MARKOV DECISION PROCESSES
AND
AN APPLICATION

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of
Dokuz Eylul University
In partial Fulfillment of the Requirements for
the Degree of Master of Science in Statistics

by
Umay UZUNOĞLU

February, 2001
İZMİR

Ms.Sc. THESIS EXAMINATION RESULT FORM

We certify that we have read the thesis, entitled **"MARKOV DECISION PROCESSES AND AN APPLICATION"** completed by Umay UZUNOGLU under supervision of Prof. Dr. Nilgün MORALI and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Nilgün MORALI
Supervisor

Doç.Dr. Sonay Kadeğlu

Committee Member

Y.Doç.Dr. C. Cengiz Gelibağlu

Committee Member

Approved by the
Graduate School of Natural and Applied Sciences

Prof.Dr.Cahit Helvacı
Director

ACKNOWLEDGMENTS

I wish to express my sincere gratitude to my supervisor Prof. Dr. Nilgün MORALI for her guidance throughout course of this work.

I am also grateful to Ass. Prof. Dr. Cengiz ÇELİKOĞLU for his continual encouragement and support all throughout the work.

I also wish to express my special thanks to Ass. Prof. Dr. Süleyman ALPAYKUT for his special help for obtaining the data.

Umay UZUNOĞLU

ABSTRACT

Markov decision process and dynamic programming which is a solution technique are discussed in this thesis. In application, monthly demands are considered as a random variable, and the process is expressed as a Markov decision process. As a result, according to the data, the optimal production policy is determined.

In conclusion, with this work and the model we constructed, we indicate that the solution, which is intuitively optimal, is also technically optimal.

ÖZET

Bu çalışmada Markov karar süreçleri ve bir çözüm tekniği olarak dinamik programlama incelenmiştir. Uygulamada, aylık talepler bir rasgele değişken olarak ele alınmış ve süreç bir Markov karar süreci olarak ifade edilmiştir. Sonuç olarak, eldeki verilere göre, en iyi üretim politikası belirlenmiştir.

Sonuç olarak, bu çalışma ve kurulmuş olan model ile, sezgisel olarak optimal olan çözümün teknik olarak da en iyi çözüm olduğu gösterilmiştir.

CONTENTS

Chapter Three

DYNAMIC PROGRAMMING

Chapter Four

MARKOV DECISION PROCESSES

Chapter Five

APPLICATION

Chapter Six

CONCLUSION

LIST OF TABLES

LIST OF FIGURES

page

CHAPTER ONE
INTRODUCTION

Science is founded on uncertainty
Lewis Thomas

1.1. Introduction and Overview

There is a general approach like *"the art of managing is in foreseeing"*. Probability and statistics have long since taught us that the future cannot be perfectly forecast but instead should be considered random or uncertain. The word *stochastic* includes the meaning of uncertainty. *Stochastic* is opposed to *deterministic*, and means that some data are random. The word *'stochastic'*, which is opposed to the word deterministic, includes the meaning of uncertainty. In fact, the art of managing is not only in foreseeing, but also in making the best decision under conditions of uncertainty. People make choices under uncertainty, that is choosing from some set of action in situations where we are uncertain about the actual consequences that will occur. The choice-maker need to make the best or optimal decision. Decision analysis provides a rich set of concepts and techniques to aid the decision-maker.

From this point of view, statistical decision theory has been intensively and extensively developed during past twenty years. A unified theory has been constructed during this period, and the concepts and methods have been widely applied to problems in areas of engineering and communications, economics and management, psychology and behavioral sciences, and systems and operations research. Because of these developments, interest in decision theory and its applications has greatly increased at all mathematical levels.

Life presents choices for everyone. Choice arises as a consequence of a problem and a wish to change to alleviate it. The choice is which change to make. An individual, faced with a choice, makes the choice, which presumably he thinks is best for him. How the choice is to be made includes some questions. The role of operations research begins at this point. An operations research worker may be able to help in any of a variety of ways. He may, for example, be able to contribute significantly to deciding how the choice should be made, as well as helping to predict the outcomes of different choices, designing decision support systems to help the choice process, designing systems which draw attention to the need or desirability for a choice to be made, and so on. One kind of choice situation is that in which the outcome of an alternative can be predicted given various states of nature. There are numerous difficulties of prediction. Often it will be possible to predict quite accurately the immediate, local effects of choosing an alternative, but not so confidently the long-term effects.

A particular difficulty arises here because only rarely is making a choice a single unique event. It is often one of a sequence of choices, made by the choice maker and the choices yet to be made will influence the outcome of the alternative under consideration. This leads naturally to the idea of looking at sequences of choices, or at least taking cognizance of the fact of there being a sequence.

It is usually one in a sequence of choices, so that the outcome of this choice depends on choices yet to be taken, by the decision-maker. The problem of sequences of choices can be structured by dynamic programming. In dynamic programming, the conditions that must be satisfied by an optimal time-staged decision process is studied and the question how to exploit these conditions to determine the best action is answered.

Decisions, which have both immediate and long-term consequences, must not be in isolation; today's decision impacts on tomorrow's, and tomorrow's on the next day's. By not accounting for the relationship between present and future outcomes, it is impossible to achieve good overall performance. For example, in a long race,

deciding to sprint at the beginning may deplete energy reserves quickly and result in a poor finish.

In sequential decision-making, when outcomes are uncertain, Markov Decision Process, which is also called stochastic dynamic programs, is used. The past decade has seen a notable resurgence in both applied and theoretical research on Markov decision processes. Branching out from operations research roots of the 1950's, Markov decision process models have gained recognition in such diverse field as ecology, economics, and communications engineering. These new applications have been accompanied by many theoretical advances. The aim of this thesis is to explain the Markov decision process and to find numerical solutions to those kinds of problems, which the optimal policy has no special form that can be exploited.

In the following chapter, the main concepts about stochastic process and types of stochastic process are given. In the third chapter of this study, dynamic programming, which is a solving technique for sequential decision problems, is explained. There is an extensive review of Markov Decision Process in chapter four. And finally, chapter five includes an application of the Markov Decision Process on an inventory problem and conclusions.

CHAPTER TWO
STOCHASTIC PROCESSES

Consider a random variable, such as the number of jobs, N, waiting to be processed in a computer center, it isn't allowed for the fact that the probability distribution of N may change with time. That is, if we let N_1 be the number of jobs in the job queue at 8 A.M. and N_2 the corresponding number at 11 A.M., then N_1 and N_2 may have different probability distributions. (To investigate the nature of the distribution of N_1, the number of jobs at 8 A.M should be noted each day for a number of days; for N_2 the same thing should be done at 11 A.M.) Thus we have a family of random variables $\{N(t), t \in T\}$, where T is the set of all times during the day that the computer center is in operation. Such a family of random variables is called a *stochastic process.* In this chapter, some definitions will be given related with the stochastic processes.

2.1. Stochastic Processes

Consideration of the time element cannot be ignored in many random phenomena observed in engineering and physical sciences. A random phenomenon is defined as an empirical phenomenon that obeys probabilistic, rather than deterministic, laws. Just a few examples are (1) particle movement in Brownian Motion, (2) emissions from a radioactive source, (3) fluctuating current in an electric circuit, (4) wave profile in the ocean, (5) response of an airplane to a wind gust and motion of a ship in a seaway, and (6) vibration of a building caused by an earthquake. In order to evaluate the statistical characteristics of these random phenomena, it is necessary to consider the concept of a family of random variables that is a function of time.

Random phenomena whose characteristics can be determined by this concept are called stochastic or random process. A *stochastic process* is defined to be simply an indexed collection of random variables $\{X_t\}$, where t is a parameter belonging to an index set T, and is denoted by $\{X_t,\ t \in T\}$. The parameter t is most commonly interpreted as time, so T is taken to be the set of nonnegative integers. X_t represents a measurable characteristic of interest at time t. For example the stochastic process, $X_1, X_2, X_3, \ldots$, can represent the collection of weekly (or monthly) inventory levels of a given product, or it can represent the weekly (or monthly) demands for this product.

Since the stochastic processes can be distinguished in the nature of the state space S, and parameter space T, the state and parameter space are defined first.

2.1.1. State space *S*

At particular points of time t labelled 0,1,..., the system is found in exactly one of a finite number of mutually exclusive and exhaustive categories or *states* labelled 0,1,...,M. The points in time may be spaced equally, or their spacing may depend upon the overall behaviour of the physical system in which the stochastic process is imbedded, e.g., the time between occurrences of some phenomenon of interest. Although the states may constitute a qualitative as well as a quantitative characterisation of the system, no loss of generality is entailed by the numerical labels 0,1,...,M, which are used to denote the possible states of the system. Thus the mathematical representation of the physical system is that of a stochastic process $\{X_t\}$, where the random variables are observed at $t=0,1,2,\ldots$, and where each random variable may take on any one of the $(M+1)$ integers $0,1,\ldots,M$. These integers are the characterisation of the $(M+1)$ states of the process.

This is the space in which the possible values of each X_t lie. In the case that $S = \{0,1,2,\ldots\}$, we refer to the process at hand as integer valued, or alternately as a

discrete state process. If S equals the real line $(-\infty,\infty)$, then we call X_t a real valued stochastic process. If S is Euclidean k space then X_t is said to be a k-vector process.

2.1.2. Index Parameter T

In most applications we can interpret t as the time parameter. Then T is the time interval involved and X_t is the observation at time t. "If $T = \{0,1,2,...\}$ then we shall always say that X_t is a discrete time stochastic process. Often when T is discrete we shall write X_n instead of X_t. If $T = [0,\infty)$ or $T = \{t : -\infty < t < \infty\}$ then X_t is called a continuous time process." (Karlin et al., 1975, p.26)

Since the state space S and the index parameter T can be both discrete and continuous, there are four possibilities:

a- Discrete state space S, discrete index parameter T
b- Discrete state space S, continuous index parameter T
c- Continuous state space S, discrete index parameter T
d- Continuous state space S, continuous index parameter T

This can be shown by a pictorial sketch as in Figure 2.1.

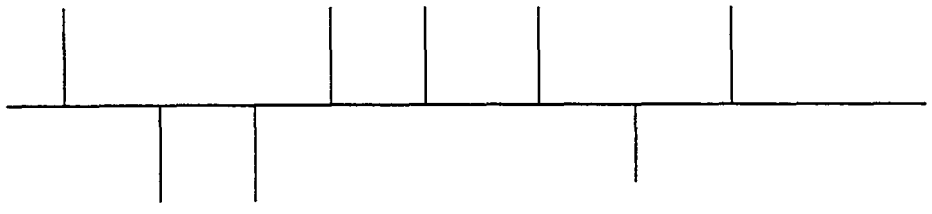

Figure 2.1.a. Pictorial sketch illustrating stochastic process of discrete state, discrete time.

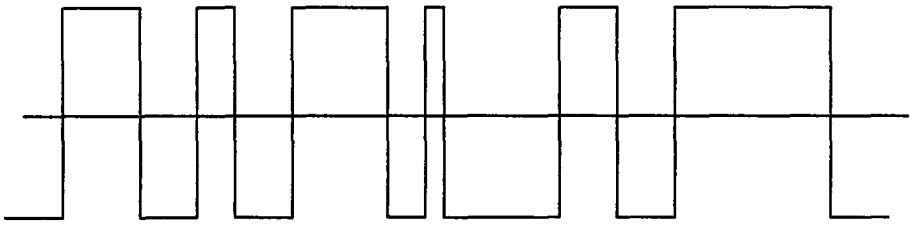

Figure 2.1.b. Pictorial sketch illustrating stochastic process of discrete state, continuous time.

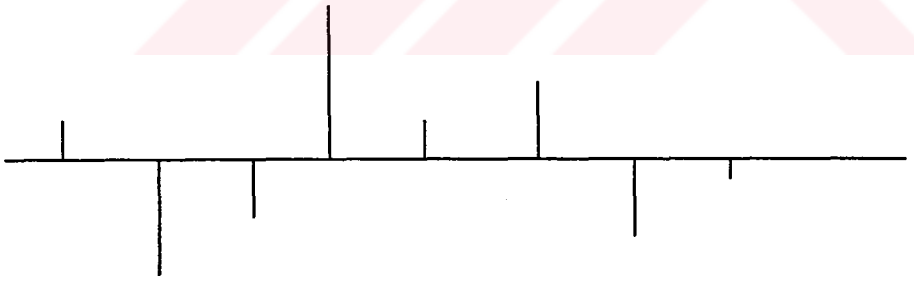

Figure 2.1.c. Pictorial sketch illustrating stochastic process of continuous state, discrete time.

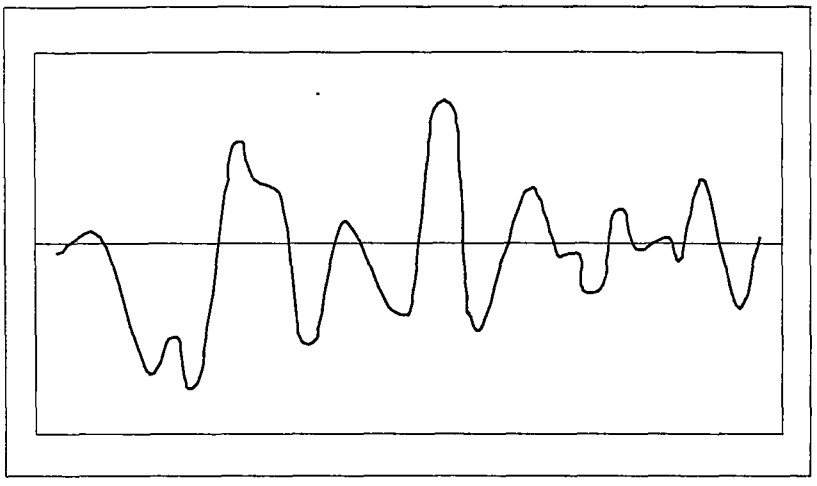

Figure 2.1.d. Pictorial sketch illustrating stochastic process of continuous state, continuous time.

2.2. The Types of Stochastic Processes

The main elements distinguishing stochastic processes are in the nature of the *state space*, the *index parameter T*, and the *dependence relations among the random variables* X_t. It will be described some of the classical types of stochastic processes characterized by different dependence relationships among X_t.

According to the dependence of the relationships among the random variables X_t, some characterizations can be given as follows:

2.2.1. Independent Increments

A stochastic process X_t is said to be an independent increment process if $X_{t_n} - X_{t_{n-1}}$, where $i=0,1,2,...$, is statistically independent. In other words, for all $t_0 < t_1 < t_2 < ... < t_n$, the random variables $X_{t_1} - X_{t_0}$, $X_{t_2} - X_{t_1}, ..., X_{t_n} - X_{t_{n-1}}$ are independent.

Some processes are stationary as well as independent increments that possess both properties.

2.2.2. Stationary Process

"A stationary process is a stochastic process whose probabilistic laws remain unchanged through shifts in time. It is an appropriate assumption for a variety of processes in communication theory, astronomy, biology, ecology, and economics." (Takācs, 1960, p.29)

A stationary process is a stochastic process $\{X_t,\ t \in T\}$ with the property that for any positive integer k and any points $t_1, t_2, ..., t_k$ and h in T, the joint distribution of

$$\{X_{t_1}, X_{t_2}, ..., X_{t_k}\}$$

is the same distribution as the joint distribution of

$$\{X_{(t_1+h)}, X_{(t_2+h)} ..., X_{(t_k+h)}\}.$$

In other words, the process is stationary if choosing any fixed point as the origin, the ensuring process has the same probability law.

In particular, the distribution of X_t is the same for each t. "If a stochastic process does not satisfy this condition, the process is called *evolutionary stochastic process.*" (Ochi, 1990,p.204). In order to clarify the difference between stationary and evolutionary stochastic processes, it is useful to consider the number of emissions from a radioactive source or the number of floodings due to storm. The emissions as well as the floodings occur randomly in time. If we consider the number of occurrences during a period from the beginning of the observation until time t, denoted by $N(t)$, then $N(t)$ is an evolutionary stochastic process since it depends on time t. On the other hand, if we consider the number of occurrences during a specified time interval h, then the number $\{N(t+h) - N(t)\}$ may not depend on time t and thereby it may be a stationary random process.

Economic time series, such as unemployment, gross national product, national income, etc., are often assumed to correspond to a stationary process, at least after some correction for long-term growth has been made.

