ATTAINING OPTIMAL INVESTMENT ALLOCATION USING DISCRETE DYNAMIC PROGRAMMING TECHNIQUE.

BY

ISAAC VANDERPUIJE (Bsc. MECHANICAL ENGINEERING)

PG6325711

A THESIS SUBMITTED TO THE DEPARTMENT OF MATHEMATICS, KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY IN PARTIAL FULFILLMENT FOR THE DEGREE OF

MASTER OF SCIENCE Industrial mathematics

College of Science / Institute of Distance Learning

JULY, 2015

DECLARATION

I hereby declare that this submission is my own work towards the MSc and that, to the best of my knowledge, it contains no material previously published by another person nor material which has been accepted for the award of any other degree of the University, except where due acknowledgement has been made in the text.

ISAAC VANDERPUIJE

PG6325711

------------------------------	--------------------------	-------------------
Student Name & ID	Signature	Date

Certify by:

PROF. S. K AMPONSAH

-----------------------------	--------------------------	--------------------
Supervisor(s) Name	Signature	Date

Certified by:

PROF. S. K AMPONSAH

---------------------------	------------------------	--------------------
Head of Dept. Name	Signature	Date

Certified by:

PROF. I.K DONTWI

-----------------------------	-------------------------------	--------------------
Dean, IDL	Signature	Date

DEDICATION

This work is dedicated to my mum, the late Mrs. Regina N. O. Vanderpuije and my dad, Mr.Nathaniel N.O Vanderpuije.

ACKNOWLEGEMENTS

I am grateful to God Almighty for the successful completion of this study.

My heartfelt gratitude goes to Professor S.K Amponsah for supervising this work and prompting me correct my mistakes.

My final appreciation goes to the fixed deposit desk officials of Ghana Commercial Bank, Fidelity Bank and Barclays Bank of Ghana limited who provided me with data for the study.

ABSTRACT

Everyone uses money. We all want it, work for it and think about it. Money has been one of the strongest supporting pillars in the lives of humans. People usually have the desire to earn more money in order to achieve set goals in their lives, but the decision on how, where and how long to invest for maximum profit becomes a problem to some . This study examines which of the three types of fixed deposits on sale from three financial institutions ; Barclays Bank Of Ghana limited, Ghana Commercial Bank and Fidelity Bank Limited at defined rates and periods when purchased yields the best returns. The method employed was discrete dynamic programming technique. Results of the study indicated that investment at Fidelity Bank Limited yields more returns comparable to Barclays Bank of Ghana Limited and Ghana Commercial Bank.

CONTENTS

DECLARATION....................ii
DEDICATION....................iii
ACKNOWLEGEMENTS....................iii
ABSTRACT....................iv

CHAPTER ONE ....
1 INTRODUCTION
.. 1

1.1 BACKGROUND OF STUDY 1

1.2 INVESTMENT 1

1.3 STATEMENT OF THE PROBLEM 3

1.4 OBJECTIVES OF THE STUDY 4

1.5 METHODOLOGY... 4

1.6 JUSTIFICATION OF THE STUDY 5

1.7 LIMITATIONS OF THE STUDY 5

1.8 ORGANIZATION OF THE STUDY 5

CHAPTER TWO ....
7 LITERATURE REVIEW
... 7

2.1 INTRODUCTION 7

2.2 LITERATURE REVIEW 7

CHAPTER THREE ....
18 METHODOLOGY
.. 18

3.0 INTRODUCTION 18

3.1 DYNAMIC PROGRAMMING 18

3.2 DYNAMIC PROGRAMMING IN MATHEMATICAL OPTIMIZATION 20

3.2.1 DYNAMIC PROGRAMMING IN COMPUTER PROGRAMMING 20

3.2.2 CHARACTERISTICS OF DYNAMIC PROGRAMMING APPLICATIONS 22

3.2.3 COMPUTATIONAL EFFICIENCY OF DYNAMIC PROGRAMMING 23

3.2.4 DETERMINISTIC VERSUS STOCHASTIC DYNAMIC PROGRAMMING 24

3.3 INTEGER PROGRAMMING 25

3.4 APPLICATIONS OF DYNAMIC PROGRAMMING 25

3.4.1 OPTIMAL CONSUMPTION AND SAVING PROBLEMS 25

3.5 DIJKSTRA'S ALGORITHM FOR THE SHORTEST PATH PROBLEM 27
3.6 THE KNAPSACK PROBLEM 32
3.7 EQUIPMENT REPLACEMENT PROBLEMS 33
3.8 AN INVENTORY PROBLEM 36
3.9 RESOURCE ALLOCATION PROBLEMS 36

CHAPTER FOUR .... 38 DATA COLLECTION AND ANALYSIS 38
4.0 INTRODUCTION 38
4.1 DATA COLLECTION 38
4.2 ANALYSIS 39

CHAPTER FIVE... 64 SUMMARY OF FINDINGS, CONCLUSIONS AND RECOMMENDATIONS 64
5.0 INTRODUCTION 64
5.1 SUMMARY OF FINDINGS... 64
5.2 CONCLUSIONS 64
5.3 RECOMMENDATIONS 65
REFERENCES.. 66

LIST OF TABLES

Table 3.1: Road Network Calculations 30
Table 3.1.1: Stage 2 Of Road Network Calculations 30
Table 3.1.2: Stage 1 Of Road Network Calculations 30
Table 3.2: Knapsack Problem 32
Table 3.3.1: Stage 5 Of Equipment Replacement Problem 35
Table 3.3.2: Stage 4 Of Equipment Replacement Problem 35
Table 3.3.3: Stage 3 Of Equipment Replacement Problem 35
Table 3.3.4: Stage 2 Of Equipment Replacement Problem 35

Table 3.3.5: Stage 1 Of Equipment Replacement Problem 35

Table 4.1: Interest Rates -Barclays Bank 38

Table 4.2: Interest Rates -Ghana Commercial Bank 39

Table 4.3: Interest Rates- Fidelity Bank 39

Table 4.4: Annual Returns From The Various Periods 40

Table 4.5: Returns Of Investment. 42

Table 4.6: Optimal Allocation Of Ghc 600 63

LIST OF FIGURES

FIGURE 3.1: ROAD NETWORK 1 28

CHAPTER ONE

INTRODUCTION

1.1 BACKGROUND OF STUDY

Human beings since creation have sought to look for avenues to increase their wealth. Investment has become a vehicle that is usually used to improve wealth. It has been used by man to move from the cave to skyscrapers. Individuals, entities, organizations and cooperate bodies make investments towards the future. One difficulty faced by all is where, how, when and for how long to invest, in the mist of all opportunities, to obtain the maximum satisfaction from the investment made. Usually investors might have a certain amount of money to invest. The investor may have various options of investment based on the returns. Any time an investor makes an investment he must decide on the optimal investment strategy. Two very important strategies are active portfolio management and long term investing. The strategies that one chooses for an optimal strategy will depend on the investors investment goals.

1.2 INVESTMENT

Investment has been defined by many and in many fields. It is usually an act of forgoing consumption. In a purely agrarian society, early humans had to choose how much grain to eat after the harvest and how much to save for future planting. That was an act of investment. Investing is the act of investing; that is laying out money or capital in an enterprise with the expectation of profit. Money is invested with an expectation of profit. Investment is the commitment of something other than money (time, energy, or effort) to a project with the expectation of some worthwhile result. Investment is the commitment of money or capital to purchase financial instruments or other assets in order to gain profitable returns in the form of interest, income, or appreciation of the value of the instrument. It is related to saving or deferring consumption. An investment involves the choice by an individual or an organization such as a pension fund, after some analysis or thought, to place or lend money in a vehicle, instrument or asset, such as property, commodity, stock, bond, financial derivatives (e.g. futures or options), or the foreign asset denominated in foreign currency, that has certain level of risk and provides the possibility of generating returns over a period of time. When an asset is bought or a given amount of money is invested in the bank, there is anticipation that some return will be received from the investment in the future.

Investment is a term frequently used in the fields of economics, business management and finance. It can mean savings alone, or savings made through delayed consumption. Investment can be divided into different types according to various theories and principles. According to economic theories, investment is defined as the per-unit production of goods, which have not been consumed, but will however, be used for the purpose of future production. Examples of this type of investments are tangible goods like construction of a factory or bridge and intangible goods like six (6) months of on-the-job training. In terms of national production and income, Gross Domestic Product (GDP) has an essential constituent, known as gross investment.

According to business management theories, investment refers to tangible assets like machinery and equipments and buildings and intangible assets like copyrights or patents and goodwill. The decision for investment is also known as capital budgeting decision, which is regarded as one of the key decision. In finance, investment refers to the purchasing of securities or other financial assets from the capital market. It also means buying money market or real properties with high market liquidity. Some examples are gold, silver, real properties, and precious items. Financial investments are in stocks, bonds, and other types of security investments. Indirect financial investments can also be done with the help of mediators or third parties, such as pension funds, mutual funds, commercial banks, and insurance companies. According to personal finance theories, an investment is the implementation of money for buying shares, mutual funds or assets with capital risk. Usually a combination of any of the above investment possibilities may be considered by an investor or an individual. Any time an investor wishes to make an investment with a certain sum of money, say, he must decide on the optimal investment strategy to adopt. The optimal investment strategy could be long term or short term, active portfolio management or long term investing. The choice of a particular investment should be based on the cumulative returns on all the investments.

1.3 STATEMENT OF THE PROBLEM

Suppose an investor or a customer wants to invest in fixed deposits. The investor may be faced with the problem of which of the financial institutions and which period of fixed deposits should be made. Usually there are several opportunities available as a result of which the problem of choice/allocation arises. It would be realized that each opportunity requires deposits in financial terms and an expected return. The investor may allocate all the money to just one opportunity or split the money between the alternatives of investments all with the aim of obtaining the optimal returns from the investment made.

Let $f_i(x),\ i = 1, 2, 3, \dots, n$ denote the return from investment i when x units of money are invested. We define $x(x = 1, 2, 3, \dots, n)$ as the amount of money invested in investment i.

The problem of determining how much to invest in each investment in order to maximize total returns can be approached through a multi-stage decision process by modeling a Mathematical program to find the optimal policy using Dynamic Programming.

Maximize $\sum f_i(x)$

Subject to $\sum x \leq b$

$x \geq 0$

$i = 1, 2, 3, \dots, n$

It must be noted that $f_i(x)$ are functions of a single variable, b is a known nonnegative integer.

1.4 OBJECTIVES OF THE STUDY

The objectives of this thesis are:

(i) to identify the various types of fixed deposits and their annual returns.

(ii) to use dynamic programming to obtain an optimal solution.

(iii) to identify where to invest and how much to invest at any point in time.

1.5 METHODOLOGY

The mathematical technique of discrete Dynamic Programming was used in the study. A secondary data consisting of Interest rates of fixed deposits for various periods; 91-day fixed deposit, 182-day fixed deposit and 1 year fixed deposits, in three financial institutions; Barclays Bank, Ghana Commercial Bank and Fidelity Bank was collected and used. Dynamic Programming was applied to determine the optimal investment and the appropriate investment allocations to be made.

1.6 JUSTIFICATION OF THE STUDY

Many are those who have resources and would like to invest, but are not sure of where, when and how to put their resources in order to accrue the maximum returns. To justify the products in which to invest, we need to look out for the various forms of investments available, the expected returns from each investment and the associated cost. Financial institutions would like to know where to keep their excess cash flows to make the maximum returns. All the above can be modeled as dynamic programming problem. It is known that dynamic programming solves problems in stages and is quicker and less time consuming far less than total enumeration

1.7 LIMITATIONS OF THE STUDY

The problem to be considered in this study is the Bellman's Principle of Optimality using Dynamic programming. I considered only three banks and three fixed deposit periods with five groups of values. I selected two figures at random from each group to arrive at a nine by eleven matrix. The selections were made due to the researcher's easy access to the information/data.

1.8 ORGANIZATION OF THE STUDY

The study is organized into five chapters.

Chapter One is the introduction of investment in general, background, problem Statement, objectives, methodology , justification and limitations of the study. In Chapter Two, there is a review of related literature on Dynamic programming applications and its variants. Chapter three outlines some algorithms solution of Shortest Path problems, Knapsack Problems, Equipment Replacement problems using total enumeration and Dynamic Programming. It considers cases where there is total enumeration and compares the time and stages used in solving a problem. Chapter Four deals with the analysis and interpretation of the data. Data on Investment made and the returns were collected from three financial institutions and Dynamic programming used to determine the optimal investment returns and the corresponding investments to be made. Chapter Five is the concluding part of the study It draws the Summary of the findings, gives the appropriate conclusions and recommendations of the study.

