	(5 5)	(0 0 5)	185.89

* Minimum expected long-run total discounted cost

Table II-8: Solution to Infinite-horizon Problem with SDD (for t=113; N=∞).

PERIOD t (a)	STATE (x,y) (b)	DECISION (u,v,j) (c)	EXPECTED TOTAL DISCOUNTED COST (d)	LOWER BOUND (e)*	UPPER BOUND (f)*
8	(0 0)	(4 0 0)	126.81	214.90	214.90
8	(0 1)	(3 1 0)	124.81	212.90	212.90
8	(0 2)	(2 2 0)	122.81	210.90	210.90
8	(0 3)	(1 3 0)	120.81	208.90	208.90
8	(0 4)	(0 4 0)	118.81	206.90	206.90
8	(0 5)	(0 5 0)	117.79	205.89	205.89
8	(1 0)	(3 0 0)	120.81	208.90	208.90
8	(1 1)	(2 1 0)	118.81	206.90	206.90
8	(1 2)	(1 2 0)	116.81	204.90	204.90
8	(1 3)	(0 3 0)	114.81	202.90	202.90
8	(1 4)	(0 4 0)	113.79	201.89	201.89
8	(1 5)	(0 4 1)	113.79	201.89	201.89
8	(2 0)	(2 0 0)	114.81	202.90	202.90
8	(2 1)	(1 1 0)	112.81	200.90	200.90
8	(2 2)	(0 2 0)	110.81	198.90	198.90
8	(2 3)	(0 3 0)	109.79	197.89	197.89
8	(2 4)	(0 3 1)	109.79	197.89	197.89
8	(2 5)	(0 3 2)	109.79	197.89	197.89
8	(3 0)	(1 0 0)	108.81	196.90	196.90
8	(3 1)	(0 1 0)	106.81	194.90	194.90
8	(3 2)	(0 2 0)	105.79	193.89	193.89
8	(3 3)	(0 2 1)	105.79	193.89	193.89
8	(3 4)	(0 2 2)	105.79	193.89	193.89
8	(3 5)	(0 2 3)	105.79	193.89	193.89
8	(4 0)	(0 0 0)	102.81	190.90	190.90
8	(4 1)	(0 1 0)	101.79	189.89	189.89
8	(4 2)	(0 1 1)	101.79	189.89	189.89
8	(4 3)	(0 1 2)	101.79	189.89	189.89
8	(4 4)	(0 1 3)	101.79	189.89	189.89
8	(4 5)	(0 1 4)	101.79	189.89	189.89
8	(5 0)	(0 0 0)	97.79	185.89	185.89
8	(5 1)	(0 0 1)	97.79	185.89	185.89
8	(5 2)	(0 0 2)	97.79	185.89	185.89
8	(5 3)	(0 0 3)	97.79	185.89	185.89
8	(5 4)	(0 0 4)	97.79	185.89	185.89
8	(5 5)	(0 0 5)	97.79	185.89	185.89

* LOWER BOUND = UPPER BOUND

Table II-9: Solution to Infinite-horizon Problem
(SDD and Error bounds approach; for t=8)

CHAPTER 9 CONCLUSIONS

9.1 Discussions

Part II of the thesis analyzed the repairable-item inventory system with random demand and dynamic return processes. In the analysis of the system, the dynamic inventory problem was remodelled as a discrete-time Markov decision model with two-dimensional state and three-dimensional decision spaces. The system and decision rules defined throughout the analysis were used to eliminate unnecessary decisions before they were being considered while the transition probabilities for each and every remaining decision are derived using tools from probability. The Markov decision model was then solved for both finite-time and infinite-time planning horizons using the backward induction algorithm and the method of successive approximations, respectively. Later, two acceleration techniques, i.e., error bounds approach and State Decomposition by Dimension (SDD), were utilized for speeding up the convergence of successive approximation.

Reduction in decision space is a very important procedure in the analysis of the system (especially for those with a large number of states) because it avoids unnecessary computation time and effort by eliminating most of decisions before they are to be considered. With $\tilde{X}$=20 and $\tilde{Y}$=20, the number of decisions before reduction is 9261, which is quite large to handle. The decision space reduction procedure reduces the number of decisions to 441, that is a reduction by 95.2%.

For the finite-horizon Markov decision problem with N stages, $(\bar{X}+1)(\bar{Y}+1)$ states per stage and $(K+1)(\bar{X}+1)(\bar{Y}+1)$ decisions per state (before reduction), where $K=\min(\bar{X},\bar{Y})$, solution by the backward induction algorithm requires one addition for the stage return plus $(\bar{X}+1)(\bar{Y}+1)$ multiplications and $(\bar{X}+1)(\bar{Y}+1)$ additions for the summation term for given stage, state and decision. Thus the total number of computational steps required is $N(K+1)T^2[2T+1]$, where $T=(\bar{X}+1)(\bar{Y}+1)$. For the same problem, with the reduction procedure the total number of computational steps becomes $NT^2[2T+1]$, showing a decrease by a factor of K+1. Without using the decision space reduction procedure, inventory problems with a large number of states may not be solved even with a very sophisticated computer.

To study effects of changes in the cost parameters on policy structure for infinite-horizon problems, we have run the programs using different values of f, p, C and R. Considering the data in Section 7.4.2 as the base case, we chose each of the following parameter values: f=0,1,2; p=0,1,4,-2; C=0,4,8; and R=4,8. For each case, the algorithms terminate in a finite number of iterations with an optimal policy and the corresponding cost. However, for the problems with zero inventory holding cost (i.e., f=0) or non-zero set-up costs (i.e., $C\neq R\neq 0$) we were unable to determine any types of fixed policy structures. Determination of a fixed policy structure seems to be impossible when f=0 because junking decision is never optimal. When $C\neq R\neq 0$, interactions among these set-up costs and unit purchase and repair costs (c and r) make policy structure unstable (i.e., the repair-before-purchase (RBP) policy is not always optimal).

Somewhat interesting results include the following. When C=0 and R≠0, we have unstable policy structures since the RBP policy is not always optimal (i.e., r<c+p). When C≠0 and R=0, however, we are able to determine a fixed policy structure since the RBP policy is always optimal for this case. The set of solution parameters for such a case is different than the one introduced earlier because it requires an additional parameter regarding the purchase decision. We denote this set of extended solution parameters by $\tilde{\Phi}_\infty=(\beta_\infty,\nu_\infty,\eta_\infty,\zeta_\infty)$. The parameters regarding the repair decision (β_∞) and the junking decision (ζ_∞) have the same definition as before. The parameters regarding the purchase decision (ν_∞,η_∞) are now defined as follows: if $x_n+y_n<\nu_\infty$, purchase up to $\eta_\infty-(x_n+y_n)$; otherwise, do not purchase. An explanation follows. When the repair decision can not raise the serviceable inventory to β_∞, one of the following two purchase decisions is made: (i) if $x_n+y_n<\nu_\infty$, repair all available repairables (y_n) and then purchase $\eta_\infty-(x_n+y_n)$ serviceables so that the total number of serviceables after the repair and purchase decisions is equal to η_∞, and (ii) if $x_n+y_n\geq\nu_\infty$, do not purchase any serviceables. One should notice that the purchase decision discussed above is similar to the order-up-to (i.e., (s,S)) inventory policy for consumables, where ν_∞ and η_∞ act like s and S, respectively.

The error bound approach is an existing computational technique which accelerates the convergence of successive approximation for infinite-horizon problems. It was found that with the error bounds approach the total computation time required to find a stationary policy is reduced by approximately 93%. The SDD, a state decomposition technique developed in this thesis for speeding up the convergence of

successive approximation, can be used for both finite-horizon and infinite-horizon problems. The SDD was proved to be very efficient in computation time and very effective in the objective function values. For a 5-period finite-horizon inventory problem with $\hat{X}$=20 and $\hat{Y}$=20, the SDD reduces the total computation time by 86% with little increase (i.e., 0.86%) in total cost. Although the SDD is more efficient and effective for inventory problems with a large number of states, the successive approximation method utilizing both the SDD and the error bounds approach for infinite-horizon problems may reduce the computation time by almost 96%. It may be dangerous to make a general conclusion on the performance of the SDD since the results obtained above was based on a few examples. However, the results based on other examples we have tried (although they are not included in this thesis) also show that the validity of the SDD still remains.

9.2 Modifications to the Model

9.2.1 Junking of Serviceables

Consideration of the additional decision variable of junking of serviceables is possible in the situation where the cost of holding a serviceable unit is very high. This modification, however, does not affect the solution methodology developed in Chapter 7, and can be easily taken care of.

Let s_n be the number of serviceables junked in period n and q be the cost (or scrap value) per unit junked. A cost occurs if q>0 and a scrap value occurs if q<0. Then the control (u_n, v_n, j_n, s_n) is an element of a four dimensional control space U. The state equations (6-1) and

(6-2) become $x_{n+1} = \max[0,\ x_n+u_n+v_n-s_n-w_n]$ and $y_{n+1} = y_n-v_n-j_n+z_n$, respectively. The new constraints are: $j_n+v_n \le y_n$ and $s_n-u_n-v_n \le x_n$. The second constraint means that the net total units junked must not be greater than the beginning inventory. Also the expected cost per stage (7-1) changes to:

$$\phi_{(x,y)}(u,v,j,s) = \sigma_u C + cu + \sigma_v R + rv + pj + qs + f \max[0,y-v-j] + h\ E\{\max[0,x+u+v-s-w]\} + b\ E\{\max[0,w-x-u-v+s]\}.$$

Propositions 7.1 and 7.2 can be revised using the new system rules and then the optimal solution can be obtained using the similar analysis.

9.2.2 Backorders case

The change from the 'lost sales' case to the 'backorders' case has no impact on the solution methodology used in Chapter 7. For the 'backorders' case, however, the system equation (6-1) becomes $x_{n+1} = x_n+u_n+v_n-w_n$, indicating a negative serviceable inventory at the beginning of a period is possible. For the solution methodology previously used to work, we must introduce a negative capacity limit for the serviceable inventory, namely, $\underline{X}(>0)$. Thus, x and y are the state variables of the two-dimensional state (x,y), where $x=-\underline{X},..,-1,0,1,..,\bar{X}$ and $y=0,1,2,..,\bar{Y}$. The negative capacity limit $\underline{X}$ can be considered as the maximum number of backorders the inventory manager intends to make for unsatisfied demands. Propositions 7.1, 7.2 and 7.3 can be modified so that the number of states and the number of decisions will become $(\underline{X}+\bar{X}+1)(\bar{Y}+1)$ and $(K+1)(L+1)$, respectively, where $K=\min(\underline{X}+\bar{X},\bar{Y})$ and $L=\max(\underline{X}+\bar{X},\bar{Y})$. The expected cost per stage (7-1) will remain the same; however, b is now defined as the penalty cost per unit backordered.

9.3 Comparison with Similar Models

In this section, the inventory models studied in this thesis are compared to the similar inventory models investigated by the following three authors: Veinott [1966], Phelps [1962], and Simpson [1972][1978].

9.3.1 Veinott's Model

The inventory models studied in this thesis is more general than Veinott's model. It will be comparable with Veinott's model when the following changes are made. First, the 'backorders' case instead of the 'lost sales' case is considered (see Section 9.2.2). Second, the warranty period (M) is zero (i.e., units demanded in the current period only can be returned for repair) and the percentage of the demand for serviceables returned (θ) is a fixed constant (rather than a random variable) so that the demand/return relationship (7-5) becomes $z_n = \theta w_n$, which is a perfect correlation between the demand and return processes. Third, the junking decision is not permitted. Finally, the set-up costs of purchasing serviceables and repairing repairables are negligible (i.e., C=R=0). Thus, Veinott' model is a special case of the modified N-period model studied in this thesis which can be solved for an optimal solution by the methodology similar to the one used in this thesis.

9.3.2 Phelps' Model

Phelps' model differs in three ways from the steady-state inventory model studied in this thesis. First, an additional decision variable of junking serviceables is included. Second, as in Veinott's model a perfect correlation between the demand and return processes is

assumed. Finally, the set-up costs of purchasing serviceables and repairing repairables are negligible. The first difference can easily be modified in the model in this thesis (see Section 9.2.1). The second and third differences have been discussed in Section 9.3.1. Thus, Phelps' model is a special case of the modified steady-state model studied in the thesis and can be solved by the methodology similar to the one used in this thesis.