2.2.3. Stationary Independent Increments

"If the random variables

$$X_{t_2} - X_{t_1},\ X_{t_3} - X_{t_2},...,X_{t_n} - X_{t_{n-1}}$$

are independent for all choices of $t_1,...,t_n$ satisfying

$$t_1 < t_2 < ... < t_n,$$

then we say that X_t is a process with *independent increments.*" (Karlin, 1975, p.27) If the index set contains a smallest index t_0, it is also assumed that

$$X_{t_0},\ X_{t_1} - X_{t_0},\ X_{t_2} - X_{t_1},\ X_{t_3} - X_{t_2},...,X_{t_n} - X_{t_{n-1}}$$

are independent. "If the index set is discrete, that is, $T = \{0,1,2,...\}$ then a process with independent increments reduces to a sequence of independent random variables $Z_0 = X_0,\ Z_i = X_i - X_{i-1}\ \ (i = 1,2,3)$, in the sense that knowing the individual distributions of $Z_0, Z_1,...,$ enables one to determine the joint distribution of any finite set of the X_i. In fact,

$$X_i=Z_0+Z_1+...+Z_i \qquad i=0,1,2,...$$

If the distribution of increments $X_{(t_1+h)} - X_{t_1}$ depends only on the length h of the interval and not on the time t_1 the process is said to have *stationary increments.* For a process with stationary increments the distribution of $X_{(t_1+h)} - X_{t_1}$ is the same as the distribution of $X_{(t_2+h)} - X_{t_2}$, no matter the values of t_1, t_2 *and* h." (Karlin, 1975, p.27)

According to the nature of the state space S an the index parameter T, the stochastic processes can be distinguished as follows:

2.2.4. Counting Processes

An important phase of applied stochastic process theory in engineering and the physical sciences is the prediction of statistical properties associated with the occurrence of a random phenomenon (event). Prediction as to how frequently the random phenomenon takes place and what is the time interval between successive occurrences of the event, and so on, provides information vital for understanding the stochastic behavior of the event. Stochastic processes that deal with the frequency of occurrence of random events are called counting processes. "An integer-valued continuous-time stochastic process *N(t)* is called a *counting process* of the series of events if *N(t)* represents the total number of occurrences of the event in the time interval *t=0* to *t*." (Ochi, 1990, p.211) A stochastic process $\{N(t),\ t \geq 0\}$ is said to be a counting process if *N(t)* represents the total number of "events" that have occurred up to time *t*. Hence, a counting process *N(t)* must satisfy:

(i) $N(t) \geq 0$

(ii) *N(t)* is integer valued.

(iii) If $s < t$, then $N(s) \leq N(t)$.

(iv) For $s < t$, $N(t) - N(s)$ equals the number of events that have occurred in the interval $(s,t]$.

"A counting process is said to possess *independent increments* if the numbers of events that occur in disjoint time intervals are independent. For example this means that the number of events that have occurred by time t (that is, *N(t)*) must be independent of the number of events occurring between times t and $t+s$, that is $N(t+s) - N(t)$." (Ross, 1983, p.31)

A counting process is said to possess *stationary increments* if the distribution of the number of events that occur in any interval of time depends only on the length of the time interval. "In other words, the process has stationary increments if the number of events in the interval $(t_1+s,\ t_2+s]$. This means that, $N(t_2+s) - N(t_1+s)$

has the same distribution as the number of events in the interval $(t_1,t_2]$. That is, $N(t_2)-N(t_1)$ for all $t_1<t_2$, *and* $s>0$." (Ross, 1983, p.31)

Figure 2.5 shows a pictorial sketch of a counting process. "The time intervals between successive occurrences of the events

$$T_1 = t_1,\ T_2 = t_2 - t_1,\ T_3 = t_3 - t_2,\ldots$$

are defined as the *interarrival times*." (Ochi, 1990, p.211). If the interarrival times are independent, identically distributed random variables, then the process is called a *renewal process* (or *renewal counting process*). In particular, if the interarrival times obey an exponential distribution, the stochastic process is called a *Poisson process*. One of the most important types of counting process is the Poisson process. A typical sample path of $N(t)$ is sketched in Figure 2.5. The jumps in the sample function of a counting process mark the arrivals and the number of arrivals in the interval $(t_0,t_1]$ is just $N(t_1)-N(t_0)$.

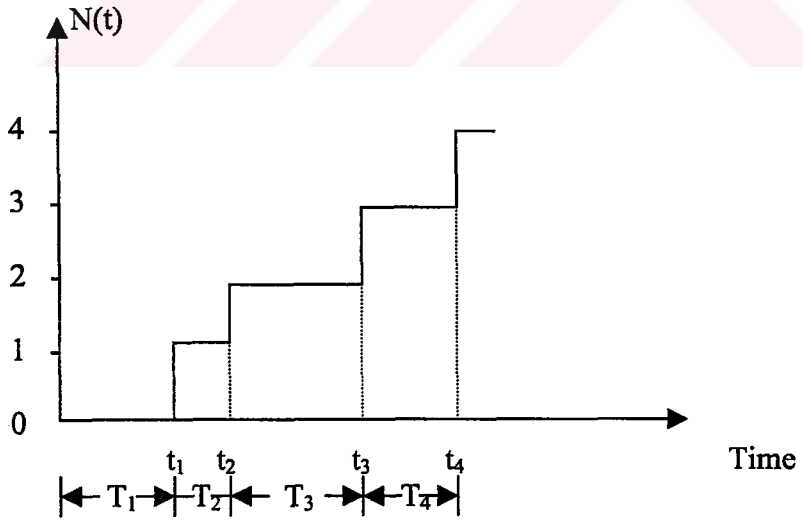

Figure 2.5. Pictorial sketch indicating counting process $N(t)$.

2.2.5. Poisson Processes

The Poisson process arises in many applications, especially as a model for the arrival process at a store, for the arrivals of calls at a telephone exchange, the occurrence of errors in a page of typing, breakdowns on a machine, and the arrival of customers for service, for the arrivals of the radioactive particles at a Geiger counter, etc. The justification for viewing these examples as Poisson processes is based on the concept of the law of rare events. Under these conditions, it is a familiar theorem that the actual number of events occurring follows a Poisson law. The Poisson approximation is excellent if the period of observation is very short. Suppose we are measuring the radioactivity of a certain quantity of radioactive material using Geiger counter. The counter actually records at any instant the total number of particles emitted in the interval from the time it was first set in operation. Let *N(t)* be the total number of particles recorded up to time *t*, where this total number of particles occur at random. Specifying what is meant by "random" quickly supplies a mathematical model of the whole process. For example, assume that the chance of a new particle being recorded in any short interval is independent, not only of the previous states of the system, but also of the present state (this would be a satisfactory assumption for periods of time that were relatively short compared with the half-life of the radioactive substance under investigation). The sample function *N(t)* counts the number of times a specified event occurs during the time period from 0 to *t*. Thus, each possible *N(t)* is represented as a nondecreasing step function.

"The counting process $\{N(t),\ t \geq 0\}$ is said to be a Poisson process having rate λ, $\lambda>0$, if:

(i) $N(0) = 0$

(ii) The process has independent increments.

(iii) The number of events in any interval of length t is Poisson distributed with mean λt. That is, for all s, $t \geq 0$," (Ochi, 1990, p.216)

$$P\{N(t+s) - N(s) = n\} = e^{-\lambda t}\frac{(\lambda t)^n}{n!} \quad n = 0,1,... \tag{2.1}$$

This is a stochastic process with continuous time parameter and discrete state space.

"From the condition *(iii)*, Poisson process has stationary increments. And also

$$E[N(t)] = \lambda t,$$

which explains why λ is called the rate of process." (Ross, 1983, p.32)

Condition *(i)*, states that the counting of events begins at time $t=0$. Condition *(ii)* expresses the number of arrivals in $(t,t+s]$ independent from the past history of the process until time t. In particular, $N(t+s)-N(t)$ is independent of $N(t_1)$, $N(t_2)-N(t_1),\ldots,N(t_n)-N(t_{n-1})$, when $t_1,t_2,\ldots,t_n \leq t$. In other words, $N(t_1)$, $N(t_2)-N(t_1),\ldots,N(t_n)-N(t_{n-1})$ are independent; that is the number of arrivals in disjoint time intervals are independent.

The independent intervals property of the Poisson process must hold even for very small intervals. For example, the number of arrivals in $(t,t+h]$ must be independent of the arrival process over $[0,t]$ no matter how small we choose $h>0$.

The Poisson process is one of the most frequently employed counting processes. There are several generalized forms of the Poisson process that are also extremely important in applied stochastic analysis techniques. In order to determine if an arbitrary counting process is actually a Poisson process, it must be shown that conditions (i), (ii), and (iii) are satisfied.

"Poisson process has some special properties, for example; the sum of independent Poisson process, $N_1(t)$, $N_2(t),\ldots,N_n(t)$, with mean values $\lambda_1 t, \lambda_2 t,\ldots,\lambda_n t$ respectively, is also a Poisson process with mean $(\lambda_1 + \lambda_2 + \ldots + \lambda_n)t$." (Parzen, 1969, p.119)

The difference of two independent Poisson processes $N_1(t)$ and $N_2(t)$ with mean $\lambda_1 t$ and $\lambda_2 t$ respectively, is not a Poisson process.

Essentially, the probability of an arrival during any instant is independent of the past history of the process. In this sense, the Poisson process is *memoryless*. This memoryless property can also be seen when the times between arrivals is examined. The random time X_n between arrivals n-1 and arrival n is called the *nth* interarrival time. The memoryless property of the Poisson process can also be seen in the exponential interarrival times. No matter how long we have waited for the arrival, the remaining time until the arrival always has an exponential distribution with mean $1/\lambda$.

From a sample function of $N(t)$, we can identify the interarrival times X_1, X_2 and so on. Similarly, from the interarrival times X_1, X_2,..., we can construct the sample function of the Poisson process $N(t)$. This implies that an equivalent representation of the Poisson process is the independent identically distributed random sequence X_1, X_2,... of exponentially distributed interarrival times.

2.2.6. The Brownian Motion Processes

The Poisson process is an example of a continuos time, discrete value stochastic process whereas the Brownian Motion process is a continuos time, continuos value stochastic process. The Brownian Motion process is one of the most useful stochastic processes in probability theory. Since its discovery, the process has been used in such areas as statistical goodness of fit, analyzing the prize levels on the stock market.

"A stochastic process $\{X_t,\ t \geq 0\}$ is said to be a Brownian Motion process if

i) $X_0 = 0$

ii) $\{X_t,\ t \geq 0\}$ has stationary increments

iii) for every $t>0$, X_t is normally distributed with mean 0 and variance $c^2 t$."
(Ross, 1983, p.185)

2.2.7. Renewal Processes

Renewal Processes are a specific class of counting processes like Poisson processes. A Renewal process is one, which there are repeated instants when the process starts over. These instants are called renewal points. The generality of renewal process model facilitates the analysis of complex systems.

One of the properties of the Poisson process is that the interarrival times of the process are a sequence of independent, identically distributed random variables that obey the exponential probability law. In more general case in which the interarrival times do not necessarily follow the exponential probability distribution. A counting process that possesses this property is called a renewal counting process.

A Renewal process $N(t)$ is completely characterized by the interval time. Since all the interarrival times are independent and identically distributed, the instant after an arrival, the time until the next arrival has the same distribution as the time (from time zero) until the first arrival.

In other words, after an arrival, the subsequent interarrival times are distributed identically to the original interarrival times. Effectively, the process restarts, or has a *renewal*, when an arrival occurs.

A counting process $N(t)$ that represents the total number of occurrences of an event in time between 0 and t is called a *renewal counting process* if the interarrival times are independent, identically distributed random variables. In other words, "a renewal process is a sequence T_k of independent and identically distributed positive random variables, representing the lifetimes of some "units". The first unit is placed in operation at time zero; it fails at time T_1 and is immediately replaced by a new unit which then fails at time T_1+T_2, and so on, thus motivating the name "renewal process". The time of n^{th} renewal is $S_n=T_1+T_2+...+T_n$." (Karlin, 1975, p.30)

A renewal counting process *N(t)* counts the number of renewals in the interval [*0,t*]. Formally,

$$N_t = n \ \ for \ S_n \le t < S_{n+1}, \quad n = 0,1,2,...$$

For example, since each Bernoulli trial is independent of the previous interarrivals, Bernoulli arrival process is a renewal process. And also, the interarrivals of a Poisson process are independent and identically distributed. Thus, Poisson process is a renewal process. Figure 2.6 shows a pictorial sketch of a renewal process:

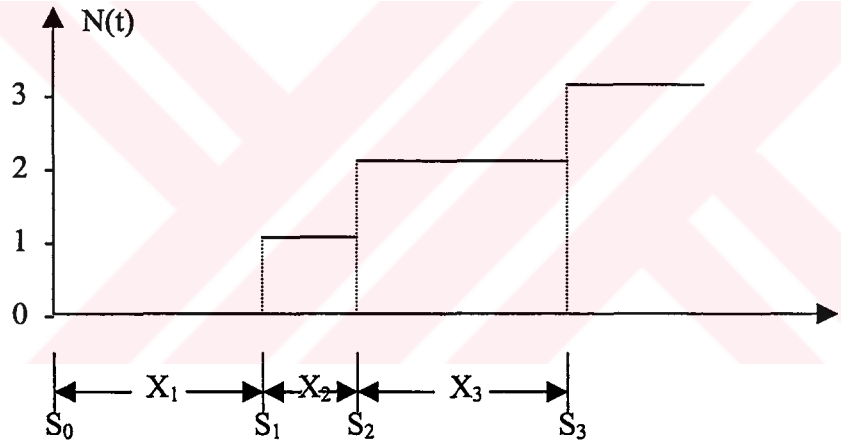

Figure 2.6. The relation between the process of independent and identically distributed random variable's $\{X_n\}$ and the renewal counting process

Renewal processes occur directly in many applied areas such as management science, economics, and biology. Of equal importance, often renewal processes may be discovered embedded in other stochastic processes that, at first glance, seem quite unrelated.

2.2.8. Bernoulli Processes

Consider an experiment consisting of an infinite sequence of trials. Suppose that the trials are independent of each other and that each one has only two possible outcomes: "success" and "failure". This means, the sample space consists of all sequences of successes and failures.

The stochastic process is called a *Bernoulli process* with success probability p provided that;

(a) $X_1, X_2, \ldots$ are independent
(b) $P\{X_n=1\}=P\{X_n=0\}=1-p=q$ for all n.

Some types of stochastic processes are examined. The last one is Markov Processes which is the scope of this study is explained in the following section.

2.3. Markov Processes

Markov Process is the stochastic process, which has the *Markov property*. First of all, Markov property will be explained.

The *Markov property,* sometimes called the *memoryless property*, implies that only the state of the process at time t, X_t matters for probability statements about the future of the process, and not the path to X_t. In other words, given the value of X_t, the values of X_s, s>t, do not depend on the values of X_u, u<t; that is, the probability of any particular future behavior of the process, when its present state is known exactly, is not altered by additional knowledge concerning its past behavior. If our knowledge of the present state of the process is imprecise, than the probability of some future behavior will in general be altered by additional information relating to the past behavior of the system. In formal terms a stochastic process $\{X_t;\ t \geq 0\}$ is said to be Markov Process, if

$$\Pr\{X_{t_{n+1}} = x_{n+1} \mid X_{t_0} = x_0, X_{t_1} = x_1, X_{t_2} = x_2, ..., X_{t_n} = x_n\} = \Pr\{X_{t_{n+1}} = x_{n+1} \mid X_{t_n} = x_n\} \quad (2.2)$$

whenever $t_1 < t_2 < ... < t_n < t_{n+1}$.

Intuitively, (2.2) indicates that the future of the process depends only the present state and not upon the history of the process. That is the entire history of the process is summarized in the present state.

Depending on whether S and T are continuous or discrete four types of Markov Processes might be identified.

A Markov process is called a *Markov chain* if its state space is discrete, Markov process can thus be classified as in Table 2.1 where it is assumed the index set is time.

Table 2.1. Classification of Markov Processes

	State Space	
Type of Parameter	Discrete	Continuous
Discrete Time	Discrete Time Markov Chains	Discrete Time Markov Processes
Continuous Time	Continuous Time Markov Chains	Continuous Time Markov Processes

2.3.1. Markov Chains With Discrete Time

A Markov Process having a discrete state space is called a Markov Chain. Let $\{X_n,\ n = 0,1,2,...\}$ be a Markov chain with state space S. The finite set S whose elements called *states,* will be assumed to be numbered in a definite manner, that is;

S is restricted to non-negative real integers. The value of X_n is the outcome of the *nth* trial. For a discrete time Markov chain, it is fruitful to think of the process as making *state transitions* at times t_n, $n=1,2,3,....$ Thus the discrete time Markov Chain $\{X_n\}$ starts in an initial state, say i, when $t = t_0$ $(X_0 = i)$, and makes a state transition at the next step (time in the sequence); that is, when $t = t_1$, so that $X_1 = j$, etc.

The probability of X_{n+1} being in state j, given that X_n is in state i (called a one-step transition probability), is denoted by $P_{ij}^{(1)}$ and

$$P_{ij}^{(1)} = P\{X_{n+1} = j \mid X_n = i\}, \qquad n, i, j = 0,1,2,...$$

The notation emphasizes that in general the transition probabilities are functions not only of the initial and final state, but also of the time of transition as well. This means transition probabilities generally depend upon the time index n. When one-step transition probabilities are independent of the time variable n, the Markov chain has *stationary transition probabilities* or to be *homogeneous in time.* In other words, if $P\{X_{t+s} = j \mid X_t = i\} = P_s\{i, j\}$ is independent of $t \geq 0$ for all $i, j \in S$ *and* $s \geq 0$, the process X_t is said to be a *time-homogenous Markov process*. All discrete Markov chains in the sequel are assumed to be time homogeneous, unless the contrary is stated.