CHAPTER TWO LITERATURE REVIEW 2.1 INTRODUCTION

Dynamic programming is both a mathematical optimization method, and a computer programming method. In both contexts, it refers to simplifying a complicated problem by breaking it down into simpler sub problems in a recursive manner. While some decision problems cannot be taken apart this way, decisions that span several points in time do often break apart recursively; Bellman called this the ―Principle of Optimality‖.

If sub problems can be nested recursively inside larger problems, so that dynamic programming methods are applicable, then there is a relation between the value of the larger problem and the values of the sub problem. In the optimization literature this relationship is called the Bellman equation.

Dynamic programming is a widely used programming technique in bioinformatics. In sharp contrast to the simplicity of textbook examples, implementing a dynamic programming algorithm for a novel and non-trivial application is a tedious and error prone task. The algebraic dynamic programming approach seeks to alleviate this situation by clearly separating the dynamic programming recurrences and scoring schemes.

2.2 LITERATURE REVIEW

Based on the programming style, Steffen et al. (2005) introduced a generic product operation of scoring schemes. This lead to a remarkable variety of applications, allowing us to achieve optimizations under multiple objective functions, alternative solutions and back tracing, holistic search space analysis, ambiguity checking, and more, without additional programming effort. The authors demonstrated the method on several applications for Ribo Nucleic Acid (RNA) secondary structure prediction. The product operation as introduced here, adds a significant amount of flexibility to dynamic programming. It provides a versatile text bed for the development of new algorithmic ideas which can immediately be put to practice.

Institutional fund managers generally rebalance using adhoc methods such as calendar basis or tolerance band triggers. Sun et al. (2005) proposed a different framework that quantifies the cost of a rebalancing strategy in terms of risk-adjusted returns net of transaction costs. The authors then developed an optimal rebalancing strategy that actively seeks to minimize that cost. They used certainty equivalents and the transaction costs associated with a policy to define a cost-to-go function, and they minimized this expected cost-to-go using dynamic programming. The authors applied Monte Carlo simulations to demonstrate that their method outperforms traditional rebalancing. They also showed the robustness of our method to model error by performing sensitivity analyses.

The existence of an optimum and dynamic programming techniques was derived from abstract assumptions based on primitive utility function U and its W and M primitive aggregators. A non-positive-valued utility function U that is derived from a W dynamic aggregator and an M stochastic aggregator was constructed. The resulting examples exhibit mean growth without the distribution of unbounded support due to the few growth restrictions of non-positive objective.

Jacobson's, (2003) differential dynamic programming is a technique, based on dynamic programming rather than the calculus of variations, for determining the optimal control function of a nonlinear system. Unlike conventional dynamic programming where the optimal cost function is considered globally, differential dynamic programming applies the principle of optimality in the neighbourhood of a nominal, possibly no optimal, trajectory. This allowed the coefficients of a linear or quadratic expansion of the cost function to be computed in reverse time along the trajectory: these coefficients may then be used to yield a new improved trajectory (i.e. the algorithms are of the ―successive sweep‖ type). A class of nonlinear control problems, linear in the control variables, is studied using differential dynamic programming. It is shown that for the free-end-point problem, the first partial derivatives of the optimal cost function are continuous throughout the state space, and the second partial derivatives experience jumps at switch points of the control function. A control problem that has an analytic solution is used to illustrate these points. The fixed-end-point problem is converted into an equivalent free-end-point problem by adjoining the end-point constraints to the cost functional using Lagrange multipliers: a useful interpretation for Pontryagin's adjoint variables for this type of problem emerges from this treatment. The above results are used to devise new second- and first-order algorithms for determining the optimal bang-bang control by successively improving a nominal guessed control function. The usefulness of the proposed algorithms is illustrated by the computation of a number of control problem examples.

Dynamic programming solutions for optimal portfolios in which the solution for the portfolio vector of risky assets is constant were solved by Merton in continuous time and by Hakansson and others in discrete time. There is no case with a closed form solution where this vector of risky asset holdings changes dynamically. Tenney (1995) derived such solutions for the first time, and is thus a dynamic dynamic-programming solution as opposed to a static dynamic-programming solution for this vector. The solution is valid when there is a set of basis assets whose excess expected return is linear in the state vector, whose variance-covariance matrix is time-dependent and for which the interest rate is a quadratic function of the state vector.

Guangliang et al. (1999) solved the problem of constructing an optimal portfolio consisting of many risky assets to maximize the long-term growth rate of a representative agent‘s expected utility, subject to a set of general linear constraints on the portfolio weight vector as well as a constraint to prevent wealth drawdown‘s below a dynamic floor. The dynamic floor is defined as the time-decayed historical all-time high. Our results generalize those achieved by earlier authors, including Grossman and Zhou (1993) and Cvitannic and Karatzas (1994). Grossman and Zhou solved a special case of the authors problem by focusing on a single risky assets without portfolio weight constraints. Cvitannic and Karatzas solve a problem involving many risky assets but that ignored portfolio weight constraints and the time decay on the dynamic floor. To illustrate the usefulness of our method, the authors presented several numerical examples based on both actual and simulated (Monte Carlo) returns. Finally, they suggested applications of our results to various practical investment management problems, including the management of hedge fund portfolios and _principalprotected‘ investment strategies.

Herman D et al. (2009) developed a multiperiod investment portfolio model that includes risky farmland, risky and risk-free nonfarm assets, and debt financing on farmland in the presence of transaction costs and credit constraints. The model is formulated as a stochastic continuous-state dynamic programming problem, and is solved numerically for Southwestern Minnesota, USA. Result show that optimal investment decisions are dynamic and take into account the future decisions due to uncertainty, partial irreversibility, and the option to wait. The optimal policy includes ranges of inaction, states where the optimal policy in the current year is to wait. The risk-averse farmer makes a lower investment in risky farmland reflecting risk-avoiding behaviour. The authors found that, in addition to risk aversion, the length of the planning horizon affects risk-avoiding behavior in investment decisions. Finally, they found that higher debt financing on farmland is optimal when risky nonfarm assets can be included in the optimal investment portfolio and that the probability of exiting farming increases with the risky nonfarm investment.

Ghezzi (1997) considered an immunization problem, in which a bond portfolio is to be periodically rebalanced. Max-min optimal control is applied to the problem. The target is to maximize the final portfolio value under the worst possible evolution of interest rates. The optimal control law, obtained by means of dynamic programming, turns out to be different from any duration-based immunization policy.

Vila et al. (1991) used stochastic dynamic programming to study the inter-temporal consumption and portfolio choice of an infinitely lived agent who faces a constant opportunity set and a borrowing constraint. The authors showed that, under general assumptions on the agent‘s utility function, optimal policies exist and can be expressed as feedback functions of current wealth. They described these policies in detail, when the agent‘s utility function exhibits constant relative risk aversion.

Optimal asset allocation deals with how to divide the investor‘s wealth across some assetclasses in order to maximize the investor‘s gain. Pola et al. (2006) considered the optimal asset allocation in a multi-period investment setting: optimal dynamic asset allocation provides the (optimal) re-balancing policy to accomplish some investment‘s criteria. Given a sequence of target sets, which represent the portfolio specifications at each re-balancing time, an optimal portfolio allocation is synthesized for maximizing the joint probability for the portfolio to fulfill the target sets requirements. The approach pursued is based on dynamic programming. The optimal solution is shown to conditionally depend on the portfolio realization, thus providing a practical scheme for the dynamic portfolio rebalancing. Finally some case studies are given to show the proposed methodology.

Rudoy et al. (2008) studied the problem of optimal portfolio construction when the trading horizon consists of two consecutive decision intervals and rebalancing is permitted. It is assumed that the log-prices of the underlying assets are non-stationary, and specifically follow a discrete-time co integrated vector autoregressive model. The authors extended the classical Markowitz mean-variance optimization approach to a multi-period setting, in which the new objective is to maximize the total expected return, subject to a constraint on the total allowable risk. In contrast to traditional approaches, they adopted a definition for risk which takes into account the non-zero correlations between the inter-stage returns. This portfolio optimization problem amounts to not only determining the relative proportions of the assets to hold during each stage, but also requires one to determine the degree of portfolio leverage to assume. Due to a fixed constraint on the standard deviation of the total return, the leverage decision is equivalent to deciding how to optimally partition the allowed variance, and thus variance can be viewed as a shared resource between the stages. The authors derived the optimal portfolio weights and variance scheduling scheme for a trading strategy based on a dynamic programming approach, which is utilized in order to make the problem computationally tractable. The performance of this method is compared to other trading strategies using both Monte Carlo simulations and real data, and promising results are obtained.

Ye (2007) considered a continuous-time model of optimal life insurance, consumption and portfolio is examined by dynamic programming technique. The Hamilton-Jacobi- Bellman (HJB in short) equation with the absorbing boundary condition is derived. Then explicit solutions for Constant Relative Risk Aversion (CRRA in short) utilities with subsistence levels are obtained. Asymptotic analysis is used to analyze the model.

Dijkhuizen et al. (1993) used a personal computer-based stochastic dynamic programming (DP) model for the determination of the optimal replacement policy in swine breeding is evaluated. The model provides the maximal expected annual net returns of current herd sows and subsequent replacements over time. The DP-based system was seen to be viable in modeling such factors as biological variations, but are limited by hardware requirements. Result accuracy is effected by the number of DP runs achieved.

Lubbecke et al. (2005) used column generation method to solve linear programs with a large number of variables. Dynamic program algorithms are used for column generation and a simple technique is used to reduce the state space of these algorithms.

Dynamic Programming has been applied to a number of digital signal processing problems. In this paper Rader et al, (2003) discussed it's well known application of determining the optimum order of sections in a digital filter realization. The authors showed that the method is quite insensitive to the specific details of the problem; it is applicable over a wide range of possible optimality criteria, various kinds of arithmetic, scaling options, etc. This is characteristic of the application of dynamic programming to many signal processing problems. Also, since a problem, to be solved by dynamic programming, must be represented as the traversal of a directed graph, we usually discover unsuspected structure in the problem when we attempt to solve it using dynamic programming. Quite often it is necessary to recognize this structure in order to solve the problem efficiently. In the case of ordering of filter section the structure leads to an efficient utilization of memory.

A system and method are disclosed for capturing the full dynamic and multi-dimensional nature of the asset allocation problem through applications of stochastic dynamic programming and stochastic programming techniques. The system and method provide a novel approach to asset allocation and based on stochastic dynamic programming and Monte Carlo sampling that permit one to consider many rebalancing periods, many asset classes, dynamic cash flows, and a general representation of investor risk preference. The system and method further provide a novel approach of representing utility by directly modeling risk aversion as a function of wealth, and thus provide a general framework for representing investor preference. They system and method demonstrate how the optimal asset allocation depends on the investment horizon, wealth, and the investors risk preference and how optimal asset allocation therefore changes over time depending on cash flow and the returns achieved and how dynamic asset allocation leads to superior results compared to static or myopic techniques. Examples of dynamic strategies for various typical risk preferences and multiple asset classes are described.

The dramatic growth in institutionally managed assets, coupled with the advent of Internet trading and electronic brokerage for retail investors, has led to a surge in the size and volume of trading. At the same time, competition in the asset-management industry has increased to where fractions of a percent in performance can separate the top funds from those in the next tier. In this environment, portfolio managers have begun to explore active management of trading to boost returns. Controlling execution cost can be viewed as a stochastic dynamic optimization problem because trading takes time, stock prices exhibit random fluctuations, and execution prices depend on trade size, order flow, and market conditions. In this article, the authors apply stochastic dynamic programming to derive trading strategies that minimize the expected cost of executing a portfolio of securities over a fixed time period. That is, they derive the optimal sequence of trade as a function of prices, quantities, and other market conditions. To illustrate the practical relevance of these methods, Bertsimas et al, (1999) apply them to a hypothetical portfolio of 25 stocks. They estimate the methods' price-impact functions using 1996 trade data and derive the optimal execution strategies. The authors also perform several Monte Carlo simulations to compare the optimal strategy's performance to that of several alternatives.

Battocchio et al. considered a stochastic model for a defined-contribution pension fund in continuous time. In particular, we focus on the portfolio problem of a fund manager who wants to maximize the expected utility of his terminal wealth in a complete financial market with stochastic interest rate. The fund manager must cope with a set-off stochastic investment opportunities and two background risks: the salary risk and the inflation risk. We use the stochastic dynamic programming approach. We show that the presence of the inflation risk can solve some problems linked to the use of the stochastic dynamic programming technique, and namely to the stochastic partial differential equation deriving from it. The authours found a closed form solution to the asset allocation problem, without specifying any functional form for the coefficients of the diffusion processes involved in the problem. Finally, the derivation of a closed form solution allows us to analyze in detail the behavior of the optimal portfolio with respect to salary and inflation.