9.3.3 Simpson's Model

Simpson and this thesis investigate the similar problem and solve for optimal solutions to both finite horizon and infinite horizon problems. Simpson's models and the models studied in this thesis differ in the following ways. First, Simpson considers the 'backorders' case for the unsatisfied demand for serviceables while this thesis assumes the 'lost sales' case. Second, Simpson's models do not allow the set-up cost of purchasing serviceables and of repairing repairables while the models in the thesis do. Third, Simpson assumes zero salvage value (or zero cost) for repairables to be junked while this thesis allows a constant salvage value (or a constant cost) per unit. Fourth, this thesis considers a more realistic relationship between the demand for serviceables and returns of repairables. A joint probability density function between the demand and returns is known (i.e., assumed) for each period in Simpson's models while it is derived in this thesis from the dynamic demand/return relationship, where not only a random proportion of the demand in the current period but those in the previous periods may be returned for repair. Finally, Simpson utilizes a dynamic

programming technique with the Kuhn-Tucker saddle point theorems to determine a definitive optimal solution structure. This thesis, however, uses a Markov decision process with a decision space reduction technique and the method of successive approximations to obtain an optimal solution which the corresponding optimal solution structure can be derived from. While the solution methodology used by Simpson is not applicable to the 'lost sales' case, the approach used in this thesis is applicable to the 'backorders' case (see Section 9.2.2).

The dynamic return process considered in this thesis also leads to instability of policy structures. However, the policy structures obtained (with C=R=0) in this thesis for the infinite-horizon problem seem to be consistent with those in Simpson. It would be quite interesting to know whether both the 'lost sales' and 'backorders' cases constitute the same policy structures for the proposed inventory problem.

APPENDIX II-A: CLASSIFICATIONS OF REPAIRABLE-ITEM INVENTORY MODELS WITH RETURNS

1. Demand/Return Relationships

a) Perfect Correlation (Complete Dependency): - A known proportion of the demand for serviceables in a certain period are returned for repair in the same period.

Phelps [1962], Veinott [1966], Schrady [1967], Allen & D'Esopo [1968], Brown et al. [1971]

b) Perfect Non-Correlation (Complete Independency):

Simpson [1970], Heyman [1977][1978], Isaac [1979], Muckstadt & Isaac [1981], Albright & Soni [1988]

c) Joint Probability Density Function between Demand and Returns (for a given period):

Simpson [1972][1978]

d) Dynamic Correlation: - Random proportions of serviceables demanded in the current and previous periods are returned.

Cho [thesis]

2. Types of Review Policy

a) Continuous Review Policy:

Schrady [1967], Allen & D'Esopo [1968], Heyman [1977][1978], Isaac [1979], Muckstadt & Isaac [1981], Albright & Soni [1988]

b) Periodic Review Policy:

Phelps [1962], Veinott [1966], Simpson [1970], Brown et al.[1971], Simpson [1972][1978], Cho [thesis]

3. Behavior of Demand/Return Processes

a) Deterministic:

Schrady [1967]

b) Stochastic:

Phelps [1962], Veinott [1966], Allen & D'Esopo [1968], Simpson [1970], Brown et al. [1971], Simpson [1972][1978], Heyman [1977] [1978], Isaac [1979], Muckstadt & Isaac [1981], Albright & Soni [1988], Cho [thesis]

4. Purchase Lead Times

a) Instantaneous (zero times):

Phelps [1962], Veinott [1966], Simpson [1972][1978], Heyman [1977] [1978], Cho [thesis]

b) Constant:

Schrady [1967], Allen & D'Esopo [1968], Brown et al. [1971], Isaac [1979], Muckstadt & Isaac [1981], Albright & Soni [1988]

c) Stochastic:

Simpson [1970]

5. Repair Times

a) Instantaneous (zero times):

Phelps [1962], Veinott [1966], Simpson [1972][1978], Heyman [1977] [1978], Cho [thesis]

b) Constant:

Schrady [1967], Allen & D'Esopo [1968], Brown et al. [1971]

c) Stochastic:

Simpson [1970], Isaac [1979], Muckstadt & Isaac [1981], Albright & Soni [1988]

6. Set-up Costs for Purchase and Repair

a) No set-up costs:

Phelps [1962], Veinott [1966], Schrady [1967], Allen & D'Esopo [1968], Simpson [1970], Brown et al. [1971], Simpson [1972][1978], Heyman [1977][1978], Isaac [1979], Muckstadt & Isaac [1981], Albright & Soni [1988]

b) Constant set-up costs for purchase and repair:

Cho [thesis]

7. Control Variables

a) Purchase Decision:

Allen & D'Esopo [1968], Simpson [1970], Brown et al. [1971], Heyman [1977][1978], Isaac [1979], Muckstadt & Isaac [1981]

b) Purchase and Repair:

Veinott [1966], Schrady [1967], Albright & Soni [1988]

c) Purchase, Repair, and Junk (Scrap):

Phelps [1962], Simpson [1972][1978], Cho [thesis]

8. Planning Time Horizon

a) Finite Time:

Veinott [1966], Brown et al. [1971], Simpson [1972][1978], Cho [thesis]

b) Infinite-Time (Steady-state):

Phelps [1962], Schrady [1967], Allen & D'Esopo [1968], Simpson [1970][1972][1978], Heyman [1977][1978], Isaac [1979], Muckstadt & Isaac [1981], Albright & Soni [1988], Cho [thesis]

9. Solution Type

a) Optimal Solution:

Phelps [1962], Veinott [1966], Schrady [1967], Allen & D'Esopo [1968], Brown et al. [1971], Simpson [1972][1978], Heyman [1977] [1978] (for M/M/1/N), Cho [thesis]

b) Approximation:

Simpson [1970], Heyman [1977][1978] (for G/G/1/N), Isaac [1979], Muckstadt & Isaac [1981], Albright & Soni [1988]

APPENDIX II-B: COMPARISON OF FIVE RELEVANT INVENTORY MODELS

A brief description of each of the models in the literature, directly related to this research is given. In order to compare these models, we will look into the following areas.

(a) Model Structure (Assumptions):

1) Demand/Return Processes
2) Demand/Return Relationship
3) Type of Review Policy
4) Backorders or Lost Sales (for Unsatisfied Demand)
5) State Variables
6) Control Variables
7) Purchase/Repair Times
8) Distinct Cost Consideration
9) Planning Horizon

(b) Solution Approach:

10) Solution Methodologies
11) Solution Type/Structure
12) Other comments

Phelps [1962]:

1) Stochastic demand and repair processes.
2) Perfect correlation (complete dependency): - a fixed portion of demand is returned.
3) Periodic review policy.
4) Lost sales
5) Two state variables: serviceable and repairable inventories.
6) Four control variables: purchase, repair, junk(S) & junk(R).
7) Instantaneous purchase/repair.
8) Infinite-time (steady-state).
9) Nil.
10) Decision analysis.
11) Optimal policy is specified by seven regions in the state plane.
No closed-form solution; No algorithms.
No determination of the boundaries of the solution structure.

Veinott [1966]:

1) Stochastic demand and repair processes.
2) Perfect correlation.
3) Periodic review policy.

4) Backorders.
5) Two state variables: serviceables and repairables.
6) Two control variables: purchase & repair.
7) Instantaneous purchase/repair.
8) Finite-time.
9) Nil.
10) Decision analysis.
11) Optimal policy is specified by three regions in the state plane.
No closed-form solution; No algorithms.
No determination of the boundaries of the solution structure.

Simpson [1972][1978]:

1) Stochastic demand/return processes.
2) Joint probability density between demand and returns.
3) Periodic review policy.
4) Backorders.
5) Two state variables: serviceables and repairables.
6) Three control variables: purchase, repair & junk.
7) Instantaneous purchase/repair.
8) Both finite-time and infinite-time.
9) Negligible salvage value from junking a unit.
10) Dynamic programming with K-T Theorem.
11) Optimal solution is described by seven regions.
Optimality has been proved.
Determination of the boundaries of the solution structure.
No algorithms for the generalized model.
Algorithms for the specialized N-period and steady-state models.
Closed-form solutions for special cases of the steady-state model.
12) The solution methodology is inapplicable to the "lost sales" case.

Isaac [1979], Muckstadt and Isaac [1981]:

1) Both demand and returns are Poisson processes.
2) Perfect non-correlation: two independent random variables.
3) Continuous review policy.
4) Backorders.
5) One state variable: net inventory position.
6) One control variable: level of procurement.
7) Constant procurement and Stochastic repair times.
8) Infinite-time (steady-state).
9) Nil.
10) Decision Analysis - use of a normally distributed random variable
for an approximation of the stationary distribution of net inventory.
11) Assumption of a continuous (r,Q) procurement policy.
Approximated closed-form solution.
12) Decision-maker has no control over the quantity of repair.

Cho [thesis]:

1) Stochastic demand and return processes.
2) Dynamic correlation: Quantity to be returned in the current period depends on the demands in both the current and previous periods.
3) Periodic review policy.
4) Lost sales.
5) Two state variables: serviceables and repairables.
6) Three control variables: levels of purchase, repair, and junk.
7) Instantaneous purchase/repair.
8) Both finite-time and infinite-time (steady-state).
9) Constant set-up costs for purchase and repair.
10) Markov decision process (Dynamic programming in Markov Chains). Successive approximation.
11) Optimal solution.
12) Development of acceleration techniques. Applicable to the 'backorders' case.

APPENDIX II-C: PROOF OF PROPOSITION 7.3

Lemma 7.1: The total number of decisions before some decisions have been eliminated by the decision and system rules is $(K+1)(\bar{X}+1)(\bar{Y}+1)$.

Proof: $\bar{U}=\bar{X}$, $\bar{V}=K$ and $\bar{J}=\bar{Y}$, where $K=\min(\bar{X},\bar{Y})$, by Proposition 7.2. Since u may take any one of $(\bar{X}+1)$ different values (i.e., $u=0,1,2,\ldots,\bar{X}$), v may take any one of $(K+1)$ different values (i.e., $v=0,1,2,\ldots,\bar{K}$), and j may take any one of $(\bar{Y}+1)$ different values (i.e., $j=0,1,2,\ldots,\bar{Y}$), the total number of decisions before some decisions have been eliminated by the decision and system rules is $(K+1)(\bar{X}+1)(\bar{Y}+1)$. Q.E.D.

Lemma 7.2: Two types of decisions, $(u>0,v>0,j>0)$ and $(u>0,v=0,j>0)$, would never be optimal.

Proof: There are eight types of decisions based on the value (positive or zero) each of three decision variables u, v, and j takes. These are:

(i)	$(u>0,v>0,j>0)$	(ii)	$(u>0,v>0,j=0)$	(iii)	$(u>0,v=0,j>0)$
(iv)	$(u>0,v=0,j=0)$	(v)	$(u=0,v>0,j>0)$	(vi)	$(u=0,v>0,j=0)$
(vii)	$(u=0,v=0,j>0)$	(viii)	$(u=0,v=0,j=0)$		

We will prove that two types of decisions (i) and (iii) would never be optimal by showing that they are dominated by other types of decisions or are invalid in some situations. In Section 6.2, we assumed that $r<c+p$, that is, the cost of repairing a repairable unit is always less than the net cost of junking the unit and purchasing a new serviceable unit. Decision $(u>0,v>0,j>0)$ would never be optimal because it is always dominated by decision $(u>0,v>0,j=0)$. Let y be the number

of repairables in the repair facility at the beginning of a period and s be the number of serviceables replenished to the storage facility to meet demand either by repairing repairables or purchasing serviceables or by both. Assume that v_1 units are repaired (i.e., $v=v_1<y$) and $y-v_1$ units are junked (i.e., $j=y-v_1$). Thus $s-v_1$ units must be purchased (i.e., $u=s-v_1$) so that the serviceable inventory will increase by s. The cost of decision ($u>0, v>0, j>0$) is then $R+rv_1+C+c(s-v_1)+p(y-v_1)$. If one follows decision ($u>0, v>0, j=0$), all units in the repair facility are repaired (i.e., $v=y$) and s-y units must be purchased (i.e., $u=s-y$). Of course, there are no units to be junked. The cost of this decision is $R+ry+C+c(s-y)$. Let

Cost of ($u>0, v>0, j>0$)	<	Cost of ($u>0, v>0, j=0$)
$R + rv_1 + C + c(s-v_1) + p(y-v_1)$	<	$R + ry + C + c(s-y)$
$rv_1 - v_1c + py - pv_1$	<	$ry - cy$
$(r-c-p)v_1$	<	$(r-c-p)y$

Since $r-c-p<0$ from the cost function (i.e, $r<c+p$), after simplifying we get $v_1>y$, which violates our assumption $v_1<y$. Thus, the cost of ($u>0$, $v>0, j=0$) is less than that of ($u>0, v>0, j>0$), implying that decision ($u>0, v>0, j=0$) is always superior to decision ($u>0, v>0, j>0$).