In this case, P_{ij} is independent of n, and P_{ij} is the probability that the state value undergoes a transition from i to j in one trial. It is customary to arrange these numbers P_{ij} as a square matrix

$$P = \begin{bmatrix} p_{00} & p_{01} & p_{02} & p_{03} & \cdots \\ p_{10} & p_{11} & p_{12} & p_{13} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \\ p_{i0} & p_{i1} & p_{i2} & p_{i3} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \end{bmatrix}$$

P is called as the *Markov matrix* or *transition probability matrix.* If the number of states is finite, then P is a finite matrix whose order (number of rows) is equal to the number of states; otherwise the matrix will be infinite. P_{ij} must satisfy

$$P_{ij} \geq 0, \qquad i, j = 0,1,2,... \tag{2.3}$$

and

$$\sum_{j=0}^{\infty} P_{ij} = 1 \qquad i, j = 0,1,2,... \tag{2.4}$$

Equation (2.4) is true because each P_{ij} is a transition probability. The *(i+1)st* row of P represents the probabilities that the process will make the transition from *state i* to *state j 0,1,2,...*, at the next step. That is this row gives the probability distribution of X_{n+1} given that $X_n = i$. Thus, (2.4) must hold, since some transition occurs (possibly back to state *i*).

The *n*-step transition probabilities $P_{ij}^{(n)}$ will be

$$P_{ij}^{(n)} = P[X_n = j \mid X_0 = i]. \tag{2.5}$$

"Since the Markov chain $\{X_n\}$ has stationary transition probabilities,

$$P_{ij}^{(n)} = P[X_{m+n} = j \mid X_m = i] \quad \text{for all } m \geq 0 \text{ and } n > 0$$

The *n*-step transition probabilities can be computed using the *Chapman-Kolmogorov equations,* that are" (Nemhauser, 1967, p.159)

$$P_{ij}^{(n+m)} = \sum_{k=0}^{\infty} P_{ik}^{(n)} P_{kj}^{(m)} \qquad \text{for all } n, m, i, j \geq 0$$

"To see that the Chapman-Kolmogorov equations are true we reason as follows: $P_{ik}^{(n)}P_{kj}^{(m)}$ is the probability that, starting in state i the process will go to state j in $n+m$ transitions through a path that takes it into state k at the *nth* transition. Hence, summing over all the intermediate states k provides the probability that the process will be in state j after $n+m$ transitions. That is,

$$P_{ij}^{(n+m)} = P[X_{n+m} = j \mid X_0 = i]$$

$$= \sum_{k=0}^{\infty} P[X_{n+m} = j, X_n = k \mid X_0 = i]$$

$$= \sum_{k=0}^{\infty} P[X_{n+m} = j \mid X_n = k, X_0 = i]P[X_n = k \mid X_0 = i]$$

$$= \sum_{k=0}^{\infty} P_{ik}^{(n)}P_{kj}^{(m)}$$." (Nemhauser,1967, p.160)

The process is completely determined once the one-step transition probabilities and the value of X_0 are specified. For example, $P_{ij}^{(n)}$ is the probability that the state at time n is j, given that the state at time 0 is i. To calculate the unconditional distribution of the state at time n, the initial distribution of the states $0,1,2,...$ must be specified. That is,

$$P[X_0 = i] = \pi_i, \qquad \text{for } i \geq 0,$$

where,

$$\sum_{i=0}^{\infty} \pi_i = 1.$$

In analyzing systems modeled as Markov chains, classification of states based on the transition characteristics of Markov chains is the first major step. State j, of a Markov chain $\{X_n\}$ is said to be *reachable* or *accessible*, from state i if it is possible for the chain to proceed from state i to state j in a finite number of transitions. That is

$P_{ij}^{n} > 0$ for some $n \geq 0$. If every state is reachable from every other state, the chain is said to be *irreducible.*

If two states i and j are *accessible* or *reachable* to each other; that is, there exist integers m and n such that $P_{ji}^{(m)} > 0$ and $P_{ij}^{(n)} > 0$, they are said to *communicate.* This is indicated by writing $i \leftrightarrow j$. The relation $i \leftrightarrow j$ is an *equivalence relation* exhibiting properties of

(a) $i \leftrightarrow i$ for each state i	(reflexivity)
(b) If $i \leftrightarrow j$, then $j \leftrightarrow i$	(symmetry)
(c) If $i \leftrightarrow j$ and $j \leftrightarrow k$, then $i \leftrightarrow k$	(transitivity)

Thus, the states of a Markov chain can be partitioned into equivalence classes of states such that two states i and j are in the same class if and only if $i \leftrightarrow j$. In other words, sets of states, which communicate with each other as forming an *equivalence class*. States belonging to different equivalence classes do not communicate with each other even though one-way accessibility is possible. A Markov chain is *irreducible* if and only if there is exactly one equivalence class. Or, it is possible to say; if a Markov chain has all its states belonging to one equivalence class, it is *irreducible.*

Based on the inherent nature of Markov chain with respect to different states, the states can be classified.

"For each state i of a Markov chain $f_i^{(n)}$ is defined to be the probability that the first return to state i occurs n steps or transitions after leaving i. That is,

$$f_i^{(n)} = P\left[X_n = i, X_k \neq i \ \textit{for} \ k = 1,2,...,n-1 \mid X_0 = i\right].$$

$$f^{(0)} = 1 \text{ for all } i.$$

The probability of ever returning to state *i* is given by

$$f_i = \sum_{n=1}^{\infty} f_i^{(n)} .$$

If $f_i<1$, then *i* is said to be a *transient state.* If $f_i=1$, then state *i* is to be *recurrent.*" (Nemhauser, 1967) A state *i* is recurrent if and only if, starting from state i, eventual return of the Markov chain to this state is certain. "If state *i* is recurrent the *mean recurrence time of i,* that is the average time to return to state *i* is defined by

$$m_i = \sum_{n=1}^{\infty} n f_i^{(n)} .$$

If $m_i = \infty$, then state *i* is said to be *recurrent null*, while if $m_i < \infty$, then state *i* is said to be *positive recurrent* or *recurrent nonnull.* Thus, each recurrent state i is either positive recurrent or recurrent null." (Nemhauser, 1967)

Some Markov chain may exhibit periodicity in their behavior. The period of a state i is defined as the greatest common divisor of all integer $n \geq 1$ for which $P_{ij}^{(n)} > 0$. If $P_{ii}^{(n)} = 0$ for all $n \geq 1$ define $d(i)=0$. If $d(i)>1$, state *i* is said to be *periodic,* while if $d(i)=1$, then *i* is an *aperiodic state.*

Periodicity is also a class property and when the period is 1, the state or the class is *aperiodic.*

An aperodic irreducible positive recurrent chain is known as *ergodic.*

Using the communication relation, it is shown that recurrence and transience are class properties. Thus when equivalence classes are identified in a Markov chain model of a system, classification of one state in a class defines the property of the class. A special positive recurrent state is an *absorbing state.* A state i is absorbing if $P_{ij}=1$; it belongs to a class of its own.

"The state and class properties lead directly to the following results, which have proved to be extremely useful in the analysis of Markov models.

1. If state i has a period d_i, $(d_i \geq 1)$, then there exist an integer n such that for all integers $n \geq N$, $P_{ii}^{(nd_i)} > 0$." (Kotz, et al., p.258)

2. Let P be the transition probability matrix of an irreducible aperiodic finite Markov chain. Then there exists an N such that for all $n \geq N$ the n-step transition probability matrix P^n has no zero elements.

3. In a finite Markov chain, as the number of steps tends to infinity, the probability that the process is in a transient state tends to zero irrespective of the state at which the process starts.

4. In a Markov chain not all states can be transient.

5. Suppose that P is the transition probability matrix for an ergodic Markov chain with state space s. Suppose that a system is at state i at the beginning. The probability of being state j after n steps is

$$\lim_{n\to\infty} P_{ij}^{(n)} = \Pi_j$$

That is; the probability of being in state j is independent from state i. If the system is at any state at the beginning, the probability of being state i for a long period, is denoted by π_1. The following transition matrix shows these probabilities.

$$\lim_{n\to\infty} P^{(n)} = \begin{bmatrix} \pi_1 & \pi_2 & \cdots & \pi_s \\ \pi_1 & \pi_1 & \cdots & \pi_s \\ \vdots & \vdots & & \vdots \\ \pi_1 & \pi_2 & \cdots & \pi_s \end{bmatrix}$$

Then, these probabilities can be denoted by a vector:

$$\pi = [\pi_1 \quad \pi_2 \quad \cdots \quad \pi_s]$$

These probabilities are called *steady state probabilities* and the vector π is called the *steady state probability vector.*

2.3.2. Markov Chains With Continuous Time

Consider a Markov chain $\{X_n,\ n = 0,1,2,...\}$ for which the state space S is the real line $(-\infty,\infty)$. Assuming that the chain is the homogenous the transition distribution function can be defined as

$$F_n(x,y) = P[X_{m+n} \le y \mid X_m = x]$$

In this case, the Chapman-Kolmogorov relation takes the form

$$F_{m+n}(x,y) = \int_{-\infty}^{\infty} d_z F_m(x,z) F_n(z,y)$$

2.3.3. Markov Process With Discrete State Space, Continuous Time

A Markov process with discrete state space, continuous time, (continuous parameter space) and stationary transition probabilities as a S*imple Markov Process.* A continuous time Markov chain X_t ,$t > 0$ is a Markov process on the state space $S = \{0,1,2,...\}$. The transition probabilities are stationary

$$P_{ij}(t) = P\{X_{t+s} = j \mid X_s = i\} \qquad i, j \in S$$

this means

$$P_{ij}(t) = P\{X(t) = j \mid X(0) = i\}$$

The probabilities $P_{ij}(t)$ have the following properties for $t>0$

1. $P_{ij}(t) \geq 0$
2. $\sum_{j \in S} P_{ij}(t) = 1$
3. $P_{ij}(t+s) = \sum_{k \in S} P_{ij}(t) P_{kj}(s), \qquad s > 0$
4. $P_{ij}(t)$ is continuous.
5. $$\lim_{t \to 0} P_{ij}(t) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

2.3.4. Markov Process With Continuous State Space, Continuous Time

This is the most general class of Markov process and to derive meaningful results imposition of some structure becomes necessary. There will be three different classes:

- purely discontinuous processes in which changes of state occur only by jumps.
- diffusion processes in which changes of state occur continually
- general discontinuous processes in which changes of state can occur by jumps (of more than one kind) as well as otherwise.

Of all stochastic processes, without any doubt, Markov processes are the ones used in most in applications. There are two major reasons for this. In simpler forms,

the analysis and the results are simple enough to be meaningful even for applied researchers lacking in mathematical sophistication. Secondly, the first order dependence of a Markov process goes quite far in representing real phenomena. Thus Markov processes have found variety of applications as mathematical models in disciplines such as biology physics, chemistry, astronomy, operations research, and computer science.

The influence of operations research on the development of the theory of Markov processes is clearly seen in the growth of the topic known as *Markov decision process* (also called *Markovian decision process).* It can also be considered as the meeting point between statistical decision theory and the general area of optimization developed under applied mathematics and operations research. A Markov decision problem aims at determining optimal policies for decision making in a Markovian setting.

CHAPTER THREE
DYNAMIC PROGRAMMING

In most operations research problems the objective is to find the optimal values of the 'decision variables'. It is probable to be faced with problems in which it might be possible to reach the objective breaking our decisions up into smaller components or parts. This approach is called *multistage problem solving*, and dynamic programming is a systematic technique for reaching an answer in problems of this nature. In this chapter, dynamic programming is examined.

3.1. Overview

Dynamic programming is a useful mathematical technique for making a sequence of interrelated decisions. It provides a systematic procedure for determining the optimal combination of decisions.

Scientific decision-making involves model building and then solving the model to determine an optimal solution. Many models, encompassing different disciplines and areas of application, are amenable to solution by dynamic programming. These models contain many decision variables and have a mathematical structure, which is such that calculations of the optimal decision can be done sequentially. When and how the calculations can be done sequentially is the essence of dynamic programming.

Together with the rapid evolution of high-speed digital computers, pioneer operation analysts simultaneously developed models and solution techniques. Given a machine that could do thousands of calculations per second, it became practical to think about solving problems containing hundreds or even thousands of variables. This realization stimulated the study of iterative optimization schemes and eventually led to the development of linear and nonlinear programming, dynamic programming, and various search methods.

Most important, the size of an optimization model, in terms of separate constraints, must be reduced drastically as compared to typical linear programming models. There are practical advantages to the approach, however, in addition to the value of new results. One is that the span of the planning horizon can be made unbounded. Another is that the dynamic programming opens the way toward solving certain small-scale, yet important cases of nonlinear programming models.

The goal is learning how to characterize the models in such a way as to clarify their dynamic properties. The common characteristic of all dynamic programming models is expressing the decision problem by means of a recursive formulation. There are no simple rules that can be applied mechanically to all problems so as to expose their dynamic properties.

Dynamic programming applications can be seen in many industry areas. Whereas most industrial applications of the linear programming models are oriented to planning decisions in the face of large-scale complex situations, dynamic programming models are typically applied to much smaller-scale phenomena. The following illustrations typify dynamic programming decision models:

a) Inventory reordering rules indicating when to replenish an item and by what amount
b) Production scheduling and employment smoothing doctrines applicable to an environment with fluctuating demand requirements

c) Spare parts level determination to guarantee high-efficiency utilization of expensive equipment
d) Capital budgeting procedures for allocating scarce resources to new ventures.
e) Selection of advertising media to promote wide public exposure to a company's product.
f) Systematic plan or search to discover the whereabouts of a valuable resource.
g) Scheduling methods for routine and major overhauls on complex machinery.
h) Long-ranger strategy for replacing depreciating assets. (DeGroot, 1970, p.211)

3.2. Mathematical Model Description

A mathematical model is a symbolic representation of relations among the factors in a problem of decision-making. A model consists of the *decision variables* (the factors that can be manipulated to achieve the desired objective), the *parameters* (the factors that affect the objective but not controllable) and the *measure of effectiveness*. The measure of effectiveness is the return associated with particular values of the decision variables and parameters. It is alternatively called the utility measure, criterion function, objective function or return function.

In most circumstances, the decision variables are limited in the values. These limitations are generally given by specifying a region of feasibility or constraint set. Sometimes it is possible to represent all or a part of the constraint set by equations or inequalities.

The decision-making problem is known as a feasible solution to the model that yields the greatest possible return or minimum cost.

A great advantage of a mathematical model is its generality and ease of manipulation. Any sort of sensitivity analysis, such as changing values of parameters, variables, constraints, or even changing the functional relationships, is most easily accomplished when there is a mathematical model of a system.

It is difficult to present a mutually exclusive and collectively exhaustive classification for mathematical models of decision-making. But it is useful to make some distinctions. In *deterministic* models, the return is given unambiguously by specifying values for the decision variables. There are no uncontrollable or random variables. In contrast, *stochastic* or *probabilistic* models contain random variables that cannot be controlled and whose values are given by probability distributions. A deterministic model can be considered as a special case of a stochastic model, in which each random variable assumes a particular value with probability 1 and all other values with probability 0.

Different forms of the objective function and constraints yield still another division of mathematical models. It is made very crucial distinction between objective functions several variables that are separable and those are not. Closely related to the notion of separability is single-stage versus multi-stage model. In a single stage decision process all decisions are made simultaneously, while in a multistage decision process the decisions are made sequentially. The division is not based entirely on physical characteristics of the process, since it is often possible to create artificially a multistage process from a single stage one. There are considerable computational advantages to making decisions once at a time, rather than all at once.

Dynamic programming is used to solve complex optimization problems. In most applications, dynamic programming obtains solutions by working backward from the end of a problem toward the beginning, thus breaking up a large unwieldy problem into a series of smaller, more tractable problems. In the following section general approach is given.

3.3. General Approach of Dynamic Programming

Dynamic Programming is an approach to optimization. Optimization means finding a best solution among several feasible alternatives.

There does not exist a standard mathematical formulation of the dynamic programming problem. Rather, dynamic programming is a general type of approach to problem solving, and the particular equations used must be developed to fit each individual situation.

Having constructed an appropriate mathematical model there must be chosen an optimization technique to solve the model. The way to determine an optimal solution depends on the form of the objective function and constraints, the nature and number of variables, the kind of computational facilities available, taste, and experience.

Basically dynamic programming is such a transformation. It takes a sequential or multistage decision process containing many interdependent variables and converts it into a series of single-stage problems, each containing only a few variables. The transformation is invariant in that, the number of feasible solutions and the value of the objective function associated with each feasible solution are preserved. The transformation is based on the intuitively obvious principle that *an optimal set of decisions has the property that whatever the first decision is, the remaining decisions must be optimal with respect to the outcome, which results from the first decision.*

It can be said that a problem with n decision variables can be transformed into n sub problems, each containing only one decision variable. As a rule of thumb, the computations increase exponentially with the number of variables, but only linearly with the number of sub problems. Thus, there can be great computational savings.

