KNAPSACK ALGORITHM

A CASE STUDY OF GARDEN CITY RADIO

(A LOCAL RADIO STATION IN KUMASI)

BY

KWAME OFOSUHENE PEASAH (BSc COMPUTER SCIENCE)

A thesis submitted to the Department of Mathematics,

Kwame Nkrumah University of Science and Technology

In partial fulfillment of the requirement for the degree of

MASTER OF SCIENCE

Faculty of Physical Science, College of Science

December 2009

DECLARATION

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

Kwame Ofosuhene Peasah, PG1837407 ………………… …………………………

Student's Name & Number Signature Date

Certified By

Dr S. K. Amponsah ………………….. ………………

(Supervisor) Signature Date

Certified by

Dr S. K. Amponsah ………………….. ………………

(Head of Department's Name) Signature Date

DEDICATION

To:

To my darling wife, Bernice and my four children; Emmanuella, Isabella, Eirena and Edwyna. I thank you for the support, understanding and your sacrifices. I love you dearly.

To my supervisor and his wife, Dr and Mrs S. K. Amponsah. You are wonderful couple. God bless you. May your marriage continue to be sweet and always be an example to us.

Finally, to the Lord Jesus for having mercy on me and giving me wisdom to carry out this project.

ACKNOWLEDGEMENT

My heart is full of thanks and gratitude to the Almighty God for granting me wisdom, strength, good health and seeing me through my education.

Secondly, I am very grateful to Dr S. K. Amponsah who read this thesis and pointed out all the mistakes. Dr S. K. Amponsah did not only supervise this project, he also laid bear his rich experience for me to tap. He was also a source of motivation and encouragement to me.

I also want to acknowledge all lectures of Mathematics and Computer Science Department especially Dr F. T. Oduro who encouraged me and assisted in diverse ways.

The staff and management of Garden City radio cannot be left out in this acknowledgement. Thank you so much for your kindness.

I am also indebted to my parents and wife who encourage me to do this course.

I owe special thanks to Eric Warren Acquah who help me in typing.

May God bless you.

ABSTRACT

The knapsack or rucksack problem is a problem in combinatorial optimization. It derives its name from the maximization problem of the best choice of essentials that can fit into one bag to be carried on a trip. Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than a given limit and the total value is as large as possible. The decision problem of the knapsack problem is the question "can a value of at least V be achieved without exceeding the weight W?" The problem often arises in resource allocation with constraints.

The practice of selecting commercials to be played on air from a pile of commercials in most radio stations to generate the maximum revenue given a fixed time is a clear case of Knapsack Problem.

In this project, we shall explore ways of effectively and efficiently selecting commercials from a pile of commercials within a fixed time to achieve optimal use of air time in order to maximize space and airtime in the FM stations in Ghana, using knapsack algorithm and also to develop a software for the knapsack algorithm using Visual Basic dot NET, which can be used by any researcher or radio station. The software will also be modeled to solve many industrial problems: capital budgeting, cargo loading, cutting stock, to mention the most classical applications. Our project will also serve as reference material in the libraries and the internet for students who wish to undertake research into the piled-up problems of commercials and programmes. The project may also serve as a guide for policy makers and the general public. In this project, different algorithm used to solve Knapsack problem is briefly discussed but the Heuristic Scheme will be used to solve the Knapsack Problem.

CHAPTER ONE

1.0 INTRODUCTION

Packing is action of putting things together, especially of putting clothes into a suitcase for a journey or surround with something crammed tightly. Packing problem forms integral part in a man's life and cannot be ignored outright. Almost everyone is involved in packing. Packing is one of the most important aspects of household moving, but it is often overlooked. Items need to be well packed and protected to survive the move unscathed. When it is done efficiently, at least to its optimal level, space and time are saved. A room may become spacious when furniture, bags and other household items are packed very well. The end result is beauty and attractiveness. Data structures, packing can be defined as the way data is arranged and accessed in computer memory. It consists of two separate but related issues: data alignment and data structure padding. When modern computers read from or write to a memory address, it will do this in word sized chunks (e.g. 4 byte chunks on a 32-bit system). Data alignment means putting the data at a memory offset equal to some multiple of the word size, which increases the system's performance due to the way the CPU handles memory. To align the data, it may be necessary to insert some meaningless bytes between the end of the last data structure and the start of the next, which is data structure padding.

1.1 BACKGROUND STATEMENT

1.1.1 HISTORY OF RADIO AND RADIO BROADCASTING

Radio is the transmission of signals by modulation of electromagnetic waves with frequencies below those of visible light. Electromagnetic radiation travels by means of oscillating electromagnetic fields that pass through the air and the vacuum of space. Information is carried by systematically changing (modulating) some property of the radiated waves, such as amplitude, frequency, or phase. When radio waves pass an electrical conductor, the oscillating fields induce an alternating current in the conductor. This can be detected and transformed into sound or other signals that carry information.

In 1896, Marconi was awarded the British patent 12039, Improvements in transmitting electrical impulses and signals and in apparatus there-for, for radio. In 1897 he established the world's first radio station on the Isle of Wight, England. Marconi opened the world's first "wireless" factory in Hall Street, Chelmsford, England in 1898, employing around 50 people [1].

Originally, radio or radiotelegraphy was called "wireless telegraphy", which was shortened to "wireless" by the British. The prefix *radio-* in the sense of wireless transmission, was first recorded in the word *radioconductor,* coined by the French physicist Édouard Branly in 1897 and based on the verb *to radiate* (in Latin "radius" means "spoke of a wheel, beam of light, ray"). "Radio" as a noun is said to have been coined by the advertising expert Waldo Warren (White 1944). This word also appears in a 1907 article by Lee De Forest, was adopted by the United States Navy in 1912 and became common by the time of the first commercial broadcasts in the United States in the 1920s. (The noun "broadcasting" itself came from an agricultural term, meaning "scattering seeds" widely.) The term was then adopted by other languages in Europe and Asia. British Commonwealth countries continued to mainly use the term "wireless" until the mid-20th century, though the magazine of the British Broadcasting Cooperation (BBC) in the United Kingdom has been called Radio Times ever since it was first published in the early 1920s.

In recent years the term "wireless" has gained renewed popularity through the rapid growth of short-range computer networking, e.g., Wireless Local Area Network (WLAN), WiFi, and Bluetooth, as well as mobile telephony, e.g., Global System for Mobile communication (GSM) and Universal Mobile Telecommunications System (UMTS). Today, the term "radio" often refers to the actual transceiver device or chip, whereas "wireless" refers to the system and/or method used for radio communication, hence one talks about radio transceivers and Radio Frequency Identification (RFID), but about wireless devices and wireless sensor networks [2].

Radio stations broadcast in only one way, from station to the radio device. Thus one can listen to radio but one cannot talk back. Two-way radio lets people talk to each other on radio waves. This most often used by the police, firefighters and the military officers.

1.1.2 HISTORY OF RADIO BROADCASTING IN GHANA

Broadcasting in Ghana began as a department of the Ministry of Information when it started in 1935. The Ministry was responsible for the formulation of national mass communication policies and for ensuring the full and effective use of the mass media for the dissemination of information, and for economic and social development of the nation.

Radio Broadcasting was first established in Ghana (then known by its colonial name of Gold Coast) in 1935 with approximately 300 subscribers in Accra. The number was low because radio sets were then rare and expensive, and was the privilege of only a rich few, mainly the expatriate community who had come from countries that already had these mass communication facilities. The brain behind the introduction of broadcasting into the country was the then Governor of the Gold Coast, Sir Arnold Hodson.