As far as decision ($u>0, v=0, j>0$) is concerned, it becomes invalid or is dominated by other decisions. Zero repair decision ($v=0$) may be made under any one of two circumstances, that is, $y=0$ or $y>0$. If $y=0$ (no repairables in the repair facility), then the above decision is invalid because there are no units to be junked (i.e., $j=0$). If $y>0$

(some repairables), however, decision (u>0,v=0,j>0) is dominated by decision (u=0,v=0,j>0). Because the cost of repairing a repairable unit is always less than that of junking the unit and purchasing a new serviceable unit, the case y>0 and v=0 indicates that the storage facility has enough serviceables to meet demand and there is no need to replenish more serviceables to the storage facility. Thus with decision (u>0,v=0,j>0), extra costs will be incurred due to a purchase of unnecessary serviceables. Q.E.D.

Lemma 7.3: The number of decisions eliminated by the decision rules, (u>0,v>0,j>0) or (u>0,v=0,j>0), is $(K+1)\bar{X}\bar{Y}$.

Proof: The decisions with u>0 and j>0 will be eliminated by the decision rules. Since u may take any one of $\bar{X}$ different values, v may take any one of (K+1) different values, and j may take any one of $\bar{Y}$ different values, the number of decisions eliminated by the decision rules is $(K+1)\bar{X}\bar{Y}$. Q.E.D.

Lemma 7.4: The number of decisions eliminated by the system rules, $(u+v>\bar{X})$ or $(v+j>\bar{Y})$, is $\frac{1}{6}K(K+1)(3L+K+5)$.

Proof: Number of decisions eliminated by the system rules (c) or (d) = number of decisions eliminated by (c) + number of decisions eliminated by (d) - number of decisions eliminated by both (c) and (d)

Number of decisions eliminated by (c):

If v=0, u may not take any value; j may take any one of $(\bar{Y}+1)$ values.

If v=1, u may take only one value; j may take any one of $(\bar{Y}+1)$ values.

If v=2, u may take one of 2 values; j may take any one of ($\bar{Y}$+1) values.

: : :

If v=K, u may take one of K values; j may take any one of ($\bar{Y}$+1) values.

Thus, the number of decisions eliminated by (c)

$$= (0+1+2+3+\ldots.+K)(\bar{Y}+1) = \frac{1}{2} K(K+1)(\bar{Y}+1).$$

Number of decisions eliminated by (d):

If v=0, j may not take any value; u may take any one of ($\bar{X}$+1) values.

If v=1, j may take only one value; u may take any one of ($\bar{X}$+1) values.

If v=2, j may take one of 2 values; u may take any one of ($\bar{X}$+1) values.

: : :

If v=K, j may take one of K values; u may take any one of ($\bar{X}$+1) values.

Thus, the number of decisions eliminated by (d)

$$= (0+1+2+3+\ldots.+K)(\bar{X}+1) = \frac{1}{2} K(K+1)(\bar{X}+1).$$

Number of decisions eliminated by both (c) and (d):

If v=0, u may not take any value; j may not take any values.

If v=1, u may take only one value; j may take only one value.

If v=2, u may take one of 2 values; j may take one of two values.

: : :

If v=K, u may take one of K values; j may take one of K values.

Thus, the number of decisions eliminated by both (c) and (d)

$$= 0^2 + 1^2 + 2^2 + 3^2 + \ldots + K^2 = \frac{1}{6} K(K+1)(2K+1).$$

Therefore, the number of decisions eliminated by either (c) or (d)

$$= \frac{1}{2} K(K+1)(\bar{Y}+1) + \frac{1}{2} K(K+1)(\bar{X}+1) - \frac{1}{6} K(K+1)(2K+1)$$

$$= \frac{1}{6} K(K+1)[3(\bar{X}+\bar{Y})-2K+5]$$

$$= \frac{1}{6} K(K+1)(3L+K+5) \quad \text{since } \bar{X}+\bar{Y} = K+L = L+K. \qquad \text{Q.E.D.}$$

Lemma 7.5: The number of decisions eliminated by both the decision rules, $(u>0, v>0, j>0)$ or $(u>0, v=0, j>0)$, and the system rules, $(u+v>\bar{X})$ or $(v+j>\bar{Y})$, is $\frac{1}{6} K(K+1)(3L+K-1)$.

Proof: Number of decisions eliminated by both the decision rules and the system rules (d) & (e):

If v=0: both u and j may not take any value.

If v=1: when u=0 to $\bar{X}$-1, j may take only one value;
when u=$\bar{X}$, j may take any one of $\bar{Y}$ different values.

If v=2: when u=0 to $\bar{X}$-2, j may take only one value;
when u=X-1 to $\bar{X}$, j may take any one of $\bar{Y}$ different values.

If v=3: when u=0 to $\bar{X}$-3, j may take only one value;
when u=X-2 to $\bar{X}$, j may take any one of $\bar{Y}$ different values.

: : :

If v=K: when u=0 to $\bar{X}$-K, j may take only one value;
when u=X-K to $\bar{X}$, j may take any one of $\bar{Y}$ different values.

$\therefore$ The number of decisions eliminated by both the decision rules and the system rules

$$= 0 + (\bar{X}+\bar{Y}-1) + 2(\bar{X}+\bar{Y}-2) + 3(\bar{X}+\bar{Y}-3) + \ldots\ldots + K(\bar{X}+\bar{Y}-K)$$

$$= (A-1) + 2(A-2) + 3(A-3) + \ldots\ldots + K(A-K) \quad \text{where } A=\bar{X}+\bar{Y}$$

$$= A + 2A + 3A + \ldots + KA - (1^2 + 2^2 + 3^2 + \ldots + K^2)$$

$$= \frac{A}{2} K(K+1) - \frac{1}{6} K(K+1)(2k+1)$$

$$= \frac{1}{6} K(K+1)[3(\bar{X}+\bar{Y})-2K-1]$$

$$= \frac{1}{6} K(K+1)(3L+K-1) \quad \text{since } \bar{X}+\bar{Y} = K+L = L+K.$$

Therefore, the number of decisions to be considered in the MDP

= Total number of decisions before elimination - Number of decisions eliminated by the decision rules - Number of decisions eliminated by the system rules + number of decisions eliminated by both the decision and system rules

$$= (K+1)(\bar{X}+1)(\bar{Y}+1) - (K+1)\bar{X}\bar{Y} - \frac{1}{6} K(K+1)(3L+K+5) + \frac{1}{6} K(K+1)(3L+K-1)$$

$$= (K+1)(\bar{X}+\bar{Y}+1) + \frac{1}{6} K(K+1)(3L+K-1-3L-K-5)$$

$$= (K+1)(\bar{X}+\bar{Y}+1) - K(K+1)$$

$$= (K+1)(\bar{X}+\bar{Y}-K+1)$$

$$= (K+1)(L+1) = (\bar{X}+1)(\bar{Y}+1) \quad \text{since } \bar{X}+\bar{Y} = K+L = L+K. \qquad \text{Q.E.D.}$$

APPENDIX II-D: COMPARISON OF THREE COMPUTATIONAL METHODS IN MDP

(1) POLICY ITERATION

- Not applicable for a finite horizon problem.
- Requires solution of a linear system of simultaneous equations, implying that each iteration takes a long time.
- Terminates in a finite number of iterations (for finite space sets)
- Not attractive for a model with a very large number of states, because the dimension of the system is equal to the number of points of the state space.
- The number of iterations required before termination may depend on the quality of an initial policy selected.

(2) LINEAR PROGRAMMING

- Not applicable for a finite horizon problem.
- Requires solution of a system of simultaneous linear equations.
- Terminates in a finite number of iterations (for finite space sets)
- Becomes impractical for very large M and K, where M is the number of states and K is the number of decisions.
- There are (M+1) functional constraints and KM original variables.

i.e., In our problem, if $\tilde{X}=\tilde{Y}=10$ then:

$$M = (\tilde{X}+1)(\tilde{Y}+1) = 11*11 = 121$$

& $K =$ $= 121.$

Thus, # of constraints $= 121 + 1 = 122$

& # of variables $= 121*121 = 14641.$

(3) METHOD OF SUCCESSIVE APPROXIMATION

- Probabilistic dynamic programming algorithm.
- Applicable for both finite and infinite horizon problems.
- Never requires the solution of a system of simultaneous equations, implying that each iteration can be performed simply and quickly.
- Requires less computer storage space.
- For a finite horizon problem, the method yields an optimal policy.
- For an infinite horizon, discounted cost problem, the method yields an optimal policy after a finite number of iterations.
- For an infinite horizon, discounted cost problem, many techniques are available for accelerating the convergence of the successive approximation method (i.e., error bounds approaches, etc.).

APPENDIX II-E: TRANSFORMATION OF THE RECURSIVE RELATIONSHIP FROM THE FINITE-HORIZON PROBLEM TO THE INFINITE-HORIZON PROBLEM.

Assume $J^N_{(x,y)} = \phi_N(x,y) = 0$ for $(x,y) \epsilon S$ when $N \to \infty$.

Let $Q^{N-n}_{(x,y)} = J^n_{(x,y)}$, $(x,y) \epsilon S$, $n=0,1,\ldots,N$.

Using the recursive relationship (7-9),

$$Q^0_{(x,y)} = 0, \quad (x,y) \epsilon S, \quad n=N$$

and

$$Q^{N-n}_{(x,y)} = \min_{(u,v,j)} \{ \phi_{(x,y)}(u,v,j) + \alpha \sum_{x'=0}^{\tilde{X}} \sum_{y'=0}^{\tilde{Y}} p_{(x,y)(x',y')}(u,v,j)\, Q^{N-n-1}_{(x',y')} \}$$

$$\text{for } (x,y) \epsilon S; \quad n=0,1,\ldots,N-1.$$

Replacing $Q^{N-n}_{(x,y)}$ with $J^{t+1}_{(x,y)}$, $(x,y) \epsilon S$, $t=0,1,\ldots,N-1$, the recursive relationship of the successive approximation method is stated as follow.

$$J^0_{(x,y)} = 0, \qquad (x,y) \epsilon S$$

and

$$J^{t+1}_{(x,y)} = \min_{(u,v,j)} \{ \phi_{(x,y)(u,v,j)} + \alpha \sum_{x'=0}^{\tilde{X}} \sum_{y'=0}^{\tilde{Y}} p_{(x,y)(x',y')}(u,v,j)\, J^t_{(x',y')} \}$$

$$\text{for } (x,y) \epsilon S, \quad t=0,1,\ldots,N-1,$$

where $J^t_{(x,y)}$ now is the minimum expected total discounted cost of the system starting at state (x,y) and evolving for t time periods.