Bouzaher et al. (1990), used dynamic programming algorithm design to analyze soil movement, to ensure water quality and reduce the costs of water treatment by facilitating the control of agricultural sediment pollution in surface waters. The algorithm models analyze the spatial characteristics of soil movement though a watershed and the impact of soil movement on reservoirs and water channels. The model solves this class of pollution control problems by generating sediment abatement cost frontiers. This information is valuable to watershed management and planning because it devises control strategies to reduce sediment deposition in water courses and can be used to identify special problem areas.

Greco (1990) said dynamic programming is a general technique for solving optimization problems. It is based on the division of problems into simpler sub problems that can be computed separately. In this paper, we show that Datalog with aggregates and other nonmonotonic constructs can express classical dynamic programming optimization problems in a natural fashion, and then we discuss the important classes of queries and applications that benefit from these techniques.

Ryzin's work, Van Ryzin and Vulcano considered a revenue management network capacity control problem in a setting where heterogeneous customers choose among the various products offered by a firm (for example, different light times, fare glasses and/or routings). Customers may therefore substitute if their preferred products are not offered, even buy up. Their choice model is very general, simply specifying the probability of purchase for each fare product as a function of the set of fare products offered. Overall, the value of this paper is to facilitate the understanding of more complex, and probably more realistic, models of revenue management.

Dynamic programming addresses how to make optimal decisions over time under uncertain conditions and to control a system. Most risk management situations can be analyzed assuming a discrete-state and a discrete time over a finite-horizon modeling.

Four criteria of research analyzed:

(i) A paper could consider a single product (at various prices) or multiple products (depending on purchase restrictions or independent demands for example);

(ii) A paper could consider a static policy (assuming a strict order of booking arrivals) or allow for a dynamic policy (not assuming the early birds hypothesis);

(iii) A paper could consider various forms of demand process;

(iv) A paper could consider either a single resource for 1 to *n* products or multiple resources (such as an airline network of hubs and spokes).

CHAPTER THREE

METHODOLOGY

3.0 INTRODUCTION

This chapter presents the research methodology of the study.

3.1 DYNAMIC PROGRAMMING

Dynamic Programming is a technique that can be used to solve many 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 mathematics and computer science, dynamic programming is a method for solving complex problems by breaking them down into simpler sub problems. It is applicable to problems exhibiting the properties of overlapping sub problems which are only slightly smaller and have optimal sub-structures. When applicable, the method takes far less time than naïve methods.

The key idea behind dynamic programming is quite simple. In general, to solve a given problem, we need to solve different parts of the problem (sub problems), then combine the solution of the sub problems to reach an overall solution. Often, many of these sub problems are really the same. The dynamic programming approach seeks to solve each sub problem only once, thus saving a lot of computation. This is especially useful when the number of repeating sub problems is exponentially large.

Top-down dynamic programming simply means storing the results of certain calculations, which are later used again since the completed calculation is a sub-problem of a larger calculation. Bottom-up dynamic programming involves formulating a complex calculation as a recursive series of simpler calculations.

The term dynamic programming was originally used in the 1940s by Richard Bellman to describe the process of solving problems where one needs to find the best decisions one after another. By 1953, he refined this to the modern meaning, referring specifically to nesting smaller decision problems inside larger decisions, and the field was there after recognized by the IEEE as a systems analysis and engineering topic. Bellman‘s contribution is remembered in the name of a systems of Bellman equation, a central result of dynamic programming which restates an optimization problem in recursive form.

The word dynamic was chosen by Bellman because it sounded impressive, not because it described how the method worked. The word programming referred to the use of the method to find an optimal program, in the sense of a military schedule for training or logistics. This usage is the same as that in the phrases linear programming and mathematical programming a synonym for optimization.

Dynamic programming is both a mathematical optimization method and a computer programming method. In both contexts it refers to simplifying a complicated problem by breaking it down into simpler sub-problems in a recursive manner. While some decision problems cannot be taken apart this way, decisions that span several points in time do often break apart recursively; Bellman called this the ―Principle of Optimality‖. Likewise, in computer science, a problem which can be broken down recursively is said to have optimal substructure.

If sub-problems can be nested recursively inside larger problems, so that dynamic programming methods are applicable, then there is a relation between the value of the larger problem and the values of the sub-problems. In the optimization literature this relationship is called the Bellman equation.

3.2 DYNAMIC PROGRAMMING IN MATHEMATICAL OPTIMIZATION

In terms of mathematical optimization, dynamic programming usually refers to simplifying a decision by breaking it down into a sequence of decision steps over time. This is done by defining sequence of value function V_1, V_2, ... V_n with an argument y representing the state of the system at times I from 1 to n. The definition of $V_n(y)$ is the value obtained in state y at the last time n. The values V_i at earlier times $i = n-1, n-2, \ldots, 2, 1$ can be found by working backward, using a recursive relationship called the Bellman equation, for $i = 1, 2, \ldots, n$ for those states. Finally, V_i at the initial state of the system is the value of the optimal solution. The optimal values of the decision variables can be recovered, one by one, tracking back the calculations already performed.

3.2.1 DYNAMIC PROGRAMMING IN COMPUTER PROGRAMMING

There are two key attributes that a problem must have in order for dynamic programming to be applicable: optimal substructure and overlapping sub problems which are only slightly smaller. When the overlapping problems are, say, half the size of the original problem the strategy is called ―divide and conquer‖ rather than ―dynamic programming‖. This is why merge sort, quick sort, and finding all matches of a regular expression are not classified as dynamic programming problems. Optimal substructure means that the solution to a given optimization problem can be obtained by the combination of optimal solutions to its sub problems. Consequently, the first step towards devising a dynamic programming solution is to check whether the problem exhibits such optimal substructure. Such optimal substructures are usually described by means of recursion. For example given a graph $G = (V, E)$, the shortest path p from a vertex to a vertex v exhibits optimal substructure: take any intermediate vertex w on this shortest path . If is truly the shortest path, then the path p_1 from to and from to are indeed the shortest paths between the corresponding vertices. Hence, one can easily formulate the solution for finding shortest paths in a recursive manner, which is what the Bellman-Ford algorithm does.

Overlapping sub problems means that the space of sub problems must be small, that is, any recursive algorithm solving the problem should solve the same sub problems over and over, rather than generating new sub problems.

For example, consider the recursive formulation for generating the Fibonacci series:

$F_i = F_{i-1} + F_{i-2}$, with base case $F_1 = F_2 = 1$.

Then $F_{13} = F_{12} + F_{11}$, and $F_{12} = F_{11} + F_{10}$.

Now F_{11} is being solved in the recursive sub trees if both F_{13} as well as F_{12} are known. Even though the total number of sub problems is actually small (only 13 of them), we end up solving the same problems over and over if we adopt a naïve recursive solution such as this.

Dynamic programming takes account of this fact and solves each sub-problem only once.

This can be achieved in either of two ways

- Top-down approach:
  This is the direct fall-out of the recursive formulation of any problem. If the solution to any problem can be formulated recursively using the solution to its sub problems, and if its sub problems are overlapping, then one can easily memorize or store the solutions

to the sub problems in a table. Whenever we attempt to solve a new sub problem, we first check the table to see if it is already solved. If a solution has been recorded we can use it directly, otherwise we solve the sub problem and add its solution to the table.

- Bottom-up approach:

  This is the more interesting case. Once we formulate the solution to a problem recursively as in terms of its sub problems, we can try reformulating the problem in a bottom-up fashion: try solving the sub problems first and use their solutions to buildon and arrive at solutions to bigger sub problems. This is also usually done in a tabular form by iteratively generating solutions to bigger and bigger sub problems by using the solutions to small sub problems. For example, if we already know the values of F_{41} and F_{40}, we can directly calculate the value of F_{42}.

3.2.2 **CHARACTERISTICS OF DYNAMIC PROGRAMMING APPLICATIONS**

There are a number of characteristics that are common to all problems and all dynamic programming problems.

(i) The problem can be divided into stages with a decision required at each stage. In capital budgeting problem the stages were the allocations to a single plant and the decision was how much to spend.

(ii) Each stage has a number of states associated with it. The states for a capital budgeting problem correspond to the amount spent at that point in time. In the shortest path problem the states were the node reached.

(iii)The decision at one stage transforms on state into a state in another stage. The decision of much to spend gave a total amount spent for the next stage. The decision of where to go next defined where you arrived in the next stage.

(iv) Given the current state, the optimal decision for each of the remaining state does not depend on the previous states or decisions. In the budgeting problem, it is not necessary to know how the money was spent in previous stages, only how much was spent.

In the path problem, it is not necessary to know how you got to a node, only that you did.

(v) There exist a recursive relationship that identifies the optimal decision for stage j, given that stage $j+1$ has already been solved.

(vi) The final stage must be solvable by itself. The last time properties are tied up in the recursive relationship given above.

3.2.3 COMPUTATIONAL EFFICIENCY OF DYNAMIC PROGRAMMING

In smaller networks it would be a matter of determining the shortest path from one point to another by enumerating all the possible paths (after all there are only a few paths). In larger networks however, compute enumeration is practically impossible and the use of dynamic programming is much more efficient for determining a shortest path.

In a network where there are five stages with: stage1 – 1 state, stage 2 – 3 state, stage 3 – 3 state, stage 4 – 2 state and stage 5 – 1 state. Total enumerate will result in 1(3)(3)(2)(1) = 18 paths while DP with result in 1(3)(3)(2)(1) = 18 paths.

If in another network there are seven stages with 5 states each. The total enumeration gives $5(5^5)$ paths.

3.2.4 DETERMINISTIC VERSUS STOCHASTIC DYNAMIC PROGRAMMING

There is one major difference between Stochastic Dynamic Programming and Deterministic Dynamic Programming. In Deterministic Dynamic Programming the complete decision path is known. In Stochastic Dynamic Programming the actual decision path will depend on the way the random aspects play out. Because of this solving a Stochastic Dynamic Programming problem involves giving a decision rule for every possible state, not just along an optimal path. A multi-stage decision process is Stochastic if the return associated with at least one decision in the process is random. This randomness generally enters in one of two ways; either the states are uniquely determined by the decisions but the returns associated with one or more states are uncertain or the returns are uniquely determined by the states arising from one or more decision are uncertain.

If the probability distributions governing the events are known and if the number of stages are finite, then deterministic dynamic programming approach is useful for optimizing a stochastic multistage decision process. The general procedure is to optimize the expected value of the return. In those cases where the randomness occurs exclusively in the returns associated with the states arising from the decision, this procedure has the effect of transforming a stochastic process into a deterministic one. For processes in which randomness exists in the states associated with the decisions, a policy may be exhibited as a policy table.

3.3 INTEGER PROGRAMMING

An Integer Programming problem (IP) is a Linear Programming (LP) problem in which some or all the variables are required to be nonnegative integers. An Integer programming in which all variables are required to be integers is a Pure Integer Programming problem.

Many problems can be modeled as an Integer Programming problem. The model is;

For a maximization problem

Maximize $$Z = \sum_{i=1}^{n} r_i(ax_i + b)$$

Subject to $$\sum_{i=1}^{n} x_i \leq c_i$$

$$x_i \geq 0, 1, 2, 3, 4 \dots \dots \dots . N$$

3.4 APPLICATIONS OF DYNAMIC PROGRAMMING

Dynamic programming can be applied in Consumption and savings problems, shortest path problem, The Knapsack Problems, Network Problems, Inventory Problems, Equipment replacement problems, Resource Allocation Problems etc

3.4.1 OPTIMAL CONSUMPTION AND SAVING PROBLEMS

A mathematical optimization problem that is often used in dynamic programming to economists concerns a consumer who lives over the periods t = 0, 1, 2, ..., T and must decide how much to consume and how much to save in each period.

Let c_t be consumption in period t, and assume consumption yields utility $u(c_t) = \ln(c_t)$ as long as the consumer lives. Assume the consumer is impatient, so that he discounts future utility by a factor b each period, where $0 < b < 1$. Let k_t be capital in period t. Assume initial capital is a given amount $k_0 > 0$, and suppose that this period's capital and consumption determine

next period's capital as $k_t + 1 = Ak_t^a - c_t$, where A is a positive constant and $0 < a < 1$. Assume capital cannot be negative. Then the consumer's decision problem can be written as follows:

Maximize $$Z = \sum_{t=0}^{T} \mathrm{b}^{\mathrm{t}} In\ (c_t)$$

Subject to $$k_{t+1} = Ak_t^a - c_t \geq 0$$

for all t = 0,1,2,...,T

Written this way, the problem looks complicated, because it involves solving for all the choice variables $c_0, c_1, c_2, \dots, c_T$ and $k_1, k_2, \dots, k_{T+1}$ simultaneously. (Note that k_0 is not a choice variable—the consumer's initial capital is taken as given.)