Broadcasting began in Ghana essentially as a relay service, re-broadcasting programmes from the BBC World Service. A year later, the service began to expand and a re-diffusion station was opened in Cape Coast, the Central Regional capital to cater for that part of the country. Three more stations were opened the following year and a new broadcasting house built in Accra during the Second World War in 1940. It had a small 1.3KW transmitter, with which transmissions could be broadcast to neighbouring institutions. During the 1940s, broadcasting began in four of the major Ghanaian languages - Twi, Fanti, Ga and Ewe [3].

In 1952, the then colonial government appointed a commission to advise it on ways of improving and developing broadcasting. It was to investigate among other things the establishment and maintenance of a statutory corporation to assume direction and control of broadcasting services as was the case in parent country Britain. As a result of the Commission's report, a new broadcasting system, the national service of the Gold Coast Broadcasting System was set up in 1954.

Broadcasting became a new department distinct from the Information Services to which it had previously been attached. Broadcast content at this time was mainly governmental announcements and rebroadcasts from the BBC. From 1956, locally produced programmes increased, educational broadcasts to schools and teacher training colleges were started and outside events were broadcast live into homes. When the Gold Coast became Ghana in 1957, the Gold Coast Broadcast System became the Ghana Broadcasting System, or as it was popularly known as Radio Ghana. Mass Communication was embraced as a way of changing society. Broadcasting in Ghana was thus to be a public service dedicated to the enlightenment and instruction of the people. Taking into consideration that its main model was the BBC, the pioneer of public service broadcasting, it was no surprise that the public service model was adopted from the onset.

Ghana foray into the international broadcasting scene began when in 1958 the government set up another commission to advise it on launching an external service of Radio Ghana; the External Service was inaugurated in June 1961 as a result. This was an unusually bold step for a newly independent country with a less than mature broadcasting service. The external service came about primarily because Kwame Nkrumah, the then prime minister saw broadcasting as an opportunity to propagate his pan African message to his fellow Africans. At the same time, television was being considered and GBC Television Service was launched on 31 July 1965. In 1997, GBC entered into an agreement with World Space to provide GBC with a channel on its Afristar satellite. This capability enabled GBC to provide a 24-hour, Direct Digital Broadcasting (DDB) service over a coverage area of 14m sq km, encompassing millions of radio listeners [5]. Today, due to deregulation, availability of technology and a shift in market economy, there are five television stations in Ghana and at least seventy radio stations. Broadcasting has been privatized and commercialized bringing with it the attendant competition, issues of regulation of content and of operation.

This paper set out to study the state of broadcasting in Ghana at present. The major terms of reference included news broadcasting, feedback, information sources, equipment and personnel; it also set out to look at change and development in the system. As an industry case study, this was done through interviews as well as observing people at work. Before the method is looked at in detail, it is imperative to look at the theories underpinning this study.

1.1.3 PRIVATE RADIO AND FM STATION IN GHANA

In early 1992, the urge to utilize the airwaves became more urgent than ever before. With the struggle to the country to democracy, the government at the time was hedging as to how to grant license for Frequency Modulation (FM) operation in Ghana. Its was in the midst of the Provisional National Defence Council's reluctance to issue the license for the introduction of FM stations that Charles Wereko-Brobbey "illegally" started operating 'Radio Eye' in the country much against the laws of the land (Public Agenda, 2005-01-25). Following this situation, the government started hunting down 'Radio Eye' until its operation stopped and the brain behind the station dragged to court and the equipment confiscated. Seeing the loss of political points in the radio 'Radio Eye' saga, the Ghana Broadcasting Corporation nicodemously influenced the establishment of Joy FM to operate under the aegis of the National radio without any due process through the National Communication Authority (NCA). This was in 1995 (Ghanaimage.com, 2005).

1.2 TYPES OF RADIO STATIONS

Radio stations are of several types. The best known are the A.M and FM stations (Wikipedia, free encyclopedia); these include commercial, public and non-profit varieties, as well as student-run campus radio stations found throughout the developed world. Now been eclipsed by internet distributed radio, there are many radio stations that broadcast on short waves band using A.M technology that can be recovered over thousands of miles. For example, the British Broadcasting Corporation has a full schedule transmitted via short wave. These broadcasts are very sensitive to atmospheric conditions and sunspot.

1.2.1 AMPLITUDE MODULATION (AM)

A.M stations were the earliest broadcasting stations to be developed. A.M. refers to Amplitude Modulation, a mode of broadcasting radio waves by varying the amplitude of career signals in response to the amplitude of the signal to be transmitted. One of the advantages of A.M is that, its unsophisticated signal can be detected, (tuned into sound) with simple equipment. If a signal is strong enough, not even a power source is needed; building an unpowered crystal radio receiver was a common childhood project in the early years of radio. AM stations were the earliest broadcasting stations to be developed. An AM receiver detects amplitude variations in the radio waves at a particular frequency. It then amplifies changes in the signal voltage to drive a loudspeaker or earphones. One of the advantages of AM is that, its unsophisticated signal can be detected, turned into sound with simple equipment. If a signal is strong enough, not even a power source is **needed.**

1.2.2 FREQUENCY MODULATION (FM)

Frequency modulation (FM) conveys information over a carrier wave by varying its frequency (contrast this with amplitude modulation, in which the amplitude of the carrier is varied while its frequency remains constant). In analog applications, the instantaneous frequency of the carrier is directly proportional to the instantaneous value of the input signal. Digital data can be sent by shifting the carrier's frequency among a set of discrete values, a technique known as frequency-shift keying.(Wikipedia, free encyclopedia). FM was invented by Edwin H. Armstrong in 1930s for the purpose of overcoming interferences [Wikipedia, free encyclopedia]

1.2.3 DIGITAL AND SATELITE RADIO

Digital and satellite radio are slowly emerging, but the enormous entry cost of space based satellite transmitters are restrictions on available radio spectrum licenses have restricted the growth of this market. Many other non-broadcasting types of radio stations exist. These include base stations for police, fire and ambulance networks, military base stations, dispatch base stations for taxi, trucks and careers, emergency broadcasting systems, amateur radio stations.

Radio has indeed contributed an appreciable percentage to the saying that the world is a global village. Global village in the sense that, journalists gather, evaluate and disseminate information of current interest and by the power of radio wave propagation and its widespread, people within a community, country, continent are fed with such information within the shortest possible time.

FM stations, due to their low running cost as compared to television and high patronage as compared to newspaper and even television, are able to attract a whole lot of audience. With the availability of portable radio sets one can listen to radio everywhere. In our offices, cars, market places, farms, etc the radio is seen as the friendliest medium educating, entertaining and giving braking news and also creating awareness. One may ask what then their source of funding is. They rely mostly on commercials or advertisements, sponsorships and advocacy. With the high increase of radio station in the country, the competition among these FM stations has become very keen and getting a lot of adverts to play a day to fund the station has become dependent on so many factors. Among these factors are getting a very high power antenna, having highly qualified presenters, knowing your target groups and coming out with programmes that can pull audience to the station. If all these are put in place, there will be a high probability of the inflows of adverts/commercials shooting up to the extent of pilling up. This pile-up comes as a result of so many customers wanting to advertise on particular programme and at a particular time. One can just image if a station has catchy programmes at all times. Many people would want to come in to sponsor, want their adverts to be played; for they know that is when they are going to get audience for their products and services.

This in fact becomes a problem especially when these programmes are also not well arranged. Bin Packing, an application that has recently gained grounds in the field of Mathematics has come to help solve some of these packing problems to ensure that the setbacks of FM stations, i.e. their inability to play all adverts/commercials because of poor packing of programmes, may be addressed. Their goal is to arrange programmes in such a way that optimal arrangement of programmes is attained within a fixed frame. This is the aim of this research.