Certain problem areas, such as inventory theory, allocation, control theory, and chemical engineering design, have been particularly fertile for dynamic programming applications. The property basic to these problems is that decisions can be calculated sequentially. The method of sequential calculation is the essence of dynamic programming.

Dynamic programming was certainly practiced long before it was named. Richard Bellman is the father of dynamic programming. He invented the rather undescriptive

but alluring name for the approach *dynamic programming*. A more representative but less glamorous name would be *recursive optimization,* and also *backward induction* is also used.

When the decision-maker employs the technique of dynamic programming, he begins with considering the final stage of observation and then he works backward to that first stage of observation. Informally, the reason for using this technique is easily described. In order for the decision-maker to decide whether he should choose a decision in D without any observation at all or he should observe a value X_1, he must know how he will use X_1 if it is observed. The first question, which he must consider is: Will he stop sampling after X_1 has been observed or will he continue and observe X_2? The answer to this question depends, in turn, on the benefit to be gained by observing X_2. His answer leads to the next question: If X_2 is observed, will sampling be terminated then or will it be continued with the observation of X_3? Continuing in this way, the decision-maker is ultimately led to the following question: If the values of $X_1,...,X_{n-1}$ have been observed, should sampling be terminated without another observation or should the final observation X_n be taken?

Usually, it is not difficult to answer the last question. If X_n is observed, then the fixed limit on the number of observations makes it compulsory to terminate the sampling and to choose a decision D. Hence, if sampling has not been stopped earlier, the decision-maker must determine at this final stage whether to choose a decision in D based on the values of $X_1,...,X_{n-1}$ or to take exactly one more observation and then choose a decision in D. The optimal decision will usually depend on the values of $X_1,...,X_{n-1}$ that have been observed.

After the decision-maker has determined, for each set of values of the observations $X_1,...,X_{n-1}$, whether it is worthwhile to take the final observation X_n, he can begin to work backward. Knowing the values of the observations $X_1,...,X_{n-2}$ at the next to last stage, he can now decide whether it is worthwhile to take the next

observation X_{n-1} is observed. By working backward in this way to the first stage, the decision-maker determines, for each possible value of X_1, the optimal continuation throughout the remaining stages. He can therefore evaluate the risk from observing X_1 and continuing in an optimal fashion thereafter and can compare this risk with the risk from the immediate choice of a decision in D without any observations. These comparisons, at the first and subsequent stages, determine the optimal sequential decision procedure.

It is possible to explain sequential decision process with an example.

3.3.1. A Puzzle

A recipe calls for seven ounces of milk, but the cook only has one five-ounce and one eight-ounce cup. How can he measure exactly seven ounces of milk into one of the cups from a large can of milk? No other containers are available.

This is a tricky problem, which requires some deliberation. If this problem is tried to solve keeping in mind the basic idea of multistage analysis, the following results can be obtained.

Clearly, if seven ounces of milk is to be contained in one of the cups, it must be the eight-ounce cup. The problem is to determine how to obtain the final state of seven ounces of milk in the eight-ounce cup.

Suppose the problem were solved and there were seven ounces of milk in the eight-ounce cup. How did it get there? If there were already two ounces of milk in the eight-ounce cup, it would be easy since the last five ounces could be transferred from a full five-ounce cup. Then the problem becomes one of getting two ounces of milk in the eight-ounce cup. How could this be done? Presumably by filling one of the cups and then pouring some out. But if this were to be done using a full eight-ounce cup, six ounces would have to be poured off. That doesn't seem easy. Perhaps three ounces could be poured from a five-ounce cup. That is obviously the answer. If

the eight-ounce cup contained five ounces of milk and if it were then filled to capacity from a full five-ounce cup, the five-ounce cup would contain exactly two ounces of milk. The problem is solved. All that we need to do is trace our steps back until the eight-ounce cup contains seven ounces of milk.

By starting at final state, the key point of getting two ounces of milk in the eight ounce cup is discovered most easily. If we start with the initial state, it appears that more trial and error is required.

This way of using multistage analysis to solve the puzzle is suggestive of an approach to all kind of planning and scheduling problems, from determining the time that various parts of a meal must be prepared in order to have the whole dinner ready at a given time to scheduling research and development activities for a missile that must be operational by a certain date. It is common sense that, when the success of an entire venture depends on the identification and integration of several parts, a step-by-step analysis of each sub problem is essential to the solution of the whole problem.

When solving a complex problem, it often broke into a series of smaller problems and then combined the results from the solutions of the smaller problems to obtain the solution of the whole problem. This is in contrast to solving the problem in one step. Of course if the problem can be solved in one step there is no need to use a multistage approach. But when the problem is complex, a suitable decomposition into subproblems can be found and the difficulty of having to determine the whole solution at once will be overcome with the multistage approach.

"There are three basic concepts used in dynamic programming. The first one is that, dividing the problem into a number of sub problems. The second is, working backward from the natural end of the problem and solving each sub problem in turn. And the last one is that, after solving each sub problem, recording the answer." (Nemhauser, 1967,p.172)

These are main concepts of the dynamic programming technique but there are more than three characteristics, which will be explained below.

3.4. Characteristics of Dynamic Programming

***Characteristic 1*:** The problem can be divided into *stages* with a policy decision required at each stage. Dynamic programming problems require making a sequence of interrelated decisions, where each decision corresponds to one stage of the problem.

***Characteristic 2*:** Each stage has a number of *states* associated with it. By a state, it meant the information that is needed at any stage to make an optimal decision. In general, the states are the various possible conditions in which the system might be at that stage of the problem. The number of states may be either finite or infinite. In order to make correct decision at any stage how we get to the present location is not important. For example, if the system is in *stage n* then the remaining decisions do not depend on how we got to *stage n*. The future decisions just depend on the fact that the system is now in *stage n*.

***Characteristic 3*:** The effect of the policy decision at each stage is to transform the current state into a state associated with the next stage possibly according to a probability distribution. This means the decision chosen at any stage describes how the state at the current stage is transformed into the state at the next stage. In many problems, however, a decision does not determine the next stage's state with certainty; instead, the current decision only determines the probability distribution of the state at the next stage.

***Characteristic 4*:** The solution procedure is designed to find an *optimal policy* for the overall problem. Dynamic programming provides a policy prescription of what to do under every possible circumstance (which is why the actual decision made upon reaching a particular state at a given stage is referred to as policy decision).

***Characteristic 5*:** Given the current state, the optimal decision for each of the remaining stages must not depend on previously reached states or previously decisions. In other words, given the current state, an optimal policy for the remaining stages is independent of the policy adapted in previous stages. This idea is known as the principle of optimality. For dynamic programming problems in general, knowledge of the current state of the system conveys all the information about its previous behavior necessary for determining the optimal policy henceforth. This property is known as Markovian property. Any problem lacking this property cannot be formulated as a dynamic programming.

***Characteristic 6*:** The solution procedure begins by finding the optimal policy for the last stage. The optimal policy for the last stage prescribes the optimal policy decision for each of the possible states at that stage.

***Characteristic 7*:** A *recursive relationship* that identifies the optimal policy for stage n given the optimal policy for stage $(n+1)$, is available. In essence, the recursion formalizes the working backward procedure.

"The precise form of the recursive relationship differs somewhat among dynamic programming problems. However it should be introduced the notation:

N	:number of stages
n	:label for current stage
i_n	:current state for stage n
x_n	:decision variable for stage n
$x_n^*(i_n)$	:optimal value of x_n (given i_n)
$f_n(i_n, x_n^*)$	:contribution of stages n, $n+1,...,N$ to the objective function if the system starts in state i_n at stage n, the immediate decision is x_n, and optimal decisions are made thereafter." (Hiller et al., 1990, p.400)

$$f_n^*(i_n) = f_n(i_n, x_n^*) \tag{3.1}$$

"The recursive relationship will always be of the form

$$f_n^*(i_n) = \max_{x_n} \{ f_n(i_n, x_n) \} \tag{3.2}$$

or

$$f_n^*(i_n) = \min_{x_n} \{ f_n(i_n, x_n) \} \tag{3.3}$$

where $f_n(i_n, x_n)$ would be written in terms of i_n, x_n, $f_{n+1}^*(i_{n+1})$ and probably some measure of the first-stage effectiveness (or ineffectiveness) of x_n.

The *recursive relationship* is given because it keeps recurring as we move backward stage by stage." (Hiller et al., 1990, p.401)

When the current stage number n is decreased by one, the new $f_n^*(i_n)$ function is derived by using the $f_{n+1}^*(i_{n+1})$ function that was just derived during the preceding iteration, and then this process keeps repeating. This property emphasized in the following characteristic.

Characteristic 8: When this recursive relationship is used, the solution procedure moves *backward* stage by stage. Each time it is found the optimal policy for that stage until the optimal policy is found starting at the initial stage.

For all dynamic programming problems, a table such as the following one would be obtained for each stage.

x_n / i_n	$f_n(i_n, x_n)$	$f_n^*(i_n)$	x_n^*

When this table is finally obtained for the initial stage *(n=1)*, the problem of interest is solved.

3.5. Formalizing The Dynamic Programming Technique

If there have been multistage problems containing one solution, decision-making would be missing. Decision-making arises when there is more than one feasible solution to a problem. Associated with each solution is a number of measuring its return. The decision-making or optimization problem is to find the solution yielding maximum return.

The dynamic programming technique is formalized by referring to stages, state variables, optimal decision rules, and optimal policies. A *stage* refers to the particular decision that is faced. A *state variable* is a variable defining the current situation at any stage. An *optimal decision rule* specifies which decision to make, as a function of the state variable and the stage number. An *optimal policy* is a set of optimal decision rules, which guides one's decisions through all stages of the entire problem.

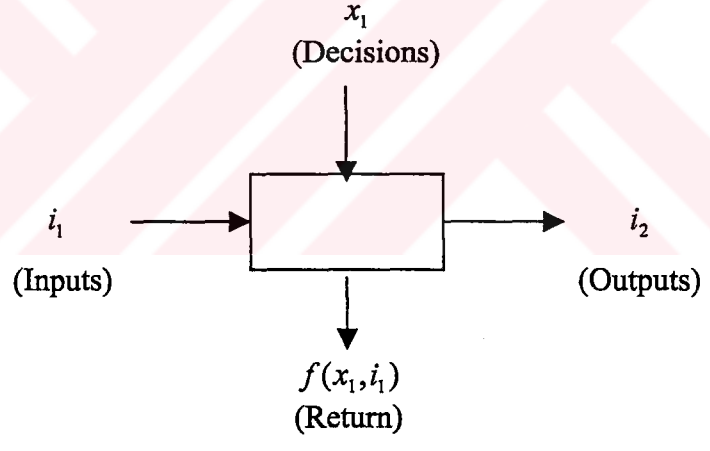

Figure 3.1. A one-stage decision system

An *input state X* that gives the relevant information about inputs to the box (*X* is called the initial state, as it gives a description of the system at the beginning of the stage.)

An *output state* Y that gives the all relevant information about outputs from the box. (Y is called the final state, as it gives the a description of the system at the end of the stage.)

A *decision variable* D is controls the operation of the box.

A *stage return* r, a scalar variable is measures the utility of the box.

3.6. **Deterministic Dynamic Programming**

This section elaborates upon the dynamic programming approach to deterministic problems, where the *state* at the *next stage* is completely determined by the *state* and *policy decision* at the *current stage.*

Deterministic dynamic programming can be described diagrammatically as shown in Figure 3.2. Thus at stage n the process will be in some state i_n. Making policy decision x_n then moves the process to some state i_{n+1} at stage $(n+1)$. The contribution thereafter to the objective function under an optimal policy has been previously calculated to be $f_{n+1}^{*}(i_{n+1})$. The policy decision x_n also makes some contribution to the objective function. Combining these two quantities in an appropriate way provides $f_n(i_n, x_n)$, the contribution of stages n onward to the objective function. Optimizing with respect to x_n then gives $f_n^{*}(i_n) = f_n(i_n, x_n^{*})$. After finding x_n^{*} and $f_n^{*}(i_n)$ for each possible value of i_n, the solution procedure is ready to move back one stage.

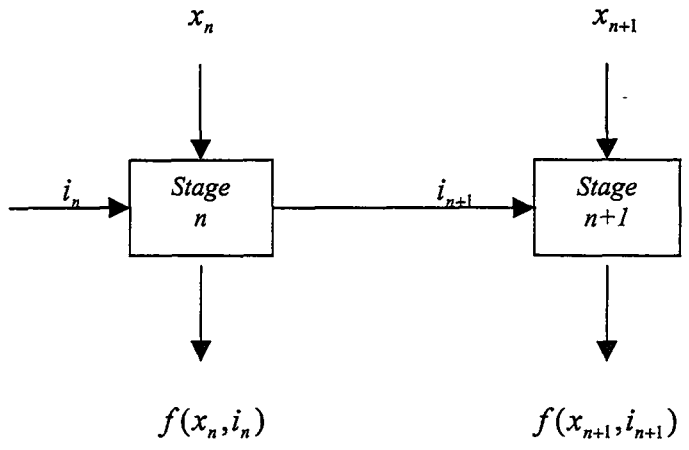

Figure 3.2. The basic structure for deterministic dynamic programming

One way of categorizing deterministic dynamic programming problems is the form of the objective function. For example, the objective might be to minimize the sum of the contributions from the individual states or to maximize such a sum, or to minimize a product of such terms, and so on. Another categorization is in terms of the nature of the set of states for the respective stages. In particular, the states i_n might be representable by a *discrete* state variable, or by a *continuous* state variable.

3.7. **Probabilistic Dynamic Programming**

Probabilistic dynamic programming differs in that the state at next state at the next stage is not completely determined by the state and policy decision at the current stage. Rather, there is a probability distribution for what the next state will be. However, this probability distribution still is completely determined by the state and policy decision at the current stage. The resulting basic structure for probabilistic dynamic programming is described diagrammatically in Figure 3.4.

The system goes to state i with probability p_i $(i = 1,2,...,S)$ given the state i_n and decision x_n at stage n. If the system goes to state i, C_i is the contribution of stage n to the objective function.

When Figure 3.4 is expanded to include all the possible states and decisions at all the stages, it is sometimes referred to as a *decision tree*. If the decision tree is not too large, it provides a useful way of summarizing the various possibilities that may occur.

Because of the probabilistic structure, the relationship between $f_n(i_n, x_n)$ and the $f_{n+1}^*(i_{n+1})$ necessarily is somewhat more complicated than for deterministic dynamic programming. The precise form of this relationship will depend upon the form of the overall objective function. If the objective function were to minimize the expected sum of the contributions from the individual stages, $f_n(i_n, x_n)$ would represent the minimum expected sum from stage *n* onward, given that the state and policy decision at stage *n* are i_n and x_n respectively. Consequently;

$$f_n(i_n, x_n) = \sum_{k=1}^{S} p_k \left[C_k + f_{n+1}^*(k) \right] \tag{3.4}$$

with

$$f_{n+1}^*(k) = \min_{x_{n+1}} f_{n+1}(k, x_{n+1}), \tag{3.5}$$

where this minimization is taken over the feasible values of x_{n+1}.

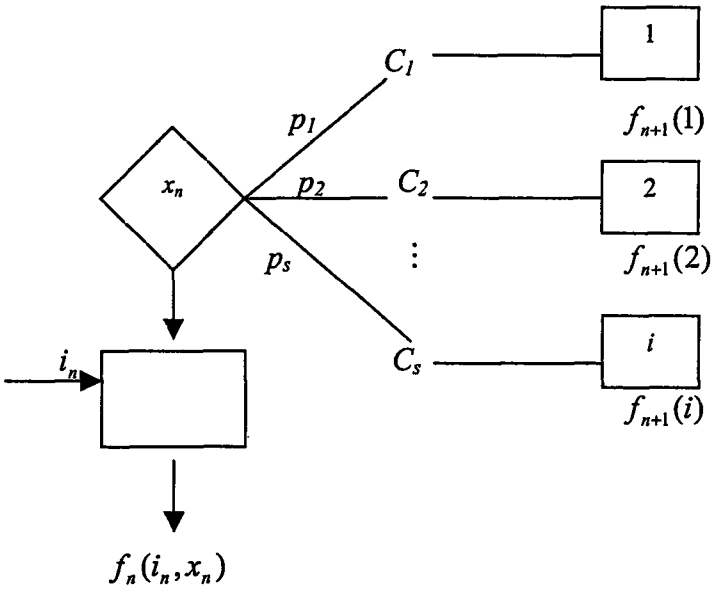

Figure 3.4. The basic structure for probabilistic dynamic programming

Consequently, dynamic programming is a useful technique for making a sequence of interrelated decisions. This technique requires a suitable recursive relationship for each individual problem. It provides a great computational savings, especially for large problems. For instance, if a problem has 10 stages with 10 states and 10 possible decisions at each stage, then exhaustive enumeration must consider up to 10^{10} combinations, whereas dynamic programming need make no more than 10^3 calculations (10 for each state at each stage).