APPENDIX II-F: COMPUTER PROGRAM - BACKWARD INDUCTION ALGORITHM (BIA)

```
REM * THE PROGRAM "THE21FIN.BAS" FINDS OPTIMAL POLICY, WHICH MINIMIZES *
REM * THE TOTAL EXPECTED DISCOUNTED COST FOR A FINITE-TIME REPAIRABLE  *
REM * INVENTORY MODEL USING THE BACKWARD INDUCTION ALGORITHM.          *
REM *                                                                  *
REM *                                                                  *
REM *    Demand function:  p(w) = { 1/(wbar+1)   w=0,1,...,wbar        *
REM *                            { 0             otherwise             *
REM *                                                                  *
REM *                                                                  *
REM *        - ( Developed  6/25/91    Revised  3/19/92 )  -          *

' Define dimensions.

DIM isymbol$(500), ksymbol$(500)
DIM usymbol$(500), vsymbol$(500), jsymbol$(500)
DIM preoptcost(500), optcost(500), avgsmcost(500), ptrans(500)
DIM probw(20), probgew(20), probz(20, 10, 20), probgez(20, 10, 20)

' Input data.

tf = 10                 'final time
alpha = .9              'discount factor
capC = 4                'fixed ordering cost
c = 6                   'unit ordering cost
capR = 4                'fixed repair facility setup cost
r = 4                   'unit repair cost
f = 1                   'holding cost for repairable items
h = 2                   'holding cost for serviceable items
p = 0                   'junking cost
b = 15                  'penalty cost of unsatisfied demand
xbar = 3                'maximum storage capacity
ybar = 2                'maximum repair capacity
wbar = 3                'maximum possible demand
delta = .2              'proportion of units demanded returned
maxwarranty = 2         'warranty periods for goods demanded

IF xbar >= ybar THEN
  lvalue = xbar
  kvalue = ybar
ELSE
  lvalue = ybar
  kvalue = xbar
END IF

ubar = xbar
vbar = kvalue
jbar = ybar
```

```
totnstate = (xbar + 1) * (ybar + 1)
totndec = (ubar + 1) * (vbar + 1) * (jbar + 1)
ndecae = (kvalue + 1) * (lvalue + 1)

' Set optimal cost at the final stage to zero.

FOR i% = 1 TO totnstate
  preoptcost(i%) = 0
NEXT i%

' Define the demand process (i.e., p.m.f. & c.d.f.).

cummprobw = 0
FOR w% = 0 TO wbar
  probw(w%) = 1 / (wbar + 1)
  cummprobw = cummprobw + probw(w%)
  probgew(w%) = 1 - cummprobw + probw(w%)
  'PRINT w%; probw(w%); probgew(w%)
NEXT w%

' Define the return process (i.e., p.m.f. & c.d.f.)
CLS

zbar = (maxwarranty + 1) * wbar
sbar = maxwarranty * wbar
zbarfactor# = 1

FOR s% = 0 TO sbar
  FOR w% = 0 TO wbar
    zcombination = 0
    cummprobz = 0
    FOR z% = 0 TO zbar
      diffzw% = z% - w%
      IF diffzw% > s% THEN
        probz(s%, w%, z%) = 0
        cummprobz = cummprobz + probz(s%, w%, z%)
        probgez(s%, w%, z%) = 1 - cummprobz + probz(s%, w%, z%)
      ELSE
        totsw = s% + w%
        swfactor# = 1
        IF totsw = 0 THEN
          swfactor# = 1
        ELSE
          FOR fact% = 1 TO totsw
            swfactor# = swfactor# * fact%
          NEXT fact%
        END IF

        zfactor# = 1
        IF z% = 0 THEN
          zfactor# = 1
        ELSE
```

```
          FOR fact% = 1 TO z%
            zfactor# = zfactor# * fact%
          NEXT fact%
        END IF

        zdiff% = totsw - z%
        zdifffactor# = 1
        IF z% = totsw THEN
          zdifffactor# = 1
        ELSE
          FOR fact% = 1 TO zdiff%
            zdifffactor# = zdifffactor# * fact%
          NEXT fact%
        END IF

        zcombination = swfactor# / (zfactor# * zdifffactor#)
        probz(s%, w%, z%) = zcombination * delta ^ z%
                                          * (1 - delta) ^ zdiff%
        cummprobz = cummprobz + probz(s%, w%, z%)
        probgez(s%, w%, z%) = 1 - cummprobz + probz(s%, w%, z%)
      END IF

    NEXT z%
  NEXT w%
NEXT s%

' Print out input data.

'CLS
PRINT "< INPUT DATA FOR FINITE-HORIZON PROBLEM: THE21FIN.BAS >"
PRINT "     "
PRINT "tf ="; tf, "final time"
PRINT "alpha ="; alpha, "discount factor"
PRINT "capC ="; capC, "fixed ordering cost"
PRINT "c ="; c, "unit ordering cost"
PRINT "capR ="; capR, "fixed repair facility setup cost"
PRINT "r ="; r, "unit repair cost"
PRINT "f ="; f, "holding cost for repairable items"
PRINT "h ="; h, "holding cost for serviceable items"
PRINT "p ="; p, "junking cost"
PRINT "b ="; b, "penalty cost of unsatisfied demand"
PRINT "       "
PRINT "xbar ="; xbar, "maximum storage capacity"
PRINT "ybar ="; ybar, "maximum repair capacity"
PRINT "      "
PRINT "wbar ="; wbar, "maximum possible demand"
PRINT "delta ="; delta, "proportion of units demanded returned"
PRINT "warranty ="; maxwarranty, "warranty periods for units demanded"

LOCATE 24, 24: PRINT "Press any key to continue ....."; INPUT$(1)

TIME$ = "00:00"
```

```
' Determine a state space and a decision space.

k% = 0
FOR u% = 0 TO ubar
  FOR v% = 0 TO vbar
    FOR j% = 0 TO jbar
      deltax% = u% + v%
      deltay% = v% + j%

      IF deltax% > xbar OR deltay% > ybar THEN
        k% = k%
      ELSEIF u% > 0 AND j% > 0 THEN
        k% = k%
      ELSE
        k% = k% + 1
        i% = 0
        FOR x% = 0 TO xbar
          FOR y% = 0 TO ybar
            i% = i% + 1
            isymbol$(i%) = STR$(x%) + STR$(y%)
            IF x% + deltax% > xbar OR deltay% > y% THEN
              u% = u%
              v% = v%
              j% = j%
            ELSE
              usymbol$(k%) = STR$(u%)
              vsymbol$(k%) = STR$(v%)
              jsymbol$(k%) = STR$(j%)
              ksymbol$(k%) = usymbol$(k%) + vsymbol$(k%) + jsymbol$(k%)
            END IF
          NEXT y%
        NEXT x%
      END IF

    NEXT j%
  NEXT v%
NEXT u%

' Determine optimal policy at each state and period.

FOR t% = tf TO 1 STEP -1

  IF t% = tf THEN
    CLS
    PRINT "OPTIMAL SOLUTION TO THE"; tf; "- PERIOD REPAIRABLE ITEM
           INVENTORY MODEL:"
    PRINT "      "
    PRINT SPC(0); "< PERIOD >"; SPC(2); "< STATE (x,y) >"; SPC(4);"<S>";
          SPC(3); "< DECISION (u,v,j) >"; SPC(3); "< EXPECTED COST >"
    PRINT "      "
  ELSE
    PRINT "      "
```

```
  PRINT SPC(0); "< PERIOD >"; SPC(2); "< STATE (x,y) >"; SPC(4);"<S>";
        SPC(3); "< DECISION (u,v,j) >"; SPC(3); "< EXPECTED COST >"
  PRINT "       "
END IF

i% = 0
sumoptcost = 0
FOR x% = 0 TO xbar
  FOR y% = 0 TO ybar
    i% = i% + 1
    totsmcost = 0
    FOR s% = 0 TO sbar
      FOR k% = 1 TO ndecae
        u% = VAL(usymbol$(k%))
        v% = VAL(vsymbol$(k%))
        j% = VAL(jsymbol$(k%))
        netdeltax% = x% + u% + v%
        netdeltay% = y% - v% - j%
        IF netdeltax% > xbar OR netdeltay% < 0 THEN
          k% = k%
        ELSE

          ' Determine Transient Probability matrix.

          idot% = 0
          FOR xdot% = 0 TO xbar
            FOR ydot% = 0 TO ybar
              idot% = idot% + 1
              differx% = netdeltax% - xdot%
              differy% = ydot% - netdeltay%

              IF differx% < 0 OR differy% < 0 THEN
                ptrans(idot%) = 0
              ELSEIF xdot% = 0 AND ydot% = ybar THEN
                ptrans(idot%) = 0
                FOR w% = differx% TO wbar
                  ptrans(idot%) = ptrans(idot%) + probw(w%)
                                  * probgez(s%, w%, differy%)
                NEXT w%
              ELSEIF xdot% = 0 AND ydot% <> ybar THEN
                ptrans(idot%) = 0
                FOR w% = differx% TO wbar
                  ptrans(idot%) = ptrans(idot%) + probw(w%)
                                  * probz(s%, w%, differy%)
                NEXT w%
              ELSEIF xdot% <> 0 AND ydot% = ybar THEN
                ptrans(idot%) = probw(differx%)
                                * probgez(s%, differx%, differy%)
              ELSE
                ptrans(idot%) = probw(differx%)
                                * probz(s%, differx%, differy%)
              END IF
```

```
      NEXT ydot%
    NEXT xdot%

    ' Determine cost function.

    IF u% > 0 THEN
      fixorder% = 1
    ELSE
      fixorder% = 0
    END IF

    IF v% > 0 THEN
      fixrepair% = 1
    ELSE
      fixrepair% = 0
    END IF

    eholding = 0
    FOR w% = 0 TO netdeltax%
      eholding = eholding + h * ((netdeltax% - w%) * probw(w%))
    NEXT w%

    epenalty = 0
    FOR w% = netdeltax% TO wbar
      epenalty = epenalty + b * ((w% - netdeltax%) * probw(w%))
    NEXT w%
    cost = fixorder% * capC + c * u% + fixrepair% * capR + r *
           v% + p * j% + f * netdeltay% + eholding + epenalty

    sum1 = 0
    FOR idot% = 1 TO totnstate
      sum1 = sum1 + ptrans(idot%) * preoptcost(idot%)
    NEXT idot%
    sum2 = cost + alpha * sum1

    IF k% = 1 THEN
      optk% = k%
      mincost = sum2
    ELSEIF sum2 >= mincost THEN
      optk% = optk%
      mincost = mincost
    ELSE
      optk% = k%
      mincost = sum2
    END IF
  END IF
NEXT k%

decision% = optk%
optcost(i%) = mincost
```

```
        ' Print out results

        invt% = t% - 1
        IF s% = 0 THEN
          PRINT TAB(4); invt%; TAB(15); i%; TAB(19); "("; isymbol$(i%);
              " )"; TAB(33); s%; TAB(43); decision%; TAB(47); "(";
              ksymbol$(decision%); " )"; TAB(65); USING "######.##";
              optcost(i%)
        ELSE
          PRINT TAB(33); s%; TAB(43); decision%; TAB(47); "("; ksymbol$
              (decision%); " )"; TAB(65); USING "######.##"; optcost(i%)
        END IF
        totsmcost = totsmcost + optcost(i%)
      NEXT s%
      avgsmcost(i%) = totsmcost / (sbar + 1)
      sumoptcost = sumoptcost + avgsmcost(i%)
      PRINT "     "

    NEXT y%
  NEXT x%

  PRINT "       "
  avgoptcost = sumoptcost / totnstate
  PRINT TAB(19); "Average total cost per state  ="; USING "#####.##";
                 avgoptcost

  FOR i% = 1 TO totnstate
    preoptcost(i%) = avgsmcost(i%)
  NEXT i%

  PRINT "       "
  PRINT "       "

NEXT t%
PRINT "Total Processing Time = "; TIMER
PRINT "       "
```