1.3 PROBLEM STATEMENT

A thief robbing a store finds *n* items; the *i*th item is worth v_i Ghana cedis and weights w_i pounds. He wants to take as valuable a load as possible, but he can carry at most W pounds in his knapsack. Here v_i, w_i and W are all positive integers. What items should be taken?

Again, imagine radio station wanting to broadcast *n* programmes. Each programme worth v_i Ghana cedis and a duration (time to broadcast) of w_i minutes. The station wants to broadcast as valuable programmes as possible, but they can broadcast at most W programmes in a day. Here v_i, w_i and W are all positive integers. Which programme should be broadcast?

This is called the 0-1 knapsack problem because each item must either be taken or left behind; the thief cannot take a fractional amount of an item or take an item more than once. The knapsack problem is an abstraction of many real problems, from investing to telephone routing.

1.4 KNAPSACK ALGORITHM

The knapsack problem or rucksack problem is a problem in combinatorial optimization. It derives its name from the following maximization problem of the best choice of essentials that can fit into one bag to be carried on a trip. Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than a given limit and the total value is as large as possible.

The decision problem form of the knapsack problem is the question "can a value of at least *V* be achieved without exceeding the weight *W*?"

1.5 OBJECTIVES

The main objective of this study is to explore ways of effectively and efficiently Select radio adverts to be played on air within a fixed time to maximize airtime in the FM stations in Ghana using knapsack algorithm and also to develop a software for the knapsack algorithm using Visual basic dot Net which can be used by any researcher or radio station. The software will be model to many industrial situations: capital budgeting, cargo loading, cutting stock, to mention the most classical applications. This work will also serve as reference material in the libraries and the internet for students who wish to undertake research into the piled-up problems of commercials and programmes. The work may also serve as a guide for policy makers and general public.

1.6 JUSTIFICATION

Radio, which is one of the fastest and highly economical media, considering what it can do as compared to other media like newspapers and television, has come to stay with us. Its impact in man's life and the entire society is just remarkable. It educates, entertains, creates employment, advertises and carries braking news etc. All these affect the life of man in one-way or another. Due to its great impact on man's life and the society as a whole, it is necessary that scientific measure be put in place to see to its continued growth. The major source revenue of these FM station happens to be advertised and advocacy. The number of people who may advertise on a station is dependent on many factors. Among these factors are

i. The status of the station. The status is determined by the type of people who listen to the station and as such the type of program and eventually the type of adverts

ii. The target group, For example, if farmers are the target group then the station must be loud and clear in near-by villages or farming communities

iii. The quality of presentation, well-trained presenters always come out big or able to give out their best

iv. The audience population level, the number of the people who tune in to a station also counts. The more audience you have the greater the probability of you getting adverts to play.

Notwithstanding all, the number of people who may listen to a station or bring adverts may not only depend on the station but also on the type of program, presenter or host and lot of other factors. These attributes sometimes bring about the unequal sharing of customers; making some station especially the so-called growing ones have more customers. This creates piled-up commercials or program in our radio station.

Commercials and advocacies may become piled-up due to poor packing of adverts or programs. Customers send adverts requesting that their adverts be played at particular time's particular times and on particular programs. These constraints if not satisfied, sometimes bring about piled-ups commercials.

The problem of piled-up commercials may result comes as result of some commercials not being aired at the right times. Immediately this happens, some uncompromising customers may looses trust in the station; some may result in law suits for breach of contract such that if pragmatic policies and scientific approaches are not used to salvage this situation the end result may be the collapse of the station, which may be a big blow to the society as a whole. This is because the society would be denied of the benefits it used to get from the station. This problem off piled-ups programs was recognized in the late 1990's and an attempt to reduce this brought about the establishment of radio stations in each regional capital and licenses were given to individuals to establish FM stations. As earlier explained, this has not been very effectives because the sharing of adverts has not been even. This accounts for the reason why there are about 21 station in Kumasi (with population of about Three hundred and ninety-nine thousand, three hundred 399,300(1990 estimate) but still the problem of piled-up commercials and its associate problems still exist. This has made the study worthwhile.

1.7 ADVANTAGES OF FM STATIONS

Radio broadcast in Ghana since its liberalization in July 1995 by the then National Democratic Congress (N.D.C) government, has seen a tremendous upsurge in its popularity among the citizens of Ghana. The initial fear of the government at that time was the potential alertness of government policies. This was known to have been the cause of much mayhem in this country, prominent among them being the seizure of Charles Wereko-Brobbey's equipment when he first came out with radio Eye in 1994 by the then N.D.C government.

Although radio can be disastrous even to the extent causing civil wars as it did in Rwanda in Eastern Africa in the mid 90's, more especially when presenters are not well trained and as such insensitive to issue that have the potential ignite violence and destabilize an entire region, it has many advantages. Among these are;

i. Radio has helped in the forging of a common sense of national identity. Contrill and Allpot referred to this integrative role of radio when they wrote "when a million people have the same subject matter, the same argument and appeals, the same music and humor, when their attention is held in the same way and at the same time to the same stimuli, it is psychologically inevitable that they should acquire in some degree common interest, common taste and common attitude (Contrill and Allpot G.W., 1935).

ii. Radio has fostered and facilitated unity among the people of this country. Certain educative and entertaing programmes attract families and friends and the entire listening public of a particular radio station. They become glued to their sets and listen to common jokes, arguments, etc. The end result may be unity among the listeners.

iii. The FM stations give gainful employments to our secondary school graduates who for one reason or another heave to suspend further education. Thus they alleviate unemployment in the country.

iv. Production companies and other business communities have had the largest share of the benefits of radio through advertisement. They make their products and services reach their customers through advertisement on radio. Customers get to know their products, location etc so they do business with them

v. Politically, certain government policies and by-laws are interpreted and made clearer to the people through radio. The radio may also be used for political campaigns, etc.

vi. Important breaking news is made known to the people/listeners a few minutes after they have occurred through the power of the radio.

vii. People do not only listen to programmes and win for themselves prizes that go a long way to enhance their lives.

viii. Listeners easily have the opportunity to easily make their grievances known to the appropriate quarters through the power of radio.

ix. Socially, important announcements, invitations reach the concerned people easily by the use of radio.

x. Radio serves as a source of entertainment and companionship for some listeners who for one reason or another may feel lonely.

xi. Criminals and their associates are easily apprehended by the police when the public relays information through the radio to the police in their quest to halt an on going crime scene.

1.8 DISADVANTAGES OF FM STATIONS

i. Radio analyst and presenters when insensitive to issues that have the potential to ignite violence or civil war can lead a country into that. An example is what happen in Rwanda in Eastern Africa in the mid 90s, where thousands of people died and many rendered homeless

ii. A wrong perception on the part of a radio presenter or analyst can mislead a lot of naïve listeners since they reach them directly.

iii. Radio can adulterate the culture of a country when it refuses to do selective borrowing from a foreign culture. This can negatively affect the life style of a group of people.

iv. In a coup de tat, the military seizes power and uses the radio to announce to the general public the seizure of power. Radio helps in that social vice.

iv. Phone-in programs, as organized by most radio stations, sometimes allow uncivilized callers to verbally attack others, use vulgar language, etc on air. This sometimes creates unhealthy tens

1.9 SUMMARY

In this chapter, the problem of optimal packing is formulated. A brief history of radio stations is given. Brief discussions of the algorithm that can be used to address these problems were also presented. Some advantages and disadvantages of radio stations were also presented. In the next chapter, we shall review some literature in the area of 0 – 1 Knapsack Problem.