The dynamic programming approach to solving this problem involves breaking it apart into a sequence of smaller decisions. To do so, we define a sequence of value functions $V_T(k)$, for $t = 0, 1, 2, \dots, T, T+1$ which represent the value of having any amount of capital k at each time t. Note that $V_{T+1}(k) = 0$, that is, there is (by assumption) no utility from having capital after death.

The value of any quantity of capital at any previous time can be calculated by backward induction using the Bellman equation. In this problem, for each $t = 0, 1, 2, \dots, T$ the Bellman equation is

$$V_t(k_t) = \max(1\text{n}(c_t) + bV_{t+1}(k_{t+1})$$

$$\text{subject to} \quad Ak_t^a - c_t \geq 0$$

This problem is much simpler than the one we wrote down before, because it involves only two decision variables, and c_t Intuitively, k_{t+1} instead of choosing his whole lifetime plan at birth, the consumer can take things one step at a time. At time t, his current capital k_t is given, and he only needs to choose current consumption c_t and saving k_{t+1}.

To actually solve this problem, we work backwards. For simplicity, the current level of capital is denoted as k $V_{T+1}(k)$. is already known, so using the Bellman equation once we can calculate $V_T(k)$, and so on until we get to $V_0(k)$, which is the value of the initial decision problem for the whole lifetime. In other words, once we know $V_{T-j+1}(k)$, we can calculate $V_{T-j}(k)$, which is the maximum of $\ln(c_{T-j}) + bV_{T-j+1}(Ak^a - c_{T-j})$, where c_{T-j} is the choice variable and working backwards, it can be shown that the value function at time t = T − j is where each V_{T-j} is a constant, and the optimal amount to consume at time t = T − j is which can be simplified to

$c_T(k) = A\mathrm{k}^a$, and $c_{T-1}(k) = (1+x)^n = \frac{Ak^a}{1+ab}$, and $c_T - 2(k) = \frac{Ak^a}{1+ab+a^2b^2}$

We see that it is optimal to consume a larger fraction of current wealth as one gets older, finally consuming all remaining wealth in period T, the last period of life.

3.5 DIJKSTRA'S ALGORITHM FOR THE SHORTEST PATH PROBLEM

From a dynamic programming point of view, Dijkstra‘s the shortest path problem is a successive approximation scheme that solves the dynamic programming functional equation for the shortest path problem by the Reaching method.

In fact, Dijkstra's explanation of the logic behind the algorithm is to find the path of minimum total length between two given nodes P and Q. We use the fact that, if R is a node on the minimal path from P to Q, knowledge of the latter implies the knowledge of the minimal path from P to R. This is a paraphrasing of Bellman‘s famous Principle of Optimality in the context of the shortest path problem.

Let's look at a particular type of shortest path problem. Suppose we wish to get from A to J in the road network of Figure 3.1.

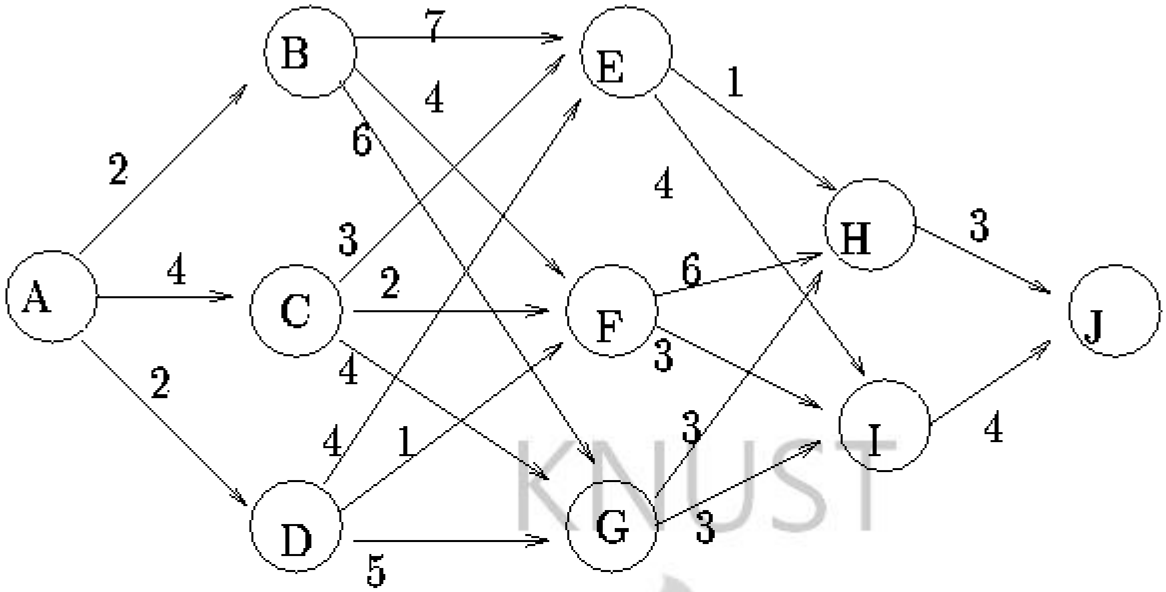

FIGURE 3.1: ROAD NETWORK

The numbers on the arcs represent distances. Due to the special structure of this problem, we can break it up into stages. Stage 1 contains node A, stage 2 contains nodes B, C, and D, stage 3 contains node E, F, and G, stage 4 contains H and I, and stage 5 contains J. The states in each stage correspond just to the node names, so stage 3 contains states E, F, and G.

If we let *S* denote a node in stage *j* and let $f_j(S)$ be the shortest distance from node *S* to the destination *J*, we can write

$$f_j(S) = \min_{\text{nodes Z in stage } j+1} \{c_{sz} + f_{j+1}(Z)\}$$

where c_{sz} denotes the length of arc *SZ*. This gives the recursion needed to solve this problem.

We begin by setting $f_5(J) = 0$ and follow with the rest of the calculations:

Stage 4:

During stage 4, there are no real decisions to make: you simply go to your destination *J*. So you get:

- $f_4(H) = 3$ by going to J,
- $f_4(I) = 4$ by going to J.

Stage 3:

Here there are more choices. Here's how to calculate $f_4(F)$. From F you can either go to H or I. The immediate cost of going to H is 6. The following cost is $f_4(H) = 3$. The total cost is 9. The immediate cost of going to I is 3. The following cost is $f_4(I) = 4$ for a total of 7. Therefore, if you are ever at F, the best thing to do is to go to I. The total cost is 7, $f_4(F) = 7$ and so on.

TABLE 3.1: ROAD NETWORK CALCULATIONS

S_3	$CS_3\ Z_3 + F_4(Z_3)$		$F_3(S_3)$	**Decision**
	H	**I**		
E	**4**	**8**	**4**	**H**
F	**9**	**7**	**7**	**I**
G	**6**	**7**	**6**	**H**

You now continue working back through the stages one by one, each time completely computing a stage before continuing to the preceding one. The results are:

TABLE 3.1.1: STAGE 2 OF ROAD NETWORK CALCULATIONS

S_2	$CS_2\ Z_2 + F_3(Z_3)$			$F_3(S_3)$	**Decision Go to**
	E	**F**	**G**		
B C	**11**	**11**	**12**	**11**	**E or F**
D	**7**	**9**	**10**	**7**	**E**
	8	**8**	**11**	**8**	**E or F**

TABLE 3.1.2: STAGE 1 OF ROAD NETWORK CALCULATIONS

S_1	$CS_1\ Z_1 + F_2(Z_1)$			$F_2(S_1)$	**Decision Go to**
	B	**C**	**D**		
A	**13**	**11**	**11**	**11**	**C or D**

There is another formulation for the knapsack problem. This illustrates how arbitrary our definitions of stages, states, and decisions are. It also points out that there is some flexibility on the rules for dynamic programming. Our definitions required a decision at a stage to take us to the next stage (which we would already have calculated through backwards recursion). In fact, it could take us to any stage we have already calculated. This gives us a bit more flexibility in our calculations.

The recursion i am about to present is a forward recursion. For a knapsack problem, let the stages be indexed by *w*, the weight filled. The decision is to determine the last item added to bring the weight to *w*. There is just one state per stage. Let *g*(*w*) be the maximum benefit that can be gained from a *w* pound knapsack. Continuing to use and as b_j the weight w_j and benefit, respectively, for item *j*, the following relates *g*(*w*) to previously calculated *g* values:

$$g(w) = \max_{j} \{b_j + g(w - w_j)\}$$

Intuitively, to fill a *w* pound knapsack, we must end off by adding some item. If we add item *j*, we end up with a knapsack of size $w - w_j$ to fill. To illustrate on the above example:

- *g*(0) = 0
- *g*(1) = 30 add item 3.
- g(2) = max$\{65 + g(0) = 65, 30 + g(1) - 60\} = 65$ add item 1.
- g(2) = max$\{65 + g(1) = 95, 80 + g(0) = 80, 30 + g(2) = 95\} = 95$ add item 1 or 3.
- g(2) = max$\{65 + g(2) = 130, 80 + g(1) = 110, 30 + g(3) = 125\} = 130$ add item 1.
- g(2) = max$\{65 + g(3) = 160, 80 + g(2) = 145, 30 + g(4) = 160\} = 160$ add item 1 or 3.

This gives a maximum of 160, which is gained by adding 2 of item 1 and 1 of item 3.

3.6 THE KNAPSACK PROBLEM

Imagine we have a homework assignment with different parts labeled A through G. Each part has a —value‖ (in points) and a —size‖ (time in hours to complete). For example, say the values and times for our assignment are:

TABLE 3.2: KNAPSACK PROBLEM

	A	B	C	D	E	F	G

Value	**7**	**9**	**5**	**12**	**14**	**6**	**12**
Time	**3**	**4**	**2**	**6**	**7**	**3**	**5**

Say we have a total of 15 hours, which parts should we do? If there was partial credit that was proportional to the amount of work done (e.g., one hour spent on problem C earns you 2.5 points) then the best approach is to work on problems in order of points/hour. But, what if there is no partial credit? In that case, which parts should you do, and what is the best total value possible? The above is an instance of the knapsack problem, formally defined as follows:

In this case, the optimal strategy is to do parts A, B, F, and G for a total of 34 points. We notice that this doesn‘t include doing part C which has the most points/hour!

In the knapsack problem we are given a set of n items, where each item i is specified by a size s_i and a value v_i. We are also given a size bound S (the size of our knapsack). The goal is to find the subset of items of maximum total value such that sum of their sizes is at most S (they all fit into the knapsack). We can solve the knapsack problem in exponential time by trying all possible subsets. With Dynamic Programming, we can reduce this to time O (*nS*).

Let‘s do this top down by starting with a simple recursive solution and then trying to memorize. it. Let‘s start by just computing the best possible total value, and we afterwards can see how to actually extract the items needed.

3.7 EQUIPMENT REPLACEMENT PROBLEMS

Suppose a shop needs to have a certain machine over the next five year period. Each new machine cost $1000, the cost of maintaining the machine during its years of operation is as follows: $c_1 = \$60,\ c_2 =$\$80, and $c_3 = \$120.$ A machine may be kept up to three years before being traded in, the trade in value after i years is $s_1 = \$800, s_2 = \$600$ as $s_3 = \$500.$ How can the shop minimize cost over the five year period?

Let the stages corresponds to each year. The state is the age of the machine for that year. The decision are whether to keep the machine or trade it in for a new machine.

Let $f_t(x)$ be the minimum cost incurred from time t to time 5, given the machine is x years old in time t.

Since we have to trade in at time 5, $f_5(x) = -s_x$

Now we consider the time periods.

If you have three year old machine in time t, you must trade in, so

$f_t(3) = -500 + 1000 + 60 + f_{t+1}(1)$

If you have a two year old machine you can either trade or keep.