APPENDIX II-G: COMPUTER PROGRAM: SUCCESSIVE APPROXIMATION METHOD (SAM) (WITHOUT/WITH ERROR BOUNDS)

```
REM *  THE PROGRAM "THE22NEW.BAS" UTILIZES THE METHOD OF SUCCESSIVE   *
REM *  APPROXIMATIONS TO DETERMINE AN OPTIMAL POLICY, WHICH MINIMIZES *
REM *  THE EXPECTED LONG-RUN TOTAL DISCOUNTED COST FOR INFINITE-TIME  *
REM *  REPAIRABLE-ITEM INVENTORY MODELS.   THE PROGRAM ALSO UTILIZES  *
REM *  AN ERROR-BOUND APPROACH TO ACCELERATE THE CONVERGENCE OF THE   *
REM *  SUCCESSIVE APPROXIMATION METHOD.                               *
REM
REM *  Assumptions:                                                   *
REM *    - Discount factor ( 0 <= alpha < 1 )                         *
REM *    - Poisson demand process                                     *
REM *    - Return process is Binomial with parameters (Ω,δ),          *
REM *              where Ω is obtained from E(w).                     *
REM
REM *          -  ( Developed  8/07/91    Revised  1/30/91 ) -       *

' Define dimensions.

DIM isymbol$(5000), ksymbol$(5000), decision%(5000)
DIM preoptcost(5000), optcost(5000), ptrans(5000)
DIM probw(100), probz(100), probgew(100), probgez(100)

' Input data.

tf = 150              'final time
alpha = .9            'discount factor
capC = 0              'fixed ordering cost
c = 6                 'unit ordering cost
capR = 0              'fixed repair facility setup cost
r = 4                 'unit repair cost
f = 1                 'holding cost for repairable items
h = 2                 'holding cost for serviceable items
p = 0                 'junking cost
b = 15                'penalty cost of unsatisfied demand
xbar = 5              'maximum storage capacity
ybar = 5              'maximum repair capacity
mratew = 3            'average demand
delta = .2            'proportion of units demanded returned
maxwarranty = 2       'warranty periods for goods demanded

' Check for the assumption of bound errors approach.

IF alpha = 1 THEN
  CLS
  LOCATE 11, 18: PRINT "Discount factor cannot be 1  for a convergence."
  LOCATE 13, 27: PRINT "Please select 'alpha' again."
  LOCATE 15, 32: PRINT "( 0 <= alpha < 1 )"
  END
```

```
ELSE
  beta = alpha / (1 - alpha)
END IF

IF xbar >= ybar THEN
  lvalue = xbar
  kvalue = ybar
ELSE
  lvalue = ybar
  kvalue = xbar
END IF

ubar = xbar
vbar = kvalue
jbar = ybar

totnstate = (xbar + 1) * (ybar + 1)
totndec = (ubar + 1) * (vbar + 1) * (jbar + 1)
ndecae = (kvalue + 1) * (lvalue + 1)

' Set optimal cost at the final stage to zero.

FOR i% = 1 TO totnstate
  preoptcost(i%) = 0
NEXT i%

' Define demand and return processes.

cummprobw = 0
w% = 0
DO WHILE cummprobw < .999999
  wfactor# = 1
  IF w% = 0 THEN
    wfactor# = 1
  ELSE
    FOR fact% = 1 TO w%
      wfactor# = wfactor# * fact%
    NEXT fact%
  END IF
  probw(w%) = mratew ^ w% * EXP(-mratew) / wfactor#
  cummprobw = cummprobw + probw(w%)
  probgew(w%) = 1 - cummprobw + probw(w%)
  w% = w% + 1
LOOP
wbar = w% - 1

' Total demand during the warranty periods is calculated from E(w).

totsales% = (maxwarranty + 1) * mratew
zbar = totsales%
```

```
zbarfactor# = 1
IF zbar = 0 THEN
  zbarfactor# = 1
ELSE
  FOR fact% = 1 TO zbar
    zbarfactor# = zbarfactor# * fact%
  NEXT fact%
END IF

zcombination = 0
cummprobz = 0
FOR z% = 0 TO zbar
  zfactor# = 1
  IF z% = 0 THEN
    zfactor# = 1
  ELSE
    FOR fact% = 1 TO z%
      zfactor# = zfactor# * fact%
    NEXT fact%
  END IF

  zdiff% = zbar - z%
  zdifffactor# = 1
  IF z% = zbar THEN
    zdifffactor# = 1
  ELSE
    FOR fact% = 1 TO zdiff%
      zdifffactor# = zdifffactor# * fact%
    NEXT fact%
  END IF

  zcombination = zbarfactor# / (zfactor# * zdifffactor#)
  probz(z%) = zcombination * delta ^ z% * (1 - delta) ^ zdiff%
  cummprobz = cummprobz + probz(z%)
  probgez(z%) = 1 - cummprobz + probz(z%)

NEXT z%

' Print out input data.

CLS
PRINT "< INPUT DATA >"
PRINT "     "
PRINT "tf ="; tf, "final time"
PRINT "alpha ="; alpha, "discount factor"
PRINT "capC ="; capC, "fixed ordering cost"
PRINT "c ="; c, "unit ordering cost"
PRINT "capR ="; capR, "fixed repair facility setup cost"
PRINT "r ="; r, "unit repair cost"
PRINT "f ="; f, "holding cost for repairable items"
PRINT "h ="; h, "holding cost for serviceable items"
PRINT "p ="; p, "junking cost"
```

```
PRINT "b ="; b, "penalty cost of unsatisfied demand"
PRINT "       "
PRINT "xbar ="; xbar, "maximum storage capacity"
PRINT "ybar ="; ybar, "maximum repair capacity"
PRINT "       "
PRINT "mratew ="; mratew, "average demand"
PRINT "delta ="; delta, "proportion of units demanded returned"
PRINT "warranty ="; maxwarranty, "warranty periods for units demanded"

LOCATE 24, 24: PRINT "Press any key to continue ....."; INPUT$(1)

TIME$ = "00:00"

' Define states, decisions, and determine optimal policy at each state
  and each period.

FOR t% = 1 TO tf

  IF t% = 1 THEN
    CLS
    LOCATE 11, 6: PRINT "OPTIMAL SOLUTION TO THE"; tf; "- PERIOD
                  REPAIRABLE ITEM INVENTORY MODEL:"
    LOCATE 25, 65: PRINT "please wait ..."
  ELSE
    LOCATE 25, 65: PRINT "please wait ..."
  END IF

  i% = 0
  FOR x% = 0 TO xbar
    FOR y% = 0 TO ybar
      i% = i% + 1
      isymbol$(i%) = STR$(x%) + STR$(y%)

      k% = 0
      FOR u% = 0 TO ubar
        FOR v% = 0 TO vbar
          FOR j% = 0 TO jbar
            DELTAX% = u% + v%
            DELTAY% = v% + j%

            IF DELTAX% > xbar OR DELTAY% > ybar THEN
              decision%(i%) = 0
            ELSEIF u% > 0 AND j% > 0 THEN
              decision%(i%) = 0
            ELSEIF x% + DELTAX% > xbar OR DELTAY% > y% THEN
              k% = k% + 1
              decision%(i%) = 0
            ELSE
              k% = k% + 1
              decision%(i%) = k%
              ksymbol$(k%) = STR$(u%) + STR$(v%) + STR$(j%)
```

```
' Determine Transient Probability matrix.

netdeltax% = x% + u% + v%
netdeltay% = y% - v% - j%

idot% = 0
FOR xdot% = 0 TO xbar
  FOR ydot% = 0 TO ybar
    idot% = idot% + 1
    differx% = netdeltax% - xdot%
    differy% = ydot% - netdeltay%

    IF differx% < 0 OR differy% < 0 THEN
      ptrans(idot%) = 0
    ELSEIF xdot% = 0 AND ydot% = ybar THEN
      ptrans(idot%) = 0
      FOR w% = differx% TO wbar
        ptrans(idot%) = ptrans(idot%) + probw(w%)
                                      * probgez(differy%)
      NEXT w%
    ELSEIF xdot% = 0 AND ydot% <> ybar THEN
      ptrans(idot%) = 0
      FOR w% = differx% TO wbar
        ptrans(idot%) = ptrans(idot%) + probw(w%)
                                      * probz(differy%)
      NEXT w%
    ELSEIF xdot% <> 0 AND ydot% = ybar THEN
      ptrans(idot%) = probw(differx%) * probgez(differy%)
    ELSE
      ptrans(idot%) = probw(differx%) * probz(differy%)
    END IF
  NEXT ydot%
NEXT xdot%

' Determine cost function.

IF u% > 0 THEN
  fixorder% = 1
ELSE
  fixorder% = 0
END IF

IF v% > 0 THEN
  fixrepair% = 1
ELSE
  fixrepair% = 0
END IF

eholding = 0
FOR w% = 0 TO netdeltax%
  eholding = eholding + h* ((netdeltax% - w%) * probw(w%))
NEXT w%
```

```
          epenalty = 0
          FOR w% = netdeltax% TO wbar
            epenalty = epenalty + b* ((w% - netdeltax%) * probw(w%))
          NEXT w%

          cost = fixorder% * capC + c * u% + fixrepair% * capR + r *
                 v% + p * j% + f * netdeltay% + eholding + epenalty

          sum1 = 0
          FOR idot% = 1 TO totnstate
            sum1 = sum1 + ptrans(idot%) * preoptcost(idot%)
          NEXT idot%
          sum2 = cost + alpha * sum1

          IF k% = 1 THEN
            optk% = k%
            mincost = sum2
          ELSEIF sum2 > mincost THEN
            optk% = optk%
            mincost = mincost
          ELSE
            optk% = k%
            mincost = sum2
          END IF
        END IF

      NEXT j%
    NEXT v%
  NEXT u%

  decision%(i%) = optk%
  optcost(i%) = mincost

  ' Calculate error bounds.

  differror = optcost(i%) - preoptcost(i%)
  IF i% = 1 THEN
    minerror = differror
    maxerror = differror
  ELSEIF differror <= minerror THEN
    minerror = differror
    maxerror = maxerror
  ELSEIF differror >= maxerror THEN
    minerror = minerror
    maxerror = differror
  ELSE
    minerror = minerror
    maxerror = maxerror
  END IF

  lowerror = beta * minerror
  uperror = beta * maxerror
```

```
      NEXT y%
    NEXT x%

    ' Print out results

    PRINT SPC(0); "<T>"; SPC(4); "<STATE (x,y)>"; SPC(2);
          "<DECISION (u,v,j)>"; SPC(2); "<EXPECTED COST>"; SPC(3);
          "<LBOUND>"; SPC(3); "<UBOUND>"
    PRINT "      "

    sumoptcost = 0

    FOR i% = 1 TO totnstate
      invt% = tf - t% + 1
      lowbound = optcost(i%) + lowerror
      upbound = optcost(i%) + uperror
      PRINT TAB(1); t%; TAB(8); i%; TAB(13); "("; isymbol$(i%); " )";
            TAB(25); decision%(i%); TAB(30); "("; ksymbol$(decision%(i%));
            " )"; TAB(41); USING "##########.##"; optcost(i%); SPC(3);
            lowbound; upbound
      devision = i% / (ybar + 1)
      IF devision = i% \ (ybar + 1) THEN
        PRINT "      "
      ELSE
        devision = devision
      END IF
      preoptcost(i%) = optcost(i%)
      sumoptcost = sumoptcost + optcost(i%)
    NEXT i%

    PRINT "      "

    avgoptcost = sumoptcost / totnstate
    PRINT TAB(20); "Average total cost per state  ="; USING "######.##";
                   avgoptcost
    PRINT "      "

NEXT t%
PRINT "Total Processing Time = "; TIMER
PRINT "      "
```