CHAPTER TWO

2.1 LITERATURE REVIEW

In this chapter, we shall review some literature pertinent in the area of knapsack problems. Several types of large-sized 0-1 Knapsack Problems (KP) may be easily solved, but in such cases most of the computational effort is used for sorting and reduction. In order to avoid this problem it has been proposed to solve the so-called *core* of the problem: a Knapsack Problem defined on a small subset of the variables. The exact core cannot, however, be identified before KP is solved to optimality, thus, previous algorithms had to rely on approximate core sizes.

Pisinger [58] presented an algorithm for KP where the enumerated core size is minimal, and the computational effort for sorting and reduction also is limited according to a hierarchy. The algorithm is based on a dynamic programming approach, where the core size is extended by need, and the sorting and reduction is performed in a similar "lazy" way. Computational experiments are presented for several commonly occurring types of data instances. Experience from these tests indicates that the presented approach outperforms any known algorithm for KP, having very stable solution times.

Martello and Toth [59] presented a new algorithm for the optimal solution of the 0-1 Knapsack problem, which is particularly effective for large-size problems. The algorithm is based on determination of an appropriate small subset of items and the solution of the corresponding "core problem": from this they derived a heuristic solution for the original problem which, with high probability, can be proved to be optimal. The algorithm incorporates a new method of computation of upper bounds and efficient implementations of reduction procedures. They also reported computational experiments on small-size and large-size random problems, comparing the proposed code with all those available in the literature.

Munapo [60] presented an approach that enhances the performance of the branch and bound algorithm for the knapsack model. This is achieved by generating and adding new objective function and constraint to knapsack model which single constrained. The branch and bound algorithm is then applied and the total numbers of sub-problems are reduced.

Majority of algorithms for solving knapsack problems typically use implicit enumeration approaches. Different bounds based on the remaining capacity of the knapsack and items not yet included at certain iteration have been proposed for use in these algorithms. Similar methods may be used for a nested knapsack problem as long as there is an established procedure for testing whether an item inserted into a knapsack at one stage can also be inserted at the following stages. Given n different items and a knapsack of capacity Cáceres and Nishibe [61] algorithm solves the 0-1 Knapsack Problem using $O(nWp)$ local computation time with $O(p)$ communication rounds. Using Dynamic Programming, their algorithm solves locally pieces of the Knapsack Problem in each processor and uses a wave front approach in order to solve the whole problem. The algorithm was implemented in a Beowulf and the obtained times showed good speed-up and scalability Robert [60].

The binary knapsack problem is a combinatorial optimization problem in which a subset of a given set of elements needs to be chosen in order to maximize profit, given a budget constraint. Das [62] used a stochastic version of the problem in which the budget is random. They proposed two different formulations of this problem, based on different ways of handling infeasibility, and propose an exact algorithm and a local search-based heuristic to solve the problems represented by these formulations. The results were presented from some computational experiments.

Yu-Hsien [63] put forward Knapsack problem and its generalizations intensively studied during the last three decades with a rich literature of research reports. Over the years, surveys and reviews have been conducted mostly on the standard knapsack problems, namely, the single-constraint linear model. The paper reports the solution approaches developed for some non-standard knapsack problems with wide range of applications through a bibliographical review. The non-standard knapsack problems reviewed in the paper include the multidimensional knapsack problem.

The MULKNAP algorithm, which is based on the MTM framework of Martello and Toth, is the first algorithm capable of solving large sized cases up to n=100,000 with data range of as much as R=10,000. MULKNAP uses specialized algorithms to derive both lower bounds and upper bounds as well as solve the Subset-sum Problem. It has been demonstrated that despite the unsuitability of Multiple Knapsack Problem in dynamic programming, dynamic programming algorithms can be used to provide the needed solutions David [64].

Probabilistic and stochastic algorithms have been used to solve many hard optimization problems since they can provide solutions to problems where often standard algorithms have failed. These algorithms basically search through a space of potential solutions using randomness as a major factor to make decisions. In their research, the knapsack problem (optimization problem) is solved using a genetic algorithm approach. Subsequently, comparisons are made with a greedy method and a heuristic algorithm. The knapsack problem is recognized to be NP-hard. Genetic algorithms are among search procedures based on natural selection and natural genetics. They randomly create an initial population of individuals. Then, they use genetic operators to yield new offspring. In their research, a genetic algorithm is used to solve the 0/1 knapsack problem. Special consideration is given to the penalty function where constant and self-adaptive penalty functions are adopted Zoheir [65].

The knapsack problem is believed to be one of the "easier" -hard problems. Not only can it be solved in pseudo-polynomial time, but also decades of algorithmic improvements have made it possible to solve nearly all standard instances from the literature. Pisinger [66] gave an overview of all recent exact solution approaches, and to show that the knapsack problem still is hard to solve for these algorithms for a variety of new test problems. These problems are constructed either by using standard benchmark instances with larger coefficients, or by introducing new classes of instances for which most upper bounds perform badly. The first group of problems challenges the dynamic programming algorithms while the other groups of problems are focused towards branch-and-bound algorithms. Numerous computational experiments with all recent state-of-the-art codes are used to show that (KP) is still difficult to solve for a wide number of problems. One could say that the previous benchmark tests were limited to a few highly structured instances, which do not show the full characteristics of knapsack problems.

The 0/1 knapsack problem is a well-known and it appears in many real domains with practical importance. The problem is NP-complete. The multiobjective 0/1 knapsack problem is a generalization of the 0/1 knapsack problem in which many knapsacks are considered. Many algorithms have been proposed in the past four decades for both single and multiobjective knapsack problem. A new evolutionary algorithm for solving multiobjective 0/1 knapsack problem was proposed by Grosan [67]. This algorithm used a ε-dominance relation for direct comparison of two solutions. Some numerical experiments are realized using the best and recent algorithms proposed for this problem. Experimental results show that the new proposed algorithm outperforms the existing evolutionary approaches for this problem.

The knapsack problem is believed to be one of the "easier" NP-hard problems. Not only can it be solved in pseudo-polynomial time by the dynamic programming, but also decades of algorithmic improvements have made it possible to solve nearly all standard instances. The currently most successful algorithm for the knapsack problem was presented by Martello, Pisinger and Toth. The algorithm can be seen as a combination of many different concepts and is hence called Combo. But they noticed that it is difficult for Combo to solve some instances. Ken'ichi [68] proposed a new genetic algorithm for knapsack problem. The algorithm can adjust solution spaces in consideration of the stability of each item which can obtain from the greedy algorithm. They applied the proposed method to those difficult instances, and test the effectiveness.

Gong [69] introduced a computational model simulating the dynamic process of human immune response to solve multidimensional knapsack problems. The new model is a quaternion (G, I, R, At), where G denotes exterior stimulus or antigen, denotes the set of valid antibodies, R denotes the set of reaction rules describing the interactions between antibodies, and Al denotes the dynamic algorithm describing how the reaction rules are applied to antibody population. The set of antibody-adjusting rules, the set of clonal selection rules, and a dynamic algorithm, named MKP-PAISA, are designed for solving multidimensional knapsack problems. The efficiency of the proposed algorithm was validated by testing on 57 benchmark problems and comparing with three genetic algorithms. The results indicated that the proposed algorithm was suitable for solving multidimensional knapsack problems.

Shang-Hua [70] considered the design of a time-efficient and processor-efficient parallel algorithm for the integral knapsack problem. A parallel integral knapsack algorithm is presented, which is adaptive to all parameters, especially to the maximum size of items. The parallel complexity of another important packing problem, the integral exactly-packing problem, is also considered. An optimal O (log n log m) time, parallel integral exactly-packing algorithm is given. Since the partition problem has a constant time, constant processor reduction to the exactly-packing problem, their parallel integral exactly-packing algorithm can be used for job scheduling, task partition, and many other important practical problems. Moreover, the methods and techniques used their paper can be used for developing processor-efficient and time-efficient parallel algorithms for many other problems. Using the new parallel integral knapsack algorithm, the previously known parallel approximation schemes for the 0-1 knapsack problem and the binpacking problem are improved upon significantly.