A general kind of model for probabilistic dynamic programming where the stages continue to recur indefinitely is called Markov decision processes. The following chapter is devoted *Markov Decision Process.*

CHAPTER FOUR

MARKOV DECISION PROCESSES

Markov decision processes, also referred to as *stochastic dynamic programs* or *stochastic control problems*, are models for sequential decision making when outcomes are uncertain. The Markov decision process model consists of five elements, which will be explained in this chapter. Choosing an action in a state generates a reward and determines the state at the next decision period through a transition function. Policies and strategies are prescriptions of which action to choose under any eventuality at every future decision period. Decision makers seek policies, which are optimal in some sense.

This chapter introduces main concepts of Markov decision process. The presentation of the theory of finite state-finite action Markov decision process, which forms special case of Stochastic Games will be given in this chapter.

A Markov decision process model consists of five elements:

i) decision periods
ii) states
iii) actions
iv) transition probabilities
v) rewards

It is described in detail below.

4.1. The General Framework For Markov Decision Process

In this chapter, Markovian decision problems that is a kind of sequential decision problems in which the statistician may be required to make two choices at any stage will be considered. First, the decision-maker may have to decide whether to continue experimenting or to terminate the process. Second, if he decides to continue, he may have to choose one of two or more feasible experiments that are available at that stage. The decision-maker may have to choose at each stage one random variable from an appropriate class, which he wishes to observe at that stage. Through his sequential choice of experiments, the decision-maker can exercise some control over the distributions of the observations generated during the process and, hence, over the distributions of his rewards and costs. The problem is to maximize the expected value of some appropriately defined gain function.

The problem of finding an optimal sequential decision procedure is typically difficult. Although every problem of this type has a fixed number of stages n, such a problem is still a sequential decision problem in the following respect:

The decision-maker chooses the experiments sequentially, and his choice of one of the alternatives available at any stage can depend on all observations made in the previous stages.

There is another class of sequential decision problems in which no stopping rule is required. In any problem of this class, there are an infinite number of stages in the process, that is the process continues indefinitely without ever being terminated. At each stage, the decision-maker must choose one of the available alternatives. An important feature of any problem of this class is that the decision-maker's total gain or total loss is typically the sum of an infinite series of random variables.

It will be considered a process that is observed at discrete time points $t = 0,1,2,3,...$ that will sometimes be called *stages*. At each time t, the state of the

process will be denoted by X_t. It is assumed that, X_t is a random variable that can take on values from the set $S = \{1,2,...,n\}$ which will be called state space.

It is also assumed that the process is controlled by a controller or a decision-maker who chooses an action $a \in A(i) = \{a_1, a_2, ..., a_{n-1}\}$ at time t if the process is in state i at that time t. The action chosen may be regarded as a realization of a random variable A_t denoting the controller's choice at time t. Furthermore, it is assumed that the choice of $a \in A(i)$ in state $X_t = i$ results in an immediate reward or output $r(i,a)$, and in a probabilistic transition to a new state $j \in S$. Much of the analysis will depend on the following assumption:

Stationary Markov Transition Property:

For every $i, j \in S$ and $a \in A(i)$, the probability that $X_{t+1} = j$ given that $X_t = i$ and the controller chooses action a is independent of time and any previous states and actions. That is, there exist stationary transition probabilities:

$$P(j \mid i,a) = P\{X_{t+1} = j \mid X_t = i,\ A_t = a\} \qquad \text{for all } t = 0,1,2,...$$

As another point, class of *strategies* or *controls* will be concerned. $f(i,a)$ will be defined as probability that the controller chooses an action $a \in A(i)$ in state $i \in S$ whenever i is visited.

The property that the controller's decisions in state i are invariant with respect to the time of visit to i sometimes is called the *stationarity* of the strategy and such strategies are called *stationary strategies*.

It can be easily seen that a strategy f defines a probability transition matrix. And

$$P(j \mid i, f) = \sum_{a \in A} P(j \mid i,a) f(i,a)$$

And also, the process has to move into one of the states of S at every transition, that is,

$$\sum_{i \in S} P(j \mid i,a) = 1 \qquad \text{for all } a \in A(i),\ i \in S$$

Hence, *P(f)* whose elements are the probabilities $P(j \mid i, f)$ is a stochastic matrix and defines a probability transition matrix. Moreover, *P(f)* uniquely defines a Markov chain.

A Markov decision processes is a decision process in which the transition probabilities take the form $P(j \mid i,a)$, dependent on action a taken.

The Markov decision model has many potential applications in inventory control, maintenance and manufacturing. Commonly, for most applications of Markov decision theory; the optimality criterion of the long-run average cost per unit time is appropriate. The alternative optimality criterion is the total expected discounted costs. However the average cost criterion is particularly appropriate when many transitions occur within a relatively short time. The goal is to choose a sequence of actions, which causes the system to perform optimally with respect to some predetermined performance criterion. Since the system is ongoing, the state of the system prior to tomorrow's decision depends on today's decision. The most important difference from a Markov chain is being apparent at this point. In a Markov chain, if the system is in the *i*th state at time t $(X_t = i)$, it is talked about only the probability of being in the *j*th state at time *t+1* $(X_{t+1} = j)$. If it is a Markov decision process, being in the *j*th state at the next step, depends on decision or action determined in the *i*th state. There is an action space as a difference. Depending on specified action, which decision policy (or strategy) will perform is determined. The objective of Markov decision process theory is to provide frameworks for finding a best decision rule from among a given set. Consequently, decisions must anticipate the opportunities and rewards (or costs) associated with future system states.

For dynamic systems with a given probabilistic law of motion, the simple Markov model is often appropriate. However, in many situations with uncertainty and dynamism, taking a sequence of actions can control the state transitions. The Markov decision model is a versatile and powerful tool for analyzing probabilistic sequential decision process with an infinite planning horizon.

A decision maker, agent, or controller is faced with the problem, or the opportunity, of influencing the behavior of a probabilistic system as it evolves through time. He does this by making decisions or choosing actions. His goal is to choose a sequence of actions, which causes the system to perform optimally with respect to some predetermined performance criterion. Since the system, which is modeled, is ongoing, the state of the system prior the tomorrow's decision depends on today's decision. Consequently, decisions mustn't be myopically, but must anticipate the opportunities and costs (or rewards) associated with future system states.

4.1.1. Decision Periods (Stages):

Decisions are made at points of time referred to as ***decision periods***. Let T denote the set of decision periods. T is the subset of the non-negative real line. T can be either discrete or continuous; and can be either finite or infinite set.

When it is discrete, decisions are made at all decision periods. In discrete time problems, time is divided into periods or stages. When formulating models, we assume that a decision period corresponds to the beginning of a period. The set of decision periods either finite or infinite, in which case $T=\{1,2,...,N\}$ for some integer $N<\infty$, or infinite in which case $T=\{1,2,...\}$.

When it is continuous, decisions may be made at all decision periods (continuously) or at random points of time when certain events occur, (such as arrivals to a queuing system) or at opportune times chosen by the decision-maker.

When decisions are made continuously, the sequential decision problems are best analyzed using control theory methods based on dynamic equations.

Elements of T (decision periods) will be denoted by t and usually referred to as *"time t"*. When N is finite, the decision problem will be called a *finite horizon* problem; otherwise it will be called an *infinite horizon* problem. In finite horizon problems, decisions are *not* made at decision period N. The last decision is made at decision period $N-1$.

4.1.2. States and Actions:

If the problem has a discrete set of states, S, and for each state $i \in S$, a feasible set of *actions, A(i)* exists. $P(j \mid i,a)$ is the probability that if the system is in state i, and takes action $a \in A(i)$, the next state will be j.

At each decision period (stage), the system occupies one *state*. The system occupies one of a finite set of states at the beginning of each of a decision period (stage). The possible set of system states can be denoted by S. The random variable X_t will denote the state at the beginning of the decision period t. For each state $X_t=i$, there is a feasible action (alternative) space *A(i)* where it is finite for each value of i. If, at some decision period, the decision-maker observes the system in state $i \in S$, it may be chosen action a from the set of allowable actions, in state i, *A(i)*. It should be noted that, it is assumed S and *A(i)* do not vary with t. The sets S and *A(i)* may be either finite sets or countable infinite sets.

4.1.3. Rewards and Transition Probabilities

At each decision period, there will be a *reward*. Costs or penalties may be seen as negative rewards. As a result of choosing action $a \in A(i)$ in state i, at decision period t, the decision-maker receives a reward, which is denoted by $r(i,a)$. That is the real valued function $r(i,a)$ defined for $i \in S$, and $a \in A(i)$ denote the value at *time t* of the

reward received in *period t.* When positive, $r(i,a)$ may be regarded as *income*, and when negative as *cost*. The reward might be a lump sum received at a fixed or random time, or a random quantity that depends on the system state at the subsequent decision period.

Suppose that the process is in a state $i \in S$ at some stage and that the decision-maker chooses an alternative $a \in A(i)$. Then he receives a reward or an expected reward $r_t(i,a)$, and the process moves to a new state j in the space S. It is given by the transition function $P(j \mid i,a)$. "Let $r_t(i,a,j)$ denote the value at time t of the reward received when the state of the system at decision period t is $i \in S$, action $a \in A(i)$ is selected, and the process occupies state j at decision period $t+1$. Its expected value at decision period t may be evaluated by computing

$$r_t(i,a)=\sum_{j \in S} r_t(i,a,j)p_t(j \mid i,a) \tag{4.1}$$

In (4.1), the non-negative function $p_t(j \mid i,a)$ denotes the probability that the system is in state $j \in S$ at time $t+1$, when the decision-maker chooses action $a \in A(i)$ in state i, at time t. This function is called a *transition probability function."* (Puterman, 1994, p.20) It is usually assumed

$$\sum_{j \in S} p_t(j \mid i,a) = 1 \tag{4.2}$$

It is assumed that both the transition function and the reward function depend only on the current state i of the process and on the action a selected by the decision-maker. Neither of these functions varies from stage to stage, and neither depends on the state of the process at any either stages or on the actions selected at those stages.

Processes, which meet these requirements, are often called Markov decision processes. The word 'Markov' is used because the transition probability and reward

functions depend on the past only through the current state of the system and the action selected by the decision-maker in that state.

4.1.4. Decision Rules and Policies

It will now defined a *decision rule* and a *policy*. A *decision rule* prescribes a procedure for action selection in each state at a specified decision period. For the general model it will be assumed that a system is observed at time $t = 0,1,...$, and classified into one of a finite number of states labeled $0,1,...s$. "Let $\{X_t,\ t = 0,1,...\}$ denote the sequence of observed states. After each observation, one of A possible decisions (actions), labeled $1,2,...,A$, is taken. Let $\{A_t, t = 0,1,...\}$ denote the sequence of actual decisions made." (Puterman, 1994, p.21)

A *policy* denoted by R, is a rule for making decisions at each point in time. In principle a policy could use all the previously observed information up to time t, that is the entire history of the system consisting of $X_0, X_1,..., X_t$ and $A_0, A_1,..., A_{t-1}$. However, for most problems encountered in practice, it is sufficient to confine consideration to those policies that depend upon only the observed state of the system at time t, X_t, and the possible decisions available. Hence a policy R can be viewed as a rule that prescribes decision $d_i(R)$ when the system is in state i, $i = 0,1,...,s$. Thus R is completely characterized by the values

$$\{d_0(R), d_1(R),...d_s(R)\}.$$

If there is a description that whenever the system is in state i, the decision to be made is the same for all values of t. Policies possessing this property are called *stationary policies*.

The objective of Markov decision process theory is to provide frameworks for finding best decision rule from among a given set. For each R, there will be a *reward*, which depends on the measure of performance used.

The distinction between policy and decision rule can be stated as follows: A decision rule for time period *t*, determines a probability distribution of actions over *A(i)* when the history of the system is known whereas a policy is a sequence of decision rules and tells us how to determine actions for any time unit of the process.

4.2. Finite Stage (Horizon) Markov Decision Processes

This section concerns finite-horizon, discrete-time Markov decision problems. Markovian Decision Processes constitute an important class of stochastic optimization problems. Time invariant Markov process is a stochastic process in which the state of the system at any stage depends only on the state of the system at the previous stage and on known probabilities. When talking about finite stage Markov decision processes, it is assumed there are finite numbers of states at each stage and a finite number of stages.

Let S denote the *sample space* or *state space*, of a sequential process with n stages, and suppose that at each stage, the state of the process can be represented as a point $i \in S$. Suppose also that at each stage of the process, the statistician must choose one alternative, or experiment, from a given class $A(i)$ of alternatives.

Suppose that the process is initially in a given state $i_1 \in S$ that the statistician selects an alternative $a_1 \in A(i_1)$ and receives the reward $r_1(i_1, a_1)$, and that the process moves to a new state i_2 in accordance with the transition function $p(i_2 \mid i_1, a_1)$. The statistician then observes the new state i_2, selects the alternative $a_2 \in A(i_2)$, and receives the reward $r_2(i_2, a_2)$; and the process moves to a new state i_3 in accordance with the transition function $p(i_3 \mid i_2, a_2)$. By continuing this way, the statistician ultimately observes the state i_{n-1} at stage *n-1*, selects an alternative $a_{n-1} \in A(i_{n-1})$, and receives the reward $r_{n-1}(i_{n-1}, a_{n-1})$; and the process finally moves to its terminal state i_n in accordance with the transition function $p(i_n \mid i_{n-1}, a_{n-1})$. In finite horizon Markov decision processes, no decision is made at decision period *n*.

Consequently; the reward at this time point is only a function of the state. It is denoted by $r_n(i_n)$ and sometimes referred a *salvage value* or a *scrap* value. "At last stage, the statistician receives a terminal reward $r_n(i_n)$. The sequence of rewards depending on the decision at each state can be written as: $\{r_1(i_1,a_1),r_2(i_2a_2),...,r_{n-1}(i_{n-1},a_{n-1}),r_n(i_n)\}$. The problem is to select a sequence of alternatives $a_0,a_1,...,a_{n-1}$, which maximizes the expected the total gain. This gain can be explained as follows:" (Nemhauser, 1967, p.231)

$$E[r_1(i_1,a_1)+...+r_{n-1}(i_{n-1},a_{n-1})+r_n(i_n)]$$

An optimal procedure can be readily characterized by backward induction. The decision-maker must work his way back trough the sequence $a_n,a_{n-1},...$ in order to determine the optimal choice of the initial alternative.

As a result of choosing and implementing a policy, the decision-maker receives rewards or incurs costs in periods *1,...,n*. Properties of policies should be evaluated and the question "which one is the best" should be answered. Let us take the rewards first. Since they are not known prior to implementing the policy, the decision-maker must view the reward sequence as random. The decision-maker's objective is to choose a policy so that the corresponding random reward sequence is as large as possible. This necessitates a method of comparing random reward sequences. The decision-maker might use a criterion, which explicitly takes into account both the total expected value and the variability of the sequence of rewards.

4.2.1. The Expected Total Reward Criterion:

Let $V_n(i_1)$ represent the expected total reward over the decision-making horizon if the system is in state i_1 at the first decision period. It is defined as:

$$V_n(i_1)=E_i\left\{\sum_{t=1}^{n-1}r_t(i_t,a_t)+r_n(i_n)\right\} \qquad (4.3)$$

Under the assumption that there will be a bound $M < \infty$ such that $|r_t(i,a)| \leq M < \infty$ for all states $i \in S$ and all alternatives $a \in A$, and that S, and A are discrete, $V_n(i_1)$ exists and is bounded for each finite n.

To account for the time value of rewards or costs, it is often introduced a *discount factor*. Discount factor means a scalar α, $0 \leq \alpha < 1$ which measures the value at time n of a one-unit reward received at time $n+1$. A one unit reward received t periods in the future has present value α^t. "The expected total discounted reward;

$$V_{n,\alpha}(i_1) = E_i\left\{\sum_{t=1}^{n-1} \alpha^{n-1} r_t(i_t, a_t) + \alpha^{n-1} r_n(i_n)\right\} \tag{4.4}$$

Note that the exponent of α is *(t-1)* not t because r_t represents the value at decision period t. For instance r_1 is the value at the first decision period. Substition of $\alpha = 1$ into (4.4) yields the expected total reward (4.3)." (Nemhauser, 1967,p.233)

Taking the discount factor into account does not affect any theoretical results or algorithms in the finite-horizon case but might affect the decision-maker's preference for policies. The discount factor plays a key analytic role in the infinite-horizon models.

Markov decision process theory and algorithms for finite-horizon models primarily concern determining a policy with the largest expected total reward, and characterizing this value. The policy, which gives the maximum reward, referred as an *optimal policy*. The problems defined are discrete-time, finite-horizon Markov decision problems with expected total reward criteria.

Markov decision problem theory and computation is based on using dynamic programming (backward induction) to recursively evaluate expected reward of a fixed policy.

Computing the rewards over the entire future is possible by inductively evaluating the rewards from decision period t and it is called the finite-horizon policy evaluation algorithm.

4.2.2. Linear Programming Approach:

It can be told about advantages of converting the problem to an equivalent linear programming model. The advantage in this regard is the widespread availability of computer algorithms for linear programming.