APPENDIX II-H: COMPUTER PROGRAM: BIA WITH THE SDD

```
REM * THE PROGRAM "THE23FIN.BAS" USES THE BACKWARD INDUCTION ALGORITHM *
REM * WITH A STATE DECOMPOSITION TECHNIQUE TO APPROXIMATE THE OPTIMAL  *
REM * POLICY, WHICH MINIMIZES THE TOTAL EXPECTED DISCOUNTED COST FOR A *
REM * FINITE-TIME REPAIRABLE INVENTORY MODEL.                          *
REM *                                                                  *
REM *                                                                  *
REM *    Demand function:  p(w) = { 1/(wbar+1)   w=0,1,...,wbar        *
REM *                            { 0             otherwise             *
REM *                                                                  *
REM *                                                                  *
REM *        -  ( Developed  6/25/91    Revised  3/19/92 )  -         *

' Define dimensions.

DIM isymbol$(100), ksymbol$(100), pxtrans(80), pytrans(80), ptrans(80)
DIM usymbol$(100), vsymbol$(100), jsymbol$(100), optdecision%(100)
DIM optcost(100), optxcost(100), optycost(100)
DIM prexoptcost(100), preyoptcost(100), preoptcost(100)
DIM avgsmcost(100), avgsmxcost(100), avgsmycost(100)
DIM probw(20), probgew(20), subprobz(20, 20), subprobgez(20, 20)
DIM probz(20, 20, 20), probgez(20, 20, 20)


' Input data.

tf = 10                 'final time
alpha = .9              'discount factor
capC = 4                'fixed ordering cost
c = 6                   'unit ordering cost
capR = 4                'fixed repair facility setup cost
r = 4                   'unit repair cost
f = 1                   'holding cost for repairable items
h = 2                   'holding cost for serviceable items
p = 0                   'junking cost
b = 15                  'penalty cost of unsatisfied demand
xbar = 3                'maximum storage capacity
ybar = 2                'maximum repair capacity
wbar = 3                'maximum possible demand
delta = .2              'proportion of units demanded returned
maxwarranty = 2         'warranty periods for goods demanded

' Check for the assumption of bound errors approach.

IF alpha = 1 THEN
  CLS
  LOCATE 11, 18: PRINT "Discount factor cannot be 1  for a convergence."
  LOCATE 13, 27: PRINT "Please select 'alpha' again."
  LOCATE 15, 32: PRINT "( 0 <= alpha < 1 )"
  END
```

```
ELSE
  beta = alpha / (1 - alpha)
END IF

' Print out input data.

CLS
PRINT "< INPUT DATA FOR FINITE-HORIZON PROBLEM WITH SDD; THE23FIN.BAS >"
PRINT "     "
PRINT "tf ="; tf, "final time"
PRINT "alpha ="; alpha, "discount factor"
PRINT "capC ="; capC, "fixed ordering cost"
PRINT "c ="; c, "unit ordering cost"
PRINT "capR ="; capR, "fixed repair facility setup cost"
PRINT "r ="; r, "unit repair cost"
PRINT "f ="; f, "holding cost for repairable items"
PRINT "h ="; h, "holding cost for serviceable items"
PRINT "p ="; p, "junking cost"
PRINT "b ="; b, "penalty cost of unsatisfied demand"
PRINT "      "
PRINT "xbar ="; xbar, "maximum storage capacity"
PRINT "ybar ="; ybar, "maximum repair capacity"
PRINT "      "
PRINT "mratew ="; mratew, "average demand"
PRINT "delta ="; delta, "proportion of units demanded returned"
PRINT "warranty ="; maxwarranty, "warranty periods for units demanded"

LOCATE 24, 24: PRINT "Press any key to continue ....."; INPUT$(1)

IF xbar >= ybar THEN
  lvalue = xbar
  kvalue = ybar
ELSE
  lvalue = ybar
  kvalue = xbar
END IF

ubar = xbar
vbar = kvalue
jbar = ybar

totnx = xbar + 1
totny = ybar + 1

totnstate = totnx * totny
totndec = (ubar + 1) * (vbar + 1) * (jbar + 1)
ndecae = (kvalue + 1) * (lvalue + 1)


' Define the demand process (i.e., p.m.f. & c.d.f.).

cummprobw = 0
```

```
FOR w% = 0 TO wbar
  probw(w%) = 1 / (wbar + 1)
  cummprobw = cummprobw + probw(w%)
  probgew(w%) = 1 - cummprobw + probw(w%)
NEXT w%

' Define the return process (i.e., p.m.f. & c.d.f.)

zbar = (maxwarranty + 1) * wbar
sbar = maxwarranty * wbar
zbarfactor# = 1

FOR s% = 0 TO sbar
  zcombination = 0
  cummsubprobz = 0
  FOR z% = 0 TO zbar
    subprobz(s%, z%) = 0
    FOR w% = 0 TO wbar
      diffzw% = z% - w%
      IF diffzw% > s% THEN
        subprobz(s%, z%) = 0
      ELSE
        totsw = s% + w%
        swfactor# = 1
        IF totsw = 0 THEN
          swfactor# = 1
        ELSE
          FOR fact% = 1 TO totsw
            swfactor# = swfactor# * fact%
          NEXT fact%
        END IF

        zfactor# = 1
        IF z% = 0 THEN
          zfactor# = 1
        ELSE
          FOR fact% = 1 TO z%
            zfactor# = zfactor# * fact%
          NEXT fact%
        END IF

        zdiff% = totsw - z%
        zdifffactor# = 1
        IF z% = totsw THEN
          zdifffactor# = 1
        ELSE
          FOR fact% = 1 TO zdiff%
            zdifffactor# = zdifffactor# * fact%
          NEXT fact%
        END IF

        zcombination = swfactor# / (zfactor# * zdifffactor#)
```

```
        subprobz(s%, z%) = subprobz(s%, z%) + zcombination * delta ^ z%
                                        * (1 - delta) ^ zdiff% * probw(w%)
      END IF
    NEXT w%
    cummsubprobz = cummsubprobz + subprobz(s%, z%)
    subprobgez(s%, z%) = 1 - cummsubprobz + subprobz(s%, z%)

  NEXT z%
NEXT s%

FOR s% = 0 TO sbar
  FOR w% = 0 TO wbar
    zcombination = 0
    cummprobz = 0
    FOR z% = 0 TO zbar
      diffzw% = z% - w%
      IF diffzw% > s% THEN
        probz(s%, w%, z%) = 0
        cummprobz = cummprobz + probz(s%, w%, z%)
        probgez(s%, w%, z%) = 1 - cummprobz + probz(s%, w%, z%)
      ELSE
        totsw = s% + w%
        swfactor# = 1
        IF totsw = 0 THEN
          swfactor# = 1
        ELSE
          FOR fact% = 1 TO totsw
            swfactor# = swfactor# * fact%
          NEXT fact%
        END IF

        zfactor# = 1
        IF z% = 0 THEN
          zfactor# = 1
        ELSE
          FOR fact% = 1 TO z%
            zfactor# = zfactor# * fact%
          NEXT fact%
        END IF

        zdiff% = totsw - z%
        zdifffactor# = 1
        IF z% = totsw THEN
          zdifffactor# = 1
        ELSE
          FOR fact% = 1 TO zdiff%
            zdifffactor# = zdifffactor# * fact%
          NEXT fact%
        END IF

        zcombination = swfactor# / (zfactor# * zdifffactor#)
        probz(s%, w%, z%) = zcombination * delta ^ z%
```

```
                                                    * (1 - delta) ^ zdiff%
        cummprobz = cummprobz + probz(s%, w%, z%)
        probgez(s%, w%, z%) = 1 - cummprobz + probz(s%, w%, z%)
      END IF

    NEXT z%
  NEXT w%
NEXT s%

' Set optimal cost at the final stage to zero.

FOR ix% = 1 TO totnx
  prexoptcost(ix%) = 0
NEXT ix%

FOR iy% = 1 TO totny
  preyoptcost(iy%) = 0
NEXT iy%

FOR i% = 1 TO totnstate
  preoptcost(i%) = 0
NEXT i%

TIME$ = "00:00"

' Determine a state space and a decision space.

k% = 0
FOR u% = 0 TO ubar
  FOR v% = 0 TO vbar
    FOR j% = 0 TO jbar
      deltax% = u% + v%
      deltay% = v% + j%

      IF deltax% > xbar OR deltay% > ybar THEN
        k% = k%
      ELSEIF u% > 0 AND j% > 0 THEN
        k% = k%
      ELSE
        k% = k% + 1
        i% = 0
        FOR x% = 0 TO xbar
          FOR y% = 0 TO ybar
            i% = i% + 1
            isymbol$(i%) = STR$(x%) + STR$(y%)
            IF x% + deltax% > xbar OR deltay% > y% THEN
              u% = u%
              v% = v%
              j% = j%
            ELSE
              usymbol$(k%) = STR$(u%)
              vsymbol$(k%) = STR$(v%)
```

```
            jsymbol$(k%) = STR$(j%)
            ksymbol$(k%) = usymbol$(k%) + vsymbol$(k%) + jsymbol$(k%)
          END IF
        NEXT y%
      NEXT x%
    END IF

  NEXT j%
NEXT v%
NEXT u%

' Determine optimal policy at each state and period.

FOR t% = tf TO 1 STEP -1

  IF t% = tf THEN
    CLS
    PRINT "OPTIMAL SOLUTION TO THE"; tf; "- PERIOD REPAIRABLE ITEM
          INVENTORY MODEL:"
    PRINT "      "
    PRINT SPC(0); "< PERIOD >"; SPC(2); "< STATE (x,y) >"; SPC(4);"<S>";
          SPC(3); "< DECISION (u,v,j) >"; SPC(3); "< EXPECTED COST >"
    PRINT "      "
  ELSE
    PRINT "      "
    PRINT SPC(0); "< PERIOD >"; SPC(2); "< STATE (x,y) >"; SPC(4);"<S>";
          SPC(3); "< DECISION (u,v,j) >"; SPC(3); "< EXPECTED COST >"
    PRINT "      "
  END IF

  i% = 0
  ix% = 0
  iy% = 0
  sumoptcost = 0
  FOR x% = 0 TO xbar
    ix% = ix% + 1
    FOR y% = 0 TO ybar
      totsmxcost = 0
      totsmycost = 0
      totsmcost = 0
      i% = i% + 1
      iy% = iy% + 1
      FOR s% = 0 TO sbar
        FOR k% = 1 TO ndecae
          u% = VAL(usymbol$(k%))
          v% = VAL(vsymbol$(k%))
          j% = VAL(jsymbol$(k%))
          netdeltax% = x% + u% + v%
          netdeltay% = y% - v% - j%
          IF netdeltax% > xbar OR netdeltay% < 0 THEN
            k% = k%
          ELSE
```

```
' Transient probabilities for the repair/junk decision.

iydot% = 0
FOR ydot% = 0 TO ybar
  iydot% = iydot% + 1
  differy% = ydot% - netdeltay%
  IF differy% < 0 THEN
    pytrans(iydot%) = 0
  ELSEIF ydot% = ybar THEN
    pytrans(iydot%) = subprobgez(s%, differy%)
  ELSE
    pytrans(iydot%) = subprobz(s%, differy%)
  END IF
NEXT ydot%

' Cost function for the repair/junk decision.

costy = p * j% + f * netdeltay%

sumly = 0
FOR iydot% = 1 TO totny
  sumly = sumly + pytrans(iydot%) * preyoptcost(iydot%)
NEXT iydot%
sum2y = costy + alpha * sumly

IF k% = 1 THEN
  optky% = k%
  minycost = sum2y
ELSEIF sum2y > minycost THEN
  optky% = optky%
  minycost = minycost
ELSE
  optky% = k%
  minycost = sum2y
END IF

' Transient probabilities for the purchase/repair decision.

ixdot% = 0
FOR xdot% = 0 TO xbar
  ixdot% = ixdot% + 1
  differx% = netdeltax% - xdot%
  IF differx% < 0 THEN
    pxtrans(ixdot%) = 0
  ELSEIF xdot% = 0 THEN
    pxtrans(ixdot%) = probgew(differx%)
  ELSE
    pxtrans(ixdot%) = probw(differx%)
  END IF
NEXT xdot%
```

```
' Cost function for the purchase/repair decision.

IF u% > 0 THEN
  fixorder% = 1
ELSE
  fixorder% = 0
END IF

IF v% > 0 THEN
  fixrepair% = 1
ELSE
  fixrepair% = 0
END IF

eholding = 0
FOR w% = 0 TO netdeltax%
  eholding = eholding + h * ((netdeltax% - w%) * probw(w%))
NEXT w%

epenalty = 0
FOR w% = netdeltax% TO wbar
  epenalty = epenalty + b * ((w% - netdeltax%) * probw(w%))
NEXT w%

costx = fixorder% * capC + c * u% + fixrepair% * capR
        + r * v% + eholding + epenalty

sum1x = 0
FOR ixdot% = 1 TO totnx
  sum1x = sum1x + pxtrans(ixdot%) * prexoptcost(ixdot%)
NEXT ixdot%
sum2x = costx + alpha * sum1x

IF k% = 1 THEN
  optkx% = k%
  minxcost = sum2x
ELSEIF sum2x >= minxcost THEN
  optkx% = optkx%
  minxcost = minxcost
ELSE
  optkx% = k%
  minxcost = sum2x
END IF

sum2xy = sum2x + sum2y
IF k% = 1 THEN
  optk% = k%
  mincost = sum2xy
ELSEIF sum2xy > mincost THEN
  optk% = optk%
  mincost = mincost
ELSE
```

```
      optk% = k%
      mincost = sum2xy
    END IF
  END IF
NEXT k%

optxcost(ix%) = minxcost
optycost(iy%) = minycost

totsmxcost = totsmxcost + optxcost(ix%)
totsmycost = totsmycost + optycost(iy%)

optdecision%(i%) = optk%
optsymbol$ = ksymbol$(optk%)
totcost = mincost

' Calculate total cost using the original cost function.

u% = VAL(usymbol$(optdecision%(i%)))
v% = VAL(vsymbol$(optdecision%(i%)))
j% = VAL(jsymbol$(optdecision%(i%)))
netdeltax% = x% + u% + v%
netdeltay% = y% - v% - j%

idot% = 0
FOR xdot% = 0 TO xbar
  FOR ydot% = 0 TO ybar
    idot% = idot% + 1
    differx% = netdeltax% - xdot%
    differy% = ydot% - netdeltay%

    IF differx% < 0 OR differy% < 0 THEN
      ptrans(idot%) = 0
    ELSEIF xdot% = 0 AND ydot% = ybar THEN
      ptrans(idot%) = 0
      FOR w% = differx% TO wbar
        ptrans(idot%) = ptrans(idot%) + probw(w%)
                                * probgez(s%, w%, differy%)
      NEXT w%
    ELSEIF xdot% = 0 AND ydot% <> ybar THEN
      ptrans(idot%) = 0
      FOR w% = differx% TO wbar
        ptrans(idot%) = ptrans(idot%) + probw(w%)
                               * probz(s%, w%, differy%)
      NEXT w%
    ELSEIF xdot% <> 0 AND ydot% = ybar THEN
      ptrans(idot%) = probw(differx%)
                            * probgez(s%, differx%, differy%)
    ELSE
      ptrans(idot%) = probw(differx%)
                            * probz(s%, differx%, differy%)
    END IF
```

```
    NEXT ydot%
  NEXT xdot%

  IF u% > 0 THEN
    fixorder% = 1
  ELSE
    fixorder% = 0
  END IF
  IF v% > 0 THEN
    fixrepair% = 1
  ELSE
    fixrepair% = 0
  END IF

  eholding = 0
  FOR w% = 0 TO netdeltax%
    eholding = eholding + h * ((netdeltax% - w%) * probw(w%))
  NEXT w%

  epenalty = 0
  FOR w% = netdeltax% TO wbar
    epenalty = epenalty + b * ((w% - netdeltax%) * probw(w%))
  NEXT w%
  cost = fixorder% * capC + c * u% + fixrepair% * capR + r * v%
         + p * j% + f * netdeltay% + eholding + epenalty

  suml = 0
  FOR idot% = 1 TO totnstate
    suml = suml + ptrans(idot%) * preoptcost(idot%)
  NEXT idot%
  optcost(i%) = cost + alpha * suml

  totsmcost = totsmcost + optcost(i%)

  ' Print out results

  invt% = t% - 1
  IF s% = 0 THEN
    PRINT TAB(4); invt%; TAB(15); i%; TAB(19); "("; isymbol$(i%);
          " )"; TAB(33); s%; TAB(44); optdecision%(i%); TAB(48);
          "("; ksymbol$(optdecision%(i%)); " )"; TAB(60); USING
          "#########.##"; optcost(i%)
  ELSE
    PRINT TAB(33); s%; TAB(44); optdecision%(i%); TAB(48); "(";
          ksymbol$(optdecision%(i%)); " )"; TAB(60); USING
          "#########.##"; optcost(i%)
  END IF

NEXT s%
PRINT "    "
avgsmxcost(ix%) = totsmxcost / (sbar + 1)
avgsmycost(iy%) = totsmycost / (sbar + 1)
```

```
        avgsmcost(i%) = totsmcost / (sbar + 1)
        sumoptcost = sumoptcost + avgsmcost(i%)

      NEXT y%
    NEXT x%

    FOR i% = 1 TO totnstate
      preoptcost(i%) = avgsmcost(i%)
    NEXT i%

    FOR ix% = 1 TO totnx
      prexoptcost(ix%) = avgsmxcost(ix%)
    NEXT ix%

    FOR iy% = 1 TO totny
      preyoptcost(iy%) = avgsmycost(iy%)
    NEXT iy%

    PRINT "     "
    avgoptcost = sumoptcost / totnstate
    PRINT TAB(10); "Average total cost per state  ="; USING "######.##";
                   avgoptcost
    PRINT "     "

NEXT t%
PRINT "Total Processing Time = "; TIMER
```