- Trade will cost – 600 + 1000 + 60 + $f_{t+1}(1)$
- Keep will cost 120 + $f_{t+1}(3)$

So the best thing to do with a two year old machine is the minimum of the two

$f_t(2) = \min\{-600 + 1000 + 60 + f_{t+1}(1), 120 f_{t+1}(3)\}$

For a one year old machine trade will cost -800 + 100 + 60 + f_{t+1}(1)

Keep will cost 80+$f_{t+1}(2)$

$f_t(1) = \min\{-800 + 1000 + 60 + f_t(1), 80 + f_t\ (2)\}$

For a zero year old machine we have to buy 1000 + 60 + $f_t(1)$

$f_0(0)$=1000+60+$f_1(1)$

$f_o(0)$=1060+$f_1(1)$

$f_t(1) = \min(260 + f_{t+1}(1), 80 + f_{t=1}(2)$

$f_t(2) = \min(460 + f_{t+1}(1), 120 + f_{t+1}(3))$

$f_t(3) = \min\ (560 + f_{t+1}(1))$

$f_5(x) = -5_x$

$f_5(1) = -S_1$ = - 800

$f_5(2) = -S_2 =$ - 600

$f_5(3) = -S_3$= - 500

This is solved with backwards recursion as follows:

TABLE 3.3.1: STAGE 5 OF EQUIPMENT REPLACEMENT PROBLEM

Age x	$f_5(x)$
1	**-800**
2	**-600**
3	**-500**

TABLE 3.3.2: STAGE 4 OF EQUIPMENT REPLACEMENT PROBLEM

Age	**Trade**	**Keep**	$f_4(2)$	**Decision**
1	**-540**	**-520**	**-540**	**Trade**
2	**-340**	**-380**	**-380**	**Keep**
3	**-240**	**–**	**-240**	**Trade**

TABLE 3.3.3: STAGE 3 OF EQUIPMENT REPLACEMENT PROBLEM

Age	**Trade**	**Keep**	$f_3(2)$	**Decision**
1	**-280**	**-300**	**-300**	**Keep**
2	**-80**	**-120**	**-120**	**Keep**
3	**20**		**20**	**Trade**

TABLE 3.3.4: STAGE 2 OF EQUIPMENT REPLACEMENT PROBLEM

Age	Trade	Keep	$f_1(2)$	Decision
1	**220**	**220**	**220**	**Trade or Keep**

TABLE 3.3.5: STAGE 1 OF EQUIPMENT REPLACEMENT PROBLEM

Age	Trade	Keep	$f_0(2)$	Decision
0	**–**	**1280**	**1280**	**Keep**

So the cost is 1280 and one solution is to trade in years; 1 and 2. There are other optimal solutions.

3.8 AN INVENTORY PROBLEM

In this section, we illustrate how dynamic programming can be used to solve an inventory problem with the following characteristics:

1. Time is broken up into periods, the present period being period 1, the next period 2, and the final period T. At the beginning of period 1, the demand during each period is known.
2. At the beginning of each period, the firm must determine how many units should be produced. Production capacity during each period is limited.
3. Each period‘s demand must be met on time from inventory or current production. During any period in which production takes place, a fixed cost of production as well as a variable per-unit cost is incurred.

4. The firm has limited storage capacity. This is reflected by a limit on end-of- period inventory. A per-unit holding cost is incurred on each periods ending inventory.
5. The firm‘s goal is to minimize the total cost of meeting on time the demands for periods 1, 2,………., T.

3.9 RESOURCE ALLOCATION PROBLEMS

Resource allocation problems, in which limited resources must be allocated among several activities, are often solved by dynamic programming. Recall that we have solved such problems by linear programming. To use linear programming to do resource allocation three assumptions must be made:

Assumption 1

The amount of a resource assigned to an activity may be any nonnegative number.

Assumption 2

The benefit obtained from each activity is proportional to the amount of the resource assigned to the activity.

Assumption 3

The benefit obtained from more than one activity is the sum of the benefits obtained from the individual activities.

Even if assumptions 1 and 2 do not hold, dynamic programming can be used to solve resource allocation problems efficiently when assumption 3 is valid and when the amount of the resource allocated to each activity is a member of a finite set.

CHAPTER FOUR

DATA COLLECTION AND ANALYSIS

4.0 INTRODUCTION

In this chapter, we shall put forward the data collection and analysis of the study.

4.1 DATA COLLECTION

Data on rate of fixed deposit is collected from three financial institutions; Barclays Bank, Ghana Commercial Bank and Fidelity Bank. In each of these institutions fixed deposit rates are in bands of various amounts for different categories of periods as shown in Tables 4.1, 4.2, 4.3.

TABLE 4.1: INTEREST RATES -BARCLAYS BANK

BAND (GHS)	91 DAY	182 DAY	1 YEAR
10,000 – 25,000	15.00%	13.50%	14.50%
$25,000^{+}$ – 50,000	17.50%	14.50%	15.50%
$50,000^{+}$ – 100,000	18.50%	15.50%	16.50%
$100,000^{+}$ – 500,000	19.00%	16.50%	18.00%
$500,000^{+}$	21.00%	18.00%	18.25%

TABLE 4.2: INTEREST RATES -GHANA COMMERCIAL BANK

BAND (GHS)	91 DAY	182 DAY	1 YEAR
10,000 – 25,000	9.50%	10.00%	11.00%
$25,000^{+}$ – 50,000	9.50%	10.00%	11.00%
$50,000^{+}$ – 100,000	10.00%	10.50%	11.50%
$100,000^{+}$ – 500,000	10.50%	11.00%	12.00%
$500,000^{+}$	11.50%	12.00%	13.50%

TABLE 4.3: INTEREST RATES- FIDELITY BANK

BAND (GHS)	91 DAY	182 DAY	1 YEAR
10,000 – 25,000	19.70%	23.40%	17.50%
$25,000^{+}$ – 50,000	20.70%	23.90%	18.50%
$50,000^{+}$ – 100,000	21.70%	24.40%	19.50%
$100,000^{+}$ – 500,000	22.70%	24.90%	20.50%
$500,000^{+}$	23.70%	25.40%	21.50%

4.2 ANALYSIS

A fixed deposit is a financial instrument provided by banks which provides investors with a higher rate of interest than a regular savings account, until the given maturity date.

Interest rate is the rate at which interest is paid by borrowers (debtors), for the use of money they borrow from lenders (creditors).

In each of the institutions fixed deposits are in three categories; 91- day, 182 – day and 1 year. Generally the investments are for a specified period. At the end of the period interest is paid and the investor may continue or otherwise. But for the purposes of this study we assume an annual period, meaning that investments shorter than one year are rolled over until the end of the year. For this purpose we have to annualize the returns for uniformity in periods. The rates are annualized for each of the bands and two arbitrary points in each band are chosen for the study. The return from the various amounts for the periods is as shown in Table 4.4.

TABLE 4.4: ANNUAL RETURNS FROM THE VARIOUS PERIODS

INVESTMENT	AMOUNT INVESTED (GHC'000)										
	0	10	20	30	40	60	80	200	300	500	600
Investment 1	0.00	1.59	3.17	5.60	7.47	11.83	14.95	40.79	61.19	113.56	136.27
Investment 2	0.00	0.98	1.97	2.95	3.94	6.23	7.88	21.84	32.76	60.03	72.03
Investment 3	0.00	2.12	4.24	6.71	8.95	14.12	17.89	49.41	74.12	129.45	155.34
Investment 4	0.00	1.40	2.79	4.51	6.01	9.66	12.88	34.36	51.54	94.05	112.86
Investment 5	0.00	1.03	2.05	3.08	4.10	6.47	8.62	22.61	33.91	61.80	74.16
Investment 6	0.00	2.48	4.95	7.60	10.13	15.53	20.71	52.90	79.35	135.06	162.08
Investment 7	0.00	1.45	2.90	4.65	6.20	9.90	13.20	36.00	54.00	91.25	109.50
Investment 8	0.00	1.10	2.20	3.30	4.40	6.90	9.20	24.00	36.00	67.50	81.00
Investment 9	0.00	1.75	3.50	5.55	7.40	11.70	15.60	41.00	61.50	107.50	129.00

Table 4.4 shows that there are nine investment options, three from each of the three financial institutions; Barclays Bank – 91 day, 182 day and 1 year, GCB Bank – 91 day, 182 day and 1 year and Fidelity Bank – 91 day, 182 day and 1 year.

The following notations are made for the study:

Investment 1 - Barclays Bank – 91 day - $f_1(x)$ Investment

2 - GCB – 91 day - $f_2(x)$

Investment 3 - Fidelity Bank – 91 day - $f_3(x)$

Investment 4 - Barclays Bank – 182 day - $f_4(x)$

Investment 5 - GCB Bank – 182 day - $f_5(x)$

Investment 6 - Fidelity Bank – 182 day - $f_6(x)$

Investment 7 - Barclays Bank – 1 year - $f_7(x)$

Investment 8 - GCB Bank – 1 year - $f_8(x)$

Investment 9 - Fidelity Bank – 1 year - $f_9(x)$

The amount to be invested is a maximum of ¢600 and two amounts from the various bands are selected at random. For the purpose of the study we restrict the amount invested to ¢0, ¢10, ¢20, ¢30, ¢40, ¢60, ¢80, ¢200, ¢300, ¢500 and ¢600. For the optimal return at least a total of ¢600 must be invested, (amounts are in thousand Ghana cedis).

To start the algorithm we arrange the investments in no specific order. We identify the appropriate stages, states and decisions. We define a stage such that when one stage is remained the problem will have a trivial solution.

We define Investment as the stage because;

At Investment 9: Investment 9 Returns = Investment 8 Returns + Investment 9 inputs

At Investment 8: Investment 8 Returns = Investment 7 Returns + Investment 8 inputs

At Investment 7: Investment 7 Returns = Investment 6 Returns + Investment 7 inputs

At Investment 6: Investment 6 Returns = Investment 5 Returns + Investment 6 inputs

At Investment 5: Investment 5 Returns = Investment 4 Returns + Investment 5 inputs

At Investment 4: Investment 4 Returns = Investment 3 Returns + Investment 4 inputs

At Investment 3: Investment 3 Returns = Investment 2 Returns + Investment 3 inputs

At Investment 2: Investment 2 Returns = Investment 1 Returns + Investment 2 inputs

At Investment 1: Investment 1Returns = Investment 0 Returns + Investment 1 inputs

We define the state of the stage as the action to be performed when a stage (investment) is reached. That is at each stage (investment) the investor will have to decide how much money he will have to invest. To do this we need to know only the amount of money left at the beginning of the investment (stage). Hence we define the State as the amount of money left to be invested.

We define decision as the amount to be invested to obtain the best solution at a stage.

Letting $f_i(x)$ denotes the return in (¢) from investment i when x units of money are invested in it as shown in Table 4.5.

TABLE 4.5: RETURNS OF INVESTMENT.

INVESTMENT	AMOUNT INVESTED (GHC'000)										
	0	10	20	30	40	60	80	200	300	500	600
$f_1(x)$	0.00	1.59	3.17	5.60	7.47	11.83	14.95	40.79	61.19	113.56	136.27
$f_2(x)$	0.00	0.98	1.97	2.95	3.94	6.23	7.88	21.84	32.76	60.03	72.03
$f_3(x)$	0.00	2.12	4.24	6.71	8.95	14.12	17.89	49.41	74.12	129.45	155.34
$f_4(x)$	0.00	1.40	2.79	4.51	6.01	9.66	12.88	34.36	51.54	94.05	112.86
$f_5(x)$	0.00	1.03	2.05	3.08	4.10	6.47	8.62	22.61	33.91	61.80	74.16
$f_6(x)$	0.00	2.48	4.95	7.60	10.13	15.53	20.71	52.90	79.35	156.89	162.08
$f_7(x)$	0.00	1.45	2.90	4.65	6.20	9.90	13.20	36.00	54.00	91.25	109.50
$f_8(x)$	0.00	1.10	2.20	3.30	4.40	6.90	9.20	24.00	36.00	67.50	81.00
$f_9(x)$	0.00	1.75	3.50	5.55	7.40	11.70	15.60	41.00	61.50	107.50	129.00

Table 4.5, shows the various amounts of money invested and the associated returns from the nine investment options.

Define $x(= 0, 10, 20, 30, 40, 60, 80, 200, 300, 500, 600)$ as the number of units of money invested in investment i.

Define $M_i(x)$ = the best return beginning in stage i and state x.

$d_i(x) =$ Decisions taken at a state that achieves $M_i(x)$

We note that if we invest nothing we do not get anything, hence $M_i(0) = 0$ and $d_i(0) = 0$.

The Model:

The model for solving the above problem is as shown below:

Maximize: $Z = \sum f_i(x)$

Subject to $\sum x = 600$

$i = 1, 2, 3, 4, 5, 6, 7, 8, 9$

and $x = 0, 10, 20, 30, 40, 60, 80, 200, 300, 500, 600$

SOLUTION

We begin the solution by considering the last stage of the process, stage 9. We assume that the previous stages have been completed and we are to complete the allocation of the money to the investment 9. Since we do not know how much was allocated to the previous investment (investment 8), we do not know how many units are available for investment 9. Thus we consider all possibilities.

After the first eight investments have been made there will be either ¢0, ¢10, ¢20, ¢30, ¢40, ¢60, ¢80, ¢200, ¢300, ¢500, ¢600. It is clear from the definition of $f_9(x)$ that the best way to complete the process is to allocate all available units to investments 9.