A policy R can be viewed as a rule that prescribes decision $d_i(R)$ when the system is state i. "Thus R is characterized by the values

$$\{d_0(R), d_1(R), \ldots d_n(R)\}$$

Alternatively, R can be characterized by assigning values $D_{ia} = 0$ or 1 in the matrix

$$\begin{array}{c} \\ \text{State} \end{array} \begin{array}{c} \text{Decision, a} \\ \begin{array}{cccc} 1 & 2 & \cdots & a \end{array} \\ \begin{array}{c} 0 \\ 1 \\ \vdots \\ n \end{array} \begin{bmatrix} D_{01} & D_{02} & \cdots & D_{0a} \\ D_{11} & D_{12} & \cdots & D_{1a} \\ & & \vdots & \\ D_{n1} & D_{n2} & \cdots & D_{ia} \end{bmatrix} \end{array},$$

where each row must contain a single 1 with the rest of the elements zero. When an element $D_{ia} = 1$, interpreted as calling for decision a when the system is in state i." (Hiller et al., 1990, p.775) For example;

		Decision a	
		1	2
State	0	1	0
	1	0	1
	2	0	1

The interpretation of the matrix is; in state 0, decision 1 is made; in state 1 and state 2, decision 2 is made.

The expected cost of a policy can be expressed ad a linear function of the D_{ia}, subject to linear constraints. The D_{ia} are integers (0 or 1), and continuous variables are required for a linear programming.

This requirement can be handled by expanding the interpretation of a policy. "The policy calling for $D_{ia} = 0$ or 1 is called a *deterministic policy* and the decision-maker makes the same decision every time the system is in state i another kind of policy which is called a *randomized policy* calls for determining a probability distribution for the decision to be made when the system is in state i." (Hiller et al., 1990, p.775) Thus D_{ia} can be viewed as

$$D_{ia} = P\{decision = a \mid state = i\} \qquad a=1,2,...,A \quad i=0,1,...,n$$

Randomized policies can be characterized by the matrix

$$\text{State} \begin{array}{c} \\ 0 \\ 1 \\ \vdots \\ n \end{array} \begin{array}{c} \text{Decision, a} \\ \begin{array}{cccc} 1 & 2 & \cdots & a \end{array} \\ \begin{bmatrix} D_{01} & D_{02} & \cdots & D_{0a} \\ D_{11} & D_{12} & \cdots & D_{1a} \\ & & \vdots & \\ D_{n1} & D_{n2} & \cdots & D_{ia} \end{bmatrix} \end{array},$$

where each row sums to 1, and $0 \le D_{ia} \le 1$. Each row $(D_{i1}, D_{i2}, ..., D_{ia})$ is the probability distribution for the decision to be made when the system is in state *i*. For example the following policy is randomized policy, which is given by the matrix

$$\text{State} \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \text{Decision, a} \\ \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1/4 & 1/4 & 1/2 \\ 0 & 1/2 & 1/2 \end{bmatrix} \end{array}.$$

The linear programming formulation is best expressed in terms of a variable y_{ia}, *which is related to* D_{ia} *as follows.* y_{ia} *is assumed the steady state unconditional probability that the system is in state i and decision a is made; that is*

$$y_{ia} = P\{state = i, \ and \ \ decision = a\}$$

From the rules of conditional probability,

$$y_{ia} = \Pi_i D_{ia}$$

Furthermore;

$$\Pi_i = \sum_{a=1}^{A} y_{ia} \qquad and$$

$$D_{ia} = \frac{y_{ia}}{\Pi_i} = \frac{y_{ia}}{\sum_{a=1}^{A} y_{ia}}$$

There are several constraints on y_{ia}*:*

1. *The sum of steady state probabilities of all states is equal to 1.*

$$\sum_{i=0}^{s} \Pi_i = 1$$

we know $\Pi_i = \sum_{a=1}^{A} y_{ia}$*, then*

$$\sum_{i=0}^{s} \Pi_i = \sum_{i=0}^{s} \sum_{a=1}^{A} y_{ia} = 1$$

2. *From the results on steady state probabilities;*

$$\Pi_j = \sum_{i=0}^{s} \Pi_i p_{ij} \quad \textit{so that}$$

$$\sum_{a=1}^{A} y_{ja} = \sum_{i=0}^{s} \sum_{a=1}^{A} y_{ja} p(j \mid i,a) \qquad \textit{for} \quad j = 0,1,...,s$$

3. $y_{ia} \geq 0$ *for* $i = 0,1,...,s$ *and* $a = 1,2,...A$*, the long-run expected average cost per unit time is given by*

$$E(C) = \sum_{i=0}^{s} \sum_{a=1}^{A} \Pi_i C_{ia} D_{ia} = \sum_{i=0}^{s} \sum_{a=1}^{A} C_{ia} y_{ia}$$

The problem is to choose y_{ia} *that*

$$Minimize \sum_{i=0}^{s}\sum_{a=1}^{A} C_{ia} y_{ia}$$

Subject to the constraints

$$\sum_{i=0}^{s}\sum_{a=1}^{A} y_{ia} = 1$$

$$\sum_{a=1}^{A} y_{jk} - \sum_{i=0}^{s}\sum_{a=1}^{A} y_{ia} p(j \mid i,a) = 0 \qquad j = 0,1,...s$$

$$y_{ia} \geq 0 \qquad i = 0,1,...,s$$

$$a = 1,2,...A$$

This formulation is clearly a linear programming problem that can be solved by the simplex method. Once the y_{ia} *are obtained,* D_{ia} *is easily found from*

$$D_{ia} = \frac{y_{ia}}{\sum_{a=1}^{A} y_{ia}}$$

(Hiller et al., 1990, pp.776-777)

4.2.3. Policy Improvement Algorithm

Multistage problems may be solved by analyzing a sequence of simpler inductively defined single-stage problems. In presenting and analyzing this algorithm, it is assumed discrete state space and arbitrary action sets. A key future of this algorithm is that the induction is backwards. This required because of the probabilistic nature of the Markov decision process. If the model were deterministic, policies could be evaluated by either forward or backward induction because expectations needn't be computed.

When the system is in state i and decision $d_i(R) = a$ is made following policy R, the cost C_{ia} is incurred. This cost may represent an expected rather that an actual cost.

C_{ia} : expected cost incurred during next transition if system is in state i and decision a is made.

It is evident that direct enumeration becomes cumbersome when the number of policies is large and algorithms are desirable for finding optimal policies.

An algorithm for finding optimal policies is given by a policy improvement technique. This algorithm is useful in that it often leads to finding the optimal policy quickly, and also it is applicable under more general conditions; for example the number of states may be countably infinite rather that finite. The algorithm is explained below.

Step 1: Select an arbitrarily initial policy, and let $n=0$.

Step 2: Given the trial policy, calculate the y_i, which represents the minimum expected present value, using an optimal strategy over an unbounded time horizon. y_i is according to the value determination equations:

$$y_i^n - \sum_{allj} p(j \mid i,x) \alpha y^n{}_j = c_{ix} \quad \text{for each } i \quad \text{(value determination routine)} \qquad (4.5)$$

Step 3: Test for a policy improvement by calculating the test quantity Y_i^n for each i.

$$\min_x \left[\sum_{allj} p(j \mid i,x) \alpha y_j^n + c_{ix} \right] \equiv Y_i^n \text{ for each } i \text{ (test quantity)} \qquad (4.6)$$

Step 4: Terminate the iterations if $Y_i^n = y_i^n$ for all i. Otherwise, revise the policy at each state k where $Y_k^n < y_k^n$, using a decision yielding Y_k^n in (4.7). Increase n to $n+1$, and return to *step 2* with the new trial policy.

The policy iteration algorithm has some properties as follows:

i) $y_i^{n+1} \le y_i^n$ for each state i and $y_k^{n+1} \le y_k^n$ if $Y_k^n \le y_k^n$

ii) The algorithm terminates in a finite number of iterations

iii) At termination, a policy yielding Y_i^n is optimal.

4.3. Infinite Stage Markov Decision Process

In a problem in which there is no natural terminal stage, it is often appropriate to construct a Markovian decision process with an infinite number of stages in the following manner. The state space S, the alternative or decision space A, the reward function r and the transition function p will be defined as before. It will now be defined α; $0 \le \alpha < 1$ as a factor used to discount future rewards in comparison with current rewards. The process begins in some given state $i_0 \in S$ and is continued indefinitely without ever being terminated. At the jth stage $(j = 0,1,2,...)$, the decision-maker observes the state i_j of the process, selects the alternative $a_j \in A$ and receives the reward $r_j(i_j, a_j)$; and the process moves to a new state i_{j+1} in accordance with the transition function $p(i_{j+1} \mid i_j, a_j)$. The problem is to select an infinite sequence of alternatives $a_0, a_1, a_2...$, which will maximize the expected value of the total gain.

The expected total gain reflects the role of the discount factor α. At any stage of the process, the decision-maker regards the promise of receiving a reward of one unit after n more stages have passed as being equivalent to receiving a reward of α^n units at the current stage. Discount factors are widely used in problems pertaining to economic planning and investment. A process involving an infinite number of stages and discounted rewards has the following two important features:

The first important feature is that in many problems in which there are an infinite number of stages, the decision-maker's expected total gain from an optimal procedure will nevertheless be finite. In the simplest case, there will be a bound

$M < \infty$ such that $|r(i,a)| \leq M$ for all states $i \in S$ and all alternatives $a \in A$. In such a case, the total gain from any sequential procedure cannot be greater than $M/(1-\alpha)$.

The second important feature is that the decision-maker's decision problem is stationary in time in the following sense. Suppose that at the *n*th stage the process is in the state i_n. Since the rewards he has received from his choices at stages 0 through *n-1* will not be affected by future choices, the decision-maker must now maximize his expected total gain over the remainder of the process.

In this process, the transition function remains the same from stage to stage. Hence, except for the factor α^n, the expected total gain from the *n*th stage through the remainder of the process is the same as the expected total gain over the entire process when i_n is regarded as the initial state. Therefore, at any stage the decision problem faced by the decision-maker is the same as the one he faced initially, but there is a different initial state.

It is assumed, for any initial state $i \in S$, that there exists an optimal procedure and that the expected total gain from the procedure is finite. It can be shown that there must be a *stationary* procedure that is optimal for every initial state *i*. In this context, a stationary procedure is one in which the alternative selected at any stage of the process depends only on the state of the process at that stage and is not otherwise dependent on the stage. In other words, a stationary procedure is characterized by a function δ, which assigns, to each state $i \in S$, an alternative $\delta(i) \in A$. When the decision-maker uses the stationary procedure δ, he chooses the alternative $\delta(i) \in A$ whenever the process is in the state $i \in S$.

4.3.1. The Expected Total Cost (Policy Improvement Algorithm)

Given a distribution $P\{X_0 = i\}$ over the initial states on the system and a policy *R*, a system evolves over time according to the joint effect of the probabilistic laws of

motion and the sequence of decisions made (actions taken). In particular, when the system is in state i and decision $d_i(R) = a$ is made, then the probability that the system is in state j at the next observed time period is given by $p(j|i,a)$. Furthermore a known expected cost C_{ia} is incurred. The expected total discounted cost of a system starting in state i (at the first observed time period), is denoted by y_i. Then y_i consists of two components, namely,

(1) C_{ia}, the cost incurred at the first observed time period as a result of the current state i and the decision $d_i(R) = a$ when operating under policy R;

(2) $\alpha \sum_{j=0}^{n} p(j|i,a) y_j^{n-1}$, the expected total discounted cost of the system evolving over the remaining n-1 time periods. Thus the recursive equation,

$$y_i^n = C_{ia} + \alpha \sum_{j=0}^{n} p(j|i,a) y_i^{n-1}, \qquad \text{for } i = 0,1,2,...,n \tag{4.7}$$

and $y_i^1 = C_{ia}$ for all i is obtained. As n approaches infinity, this expression converges to

$$y_i = C_{ia} + \alpha \sum_{j=0}^{n} p(j|i,a) y_j, \text{ for } i = 0,1,2,...,n, \tag{4.8}$$

where y_i can now be interpreted as the expected long-run total discounted cost for a system starting in state i and continuing indefinitely.

The aforementioned procedure not only evaluates a given policy, but also is suggestive of an algorithm to determine the optimal policy. The calculations and the steps are the same as explained in section 4.2.3.

CHAPTER FIVE
APPLICATION

Having examined the theory of Markov Decision Process, an application will be explained in this chapter.

5.1. Problem Definition

In this chapter, an inventory problem is discussed as an application of Markov Decision Process. Data is obtained from a company, which produces spare parts for cars. The production is made in a very large plant and the products are diversified to a wide extent. In our application, we studied with demand data for only a certain product. Monthly demand data for the selected product during four years from 1996 to 1999 is provided. These data are presented in Table 5.1.

The production is made of batches. A batch contains 1500 units. Since the production process is nearly the same for all kinds of production, and the production is made in large amounts, setup cost is considered negligible The production quantity is determined with respect to demand and available storage capacity. Producing a single spare part costs the company 15.71 DM. Since a batch contains 1500 units, production cost is 15.71*1500 DM. per batch. In principle, demand is to be satisfied. In some cases, it is economical to produce more than expected demand and to store the excess amount so that the excess production can be used in the following month. Storing the excess amount causes firm a certain holding cost. Monthly inventory holding cost for a single spare part is 1 DM. Since a batch contains 1500 units,

holding cost per batch is 1500 DM./month. The question is how much the firm should produce each month in order to minimize total cost of production and carrying inventory. The monthly production amount depends on two factors. One of them is monthly demand. The other is the amount of inventory in the beginning of that month, in other words, preceding month's excess production.

Table 5.1. Monthly Demands (in units) for Four Years

Months	1996	1997	1998	1999
January	12460	14885	15150	15320
February	11830	12340	13240	13340
March	12105	13530	14530	15535
April	12980	14740	14840	15640
May	13220	14630	15225	15720
June	12250	15305	15840	15610
July	16150	16210	15920	16040
August	16200	17045	15830	16055
September	15840	14140	15320	17050
October	15920	15520	15150	17125
November	13680	14330	15040	17205
December	14345	14545	15145	17530

5.2. Model Formulation

The aim of the firm is to minimize the total expected cost, under some restrictions. These restrictions are given in the following section. To describe the inventory problem in a formulation, the following descriptions, which are explained in the fourth chapter, should be done.

5.2.1. Stages

In this study, time is examined in months. For the inventory problem, it is considered that, each month corresponds to a stage. Then *T,* which denotes the set of stage is discrete.

$$T = \{1,2,...,N\}$$

Since the objective is to determine the production policy in the long run; N is considered infinite. This means the process is considered unbounded. In other words the problem is infinite horizon problem.

$$T = \{1,2,...,\}$$

5.2.2. States and Actions

States give the description of a system in each stage. In inventory problem, the beginning inventory determines the system's state in each period. State set consists of the values which the beginning inventory i can take. It is limited by the storage capacity.

The available storage capacity for the considered spare parts is limited by 8 batches with respect to the information obtained from the production plant. This also means that the inventory at the beginning of any period cannot exceed 8 batches. If the ending inventory denoted with j, this constraint can be expressed as

$$j \leq 8$$

An action is the decision to be made at each stage. Considering the beginning inventory and demand the production policy is determined. This means for the inventory problem, the action set consists of possible production amounts.

The production capacity is given as 15 batches per month.

5.2.3. Transition Probabilities

$p(X_t = j)$ is sometimes denoted as $P_j(t)$ where j symbolizes state and t symbolizes the time. It is customary to say, *"the process is in state j at time t"*. The transition probability $p(X_t = k \mid X_{t+1} = j)$ is the probability of being in state k at time t given that you are in state j at time $t+1$, where $k, j \in S$ and $t \in T$. Then a Markov chain is a sequence of random variables such that for any t, X_{t+1} is conditionally independent of $X_0, X_1, ..., X_{t-1}$ given X_t. That is the next state X_{t+1} of the process is independent of the past states $X_0, X_1, ..., X_{t-1}$ provided that the present state X_t be known. This property is called Markov property. The transition probabilities $p(X_t = i \mid X_{t+1} = j) = p_{ij}$ form the P, which is called *the transition probability matrix* for the Markov chain. The properties of this matrix are explained in the second chapter.

While considering Markov decision process, besides the state set and index set, there is also an action set as it is explained in the fourth chapter. If the process is in state i at time t an action x is chosen, the next state of the system is chosen according to the transition probabilities $p(j \mid i, x)$.

Define $p(j \mid i, x)$ as conditional probability that the state of the system becomes state j given that; at state i, decision x is made. For given i and x, the sum of the $p(j \mid i, x)$ over all possible j equals one. This definition embodies Markov property assumption. Each transition probability is influenced only by the values of i and x; and not by the specific history of the system (previous states and decisions) prior to arriving at stage i. In inventory problem, the distribution of demand, which is a random variable, is searched according to the data in Table 5.1. Suppose the beginning inventory is i, and the production amount is decided to be x. The state of the system becomes $j = i + x - d$ where d symbolizes the demand. This means, the

inventory in the beginning of the next period will be found by subtracting the demand from the summation of the inventory and the production amount of the previous period. For given values of x and i, the probability of beginning inventory is $(i+x-d)$ equal to the probability that demand is d. Then the transition probabilities will be

$$P(i+x-d \mid i,x) = P(D=d)$$

5.2.4. Objective

The objective function is to minimize the total cost. There are two kinds of costs; production and inventory holding cost. The objective of the company is to devise a schedule that minimizes the total production and inventory holding costs subject to the restriction that it meets all the demand requirements on time. The cost function depends on the past through the current state of the system and the action selected in that state.