APPENDIX II-I: COMPUTER PROGRAM: SAM WITH THE SDD

```
REM *  THE PROGRAM "THE24NEW.BAS" USES THE SUCCESSIVE APPROXIMATION   *
REM *  METHOD WITH A STATE DECOMPOSITION TECHNIQUE TO APPROXIMATE THE *
REM *  OPTIMAL POLICY, WHICH MINIMIZES THE EXPECTED LONG-RUN TOTAL    *
REM *  DISCOUNTED COST FOR INFINITE-TIME REPAIRABLE-ITEM INVENTORY    *
REM *  MODELS.  THE PROGRAM ALSO UTILIZES AN ERROR-BOUND APPROACH TO  *
REM *  ACCELERATE THE CONVERGENCE OF SUCCESSIVE APPROXIMATIONS.       *
REM
REM *  Assumptions:
REM *    - Discount factor ( 0 <= alpha < 1 )                         *
REM *    - Poisson demand process                                     *
REM *    - Return process is Binomial with parameters (Ω,δ),          *
REM *                          where Ω is obtained from E(w).         *
REM
REM *                  -  ( Revised  8/30/91 )  -                     *


' Define dimensions.

DIM usymbol$(500), vsymbol$(500), jsymbol$(500)
DIM isymbol$(500), ksymbol$(500), optdecision%(500)
DIM pxtrans(500), pytrans(500), ptrans(500)
DIM optcost(500), optxcost(500), optycost(500)
DIM prexoptcost(500), preyoptcost(500), preoptcost(500)
DIM probw(200), probz(200), probgew(200), probgez(200)

' Input data.

tf = 150              'final time
alpha = .9            'discount factor
capC = 0              'fixed ordering cost
c = 6                 'unit ordering cost
capR = 0              'fixed repair facility setup cost
r = 4                 'unit repair cost
f = 1                 'holding cost for repairable items
h = 2                 'holding cost for serviceable items
p = 0                 'junking cost
b = 15                'penalty cost of unsatisfied demand
xbar = 5              'maximum storage capacity
ybar = 5              'maximum repair capacity
mratew = 3            'average demand
delta = .2            'proportion of units demanded returned
maxwarranty = 2       'warranty periods for goods demanded

' Check for the assumption of bound errors approach.

IF alpha = 1 THEN
  CLS
  LOCATE 11, 18: PRINT "Discount factor cannot be 1  for a convergence."
  LOCATE 13, 27: PRINT "Please select 'alpha' again."
```

```
  LOCATE 15, 32: PRINT "( 0 <= alpha < 1 )"
  END
ELSE
  beta = alpha / (1 - alpha)
END IF

' Print out input data.

CLS
  PRINT "< INPUT DATA >"
PRINT "     "
PRINT "tf ="; tf, "final time"
PRINT "alpha ="; alpha, "discount factor"
PRINT "capC ="; capC, "fixed ordering cost"
PRINT "c ="; c, "unit ordering cost"
PRINT "capR ="; capR, "fixed repair facility setup cost"
PRINT "r ="; r, "unit repair cost"
PRINT "f ="; f, "holding cost for repairable items"
PRINT "h ="; h, "holding cost for serviceable items"
PRINT "p ="; p, "junking cost"
PRINT "b ="; b, "penalty cost of unsatisfied demand"
PRINT "      "
PRINT "xbar ="; xbar, "maximum storage capacity"
PRINT "ybar ="; ybar, "maximum repair capacity"
PRINT "      "
PRINT "mratew ="; mratew, "average demand"
PRINT "delta ="; delta, "proportion of units demanded returned"
PRINT "warranty ="; maxwarranty, "warranty periods for units demanded"

LOCATE 24, 24: PRINT "Press any key to continue ....."; INPUT$(1)

IF xbar >= ybar THEN
  lvalue = xbar
  kvalue = ybar
ELSE
  lvalue = ybar
  kvalue = xbar
END IF

ubar = xbar
vbar = kvalue
jbar = ybar

totnx = xbar + 1
totny = ybar + 1

totnstate = totnx * totny
totndec = (ubar + 1) * (vbar + 1) * (jbar + 1)
ndecae = (kvalue + 1) * (lvalue + 1)
```

```
' Define demand and return processes.

cummprobw = 0
w% = 0
DO WHILE cummprobw < .999999
  wfactor# = 1
  IF w% = 0 THEN
    wfactor# = 1
  ELSE
    FOR fact% = 1 TO w%
      wfactor# = wfactor# * fact%
    NEXT fact%
  END IF
  probw(w%) = mratew ^ w% * EXP(-mratew) / wfactor#
  cummprobw = cummprobw + probw(w%)
  probgew(w%) = 1 - cummprobw + probw(w%)
  w% = w% + 1
LOOP
wbar = w% - 1

' Total demand during the warranty periods is calculated from E(w).

totsales% = (maxwarranty + 1) * mratew
zbar = totsales%

zbarfactor# = 1
IF zbar = 0 THEN
  zbarfactor# = 1
ELSE
  FOR fact% = 1 TO zbar
    zbarfactor# = zbarfactor# * fact%
  NEXT fact%
END IF

zcombination = 0
cummprobz = 0
FOR z% = 0 TO zbar
zfactor# = 1
  IF z% = 0 THEN
    zfactor# = 1
  ELSE
    FOR fact% = 1 TO z%
      zfactor# = zfactor# * fact%
    NEXT fact%
  END IF

  zdiff% = zbar - z%
  zdifffactor# = 1
  IF z% = zbar THEN
    zdifffactor# = 1
```

```
    ELSE
      FOR fact% = 1 TO zdiff%
        zdifffactor# = zdifffactor# * fact%
      NEXT fact%
    END IF

    zcombination = zbarfactor# / (zfactor# * zdifffactor#)
    probz(z%) = zcombination * delta ^ z% * (1 - delta) ^ zdiff%
    cummprobz = cummprobz + probz(z%)
    probgez(z%) = 1 - cummprobz + probz(z%)

NEXT z%

' Set optimal cost at the final stage to zero.

FOR ix% = 1 TO totnx
  prexoptcost(ix%) = 0
NEXT ix%

FOR iy% = 1 TO totny
  preyoptcost(iy%) = 0
NEXT iy%

FOR i% = 1 TO totnstate
  preoptcost(i%) = 0
NEXT i%

TIME$ = "00:00"

' Determine a state space and a decision space.

k% = 0
FOR u% = 0 TO ubar
  FOR v% = 0 TO vbar
    FOR j% = 0 TO jbar
      deltax% = u% + v%
      deltay% = v% + j%

      IF deltax% > xbar OR deltay% > ybar THEN
        k% = k%
      ELSEIF u% > 0 AND j% > 0 THEN
        k% = k%
      ELSE
        k% = k% + 1
        i% = 0
        FOR x% = 0 TO xbar
          FOR y% = 0 TO ybar
            i% = i% + 1
            isymbol$(i%) = STR$(x%) + STR$(y%)
            IF x% + deltax% > xbar OR deltay% > y% THEN
              u% = u%
              v% = v%
```

```
            j% = j%
          ELSE
            usymbol$(k%) = STR$(u%)
            vsymbol$(k%) = STR$(v%)
            jsymbol$(k%) = STR$(j%)
            ksymbol$(k%) = usymbol$(k%) + vsymbol$(k%) + jsymbol$(k%)
          END IF
        NEXT y%
      NEXT x%
    END IF

  NEXT j%
 NEXT v%
NEXT u%

' Determine optimal policy at each state and period.

FOR t% = 1 TO tf

 IF t% = 1 THEN
   CLS
   LOCATE 11, 6: PRINT "OPTIMAL SOLUTION TO THE"; tf; "- PERIOD
                 REPAIRABLE ITEM INVENTORY MODEL:"
   LOCATE 25, 65: PRINT "please wait .:."
 ELSE
   LOCATE 25, 65: PRINT "please wait ..."
 END IF

 i% = 0
 ix% = 0
 iy% = 0
 sumoptcost = 0
 FOR x% = 0 TO xbar
   ix% = ix% + 1
   FOR y% = 0 TO ybar
     i% = i% + 1
     iy% = iy% + 1
     FOR k% = 1 TO ndecae
       u% = VAL(usymbol$(k%))
       v% = VAL(vsymbol$(k%))
       j% = VAL(jsymbol$(k%))
       netdeltax% = x% + u% + v%
       netdeltay% = y% - v% - j%
       IF netdeltax% > xbar OR netdeltay% < 0 THEN
         k% = k%
       ELSE

         ' Transient probabilities for the repair/junk decisions.

         iydot% = 0
         FOR ydot% = 0 TO ybar
           iydot% = iydot% + 1
```

```
    differy% = ydot% - netdeltay%
    IF differy% < 0 THEN
      pytrans(iydot%) = 0
    ELSEIF ydot% = ybar THEN
      pytrans(iydot%) = probgez(differy%)
    ELSE
      pytrans(iydot%) = probz(differy%)
    END IF
  NEXT ydot%

  ' Cost function for the repair/junk decision.

  costy = p * j% + f * netdeltay%
  sumly = 0
  FOR iydot% = 1 TO totny
    sumly = sumly + pytrans(iydot%) * preyoptcost(iydot%)
  NEXT iydot%
  sum2y = costy + alpha * sumly

  IF k% = 1 THEN
    optky% = k%
    minycost = sum2y
  ELSEIF sum2y > minycost THEN
    optky% = optky%
    minycost = minycost
  ELSE
    optky% = k%
    minycost = sum2y
  END IF

  ' Transient probabilities for the purchase/repair decisions.

  ixdot% = 0
  FOR xdot% = 0 TO xbar
    ixdot% = ixdot% + 1
    differx% = netdeltax% - xdot%
    IF differx% < 0 THEN
      pxtrans(ixdot%) = 0
    ELSEIF xdot% = 0 THEN
      pxtrans(ixdot%) = probgew(differx%)
    ELSE
      pxtrans(ixdot%) = probw(differx%)
    END IF
  NEXT xdot%

  ' Cost function for the purchase/repair decision.

  IF u% > 0 THEN
    fixorder% = 1
  ELSE
    fixorder% = 0
  END IF
```

```
      IF v% > 0 THEN
        fixrepair% = 1
      ELSE
        fixrepair% = 0
      END IF

      eholding = 0
      FOR w% = 0 TO netdeltax%
        eholding = eholding + h * ((netdeltax% - w%) * probw(w%))
      NEXT w%

      epenalty = 0
      FOR w% = netdeltax% TO wbar
        epenalty = epenalty + b * ((w% - netdeltax%) * probw(w%))
      NEXT w%

      costx = fixorder% * capC + c * u% + fixrepair% * capR + r * v%
              + eholding + epenalty

      sumlx = 0
      FOR ixdot% = 1 TO totnx
        sumlx = sumlx + pxtrans(ixdot%) * prexoptcost(ixdot%)
      NEXT ixdot%
      sum2x = costx + alpha * sumlx

      IF k% = 1 THEN
        optkx% = k%
        minxcost = sum2x
      ELSEIF sum2x > minxcost THEN
        optkx% = optkx%
        minxcost = minxcost
      ELSE
        optkx% = k%
        minxcost = sum2x
      END IF

      sum2xy = sum2x + sum2y
      IF k% = 1 THEN
        optk% = k%
        mincost = sum2xy
      ELSEIF sum2xy > mincost THEN
        optk% = optk%
        mincost = mincost
      ELSE
        optk% = k%
        mincost = sum2xy
      END IF
    END IF
  NEXT k%

  optxcost(ix%) = minxcost
  optycost(iy%) = minycost
```

```
optdecision%(i%) = optk%
optsymbol$ = ksymbol$(optk%)
totcost = mincost

' Calculate total cost using the original cost function.

u% = VAL(usymbol$(optdecision%(i%)))
v% = VAL(vsymbol$(optdecision%(i%)))
j% = VAL(jsymbol$(optdecision%(i%)))
netdeltax% = x% + u% + v%
netdeltay% = y% - v% - j%

idot% = 0
FOR xdot% = 0 TO xbar
  FOR ydot% = 0 TO ybar
    idot% = idot% + 1
    differx% = netdeltax% - xdot%
    differy% = ydot% - netdeltay%

    IF differx% < 0 OR differy% < 0 THEN
      ptrans(idot%) = 0
    ELSEIF xdot% = 0 AND ydot% = ybar THEN
      ptrans(idot%) = 0
      FOR w% = differx% TO wbar
        ptrans(idot%) = ptrans(idot%) + probw(w%)
                                        * probgez(differy%)
      NEXT w%
    ELSEIF xdot% = 0 AND ydot% <> ybar THEN
      ptrans(idot%) = 0
      FOR w% = differx% TO wbar
        ptrans(idot%) = ptrans(idot%) + probw(w%)
                                        * probz(differy%)
      NEXT w%
    ELSEIF xdot% <> 0 AND ydot% = ybar THEN
      ptrans(idot%) = probw(differx%) * probgez(differy%)
    ELSE
      ptrans(idot%) = probw(differx%) * probz(differy%)
    END IF
  NEXT ydot%
NEXT xdot%

IF u% > 0 THEN
  fixorder% = 1
ELSE
  fixorder% = 0
END IF
IF v% > 0 THEN
  fixrepair% = 1
ELSE
  fixrepair% = 0
END IF
```

```
    eholding = 0
    FOR w% = 0 TO netdeltax%
      eholding = eholding + h * ((netdeltax% - w%) * probw(w%))
    NEXT w%

    epenalty = 0
    FOR w% = netdeltax% TO wbar
      epenalty = epenalty + b * ((w% - netdeltax%) * probw(w%))
    NEXT w%
    cost = fixorder% * capC + c * u% + fixrepair% * capR + r * v%
           + p * j% + f * netdeltay% + eholding + epenalty

    suml = 0
    FOR idot% = 1 TO totnstate
      suml = suml + ptrans(idot%) * preoptcost(idot%)
    NEXT idot%
    optcost(i%) = cost + alpha * suml

    ' Calculate error bounds.

    differror = optcost(i%) - preoptcost(i%)
    IF i% = 1 THEN
      minerror = differror
      maxerror = differror
    ELSEIF differror <= minerror THEN
      minerror = differror
      maxerror = maxerror
    ELSEIF differror >= maxerror THEN
      minerror = minerror
      maxerror = differror
    ELSE
      minerror = minerror
      maxerror = maxerror
    END IF

    lowerror = beta * minerror
    uperror = beta * maxerror

  NEXT y%
NEXT x%


' Print out results

PRINT SPC(0); "<T>"; SPC(4); "<STATE (x,y)>"; SPC(2);
      "<DECISION (u,v,j)>"; SPC(2); "<EXPECTED COST>"; SPC(3);
      "<LBOUND>"; SPC(3); "<UBOUND>"
PRINT "      "

sumoptcost = 0
FOR i% = 1 TO totnstate
  invt% = tf - t% + 1
```

```
      lowbound = optcost(i%) + lowerror
      upbound = optcost(i%) + uperror
      PRINT TAB(1); t%; TAB(8); i%; TAB(13); "("; isymbol$(i%);
            " )"; TAB(25); optdecision%(i%); TAB(30); "("; ksymbol$
            (optdecision%(i%)); " )"; TAB(41); USING "##########.##";
            optcost(i%); SPC(3); lowbound; upbound
      devision = i% / (ybar + 1)
      IF devision = i% \ (ybar + 1) THEN
        PRINT "      "
      ELSE
        devision = devision
      END IF

      preoptcost(i%) = optcost(i%)
      sumoptcost = sumoptcost + optcost(i%)
    NEXT i%

    FOR ix% = 1 TO totnx
      prexoptcost(ix%) = optxcost(ix%)
    NEXT ix%

    FOR iy% = 1 TO totny
      preyoptcost(iy%) = optycost(iy%)
    NEXT iy%

    PRINT "     "
    avgoptcost = sumoptcost / totnstate
    PRINT TAB(20); "Average total cost per state  ="; USING "######.##";
                   avgoptcost

NEXT t%
PRINT "Total Processing Time = "; TIMER
PRINT "       "
```