From investment 9:

$M_9(600) = \max[f_9(0), f_9(10), f_9(20), f_9(30), f_9(40), f_9(60), f_9(80), f_9(200),$
$f_9(300), f_9(500), f_9(600)]$

$= \max[0, 1.75, 3.50, 5.55, 7.40, 11.70, 15.60, 41.00, 61.50, 107.50, 129.00] = 129$

$d_9(600) = 600$

$M_9(500) = \max[f_9(0), f_9(10), f_9(20), f_9(30), f_9(40), f_9(60), f_9(80), f_9(200), f_9(300), f_9(500)]$

$= \max[0, 1.75, 3.50, 5.55, 7.40, 11.70, 15.60, 41.00, 61.50, 107.50] = 107.50$

$d_9(500) = 500$

$M_9(300) = \max[f_9(0), f_9(10), f_9(20), f_9(30), f_9(40), f_9(60), f_9(80), f_9(200), f_9(300)]$

$= \max[0, 1.75, 3.50, 5.55, 7.40, 11.70, 15.60, 41.00, 61.50] = 61.50$

$d_9(300) = 300$

$M_9(200) = \max[f_9(0), f_9(10), f_9(20), f_9(30), f_9(40), f_9(60), f_9(80), f_9(200)]$

$= \max[0, 1.75, 3.50, 5.55, 7.40, 11.70, 15.60, 41.00] = 41.00$ $d_9(200) = 200$

$M_9(80) = \max[f_9(0), f_9(10), f_9(20), f_9(30), f_9(40), f_9(60), f_9(80)]$

$= \max[0, 1.75, 3.50, 5.55, 7.40, 11.70, 15.60] = 15.60$ $d_9(80) = 80$

$M_9(60) = \max[f_9(0), f_9(10), f_9(20), f_9(30), f_9(40), f_9(60)]$

$= \max[0, 1.75, 3.50, 5.55, 7.40,\ 11.70] = 11.70$. $d_9(60) = 60$

$M_9(40) = \max[f_9(0), f_9(10), f_9(20), f_9(30), f_9(40)]$

$= \max[0, 1.75, 3.50, 5.55, 7.40] = 7.40$.

$d_9(40) = 40$

$M_9(30) = \max[f_9(0), f_9(10), f_9(20), f_9(30)] = \max[0, 1.75, 3.50, 5.55] = 5.5$; $d_9(30) = 30$

$M_9(20) = \max[f_9(0), f_9(10), f_9(20)] = \max[0, 1.75, 3.50] = 3.50$. $d_9(20) = 20$

$M_9(10) = \max[f_9(0), f_9(10)] = \max[0, 1.75] = 1.75$ $d_9(10) = 10$

$M_9(0) = \max[f_9(0)] = \max[0] = 0$ $d_9(0) = 0$

From investment 8:

$M_8(600)$
$= \max[M_9(600) + f_8(0),\ M_9(500) + f_8(80) + f_8(20),\ M_9(500) + f_8(60) + f_8(40),$
$M_9(300) + f_8(300),\ M_9(200) + f_8(300) + f_8(80) + f_8(20),\ M_9(80) + \ f_8(500) + f_8(20),$
$M_9(20) + \ f_8(500) + f_8(80),\ M_9(0) + f_8(600)]$

$= \max[129 + 0, 107.50 + 9.20 + 2.2, 107.50 + 6.9 + 4.4, 61.5 + 36, 41 + 36 + 9.2 +$
$2.2, 15.60 + 67.50 + 2.2, 3.50 + 67.50 + 9.20, 0 + 81.00]$

$= \max[129.00, 118.90, 118.80,\ 97.50, 88.4, 85.30, 80.2, 81.00\ ] = 129.00$

$d_8(600) = 0$

$M_8(500) = \max[M_9(500) + f_8(0),\ M_9(300) + f_8(200),\ M_9(200) + f_8(300), M_9(0) +$
$f_8(500)]$

$= \max[107.50 + 0,\ 61.50 + 24.00, 41.00 + 36.00, 0 + 67.50]$

$= \max[107.50, 85.50, 77.00, 67.50] = 107.50$

$d_8(500) =_0$

$M_8(300)$
$= \max[M_9(300) + f_8(0), M_9(200) + f_8(80) + f_8(20),\ M_9(200) + f_8(60) + f_8(40),\ M_9(0) + f_8(300)]$

$= \max[61.50 + 0, 41.00 + 9.20 + 2.20, 41.00 + 6.90 + 4.40, 0 + 36.00]$

$= \max[61.50, 52.40, 52.30, 36.00] = 61.50$

$d_8(300) = 0$

$M_8(200)$

$= \max[M_9(200) + f_8(0), M_9(80) + f_8(60) + f_8(40) + f_8(20), M_9(0) + f_8(200)]$

$= \max[41.00 + 0, 15.60 + 6.90 + 4.40 + 2.20, 0 + 24.00] = \max[41.00, 29.1, 24.00] = 41.00$

$d_8(200) = 0$

$M_8(80) = \max[M_9(80) + f_8(0), M_9(60) + f_8(20), M_9(20) + f_8(60), M_9(0) + f_8(80)]$

$= \max[15.60 + 0, 11.70 + 2.20, 3.50 + 6.90, 0 + 9.20]$
$= \max[15.60, 13.90, 10.40, 9.20] = 15.60$
$d_8(80) = 0$

$M_8(60) = \max[M_9(60) + f_8(0), M_9(40) + f_8(20), M_9(20) + f_8(40), M_9(0) + f_8(60)]$

$= \max[11.70 + 0, 7.40 + 2.20, 3.50 + 4.40, 0 + 6.90]$

$= \max[11.70, 9.60, 7.90, 6.90] = 11.70$

$d_8(60) = 0$

$M_8(40)$

$= \max[M_9(40) + f_8(0), M_9(30) + f_8(10),\ M_9(20) + f_8(20),\ M_9(10) + f_8(30)]$

$= \max[7.40 + 0, 5.5 + 1.10, 3.50 + 2.20,\ 1.75 + 3.30]$

$= \max[7.40, 6.60, 5.70, 5.05] = 7.40$

$d_8(40) = 0$

$M_8(30) = \max[M_9(30) + f_8(0), M_9(20) + f_8(10),\ M_9(10) + f_8(20)]$

$= \max[5.50 + 0, 3.50 + 1.10, 1.75 + 2.20] = \max[5.50, 4.60, 3.95] = 5.50$

$d_8(30) = 0$

$M_8(20) = \max[M_9(20) + f_8(0), M_9(10) + f_8(10)]$

$= \max[3.50 + 0, 1.75 + 1.10] = \max[3.50, 2.85] = 3.50$

$d_8(20) = 0$

$M_8(10) = \max[M_9(10) + f_8(0)] = \max[1.75 + 0] = \max[1.75] = 1.75$

$d_8(10) = 0$

$M_8(0) = \max[f_9(0)] = \max[0] = 0$ $d_8(0) = 0$

From investment 7:

$M_7(600)$

$= \max[M_8(600) + f_7(0),\ M_8(500) + f_7(80) + f_7(20),\ M_8(500) + f_7(60) + f_7(40),$
$M_8(300) + f_7(300),\ M_8(200) + f_7(300) + f_7(80) + f_7(20),\ M_8(80) + f_7(500) +$
$f_7(20), M_8(20) + f_7(500) + f_7(80), M_8(0) + f_7(600)]$

$= \max[129 + (0), 107.50 + 13.20 + 2.90 \;\; 107.50 + 9.90 + 6.20, 61.5 + 54.00, 41.00 + 54.00 + 13.20 + 2.9, 15.60 + 91.25 + 2.90, 3.50 + 91.25 + 13.20, 0 + 109.50\,]$

$= \max[129.00, 123.6, 123.6, 115.5, 111.10, 109.75, 107.95, 109.50] = 129.00$;

$d_7(600) = 0$

$M_7(500) = \max[M_8(500) + f_7(0),\; M_8(300) + f_7(200),\; M_8(200) + f_7(300), M_8(0) + f_7(500)]$

$= \max[107.50 + 0, 61.50 + 36.00, 41.00 + 54, 0 + 91.25] =$

$= \max[107.50, 97.50, 95.00, 91.25] = 107.50$

$d_7(500) = 0$

$M_7(300) = \max[M_8(300) + f_7(0), M_8(200) + f_7(80) + f_7(20),\; M_8(200) + f_7(60) + f_7(40)]$

$= \max[61.50 + 0, 41.00 + 13.20 + 2.90, 41.00 + 9.90 + 6.20]$
$= \max[61.50, 57.10, 57.10\,] = 61.50$

$d_7(300) = 0$

$M_7(200) = \max[M_8(200) + f_7(0), M_8(80) + f_7(60) + f_7(40) + f_7(20)]$

$= \max[41.00 + 0, 15.60 + 9.90 + 6.20 + 2.90] = \max[41.00, 34.6] = 41.00$
$d_7(200) = 0$

$M_7(80) = \max[M_8(80) + f_7(0), M_8(60) + f_7(20), M_8(20) + f_7(60)]$

$= \max[15.60 + 0, 11.70 + 2.90, 3.50 + 9.90] = \max[15.60, 14.6, 13.4] = 15.60 \quad d_7(80) = 0$

$M_7(60) = \max[M_8(60) + f_7(0), M_8(40) + f_7(20), M_8(20) + f_7(40)]$

$= \max[11.70 + 0, 7.40 + 2.9, 3.50 + 6.20] = \max[11.70, 10.30, 9.70] = 11.70$

$d_7(60) = 0$

$M_7(40) = \max[M_8(40) + f_7(0), M_8(30) + f_7(10),\ M_8(20) + f_7(20),\ M_8(10) + f_7(30)]$

$= \max[7.40 + 0, 5.50 + 1.45, 3.50 + 2.90, 1.75 + 4.65\ ]$

$= \max[7.40, 6.95, 6.40, 6.40] = 7.40$

$d_7(40) = 0$

$M_7(30) = \max[M_8(30) + f_7(0),\ M_8(20) + f_7(10),\ M_8(10) + f_7(20)]$

$= \max[5.50 + 0, 3.50 + 1.45, 1.75 + 2.90] = \max[5.50, 4.95, 4.65] = 5.50$

$d_7(30) = 0$

$M_7(20) = \max[M_8(20) + f_7(0), M_8(10) + f_7(10)] = \max[3.50 + 0, 1.70 + 1.45\ ]$

$= \max[3.50, 3.15] = 3.50$

$d_7(20) = 0$

$M_7(10) = \max[M_8(10) + f_7(0)] = \max[1.75 + 0] = \max[1.75] = 1.75$

$d_7(10) = 0$

$M_7(0) = \max[f_7(0)] = \max[0] = 0$ $d_7(0) = 0$

From investment 6:

$M_6(600) = \max[M_7(600) + f_6(0),\ M_7(500) + f_6(80) + f_6(20),\ M_7(500) + f_6(60) + f_6(40),\ M_7(300) + f_6(300),\ M_7(200) + f_6(300) + f_6(80) + f_6(20),\ M_7(80) + f_6(500) + f_6(20),\ M_7(20) + f_6(500) + f_6(80),\ M_7(0) + f_6(600)]$

$= \max[129.00 + 0, 107.50 + 20.71 + 4.95, 107.50 + 15.53 + 10.13, 61.5 + 79.35,\ 41.00 + 79.35 + 20.71 + 4.95, 15.60 + 135.06 + 4.95, 3.5 + 135.06 + 20.71,\ 0 + 162.08]$

$= \max[129.00, 133.16, 133.16, 140.85, 146.01, 155.61.44, 159.27, 162.08] = 162.08$

$d_6(600) = 600$

$M_6(500) = \max[M_7(500) + f_6(0),\ M_7(300) + f_6(200),\ M_7(200) + f_6(300),\ M_7(0) + f_6(500)]$

$= \max[107.5 + 0, 61.50 + 52.90, 41.00 + 79.35,\ 0 + 156.89]$

$= \max[107.50, 114.40, 120.35, 156.89] = 156.89$

$d_6(500) = 500$

$M_6(300) = \max[M_7(300) + f_6(0), M_7(200) + f_6(80) + f_6(20),\ M_7(200) + f_6(60) + f_6(40), M_7(0) + f_6(300)]$

$= \max[61.5 + 0, 41.00 + 20.71 + 4.95, 41.00 + 15.53 + 10.13, 0 + 79.35]$

$= \max[61.50, 66.66, 66.66, 79.35] = 79.35$

$d_6(300) = 300$

$M_6(200) = \max[M_7(200) + f_6(0),\ M_7(80) + f_6(60) + f_6(40) + f_6(20), M_7(0) + f_6(200)]$

$= \max[41.00 + 0, 15.60 + 15.53 + 10.13 + 4.95, 52.90]$

$= \max[41.00, 46.21, 52.90] = 52.90$

$d_6(200) = 200$

$M_6(80) = \max[M_7(80) + f_6(0),\ M_7(60) + f_6(20),\ M_7(20) + f_6(60),\ M_7(0) + f_6(80)]$

$= \max[15.60 + 0, 11.70 + 4.95, 3.50 + 15.53, 0 + 20.71]$

$= \max[15.60, 16.65, 19.03, 20.71] = 20.71$

$d_6(80) = 80$

$M_6(60) = \max[M_7(60) + f_6(0),\ M_7(40) + f_6(20),\ M_7(20) + f_6(40), f_6(60)]$