In production process, because the production is performed by taking consideration of the excess amount for a previous month, sequential decisions are to be taken. As it is explained in the fourth chapter of this study, when the outcomes are uncertain and when there is a sequential decision process, Markov decision process models can be used. So it is probable to consider the inventory problem, which is going to be analyzed, in Markov decision process context. Thus, the process for the inventory problem can be defined as Markov decision process.

5.3. Solving The Inventory Problem

Having formulated the problem in a context of Markov decision process, solving the problem can be suitable by using the dynamic programming, which is a technique that is used to solve the sequential decision problems.First of all the data are examined. The descriptive statistics for these data is as follows:

Descriptive Statistics

Variable	N	Mean	Median	Tr Mean	StDev	SEMean
Y	48	14954	15188	14980	1418	205

Variable	Min	Max	Q1	Q3
Y	11830	17530	14188	15900

It can be shown that these data is normally distributed with mean 14954, and the standard deviation 1418. The histogram of data is shown in Figure 5.1:

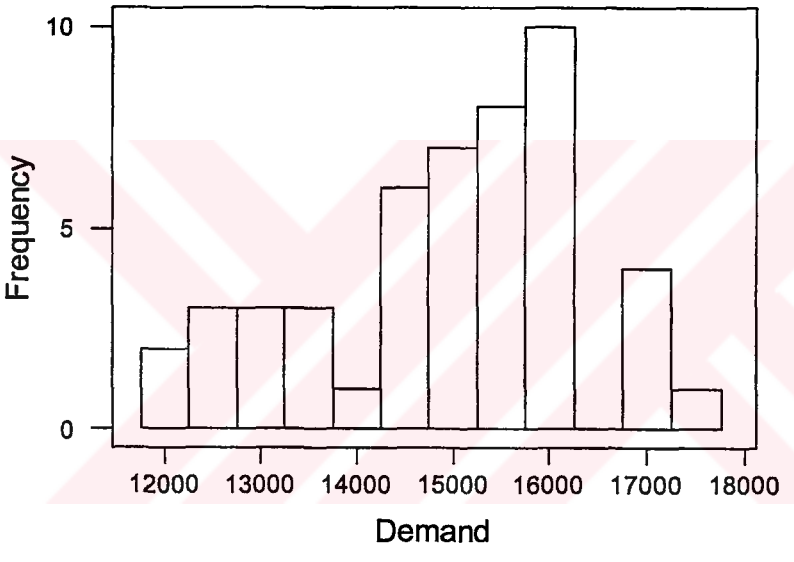

Figure 5.1. Histogram of Data

By using MINITAB, it is possible to show that the data are normally distributed in various ways. Two normality tests in MINITAB are applied to the data. For both of the tests, the null hypothesis is that the data are normal; the alternative hypothesis is that the data are not normal.

H_0: The data are normal

H_a: The data are not normal.

The normal probability plot for Anderson-Darling Normality Test is shown in Figure 5.2.

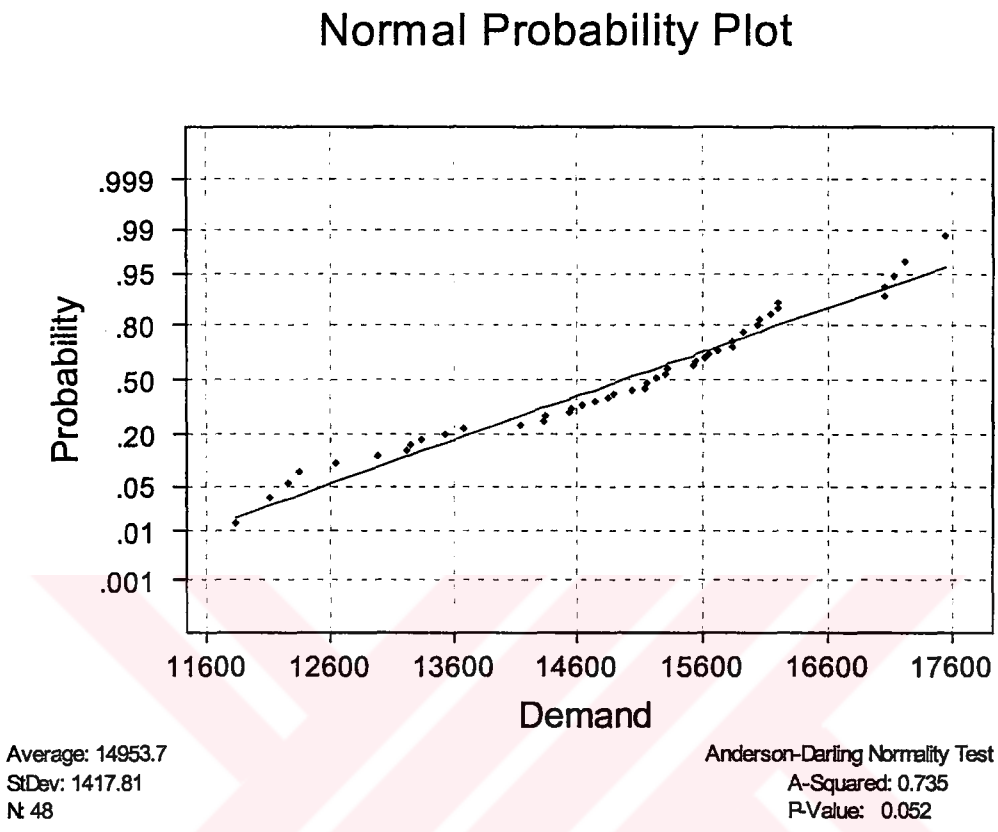

Figure 5.2. Normal Probability Plot for Anderson-Darling Normality Test

A straight or close to straight line indicates normality. A lot of curvature indicates non-normal data. The p-value, which is found as 0.052 says the null hypothesis, cannot be rejected. This means the data are normal at the significance level of 0.05.

The other test for normality that is applied to the data is Kolmogorov-Smirnov Test; which is a chi-square based test. Normal probability plot for Kolmogorov-Smirnov Test is shown in Figure 5.3.

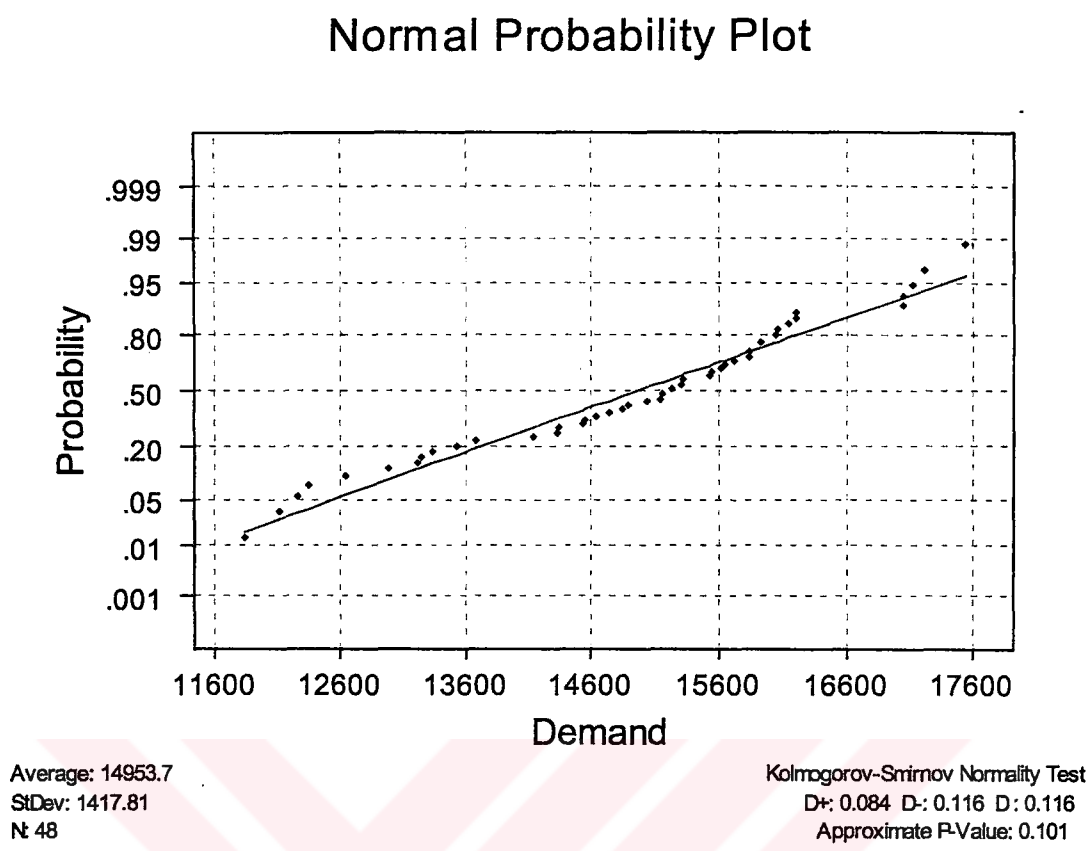

Figure 5.3. Normal Probability Plot for Kolmogorov-Smirnov Test.

If the α value is larger than the p-value displayed with graph, the hypothesis of normality is rejected. The p-value, which is found as 0.101 says the null hypothesis, cannot be rejected. This means the data are normal at the significance level of 0.05.

As a result, the data is normally distributed with mean 14954, and standard deviation 1418.

The company produces the items in batches. A batch contains 1500 units. Because it is considered the production is made in form of batches, the demand is also considered in form of batches.

Since the demands are normally distributed, the probabilities of demands can be calculated as follows:

The probability of demand is less than 10500 units is

$$P(x \leq 10500) = P(D = 7) = P(z \leq -3.14) = 0.0008$$

Since this probability is too small, it is ignored.

The probability of demand is between 10500 units and 12000 units, that is 8 batches is;

$$P(10500 \leq x \leq 12000) = P(D = 8) = P(-3.14 \leq z \leq -2.08) = 0.018$$

The probability of demand is between 12001 units and 13500 units, that is 9 batches is;

$$P(12000 \leq x \leq 13500) = P(D = 9) = P(-2.08 \leq z \leq -1.03) = 0.1327$$

The probability of demand is between 13501 units and 15000 units, that is 10 batches is;

$$P(13500 \leq x \leq 15000) = P(D = 10) = P(-1.03 \leq z \leq 0.03) = 0.3605$$

The probability of demand is between 15001 units and 16500 units, that is 11 batches is;

$$P(15000 \leq x \leq 16500) = P(D = 11) = P(0.03 \leq z \leq 1.09) = 0.3501$$

The probability of demand is between 16501 units and 18000 units, that is 12 batches is;

$$P(16500 \leq x \leq 18000) = P(D = 12) = P(1.09 \leq z \leq 2.15) = 0.1221$$

The probability of demand is greater than 18000 units that is 13 batches is

$$P(x \geq 18000) = P(D = 13) = P(z \geq 2.15) = 0.0158$$

These calculations are shown in the Table 5.2.

Table 5.2. Demands and probabilities

Demands	D_i	p_i
10500-12000	8	0.0180
12001-13500	9	0.1327
13501-15000	10	0.3605
15001-16500	11	0.3501
16501-18000	12	0.1221
18000-	13	0.0158

In the first column, there are demands in units. In the second column there are demands in batches. The frequencies and the probabilities are in the third and fourth columns respectively. These transition probabilities are given in Table 5.3. To find the mean batch of demand, expected value of D is calculated as follows:

$$E(D) = 0.0158 \times 13 + 0.1221 \times 12 + 0.3501 \times 11 + 0.3605 \times 10 + 0.1327 \times 9 + 0.018 \times 8$$

$$E(D) = 10.465$$

Table 5.3 Transition Probabilities

i	x	0	1	2	3	4	5	6	7	8	$c_x/1500$
0	13	0.0158	0.1221	0.3501	0.3605	0.1327	0.018				206.765
	14		0.0158	0.1221	0.3501	0.3605	0.1327	0.018			223.475
	15			0.0158	0.1221	0.3501	0.3605	0.1327	0.018		240.185
1	12	0.0158	0.1221	0.3501	0.3605	0.1327	0.018				191.055
	13		0.0158	0.1221	0.3501	0.3605	0.1327	0.018			207.765
	14			0.0158	0.1221	0.3501	0.3605	0.1327	0.018		224.475
	15				0.0158	0.1221	0.3501	0.3605	0.1327	0.018	241.185
2	11	0.0158	0.1221	0.3501	0.3605	0.1327	0.018				175.345
	12		0.0158	0.1221	0.3501	0.3605	0.1327	0.018			192.055
	13			0.0158	0.1221	0.3501	0.3605	0.1327	0.018		208.765
	14				0.0158	0.1221	0.3501	0.3605	0.1327	0.018	225.475
3	10	0.0158	0.1221	0.3501	0.3605	0.1327	0.018				159.635
	11		0.0158	0.1221	0.3501	0.3605	0.1327	0.018			176.345
	12			0.0158	0.1221	0.3501	0.3605	0.1327	0.018		193.055
	13				0.0158	0.1221	0.3501	0.3605	0.1327	0.018	209.765
4	9	0.0158	0.1221	0.3501	0.3605	0.1327	0.018				143.925
	10		0.0158	0.1221	0.3501	0.3605	0.1327	0.018			160.635
	11			0.0158	0.1221	0.3501	0.3605	0.1327	0.018		177.345
	12				0.0158	0.1221	0.3501	0.3605	0.1327	0.018	194.055
5	8	0.0158	0.1221	0.3501	0.3605	0.1327	0.018				128.215
	9		0.0158	0.1221	0.3501	0.3605	0.1327	0.018			144.925
	10			0.0158	0.1221	0.3501	0.3605	0.1327	0.018		161.635
	11				0.0158	0.1221	0.3501	0.3605	0.1327	0.018	178.345
6	7	0.0158	0.1221	0.3501	0.3605	0.1327	0.018				112.505
	8		0.0158	0.1221	0.3501	0.3605	0.1327	0.018			129.215
	9			0.0158	0.1221	0.3501	0.3605	0.1327	0.018		145.925
	10				0.0158	0.1221	0.3501	0.3605	0.1327	0.018	162.635
7	6	0.0158	0.1221	0.3501	0.3605	0.1327	0.018				96.795
	7		0.0158	0.1221	0.3501	0.3605	0.1327	0.018			113.505
	8			0.0158	0.1221	0.3501	0.3605	0.1327	0.018		130.215
	9				0.0158	0.1221	0.3501	0.3605	0.1327	0.018	146.925
8	5	0.0158	0.1221	0.3501	0.3605	0.1327	0.018				81.085
	6		0.0158	0.1221	0.3501	0.3605	0.1327	0.018			97.795
	7			0.0158	0.1221	0.3501	0.3605	0.1327	0.018		114.505
	8				0.0158	0.1221	0.3501	0.3605	0.1327	0.018	131.215

Define c_x as the expected cost in the current period from making decision x. The expected cost is the sum of the production cost and inventory holding cost. Producing a unit costs the firm 15.71 DM. Since a batch consists of 1500 units, producing a batch costs the firm 15.71×1500 DM. If cost of x batches is denoted by, $C(x)$, the cost function for a batch is

$$C(x) = 1500 \times 15.71x$$

The amount of inventory is found by summing the beginning inventory and production amount in that period and subtracting the mean of demand from this summation. Then the expected cost for the decision to produce x batches when entering inventory is i will be as follows:

$$c_x = C(x) + h(i + x - \mu)$$

Since the holding cost is 1DM for a unit the holding cost for a batch is 1500 DM. Since the mean batch demand is 10.465, the expected cost function will be

$$c_{ix} = C(x) + 1500(i + x - 10.465).$$

The expected costs for all i and x values are calculated and given Table 5.3.

As it is explained in chapter three, the sequential decision problems can be solved by dynamic programming technique. In the inventory problem there is a probability distribution for what the next stage will be. This means this problem characterized as an example of dynamic programming.