Mastrolilli [71] addressed the classical knapsack problem and a variant in which an upper bound is imposed on the number of items that can be selected. They showed that appropriate combinations of rounding techniques yield novel and powerful ways of rounding. As an application of these techniques, they presented a faster polynomial time approximation schemes that computes an approximate solution of any fixed accuracy in linear time. This linear complexity bounds gave a substantial improvement of the best previously known polynomial bounds.

Puchinger [72] presented a newly developed core concept for the Multidimensional Knapsack Problem (MKP) which is an extension of the classical concept for the one-dimensional case. The core for the multidimensional problem is defined in dependence of a chosen efficiency function of the items, since no single obvious efficiency measure is available for MKP. An empirical study on the cores of widely-used benchmark instances is presented, as well as experiments with different approximate core sizes. Furthermore they described a memetic algorithm and a relaxation guided variable neighborhood search for the MKP, which are applied to the original and to the core problems. The experimental results show that given a fixed run-time, the different metaheuristics as well as a general purpose integer linear programming solver yield better solution when applied to approximate core problems of fixed size.

According to Chekuri and Khanna [73] the multiple knapsack problems (MKP) is a natural and well-known generalization of the single knapsack problem and is defined as follows. Supposing a set of n items and m bins (knapsacks) are given such that each item i has a profit p(i) and a size s(i), and each bin j has a capacity c(j). The goal is to find a subset of items of maximum profit such that they have a feasible packing in the bins. MKP is a special case of the generalized assignment problem (GAP) where the profit and the size of an item can vary based on the specific bin that it is assigned to. GAP is APX-hard and a 2-approximation, for it is implicit in the work of Shmoys and Tardos [74]. This was also the best known approximation for MKP. The main result of Shmoys and Tardos [74] is a polynomial time approximation scheme (PTAS) for MKP. Apart from its inherent theoretical interest as a common generalization of the well-studied knapsack and bin packing problems, it appears to be the strongest special case of GAP that is not APX-hard. They substantiated this by showing that slight generalizations of MKP are APX-hard. Thus their results helped to demarcate the boundary at which instances of GAP become APX-hard. An interesting aspect of this approach is a PTAS-preserving reduction from an arbitrary instance of MKP to an instance with O (log n) distinct sizes and profits.

A Multiple-Choice Multi-Dimension Knapsack Problem (MMKP) in PRAM model is a variant of the classical 0-1 knapsack problem has a knapsack with multidimensional capacity constraints, groups of items, each item having a utility value and multidimensional resource constraints. The problem is to maximize the total value of the items in the knapsack with the constraint of not exceeding the knapsack capacity. Since MMKP is an NP-Hard problem, its exact solution is not suitable for real time problems, so heuristic based approximation algorithms are developed. A parallel heuristic algorithm was presented here that runs in $O\left(\log nl(\log n+\log m+\log\log l)\right)$ time in CRCW PRAM machine; taking $O\left(n\log n(\log n+ml)\right)$ operations. Experimental results showed that it achieves 95% optimal solution on average. This also means that a sequential heuristic running in $O(n\log n(\log n+ml)$ time which seems to be remarkable since M-HEU, a celebrated sequential heuristic although achieves 96% of optimal value $O(mn^2l^2)$ running time (www.drivehq.com).

The knapsack problem is a typical one of NPC problems, which is easy to be described but difficult to be solved. It is very important in theory and practice to study it. Nowadays there is a variety of research in algorithm for solving it. As the parallel processing technologies develop, the research of effective parallel algorithms for this problem attracts much attention. To run those algorithms needs high level hardware and high performance parallel computers. Using mobile agent technologies, a more effective model to solve the complex distributed problems can be established. Combines mobile agent with the traditional parallel algorithm, the process in a parallel computer can be evolved into the one performed by some ordinary computers. This can avoid the limitation of experiment conditions and provides convenience in practice. A distributed algorithm is proposed for the 0-1 knapsack problem based on the mobile agent, and it is feasible and effective in theoretical analysis Qiao [76].

Han [77] studied the following knapsack problem: Given a list of squares with profits, we are requested to pack a sub list of them into a rectangular bin (not a unit square bin) to make profits in the bin as large as possible. They first observed that there is a Polynomial Time Approximation Scheme (PTAS) for the problem of packing weighted squares into rectangular bins with large resources, then apply the PTAS to the problem of packing squares with profits into a rectangular bin and get a $\frac65+\epsilon$ approximation algorithm.

Fontanari [78] investigated the dependence of the multi-knapsack objective function on the knapsack capacities and on the number of capacity constraints P, in the case when all N objects are assigned the same profit value and the weights are uniformly distributed over the unit interval. A rigorous upper bound to the optimal profit is obtained employing the annealed approximation and then compared with the exact value obtained through the Lagrangian relaxation method. The analysis is restricted to the regime where N goes to infinity and P remains finite.

Senior managers in retail industry, make important decisions upon assortment planning, product pricing, and product promotion. While product assortment is a strategic decision taken over a long-term planning period, the latter two are both strategic and tactical: they can be used in day-to-day marketing decisions to dynamically adjust to demand variations. Within the food retail industry, the necessity, frequency, and complexity of pricing and promotion decisions are further magnified by perishability of food products. There is a strong need by retail managers for a “soft” marketing tool, which would dynamically allow them to improve sales and revenues, yet not altering product prices. For, dynamic pricing models may prescribe to change prices too often or in an “unsystematic” fashion, which contradicts discrete-time decision making, implementation costs, and retail brand image strategy. In addition, the price reduction must usually be done over all units of the product, thus losing possible profit from customers willing to pay the original, higher price. Furthermore, dynamic pricing results in a trade-off between markdowns and stockouts, since markdowns may damage producers, while stockouts may damage retailers, Jacko [80].

A revenue management model was designed, in which demand is stimulated by moving a number of product units to a promotion space, rather than by price changes. Thus, they addressed the problem of filling a promotion location with limited space to maximize the expected revenue, which we have termed the Knapsack Problem for Perishable Items (KPPI). Examples of the promotion space include shelves close to the cash register, promotion kiosks, or a depot used for selling via the Internet.

They solved the KPPI using problem decomposition into single product unit subproblems. A natural mathematical setting for the KPPI subproblems is the restless bandit model a fundamental stochastic model for resolving a trade-off between exploration and exploitation in an optimal fashion. In our model the bandits (perishable items) are restless, because they can get sold regardless of being in the knapsack or not, the time horizon is finite, and we are to select more than one item for the knapsack, which is allowed to be filled partially, due to the heterogeneity of the items. Each product unit is assigned a promotion priority index, which captures the opportunity cost of promotion, as a function of its price, lifetime, expected demand, and expected promotion power. These indices are then used for each item as objective-function value coefficients in a (classic) knapsack problem, whose solution yields a well-performing heuristic for the KPPI. They thus mix up two models: the restless bandit problem and the knapsack problem.

Optimal Dynamic Promotion under Time-Homogeneous Demand Suppose that the item perishes in T time periods, implying a final cost $c > 0$ if not sold before. Let $1 - p$ be the probability that a promoted item is sold in one period, and $1 - q$ that of a non-promoted item ($q > p$). Future costs are discounted by the one-period discount factor β. The next proposition asserts that, optimally, an item with lower probabilities of being sold when not promoted is assigned a higher promotion priority index, and that the index is nondecreasing over time. Hence, once the item is optimally chosen for promotion, then it should remain promoted until it perishes.