$= \max[11.70, 7.40 + 4.95, 3.50 + 10.13, 15.53]$

$= \max[11.70, 12.35, 13.63, 15.53] = 15.53$

$d_6(60) = 60$

$M_6(40) = \max[M_7(40) + f_6(0), M_7(30) + f_6(10),\ M_7(20) + f_6(20),\ M_7(10) + f_6(30), f_6(40)]$

$= \max[7.40, 5.50 + 2.48, 3.50 + 4.95, 1.75 + 7.60, 10.13]$

$= \max[7.40, 7.98, 8.45, 9.35, 10.13] = 10.13$

$d_6(40) = 40$

$M_6(30) = \max[M_7(30) + f_6(0),\ M_7(20) + f_6(10),\ M_7(10) + f_6(20),\ f_6(30)]$

$= \max[5.50, 3.50 + 2.48, 1.75 + 4.95, 7.60] = \max[7.60,] = 7.60$

$d_6(30) = 30$

$M_6(20) = \max[M_7(20) + f_6(0),\ M_7(10) + f_6(10),\ f_6(20)]$

$= \max[3.50, 1.75 + 2.48,\ 4.95] = \max[3.50, 4.23, 4.95\ ] = 4.95$

$d_6(20) = 20$

$M_6(10) = \max[M_7(10) + f_6(0)] = \max[1.75 + 0] = \max[1.75] = 1.75$

$d_6(10) = 0$

$M_6(0) = \max[f_6(0)] = \max[0] = 0$

$d_6(0) = 0$

From investment 5:

$M_5(600) = \max[M_6(600) + f_5(0),\ M_6(500) + f_5(80) + f_5(20),\ M_6(500) + f_5(60) + f_5(40),$
$M_6(300) + f_5(300),\ M_6(200) + f_5(300) + f_5(80) + f_5(20),\ M_6(80) + \ f_5(500) + f_5(20),$
$M_6(20) + \ f_5(500) + f_5(80),\ M_5(0) + \ f_5(600)]$

$= \max[162.08, 156.89 + 8.62 + 2.05, 156.89 + 6.47 + 4.10, 79.35 + 33.91, 52.90 + 33.91 +$
$8.62 + 2.05, 20.71 + 61.80 + 2.05, 4.95 + 61.80 + 8.62, 74.16]$

$= \max[162.08, 167.56, 167.46, 113.26, 97.48, 84.56, 75.37, 74.16] = 167.56$

$d_5\ (600) = 80, 20$

$M_5(500) = \max[M_6(500) + f_5(0),\ M_6(300) + f_5(200),\ M_6(200) + f_5(300),\ M_6(0) +$
$f_5(500)]$

$= \max[156.89, 79.35 + 22.61, 52.90 + 33.91, 61.80]$

$= \max[156.89, 101.96, 86.81, 61.80] = 156.89$ $\qquad d_5(500) = 0$

$M_5(300) = \max[M_6(300) + f_5(0),\ M_6(200) + f_5(80) + f_5(20),\ M_6(200) + f_5(60) + f_5(40),$
$M_6(0) + f_5(300)]$

$= \max[79.35, 52.90 + 8.62 + 2.05, 52.90 + 6.47 + 4.10, 33.91]$

$= \max[79.35, 63.57, 63.47, 33.91] = 79.35$ $\qquad d_5(300) = 0$

$M_5(200) = \max[M_6(200) + f_5(0),\ M_6(80) + f_5(60) + f_5(40) + f_5(20), M_6(0) + f_5(200)]$

$= \max[52.90, 20.71 + 6.47 + 4.10 + 2.05, 22.61] = \max[52.90, 33.33, 22.61] = 52.90$

$d_5(200) = 0$

$M_5(80) = \max[M_6(80) + f_5(0),\ M_6(60) + f_5(20),\ M_6(20) + f_5(60),\ f_5(80)]$
$= \max[20.71, 15.53 + 2.05, 4.95 + 6.47, 8.62]$

$= \max[20.71, 17.58, 11.42, 8.62] = 20.71$

$d_5(80) = 0$

$M_5(60) = \max[M_6(60) + f_5(0),\ M_6(40) + f_5(20),\ M_6(20) + f_5(40), f_5(60)]$

$= \max[15.53, 10.13 + 2.05, 4.95 + 4.10, 6.47] = \max[15.53, 12.18, 9.05, 6.46] = 15.53$

$d_5(60) = 0$

$M_5(40) = \max[M_6(40) + f_5(0), M_6(30) + f_5(10),\ M_6(20) + f_5(20),\ M_6(10) + f_5(30), f_5(40)]$

$= \max[10.13, 7.60 + 1.03, 4.95 + 2.05, 1.75 + 3.08, 4.10] = \max[10.13, 8.63, 7.00, 4.83, 4.10] = 10.13$

$d_5(40) = 0$

$M_5(30) = \max[M_6(30) + f_5(0),\ M_6(20) + f_5(10),\ M_6(10) + f_5(20),\ f_5(30)]$

$= \max[7.60, 4.95 + 1.03, 1.75 + 2.05, 3.08] = \max[7.60, 5.98, 3.80, 3.08\ ] = 7.60$

$d_5(30)=0$

$M_5(20) = \max[M_6(20) + f_5(0),\ M_6(10) + f_5(10),\ f_5(20)]$

$= \max[4.95, 1.75 + 1.03, 2.05] = \max[4.95, 2.78, 2.05] = 4.95$

$d_5(20)=0$

$M_5(10) = \max[M_6(10) + f_5(0)] = \max[1.75 + 0] = \max[1.75] = 1.75$

$d_5(10) = 0$

$M_5(0) = \max[f_5(0)] = \max[0] = 0$ $\qquad d_5(0) = 0$

From investment 4:

$M_4(600) = \max[M_5(600) + f_4(0),\ M_5(500) + f_4(80) + f_4(20),\ M_5(500) + f_4(60) + f_4(40),$
$M_5(300) + f_4(300),\ M_5(200) + f_4(300) + f_4(80) + f_4(20),\ M_5(80) + \ f_4(500) + f_4(20),$
$M_5(20) + \ f_4(500) + f_4(80),\ M_5(0) + \ f_4(600)]$

$= \max[162.08, 156.89 + 12.88 + 2.79, 156.89 + 9.66 + 6.01, 79.35 + 51.54, 52.90 +$
$51.54 + 12.88 + 2.79, 20.71 + 94.05 + 2.79 , 4.95 + 94.05 + 12.88, 112.86]$

$= \max[162.08, 172.56, 172.56, 130.89, 120.11, 117.55, 111.88, 112.86] = 172.56$

$d_4(600) = 80,20\ OR\ 60,40$

$M_4(500) = \max[M_5(500) + f_4(0),\ M_5(300) + f_4(200),\ M_5(200) + f_4(300),\ M_5(0) +$
$f_4(500)]$

$= \max[156.89, 79.35 + 34.36, 52.90 + 51.54, 94.05]$

$= \max[156.89, 113.71, 104.44, 94.05] = 156.89$

$d_4(500) = 0$

$M_4(300) = \max[M_5(300) + f_4(0), M_5(200) + f_4(80) + f_4(20),\ M_5(200) + f_4(60) +$
$f_4(40), M_5(0) + f_4(300)]$

$= \max[79.35, 52.90 + 12.88 + 2.79, 52.90 + 9.66 + 6.01, 51.54]$

$= \max[79.35, 68.57, 68.57, 51.54] = 79.35$

$d_4(300) = 0$

$M_4(200) = \max[M_5(200) + f_4(0),\ M_5(80) + f_4(60) + f_4(40) + f_4(20), M_5(0) + f_4(200)]$

$= \max[52.90, 20.71 + 9.66 + 6.01 + 4.51, 34.36] = \max[52.90, 40.89, 34.36] = 52.90$

$d_4(200) = 0$

$M_4(80) = \max[M_5(80) + f_4(0),\ M_5(60) + f_4(20),\ M_5(20) + f_4(60),\ f_4(80)]$

$= \max[20.71, 15.53 + 2.79+, 4.95 + 9.66, 12.88] = \max[20.71, 18.32, 14.61, 12.88] = 20.71$

$d_4(80) = 0$

$M_4(60) = \max[M_5(60) + f_4(0),\ M_5(40) + f_4(20),\ M_5(20) + f_4(40), f_4(60)]$

$= \max[15.53, 10.13 + 2.79, 4.95 + 6.01, 9.66] = \max[15.53, 10.96, 11.94, 9.66] = 15.53$

$d_4(60) = 0$

$M_4(40) = \max[M_5(40) + f_4(0), M_5(30) + f_4(10),\ M_5(20) + f_4(20),\ M_5(10) + f_4(30),$
$f_4(40)] = \max[10.13, 7.60 + 1.40, 4.95 + 2.79, 1.75 + 4.51, 6.01]$
$= \max[10.13, 9.00, 7.74, 6.26, 6.01] = 10.13$

$d_4(40) = 0$

$M_4(30) = \max[M_5(30) + f_4(0),\ M_5(20) + f_4(10),\ M_5(10) + f_4(20),\ f_4(30)]$

$= \max[7.60, 4.95 + 1.40, 1.75 + 2.79, 4.51] = \max[7.60, 6.35, 4.54, 4.51] = 7.60$

$d_4(30) = 0$

$M_4(20) = \max[M_5(20) + f_4(0),\ M_5(10) + f_4(10),\ f_4(20)]$
$= \max[4.95, 1.75 + 1.40, 2.79] = \max[4.95, 3.15, 2.79] = 4.95$

$d_4(20) = 0$

$M_4(10) = \max[M_5(10) + f_4(0)] = \max[1.75 + 0] = \max[1.75] = 1.75$ $d_4(10) = 0$

$M_4(0) = \max[f_4(0)] = \max[0] = 0$

$d_4(0) = 0$

From investment 3:

$M_3(600) = \max[M_4(600) + f_3(0), M_4(500) + f_3(80) + f_3(20),\ M_4(500) + f_3(60) + f_3(40),$
$M_4(300) + f_3(300), M_4(200) + f_3(300) + f_3(80) + f_3(20),\ M_4(80) + \ f_3(500) + f_3(20),$
$M_4(20) + \ f_3(500) + f_3(80),\ M_4(0) + f_3(600)]$

$= \max[162.08, 156.89 + 17.89 + 4.24, 156.89 + 14.12 + 8.95, 79.35 + 74.12, 52.90 +$
$74.12 + 17.89 + 4.24, 20.71 + 129.45 + 4.24, 4.95 + 129.45 + 17.89\ , 155.34]$

$= \max[162.08, 179.02, 179.96, 153.47, 149.15, 154.40, 152.29, 155.34\ ] = 179.96$

$d_3(600) = 60{,}40$

$M_3(500) = \max[M_4(500) + f_3(0),\ M_4(300) + f_3(200),\ M_4(200) + f_3(300),\ M_4(0) +$
$f_3(500)]$

$= \max[156.89, 79.35 + 49.41, 52.90 + 74.12, 129.45] =$
$\max[156.89, 128.76, 127.02, 129.45] = 156.89$

$d_3(500) = 0$

$M_3(300) = \max[M_4(300) + f_3(0), M_4(200) + f_3(80) + f_3(20),\ M_4(200) + f_3(60) + f_3(40),$
$M_4(0) + f_3(300)]$

$= \max[79.35, 52.90 + 17.89 + 4.24, 52.90 + 14.12 + 8.95, 74.12]$

$= \max[79.35, 75.03, 75.97, 74.12] = 79.35$ $d_3(300) = 0$

$M_3(200) = \max[M_4(200) + f_3(0),\ M_4(80) + f_3(60) + f_3(40) + f_3(20),\ M_4(0) + f_3(200)]$

$= \max[52.90, 20.71 + 14.12 + 8.95 + 4.24, 49.41] = \max[52.90, 48.02, 49.41] = 52.90$

$d_3(200) = 0$

$M_3(80) = \max[M_4(80) + f_3(0),\ M_4(60) + f_3(20),\ M_4(20) + f_3(60),\ f_3(80)]$

$= \max[20.71, 15.53 + 4.24, 4.95 + 14.12, 17.89] = \max[20.71, 19.77, 19.07, 17.89] = 20.71$

$d_3(80) = 0$

$M_3(60) = \max[M_4(60) + f_3(0),\ M_4(40) + f_3(20),\ M_4(20) + f_3(40), f_3(60)]$

$= \max[15.53, 10.13 + 4.24, 4.95 + 8.95, 14.12] = \max[15.53, 14.37, 13.90, 14.12] = 15.53$

$d_3(60) = 0$

$M_3(40) = \max[M_4(40) + f_3(0), M_4(30) + f_3(10),\ M_4(20) + f_3(20),\ M_4(10) + f_3(30),$
$f_3(40)]$