Dynamic programming has its own formulation. If we think the inventory problem in dynamic formulation; we obtain the following results:

State variable, which gives a description of the system, is the entering inventory for a stage. The amount of inventory at the beginning of a period determines the state of the period. So, state variable is the amount of the beginning inventory of a period and it can be 8 units maximum.

$$i = 0,1,...,8$$

Decision variables, It is to be determined production level at each period. Production x is made large enough so that a stock out never occurs; implying the restriction

$$\text{entering inventory} + \text{production} \geq 13$$

Because the production is limited 15 units;

$$x \leq 15$$

The summation of the inventory i, and the production amount x must be 16 at most. When this summation is larger than 16, for example 17 batches; the ending inventory will be 9 batches in the case of maximum demand. And this contradicts the company's wishes because it desires that no more than 9 batches should be stocked. This restriction can be expresses as

$$i + x \leq 16$$

Then the bounds for the production amount is

$$13 - i \leq x \leq \min(15,16 - i)$$

The appropriate dynamic programming functional equation is:

$$f(i) = \min_x \left\{ C(x) + h(i+x-\mu) + \alpha \sum_{i=1}^{6} P(D=d_i) f(i+x-d_i) \right\}$$

because holding cost is 1, and μ=10.465 the equation will be:

$$f(i) = \min_x \left\{ C(x) + (i+x-10.465) + \alpha \sum_{i=1}^{6} P(D=d_i) f(i+x-d_i) \right\} \text{ for } i = 0,1,...,8$$

where the minimization is over only nonnegative integer values in the range $13-i \le x \le \min(15,16-i)$.

Since the objective is minimization, the cost related with the stage n is called the stage cost and denoted by $f_n(i_n, x_n)$. Since the horizon is unbounded, the expected present value of an optimal policy is represented by $f(i)$.

The analysis of the dynamic programming models where the number of states and decisions is finite, the horizon is unbounded, and the transition probabilities are stationary, can be done by solving the value determination equations as it is described in the fourth chapter.

The solution can be obtained by two algorithms:

(1) Value Iteration Algorithm
(2) Policy Iteration Algorithm.

Same equations are used for both algorithms.

5.3.1. Value Iteration Algorithm

The value iteration algorithm is

$$y_i^{n+1} = \min_x \left[\sum_{allj} p(j \mid i,x) \alpha y_j^n + c_x \right] \quad \text{for each } i \text{ (value iteration)} \quad (5.1)$$

The value of y_i represents the minimum expected present value, using an optimal strategy over an unbounded time horizon. Each y_i^n always will converge to a limit. If all $c_{ix} \geq 0$ and all y_i^0=0, the y_i^n increase monotonically. By starting with every y_i^0=0, the process is the same as looking at a certain finite horizon model. y_i^n represents the expected present value of an optimal strategy.

The coefficient α provides to obtain the present value and should be $0 \leq \alpha < 1$. When *i%* is the interest rate per period, $\alpha = \left[1 + \frac{i}{100}\right]^{-1}$ is called the single-period *discount factor*. The higher the interest rate *i*, the smaller value of α. For the inventory example, the solution is obtained for two different α-values; 0.8, and 0.9.

For any c_x, the convergence from (5.1) can be obtained as follows: Select a trial stationary policy, and denote each of the decisions by *x*. Then solve the set of simultaneous linear equations

$$y_i - \sum_{allj} p(j \mid i,x) \alpha y_j = c_x \qquad \text{for each } i \qquad (5.2)$$

for the initial values $y_i^0 = y_i$ to be used in (5.1).

For inventory model, all possible values for all *i* values and the related decision probabilities are given in Table 5.3. The solution is obtained assuming α=0.8 with value iteration. The results are in Table 5.4.

We started by selecting an arbitrary policy. A policy chosen which gives the minimum cost. The values of y_i^n are determined starting with y_i^0=0. Thirty iterations are performed. It can be easily seen that y_i^n increases monotonically. After numbers of iterations, y_i^n converges to the values, which are given in the last column of Table 5.5.

The optimum policy is to produce to meet the maximum demand. The optimum policy says that, the entering inventory and production amount must meet the maximum demand.

The y_i values for unbounded horizon can be found by solving the following set of equations:

$$y_0 - 0.0158\alpha y_0 - 0.1221\alpha y_1 - 0.3501\alpha y_2 - 0.3605\alpha y_3 - 0.1327\alpha y_4 - 0.018\alpha y_5 = 206.7$$

$$y_1 - 0.0158\alpha y_0 - 0.1221\alpha y_1 - 0.3501\alpha y_2 - 0.3605\alpha y_3 - 0.1327\alpha y_4 - 0.018\alpha y_5 = 191.05$$

$$y_2 - 0.0158\alpha y_0 - 0.1221\alpha y_1 - 0.3501\alpha y_2 - 0.3605\alpha y_3 - 0.1327\alpha y_4 - 0.018\alpha y_5 = 175.3$$

$$y_3 - 0.0158\alpha y_0 - 0.1221\alpha y_1 - 0.3501\alpha y_2 - 0.3605\alpha y_3 - 0.1327\alpha y_4 - 0.018\alpha y_5 = 159.6$$

$$y_4 - 0.0158\alpha y_0 - 0.1221\alpha y_1 - 0.3501\alpha y_2 - 0.3605\alpha y_3 - 0.1327\alpha y_4 - 0.018\alpha y_5 = 143.9$$

$$y_5 - 0.0158\alpha y_0 - 0.1221\alpha y_1 - 0.3501\alpha y_2 - 0.3605\alpha y_3 - 0.1327\alpha y_4 - 0.018\alpha y_5 = 128.2$$

$$y_6 - 0.0158\alpha y_0 - 0.1221\alpha y_1 - 0.3501\alpha y_2 - 0.3605\alpha y_3 - 0.1327\alpha y_4 - 0.018\alpha y_5 = 112.5$$

$$y_7 - 0.0158\alpha y_0 - 0.1221\alpha y_1 - 0.3501\alpha y_2 - 0.3605\alpha y_3 - 0.1327\alpha y_4 - 0.018\alpha y_5 = 96.79$$

$$y_8 - 0.0158\alpha y_0 - 0.1221\alpha y_1 - 0.3501\alpha y_2 - 0.3605\alpha y_3 - 0.1327\alpha y_4 - 0.018\alpha y_5 = 81.08$$

As a result, the production amount should be the difference of the maximum demand and the entering inventory. In other words;

$$\text{produce } x=\{13-i \qquad i=0,1,\ldots,8$$

The optimal policy yields the following costs for each inventory.

y_0=872.387

y_1=856.677

y_2=840.967

y_3=825.257

y_4=809.547

y_5=793.837

y_6=778.127

y_7=762.417

y_8=746.707

Table 5.4. Value Iteration Approximation for Inventory Problem when $\alpha=0.8$

Entering inventory	*n = 1*		*n = 2*		*n = 3*		*n=5*	*n=10*	*n=20*	*n=30*	*Unbounded Horizon*	
i	$x^1(i)$	y_i^1	$x^2(i)$	y_i^2	$x^3(i)$	y_i^3	y_i^5	y_i^{10}	y_i^{20}	y_i^{30}	$x_\infty(i)$	y_i
0	13	206.765	13	340.315	13	447.070	586.788	779.176	862.459	871.330	**13**	**872.387**
1	12	191.055	12	324.605	12	431.360	571.078	763.466	846.749	855.620	**12**	**856.677**
2	11	175.345	11	308.895	11	415.650	555.368	747.756	831.039	839.910	**11**	**840.967**
3	10	159.635	10	293.185	10	339.940	539.658	732.046	815.329	824.200	**10**	**825.257**
4	9	143.925	9	277.475	9	384.230	523.948	716.336	799.619	808.490	**9**	**809.547**
5	8	128.215	8	261.765	8	368.520	508.238	700.626	783.909	792.780	**8**	**793.837**
6	7	112.505	7	246.055	7	352.810	492.528	684.916	768.199	777.070	**7**	**778.127**
7	6	96.795	6	230.345	6	337.100	476.818	669.206	752.489	761.360	**6**	**762.417**
8	5	81.085	5	214.635	5	321.390	461.108	653.496	736.779	745.650	**5**	**746.707**

5.3.2. Policy Iteration Algorithm

The same results can be obtained by using Policy Iteration. The inventory problem is solved also by using the Policy Iteration Algorithm. Firstly, the procedure is given step by step.

Step 1: Select an arbitrary initial policy, and let $n = 0$.

Step 2: Given the trial policy, calculate the y_i^n according to the value determination equations:

$$y_i^n - \sum_{allj} p(j \mid i,x) \alpha y^n{}_j = c_{ix} \text{ for each } i \qquad \text{(value determination routine)} \qquad (5.2)$$

where x represents the decision at state i.

Step 3: Test for a policy improvement by calculating

$$\min_x \left[\sum_{allj} p(j \mid i,x) \alpha y_j^n + c_{ix} \right] \equiv Y_i^n \text{ for each } i \quad \text{(test quantity)} \qquad (5.3)$$

Step 4: Terminate the iterations if $Y_i^n = y_i^n$ for all i. Otherwise, revise the policy at each state k where $Y_k^n < y_k^n$, using a decision yielding Y_k^n in (5.3). Increase n to $n+1$, and return to *step 2* with the new trial policy.

The policy iteration algorithm is applied to the inventory problem and it is explained step by step as follows:

Step 1: An arbitrary policy is selected. It is possible to start with the policy, which gives the minimum costs for each i; but in that case the algorithm ends in the first iteration. Since the policy iteration algorithm is going to be explained, it is not preferred the algorithm ends in the first iteration. Because of this fact, an arbitrary

policy is selected. The arbitrarily selected values are given in the second column of Table 5.5.

Step 2: Given the trial policy, for all x and i values, equation (5.2) is obtained, and from these set of equations, y_i^0 values are calculated. These values are given in the third column of the Table 5.5.

Step 3: For all i values, Y_i^n are calculated for $n=0$ as follows:

Y_0^0=897.322

Y_1^0=881.612

Y_2^0=865.902

Y_3^0=850.192

Y_4^0=834.482

Y_5^0=818.772

Y_6^0=803.062

Y_7^0=787.352

Y_8^0=771.642

Step 4: y_i^0 values are compared to the Y_i^0 values. If $Y_i^0 = y_i^0$ for all i, optimum policy is obtained. Otherwise the policy is revised. If $Y_k^0 < y_k^0$, the x-value, which yields Y_k^n, is included in the new policy. In other words, Y_i^0 values determines the new policy that should be chosen. The comparisons and the new policies for all i are as follows:

y_0^0=897.323, Y_0^0=897.322 $\Rightarrow$ $x^1(0)=13$

y_1^0=885.817, Y_1^0=881.612 $\Rightarrow$ $x^1(1)=12$

y_2^0=870.107, Y_2^0=865.902 $\Rightarrow$ $x^1(2)=11$

y_3^0=859.618, Y_3^0=850.192 $\Rightarrow$ $x^1(3)=10$

y_4^0=841.041, Y_4^0=834.482 $\Rightarrow$ $x^1(4)=9$

y_5^0=818.773, Y_5^0=818.772 $\Rightarrow$ $x^1(5)=8$

y_6^0=807.267, Y_6^0=803.062 $\Rightarrow$ $x^1(6)=7$

y_7^0=787.353, Y_7^0=787.352 $\Rightarrow$ $x^1(7)=6$

y_8^0=771.643, Y_8^0=771.642 $\Rightarrow$ $x^1(8)=5$

Then we return to *step 2* and obtain the y_i^1 values, which are given in the last column of the Table 5.6. To test the y_i^1 values are optimal or not, Y_i^1 values are calculated as follows:

Y_0^1=872.387

Y_1^1=856.677

Y_2^1=840.967

Y_3^1=825.257

Y_4^1=809.547

Y_5^1=793.837

Y_6^1=778.127

Y_7^1=762.417

Y_8^1=746.707

These Y_i^1 values equal to y_i^1 values. This means optimum solution is obtained. The results are given in Table 5.5.

Table 5.5. Optimum Solution for α=0.8

Entering Inventory	*n = 0*		*n = 1*	
i	$x^0(i)$	y_i^0	$x^1(i)$	y_i^1
0	13	897.323	13	872.387
1	13	885.817	12	856.677
2	12	870.107	11	840.967
3	13	859.618	10	825.257
4	11	841.041	9	809.547
5	8	818.773	8	793.837
6	8	807.267	7	778.127
7	6	787.353	6	762.417
8	5	771.643	5	746.707

The solution of the same problem is searched with policy iteration assuming α=0.9 and following results are obtained.

Table 5.6. Optimum Solution for α=0.9

Entering Inventory	*n = 0*		*n = 1*	
i	$x^0(i)$	y_i^0	$x^1(i)$	y_i^1
0	14	1728.42	13	1703.08
1	12	1710.43	12	1687.37
2	13	1700.14	11	1671.66
3	11	1681.29	10	1655.95
4	10	1665.58	9	1640.24
5	9	1649	8	1624.53
6	10	1641.66	7	1608.82
7	8	1621.59	6	1593.11
8	5	1600.46	5	1577.4

The comparison between optimal solutions with different α-values are given in Table 5.8.As it is mentioned before, α is called as discount factor, and it is calculated according to the interest rate *i*. The higher the interest rate *i* the smaller the value of α. Firstly the present value is calculated with interest rate 25%; this means α=0.8. Secondly, the calculation is performed according to the interest rate 10%. In this case, α=0.9. The results are given in Table 5.7.

The optimal policy for both cases is the same. Although holding cost is low, optimal policy suggests that production quantity should be determined as the amount to satisfy the maximum possible demand; that is there is no need to produce for stock.

Table 5.7. Comparison Between Optimal Solutions

Entering Inventory	$\alpha = 0.8$		$\alpha = 0.9$	
i	$x_\infty(i)$	y_i	$x_\infty(i)$	y_i
0	13	872.387	13	1703.08
1	12	856.677	12	1687.37
2	11	840.967	11	1671.66
3	10	825.257	10	1655.95
4	9	809.547	9	1640.24
5	8	793.837	8	1624.53
6	7	778.127	7	1608.82
7	6	762.417	6	1593.11
8	5	746.707	5	1577.4

CHAPTER SIX
CONCLUSION

6.1. Conclusions

In this study, an inventory problem is formulated as a Markov decision process and solved by the algorithms of infinite horizon dynamic programming technique. The solution of the problem is obtained for two discount factor; that is two interest rate. When the interest rate is *i=10%*, from the equation $\alpha = \left[1+\frac{i}{100}\right]^{-1}$; the discount factor is found as $\alpha = 0.9$. When the interest rate is *i=25%*, from the same equation, the discount factor is found as $\alpha = 0.8$. In this case, the holding cost will be $h = 15.71 \times 0.25 = 3.93$. The optimal policy is obtained for two discount factor in this study. When the holding cost is less than 3.93 DM, it is expected that the production will be in a way that it allows holding inventory. Contrary to this expectation, the results not suggest working with inventory as an optimal solution. The reason is that the set up cost is accepted zero. Since the plat is too large, the production is made in a large variety of scale and the production is made without stopping, there is no set up cost. For this reason, even the holding cost is cheap, the production is made continuously, and the optimal policy is determined in a way that there is no holding inventory.

Determining the optimal policy in such a way is because not only the set up cost is zero, but also the storage capacity is 8, maximum possible demand is 13. This

means the production should be all the time. Therefore, determining the production amount greater that *(13-i)* causes the holding cost to increase.

Consequently, with this work and the model we constructed, we indicate that the solution, which is intuitively optimal, is also technically optimal.

REFERENCES

Allen A. O. (1990). Probability, Statistics and queuing Theory with Computer Science Applications. Academic Press.

Bierman, Bonini&Hausman. (1981). Quantitive Analysis for Business Decisions.(6th ed.). Ontario. Richard Irwin Inc.

Birge J.R., Louveaux F.. (1997). Introduction to Stochastic Programming. NewYork. Springer-Verlag.

Bailey, Norman J.. (1990). The Elements of Stochastic Processes. USA. John Wiley & Sons.

Cinlar E. (1975). Introduction to Stochastic Processes. Englewood Cliffs (USA). Prentice-Hall.

DeGroot M.H.. (1970). Optimal Statistical Decisions. USA. McGraw Hill.

Filar J., Vrieze K. (1997). Competitive Markov Decision Process. NewYork. Spring-Verlag.

Hiller F.S. & Lieberman G.J. (1990). Introduction to Stochastic Models in Operations Research. USA. McGraw Hill.

Karlin S., Taylor H.M..(1975). A first Course in Stochastic Processes. Academic Press.

Kotz S., Jhonson N. L. (1988). Encyclopedia of Statistical Sciences. New York. John Wiley & Sons.

Mitchell G..(1993). The Practice of Operational Research. Chichester. John Wiley & Sons.

Nemhauser G..(1967). Introduction to Dynamic Programming. John Wiley & Sons.

Ochi M.K. (1990). Applied Probability & Stochastic Processes-In Engineering & Physical Sciences. Canada. John Wiley & Sons.

Parzen E. (1969). Stochastic Processes. Holden-Day.

Puterman M. L. (1994). Markov Decision Processes. Canada. John Wiley and Sons.

Ross S. (1983). Stochastic Processes. Canada. John Wiley & Sons.

Sahinoglu M. (1992). Applied Stochastic Process. Ankara. ODTU.

Takacs L.. (1960). Stochastic Processes. New York. John Wiley & Sons.

Wagner H M.. (1969). Principles of Operations Research With Applications to Managerial Decisions. Prentice-Hall.

White D.J.. (1993). Markov Decision Processes. Chicherter. John Wiley & Sons.

Yates R. D., Goodman D. J.. (1998). Probability and Stochastic Processes-A Friendly Introduction for Electrical and Computer Engineers. John Wiley & Sons.