Krass [81] proposed a new class of knapsack problems by assuming that the sizes of the items to be put into a knapsack are known to be elements 0f a given subset S of the positive integers Z'^. The set S is treated as a parameter. They showed that the family of knapsack problems obtained by varying the parameter S in the power set of Z'^ contains polynomially solvable problems and NP-complete problems, even when they restrict S to the class of polynomially recognizable sets.

For the minimization knapsack problem with Boolean variables, primal and dual greedy algorithms are formally described. Their relations to the corresponding algorithms for the maximization knapsack problem are shown. The average behavior of primal and dual algorithms for the minimization problem is analyzed. It is assumed that the coefficients of the objective function and the constraint are independent identically distributed random variables on [0, 1] with an arbitrary distribution having a density and that the right-hand side d is deterministic and proportional to the number of variables i.e. $\left(d = \mu n\right)$. A condition on μ is found under which the primal and dual greedy algorithms have an asymptotic error of t Diubinv [82].

In the last decade Mixed-Integer Programming solvers have evolved enormously contributing to the widespread application of optimization in real world problems in industry. Nonetheless, it is paramount for practitioners to have basic knowledge on how these solvers work and to be able to identify model structures, so one can take full advantage of the machinery at hand. In this paper we present a reformulation to a simple problem that appears as sub-problem in many supply chain models, and they showed the advantage of using suitable mathematical structures in the form of cascading knapsack inequalities to solve it. Moreover, they introduced new reformulations to some special cases, producing tighter linear relaxation and faster solution times [83]. Specifically, they presented and tested reformulations to an inventory-production distribution problem that arises frequently as a subproblem in supply chain optimization models namely, the inventory-production-distribution problem. A company has several industrial plants producing parts that are shipped to its distribution centers to supply its customers. The production of parts per time period is known. The company wants to determine the best shipment schedule in order to satisfy the customer demands and avoid inventory shortfalls at the industrial plants, as well as at the distribution centers in the planning horizon. The customer demands are known in advance, and due to the company shipping policy the trucks must be loaded to full capacity. Additionally, they assumed that the number of shipments per time period between each pair of plant-distribution center is limited to at most one, and that the number of trucks in each type of truck is unlimited. They showed that through an inventory reformulation in the above problem, the special structure of Cascading Knapsack Inequalities, hidden in the initial formulation, is identified. This structure is of great value as the knapsack inequalities have been studied since the beginning of the integer programming area and nowadays all solvers have implemented very sophisticated techniques to exploit such structures. Besides, the cascading form allows us to derive tighter reformulations for some special cases. This in turn allows us to use an off-the-shelf MIP solver to solve instances that were out of reach by the Initial Model. Furthermore, capitalizing on this special structure, they proposed tighter reformulations for some special cases of this problem, reducing further the solution times for a great number of instances. A main lesson of this work is that although they have nowadays sophisticated MIP solvers capable of solving problems never envisaged before, it is still paramount for the users of MIP solvers to have an understanding not only on the modeling but also on how the MIP solvers work if they are to be able to solve more challenging problems.

Recently, a knapsack problem with precedence constraints imposed on pairs of items, known as the precedence constrained knapsack problem (PCKP) was considered. This problem has applications in management and machine scheduling, and also appears as a sub problem in decomposition techniques for network design and other related problems. They presented a new approach for determining facets of the PCKP polyhedron based on clique inequalities. A comparison with existing techniques, that lift knapsack cover inequalities for the PCKP, is also presented. It is shown that the clique-based approach generates facets that cannot be found through the existing cover-based approaches, and that the addition of clique-based inequalities for the PCKP can be computationally beneficial [84].

The static stochastic knapsack problem requires assigning items with random sizes to a knapsack whose capacity to be any value between a lower and an upper bound. An item's value is determined by multiplying a per-unit revenue term by the realization of the item's size. When the realization of the sum of selected item sizes exceeds the knapsack capacity, a penalty cost is assessed for each unit of overflow. They seeked to maximize the expected net value resulting from the assignment of items to the knapsack. They provided an exact solution method for a linear relaxation of the problem when item sizes are independent and normally distributed, which is used in a branch-and-bound algorithm for obtaining an optimal solution satisfying the binary variable requirements [85].

For the approximation for four kinds of knapsack problems with multiple constraints is studied: 0/1 Multiple Constraint Knapsack Problem (0/1 MCKP), Integer Multiple Constraint Knapsack Problem (Integer MCKP), 0/1k-Constraillt Knapsack Problem (0/1 k-CKP) and Integer k-Constraint KnapsackProblem (Integer k-CKP). The following results are obtained:

1) Unless$NP=co-R$, no polynomial time algorithm approximates 0/1 MCKP or Integer MCKP within a factor$k^{(1/2)-\sigma}$ for any σ>0; unless$NP=P$, no polynomial time algorithm approximates 0/1 MCKP or Integer MCKP within a factor$k^{(1/4)-\sigma}$ for any σ>0, where$k$ stands for the number of constraints.

2) For any fixed positive integer$k$, 0/1k-CKP has a fully polynomial time approximation scheme (FPTAS).

3) For any fixed positive integer$k$, Integer$k$-CKP has a fast FPTAS which has time complexity $0(n+\frac{1}{\varepsilon^3}+\frac{1}{\varepsilon^{2^{k+1}-2}})$ and space complexity $0(n+(1/\varepsilon^3))$, and finds an approximate solution to within ε of the optimal solution [87].

According to David [86] the Multiple-Choice Knapsack Problem is dened as a 0-1 Knapsack Problem with the addition of disjoined multiple-choice constraints. As for other knapsack problems most of the computational effort in the solution of these problems is used for sorting and reduction. But although O (n) algorithms which solve the linear Multiple-Choice Knapsack Problem without sorting have been known for more than a decade, such techniques have not been used in enumerative algorithms. In their paper, they presented a simple O (n) partitioning algorithm for deriving the optimal linear solution, and show how it may be incorporated in a dynamic programming algorithm such that a minimal number of classes are enumerated, sorted and reduced. Computational experiments indicate that this approach leads to a very efficient algorithm which outperforms any known algorithm for the problem.

A novel branch-and-bound procedure for solving the 0-1 knapsack problem is introduced [86]. The problem which concerns the placement of n items into a knapsack of given capacity c, requires the maximization of the profit sum under the constraint that the total weight of the items is less than the c. The new algorithm, which has been dubbed EXPKNAP, eliminates sorting and reduction steps to yield an exact solution in less time than other methods.

CHAPTER THREE

3.0 KNAPSACK PROBLEM

Radio stations, especially FM stations in Kumasi and for that matter Ghana has shot up in terms of numbers over the past decade and it appears that the Garden City (Kumasi), which now has eighteen (18) FM stations will soon be challenging Accra's twenty four (24) stations. This has really called for the need to adopt careful and optimal packaging strategies to enhance the achievement of the following;

(i) Customer confidence, goodwill and trust

(ii) Optimum use of airtime and avoidance of pile up commercials and programs

(iii) Increase production and maximization of profit, and

(iv) Increase in the stations' growth rate

In reaching the above goals, the knapsack algorithm will be used to explore for optimal and efficient arrangement of commercials and programs.

The knapsack problems are among the simplest integer programs which are NP-hard. Problems in this class are typically concerned with selecting from a set of given items each with a specified weight and value, a subset of items whose weight sum does not exceed a prescribed capacity and whose value is maximum. The specific that arises depends on the number of knapsacks (single or multiple) to be filled and the number of available items of each type (bounded or unbounded).