$= \max[10.13, 7.60 + 2.12, 4.95 + 4.24, 1.75 + 6.71, 8.95]$

$= \max[10.13, 9.72, 9.19, 8.46, 8.95] = 10.13$

$d_3(40) = 0$

$M_3(30) = \max[M_4(30) + f_3(0),\ M_4(20) + f_3(10),\ M_4(10) + f_3(20),\ f_3(30)]$

$= \max[7.60, 4.95 + 2.12, 1.75 + 4.24, 6.71] = \max[7.60, 7.07, 5.99, 6.71] = 7.60$

$d_3(30) = 0$

$M_3(20) = \max[M_4(20) + f_3(0),\ M_4(10) + f_3(10),\ f_3(20)]$

$= \max[4.95, 1.75 + 2.12,\ 4.24] = \max[4.95, 3.87, 4.24] = 4.95$

$d_3(20) = 0$

$M_3(10) = \max[M_4(10) + f_3(0)] = \max[1.75 + 0] = \max[1.75] = 1.75$

$d_3(10) = 0$

$M_3(0) = \max[f_3(0)] = \max[0] = 0$

$d_3(0) = 0$

From investment 2:

$M_2(600) = \max[M_3(600) + f_2(0),\ M_3(500) + f_2(80) + f_2(20),\ M_3(500) + f_2(60) + f_2(40),$
$M_3(300) + f_2(300),\ M_3(200) + f_2(300), + f_2(80) + f_2(20),,\ M_3(80) +\ f_2(500) + f_2(20),$
$M_3(20) +\ f_2(500) + f_2(80),\ M_3(0) +\ f_2(600)]$

$= \max[162.08, 156.89 + 7.88 + 1.97, 156.89 + 6.23 + 3.94, 79.35 + 32.76, 52.90 + 32.76 +$
$7.88 + 1.97, 20.71 + 60.03 + 1.97, 4.95 + 60.03 + 1.97, 72.03]$

$= \max[\ 162.08, 166.74, 167.06, 112.11, 95.51, 82.71, 66.95, 72.03] = 167.06$

$d_2(600) = 60{,}40$

$M_2(500) = \max[M_3(500) + f_2(0),\ M_3(300) + f_2(200),\ M_3(200) + f_2(300),\ M_3(0) +$
$f_2(500)]$

$= \max[156.89, 79.35 + 21.84, 52.90 + 32.76, 60.02] = \max[156.89, 101.19, 85.66, 60.02] =$
156.89

$d_2(500) = 0$

$M_2(300) = \max[M_3(300) + f_2(0),\ M_3(200) + f_2(80) + f_2(20),\ M_3(200) + f_2(60) + f_2(40),$
$M_3(0) + f_2(300)]$

$= \max[79.35, 52.90 + 7.88 + 1.97, 52.90 + 6.23 + 3.94, 32.76]$

$= \max[79.35, 62.75, 63.07, 32.76] = 79.35$

$d_2(300) = 0$

$M_2(200) = \max[M_3(200) + f_2(0), M_3(80) + f_2(60) + f_2(40) + f_2(20),\ M_3(0) + f_2(200)]$

$= \max[52.90, 20.71 + 6.23 + 3.94 + 1.97, 21.84] = \max[52.90, 32.85, 21.84] = 52.90$

$d_2(200) = 0$

$M_2(80) = \max[M_3(80) + f_2(0),\ M_3(60) + f_2(20),\ M_3(20) + f_2(60),\ f_2(80)]$

$= \max[20.71, 15.53 + 1.97, 4.95 + 6.23, 7.88]$
$= \max[20.71, 17.50, 11.18, 7.88] = 20.71$

$d_2(80) = 0$
$M_2(60) = \max[M_3(60) + f_2(0),\ M_3(40) + f_2(20),\ M_3(20) + f_2(40), f_2(60)]$

$= \max[15.53, 10.13 + 1.97, 4.95 + 3.94, 6.23\ ] = \max[19.77, 12.10, 8.89.6.23] = 15.53$

$d_2(60) = 0$

$M_2(40) = \max[M_3(40) + f_2(0), M_3(30) + f_2(10),\ M_3(20) + f_2(20),\ M_3(10) + f_2(30),$
$f_2(40)]$

$= \max[10.13, 7.60 + 0.98, 4.95 + 1.97, 1.75 + 2.95, 3.94]$

$= \max[10.13, 8.58, 6.92, 4.70, 3.94] = 10.13$

$d_2(40) = 0$

$M_2(30) = \max[M_3(30) + f_2(0),\ M_3(20) + f_2(10),\ M_3(10) + f_2(20),\ f_2(30)]$

$= \max[7.60, 4.95 + 0.98, 1.75 + 1.97, 2.95] = \max[7.60, 5.93, 3.72, 2.95] = 7.60$

$d_2(30) = 0$

$M_2(20) = \max[M_3(20) + f_2(0),\ M_3(10) + f_2(10),\ f_2(20)]$

$= \max[4.95, 1.75 + 0.98, 1.97] = \max[4.95, 2.73, 1.97] = 4.95$

$d_2(30) = 0$

$M_2(10) = \max[M_3(10) + f_2(0)] = \max[1.75 + 0] = \max[1.75] = 1.75$

$d_2(10) = 0$

$M_2(0) = \max[f_2(0)] = \max[0] = 0$ $\qquad d_2(0) = 0$

From investment 1:

$M_1(600) = \max[M_2(600) + f_1(0),\ M_2(500) + f_1(80) + f_1(20),\ M_2(300) + f_1(300),\ M_2(200) + f_1(300) + f_1(80) + f_1(20),\ M_2(80) + f_1(500) + f_1(20),\ M_2(20) + f_1(500) + f_1(80),\ M_2(0) + f_1(600)]$

$= \max[162.08, 156.89 + 14.95 + 3.17, 79.35 + 61.19, 52.90 + 61.19 + 14.95 + 3.17, 20.71 + 113.56 + 3.17, 4.95 + 113.56 + 14.95, 136.27]$

$= \max[162.08, 175.01, 140.54, 132.21, 137.44, 133.46, 136.27] = 175.01$

$d_1(600) = 80{,}20$

ALLOCATION

The optimal return from the investment is 179.96 which we obtained by starting the allocation from stage 1, then to stage 2 up to stage 9 as follows:

i. With 600 units available, allocate to stage 1, $d_1(600) = 0$, leaving 600 – 0 = 600. ii. With 600 units, available allocate to stage 2, $d_2(600) = 0$, leaving 600 – 0 = 600.

iii. With 600 units, available allocate to stage 3, $d_3(600) = 60 \text{ \& } 40$,

leaving 600 – 100 = 500.

iv. With 500 units, available allocate to stage 4, $d_4(500) = 0$, leaving 500 – 0 = 500.

v. With 500 units, available allocate to stage 5, $d_5(500) = 0$, leaving 500 – 0 = 500. vi. With 500 units, available allocate to stage 6, $d_6(500) = 500$, leaving 500 – 500 = 0.

vii. With 0 units, available allocate to stage 7, $d_7(0) = 0$, leaving 0 – 0 = 0. viii. With 0 units, available allocate to stage 8, $d_8(0) = 0$, leaving 0 – 0 = 0. ix. With 0 units, available allocate to stage 9, $d_9(0) = 0$, leaving 0 – 0 = 0.

TABLE 4.6: OPTIMAL ALLOCATION OF GHC 600

INVESTMENT	AMOUNT INVESTED (GHC'000)										
	0	10	20	30	40	60	80	200	300	500	600
$f_1(x)$	0.00	1.59	3.17	5.60	7.47	11.83	14.95	40.79	61.19	113.56	136.27
$f_2(x)$	0.00	0.98	1.97	2.95	3.94	6.23	7.88	21.84	32.76	60.03	72.03
$f_3(x)$	0.00	2.12	4.24	6.71	8.95*	14.12*	17.89	49.41	74.12	129.45	155.34
$f_4(x)$	0.00	1.40	2.79	4.51	6.01	9.66	12.88	34.36	51.54	94.05	112.86
$f_5(x)$	0.00	1.03	2.05	3.08	4.10	6.47	8.62	22.61	33.91	61.80	74.16
$f_6(x)$	0.00	2.48	4.95	7.60	10.13	15.53	20.71	52.90	79.35	156.89*	162.08
$f_7(x)$	0.00	1.45	2.90	4.65	6.20	9.90	13.20	36.00	54.00	91.25	109.50
$f_8(x)$	0.00	1.10	2.20	3.30	4.40	6.90	9.20	24.00	36.00	67.50	81.00
$f_9(x)$	0.00	1.75	3.50	5.55	7.40	11.70	15.60	41.00	61.50	107.50	129.00

CHAPTER FIVE

SUMMARY OF FINDINGS, CONCLUSIONS AND RECOMMENDATIONS

5.0 INTRODUCTION

This chapter presents the summary of findings, conclusions and recommendations of the study.

5.1 SUMMARY OF FINDINGS

Table 4.4 shows how the ¢600,000 investment fund should be allocated to achieve the optimal allocation. The table shows that with ¢600,000 available for investment and given the corresponding annual fixed deposits returns for the three financial institutions for the various periods, the investor should invest only in the Fidelity Bank. The solutions shows that ¢40,000 and ¢60,000 should be placed in the 91-day fixed deposits while ¢500, 000 should be deposited in the 182-day fixed deposits. The optimal return from the investment is ¢179.96

5.2 CONCLUSIONS

Generally the annual fixed deposit is not in the interest of the investor. The results show clearly that with the current rate of fixed deposits, Fidelity Bank pays more to the investor than the other Banks; Ghana Commercial Bank and Barclays Bank. It is also beneficial to allocate resources to 91-day and 182-day fixed deposits.

5.3 RECOMMENDATIONS

I would like to recommend that one of the most important and necessary things one can do for one's financial wellbeing is to start investing. Once the habit is cultivated or instilled, it automatically develops and becomes much easier for a person to invest on a regular basis. This gives one the benefit of enjoying a higher standard of living for roughly the same amount of work.

Although people may have variety of reasons for investing their monies, I am recommending the following:

One reason why you have to begin investing is for your future education. Each year more people return to school to earn their masters or doctorate degrees. You may also consider investing for your children's educational costs to enable them climb up the educational ladder.

Another important reason why you must invest is for your retirement. The sooner you start investing for retirement, the less you will have to invest in the future.

It is also important to have emergency funds set aside to cover unexpected expenses or anticipated major expenses, which could be a sudden job loss or an unexpected car repair or medical expenses.

Finally monies invested can enable one to start his/her own business to support the economy.

REFERENCES

Amponsah, S. K (2011). Optimization Techniques I

Albota, M. A, Chen, L. W, Fan, A, Fregfogle E, Grover, J, Schouwenaars, T and Sun, W (2005). Optimal Rebalancing Strategy for Institutional Portfolios

Bellman, R. E (1957). Dynamic Programming N. J. Princeton University Press

Bertsimas, D and Wlo, A (1999). Optimal Control of Execution Costs for Portfolios

Dijkhuizen, A. A, Hurne R. B. M, Beck, P, Van, Hendricks, Th G. B. (1993). European Journal of Operational Research

Farmer, J (2008). Introduction to Dynamic Programming

Ghezzi, L. L (1997). Bond Management and Max – Min Optimal Control

Greco, S (1990). Dynamic Programming In Datalog With Aggregates

Guangliang, He, (1999). Drawdown Controlled Optimal Portfolio Selection With Linear Constraints On Portfolio Weights

Jacobson, D (2003). Differential Dynamic Programming Methods For Solving Bang – Bang Control Problems

Legal and General (Portfolio Management Services) Limited, Registered in England and Wales No. 02457525 ; Registered Office : One Coleman street, London EC2R5AA. Investing, An Informed Decision

Mohan, L. T (1995). Banking Law and Practice In India

Malkiel, B . G (2008). Interest Rates

Robinson, R. C (2013). Introduction to Mathematical Optimization

Sylla, R (1996). A history of Interest Rates

Sumant, K. M (1952). Modern Banking In India

Tenney, M. S (1995). Dynamic Dynamic Programming Solutions For The Portfolio of Risky Assets

Wright, J. S (2014). Optimization (Mathematics)

Wayne, L. W (1994). Operations Research Applications and Algorithms. Duxbury Press

Zariphopoulou, T and Vila, J. L (1991). Optimal Consumption and Portfolio Choice with Borrowing Constraints

What is investment (2013) : http//finance.mapsofworld.com/investment, http//yourdictionary.com/investment, http//financial- dictionary .the free dictionary.com/investment, http//www.investopedia.com/university/beginner1.asp (2013)

Dynamic Programming, from Wikipedia, the free encyclopedia – http//en.wikipedia.org/wiki/dynamic programming (2015)