BIBLIOGRAPHY

Albright, S.C., and Soni, A., "An Approximation to the Stationary Distribution of A Multidimensional Markov Process", IIE Transactions, 20, (1988), 111-118.

Allen, S.G., and D'Esopo, D.A., "An Ordering Policy for Repairable Stock Items", Operations Research, 16, (1968), 669-674.

Bellman, R., Dynamic Programming, Princeton University Press, Princeton, New Jersey, (1957).

-------, and Dreyfus, S., Applied Dynamic Programming, Princeton University Press, Princeton, New Jersey, (1962).

Bertsekas, D.P., Dynamic Programming and Stochastic Control, Mathematics in Science and Engineering, Vol. 125, Academic Press, (1976).

---------, Dynamic Programming: Deterministic and Stochastic Models, Prentice-Hall, (1987).

---------, and Castanon, D.A., "Adaptive Aggregation Methods for Infinite Horizon Dynamic Programming," IEEE Transactions on Automatic Control, 34, (1989), 589-598.

Brown,Jr., G.F., Corcoran, T.M., and Lloyd, R.M., "Inventory Models with Forecasting and Dependent Demand", Management Science, 17, (1971), 498-499.

Derman, C., "On Sequential Decisions and Markov Chains", Management Science, 9, (1962), 16-24.

------, Finite-State Markovian Decision Processes, Academic Press, (1970).

Heyman, D.P., "Optimal Disposal Policies for a Single-Item Inventory System with Returns", Naval Research Logistics Quarterly, 24, (1977), 385-405.

------, "Return Policies for an Inventory System with Positive and Negative Demands", Naval Research Logistics Quarterly, 25, (1978), 581-596.

------, and Sobel, M.J., Stochastic Models in Operations Research (Vol.II): Stochastic Optimization, McGraw-Hill, (1982).

Hinderer, K., Foundations of Non-Stationary Dynamic Programming with Discrete Time Parameters, Springer-Verlag, Berlin and N.Y., (1970).

Howard, R.A., Dynamic Programming and Markov Processes, MIT Press, (1960).

Isaac, M.H., "An Analysis of Single Item Inventory Systems with Returns", Ph.D. Dissertation, School of Operations Research and Industrial Engineering, Cornell University, (1979).

Kelle, P., and Silver, E.A., "Forecasting the Returns of Reusable Containers", J. of Operations Management, 8, (1989), 17-35.

Manne, A.S., "Linear Programming and Sequential Decisions", Management Science, 6, (1960), 259-267.

Muckstadt, J.A., "A Model for a Multi-Item, Multi-Echelon, Multi-Indenture Inventory System", Management Science, 20, (1973), 472-481.

Muckstadt, J.A., and Isaac, M.H., "An Analysis of Single Item Inventory Systems with Returns", Naval Research Logistics Quarterly, 28, (1981), 237-254.

Phelps, E., "Optimal Decision Rules for Procurement, Repair or Disposal of Spare Parts", RM-2920-PR, RAND Corp., Santa Monica, California, May, (1962).

Porteus, E.L., "Overview of Iterative Methods for Discounted Finite Markov and Semi-Markov Chains," In: Recent Developments in Markov Decision Processes, (R. Hartley, L.C. Thomas, and D.J. White, Eds.), New York: Academic, (1980).

Puterman, M.L., "Markov Decision Processes," In: Handbooks in OR & MS, Vol. 2, (D.P. Heyman and M.J. Sobel, Eds.), North-Holland: Elsevier Science Publishers B.V., (1990).

--------, M.L., and Shin, M.C., "Modified Policy Iteration Algorithms for Discounted Markov Decision Problems", Management Science, 24, (1979), 1127-1137.

Schrady, D.A., "A Deterministic Inventory Model for Repairable Items", Naval Research Logistics Quarterly, 14, (1967), 391-398.

Sherbrooke, C.C., "METRIC: A Multi-Echelon Technique for Recoverable-Item Control", Operations Research, 16, (1968), 122-141.

Simpson, V.P., "An Ordering Model for Recoverable Stock Items", AIIE Transactions, 2, (1970), 315-320.

-------, "Inventory Theory for Repairables," Ph.D. Dissertation, School of Engineering and Science, New York University, (1972).

-------, "Optimum Solution Structure for a Repairable Inventory Problem", Operations Research, 26, (1978), 270-281.

Veinott,Jr., A.F., "The Status of Mathematical Inventory Theory", Management Science, 12, (1966), 745-777.