3.1 INTRODUCTION

A great variety of practical problems can be represented by a set of entities, each having an associated value, from which one or more subsets has to be selected in such a way that the sum of the values of the selected entities is maximized, and some predefined conditions are respected. The most common condition is obtained by also associating a size to each entity and establishing that the sum of the entity sizes in each subset does not exceed some prefixed bound. These problems are generally called knapsack problems, since they recall the situation of a hitch-hiker having to fill up his knapsack by selecting from among various possible objects those which will give him the maximum comfort. In this project we shall adopt the following terminology. The entities will be called *items* and their numbers will be indicated by *n.* The value and size associated with the j-th item will be called *profit* and *weight*, respectively, and denoted by p_j and $w_j (j = 1,...,n)$.

The majority of problems considered in this thesis are single knapsack problems, where one container must be filled with an optimal subset of items. The capacity of such a container will be denoted by c. We shall also consider the more general case where *m* containers, of capacities $c_j (i = 1,...,n)$, are available (multiple knapsack problems). We shall suppose that profits, weights and capacities are positive integers.

Knapsack problems have been intensively studied both because they arise as subproblems in various integer programming problems and may represent many practical situations. The most typical applications are in capital budgeting and industrial production. Various capital budgeting models have been studied by Weingartner [18, 19], Weingartner and Ness [20], Cord [21] and Kaplan [22]. Among industrial applications, the classical studies a Cargo Loading Problems Bellman and Dreyfus [23] and on Cutting Stock Problems Gilmore and Gomory [24, 25, 26, 27] are worth mentioning. More detailed reviews of applications can be found in Salkin [28] and Salkin and de Kluyver [29]. Almost all books on Integer Programming or Combinational Optimization contain a chapter on knapsack problems. In particular mention is made of those of Hu [30], Garfinkel and Nemhauser [31], Salkin [32], Martello and Toth [33], Syslo, Deo and Kowalik [34].

3.2 0 – 1 KNAPSACK ALGORITHM

The Knapsack problem is the classic integer linear programming problem with a single constraint. The 0-1 Knapsack Problem (KP) is a problem of choosing a subset of the *n* items such that the corresponding profit sum is maximized without having the weight sum to exceed the capacity *b*. This may be formulated as follows:

$$\max \quad p_1x_1 + p_2x_2 + ... + p_nx_n \qquad \text{ie} \quad z = \sum_i^n p_i x_i \tag{3.1}$$

$$\text{subject to: } w_1x_1 + w_2x_2 + ... + w_nx_n \le c \qquad \text{ie} \quad \sum_i^n w_i x_i \le c \tag{3.2}$$

$$x_i = \begin{cases} 1, \; if \; the \; i^{th} \; item \; has \; been \; selected \\ 0, otherwise \end{cases}$$

Since profits and weight are positive, it will be supposed, without loss of generality, that is

$$\sum_{i=1}^{n} w_i > c \qquad (2.3)$$

$$w_i \quad \leq \quad c \;\; (i=1,\ldots,n) \qquad (2.4)$$

The c_i and a_i represents the value and weight of selecting item i respectively for inclusion in the knapsack. The constant b represents the maximum weight that the knapsack is permitted to hold.

KP is NP-hard and is well-known problem and several exact and heuristic algorithms have been proposed for its solution. The exact algorithm can be subdivided into two classes: *branch and bound* methods Kolesar[7], Greenberg and Hegerich[8], Horowitz and Sahni [9], Fayard and Plateau[10]. The performance of both classes largely depends on the size of the problems to solved. This can generally be decreased by applying reduction procedures Ingargiola and Korsh[11], Toth[12], Dembo and Hammer[13], Fayard and Plateau[10] so as to fix the value of as many variable as possible.

One of the essential ingredients of the implicit enumeration algorithms and the reduction procedures is the quality of the *upper bounds* used in the computation. The most effective upper bounds for KP will be presented is the Relaxation and upper bounds (section 3,2). The following sections will summarize the reduction procedures, the branch and bound methods for small size and large size problems and the dynamic programming approaches.

3.3 Relaxations and upper bounds

For computation of the upper bounds it is assumed that the items are ordered according to non increasing values of the profit per unit weight, that is, so that

$p_i \,/\, w_i \;\geq\; p_{j+1}, / \, w_{j+1}$ for $j = 1, \ldots, n - 1.$

If this is not the case, the sorting of the items can be performed in 0 *(n* log *n)* time through any efficient sorting procedure (see, for instance, Aho, Hopcroft and Ullman [50].

Let the *critical item s* be defined as

$$s = \min \{j: \sum_{i=1}^{j} w_i > c\}$$

(because of (3.3) and (3.4), we have 1 < s < *n).*

Given a problem instance *p*, we denote with *z(p)* the value of any optimal solution of *P.* If *P* can be formulated as an integer linear program, we denote with $\overline{p}$ the linear programming relaxation of *p.* The linear programming relaxation of KP, also called the *continuous knapsack problem,* is defined by

$\overline{KP}$ (3.1), (3.2) and $0 \le x_j \le 1$ (j = 1,…, n).

The optimal solution $\overline{x}$ of $\overline{KP}$ can easily be obtain in the following way Dantig[51]

$\overline{x}_j = 1$ for j = 1,…,s – 1

$\overline{x}_j = 0$ for j = s +1,…,n

Where $\overline{c} = c - \sum_{j=1}^{s-1} w_j$.

The optimal solution value of $\overline{KP}$ is given by

$$z(\overline{KP}) = \sum_{j=1}^{s-1} p_j + \overline{c} p_s / w_s$$

Because of the integrality of p_j and x_j, a valid upper bound for KP is

$$u_1 = [z(\overline{KP})] = \sum_{j=1}^{s-1} [p_j + \overline{c} p_s / w_s],$$

where [r] is the largest integer not greater than r**.** The computational complexity of $\overline{KP}$, and hence of the Dantzig bound u_1**,** is clearly *O(n)* if we assume that the items are already sorted, otherwise *O(n* logn).

An improved upper bound has been proposed by Martello and Toth [52] according to the following consideration: since in KP**, $\mathbf{x_s}$** cannot assume a fractional value, the optimal solution of KP can be obtained from the corresponding continuous solution $\bar{x}$ of $\overline{KP}$,either without inserting the s-th item (i.e. by setting $\bar{x}_s = 0$) or by inserting it (i.e. by setting $\bar{x}_s = 1$). In the former case, the solution value cannot exceed

$$b_1 = \sum_{j=1}^{s-1} p_j + [\bar{c} p_{s+1} / w_{s+1}],$$

which corresponds to the case of filling the residual capacity C with items having the best possible value of $p_i \ / \ w_i$ ($p_{s+1}, \ / \ w_{s+1}$). In the latter, since it is necessary to remove at least one of the first s - 1 item, the best solution value is given by

$$b_2 = \sum_{j=1}^{s-1} p_j + [p_s - (w_s - \bar{c}) p_{s-1} / w_{s-1}]$$

where it has been supposed that the item to be removed has exactly the minimum necessary value of w (i.e. $w_s - \bar{c}$) and the worst possible value of p_j / w_j (i.e. p_{s-1} / w_{s-1})**.** A valid upper bound for KP is so given by

$u_2 = \max \{b_1, b_2\}$.

It has been proved (Martello and Toth [52]) that $u_2 < u_1$**.**

The consideration on which the Martello-Toth bound is based can be further exploited to compute more restrictive upper bounds than u_2. This can be achieved by replacing the values b_1 and b_2 with stronger values, say b_3 and b_4, which take more carefully into account the exclusion and inclusion of the s-th item. Hudson[53] has proposed computing b_4 through the continuous relaxation of KP with the additional constraint $x_s = 1$, that is

$b_4 = [\, b_4 = [z(\overline{KP}$ with $x_s = 1)]$

Fayard and Plateau [11] and, independently, Villela and Bornstein [54], have proposed computing b_3 as well through continuous relaxation of KP, by imposing constraint $x_s = 0$, that is,

$b_3 = [z(\overline{KP}$ with $x_s = 0)]$

The corresponding bound is $u_3 = \max \{b_3, u_4\}$.

It can easily be proved that, since $b_1 \leq b_2$ and $b_3 \leq b_4$, we have $u_3, < u_2$. Bound u_3 , can also be seen as the result of the application of Dantzig's bound at the two terminal nodes of a decision tree having the root node corresponding to KP and two descendent nodes, say N1 and N2, corresponding to the exclusion and inclusion of the s - t h item. Clearly, the maximum among the upper bounds corresponding to all the terminal nodes of a decision tree represents a valid upper bound for the original problem corresponding to the root node. So, if b_3 and b_4 are the Dantzig bounds corresponding respectively to nodes Nl and N2, u_3 represents a valid upper bound for KP. An improved bound u_4 can be obtained by considering decision trees having more than two terminal nodes; this approach has been proposed by Martello and Toth in [55]. If, for example, nodes N1 and N2 have each two descendent nodes, corresponding to the exclusion and inclusion of the (s + 1)-th item, a valid upper bound is given by $\bar{u}_4$ = max {b_5, b_6, $b_{7,}$ b_8},

3.4 Reduction procedures

The number of binary variables (x_j) and the value of ***c*** can be decreased by applying reduction procedures ([56], [57], [14], [54]) which fix the optimal value of as many variables as possible. These procedures partition set N = (1, 2, . . . , n } into three subsets:

$J1 = \{J \in N : x_j = 1\}$ in an optimal solution to KP};

$J0 = \{J \in N : x_j = 0\}$ in an optimal solution to KP};

$F = N - (J1 \cup JO)$.

The original KP can now be transformed into the reduced form

$$\text{Maximize} \quad z = \sum_{j \in F} p_j x_j + \hat{p}$$

$$\text{Subject to} \quad \sum_{j \in F} w_j x_j \leq \hat{c}$$

$$x_j = 0 \text{ or } 1 \qquad (j \in F)$$

where $\hat{p} = \sum_{j \in J1} p_j$, $\hat{c} = c - \sum_{j \in J1} w_j$

Subsets *J1* and *JO* are obtained by considering the implications corresponding to setting a variable x_i to 0 or to 1. If setting a variable x_i to a given value a (a = 0 or 1) implies an infeasible solution t o KP or a solution not better than an existing one, variable x_i can be fixed to 1 -- a, since this is the only choice which can lead to feasible or improved solutions. The implications of what occurs by setting a variable to a given value can be derived either through dominance relations or the evaluation of upper bounds.

Let $\hat{x}$ be any feasible solution to KP and $\hat{z}$ the corresponding value ($\hat{z}$ represents a lower bound on the optimal solution value z^* to KP). Moreover, for $j \in$ N, let z_j (resp. $\overline{z}_j$) be the upper bound corresponding to KP with the additional constraint $x_j = 1$ (resp. $x_j = 0$). Then we have

$$J\,1 = \{\, j \in \mathrm{N}:\ \overline{z}_j \leq \hat{z}\,\},$$

$$J\,0 = \{\, j \in \mathrm{N}:\ \overline{z}_j \leq \hat{z}\,\},$$

The effectiveness and computational efficiency of the reduction procedures clearly depend on the techniques used in computing, $\hat{z}$, z_j and $\overline{z}_j$. The upper bounds z_j and $\overline{z}_j$ can be computed through any of the approaches of Section (3.2). In particular, bound u_1 has been used in [56] and [54],bound u_2 in [57]. Since the computation of bounds z_j (resp. $\overline{z}_j$) requires *O(n)* time to find the critical item s_j (resp. $\overline{s}_j$) in the corresponding continuous solution, the overall complexity of the reduction procedures is *O(n2)*. However, if bounds u_1, u_2, u_3 or u_4 are used, the average computing times can be greatly decreased since, as mentioned in Section 3.2, these bounds can easily be computed through parametric techniques.

3.5 0 – 1 Multiple Knapsack Problem

3.5.1. The problem

The ***0-1 Multiple Knapsack Problem (MKP)*** is a generalization of the 0-1 knapsack problem where *m* containers with capacities $c_1, \ldots, c_m$, are available. If p_i and w_j are, respectively, the profit and weight of the j-th item ($j = 1, \ldots, n$), and we introduce a boolean variable $x_{i,j}$ which is set to 1 if item *j* is inserted in container ***i***, or to 0 otherwise, the problem can be formally stated as maximize $z = \sum_{i=1}^{m} \sum_{j=1}^{n} p_{i,j} x_{i,j}$ (3.5)

subject to $\sum_{i=1}^{m} w_j x_{i,j} \leq c_i$ $(i = 1, \ldots, m)$ (3.6)

$\sum_{j=1}^{m} x_{i,j} \leq 1$ $(j = 1, \ldots, n)$ (3.7)

$X_{i,j} = 0$ or 1 $(i = 1, \ldots, m; \; j = 1, \ldots, n)$ (3.8)

The problem is clearly NP-hard, since it reduces to the 0- 1 knapsack problem when $m = 1$.

In what follows we will assume, without loss of generality

$\min \{w_j\} \leq \min \{c_i\}$;

$\max \{w_j\} \leq \max \{c_i\}$;

$\sum_{j=1}^{n} w_j > \max\{c_i\}$

Furthermore, we will assume that the items are sorted so that $p_j / w_j \geq p_{i+1} / w_{j+1}$,
$(j = 1, \ldots, n - 1)$

3.5.2 Relaxations and upper bounds

Two relaxation methods are generally employed to obtain upper bounds for MKP: the Lagrangean relaxation and the surrogate relaxation.

3.5.3 Lagrangean Relaxation

The *Lagrangean Relaxation*, relative to a nonnegative vector (λ), is

Maximize $$z_\lambda = \sum_{i=1}^{m}\sum_{i}^{n} p_j x_{ij} - \sum_{j=1}^{n}\lambda_j\left(\sum_{i=1}^{m} x_{ij} - 1\right)$$

Subject to $$\sum_{j=1}^{n} w_j x_{i,j} \le c_i \qquad (i = 1, \ldots, m)$$

$$X_{i,j} = 0 \text{ or } 1 \qquad (i = 1, \ldots, m;\ j = 1, \ldots, n).$$

Since the object function can be written as

$$z_\lambda = \sum_{j=1}^{n}\lambda_j + \sum_{i=1}^{m}\sum_{j=1}^{n}(p_j - \lambda_j)x_{i,j}$$

the relaxed problem can be decomposed into a series of *m* single knapsack problems.
For $i = 1, \ldots, m$ we can solve the single knapsack problem

Maximize $$z_i = \sum_{j=1}^{n}(p_j - \lambda_j)x_{i,j}$$

Subject to $$\sum_{j=1}^{n} w_j x_{i,j} \le c_i \qquad (i = 1, \ldots, m)$$

and then set $z_\lambda = \sum_{j=1}^{n}\lambda_j + \sum_{i=1}^{n} z_i$.

In order to find the tightest possible bound,vector (λ) should be determined so that z_λ minimized. An approximation of the optimum **(λ)** can be obtained through subgradient techniques, which are, however, time consuming. Hung and Fisk [35], who first used this relaxation to obtain upper bounds for MKP, set the λ_i's equal to the optimal dual multipliers of constraints (3.6) in the continuous relaxation given by (3.4), (3.5), (3.6) and

$$0 \leq x_{i,j} \leq 1 \qquad (I = 1,\ldots,m\ ;\ j = 1,\ldots,n)$$

that is, $\lambda_i = p_i - w_j\,(p_t/w_t)$ if $j < t$, or $\lambda_i = 0$ otherwise, where t is the smallest index such that

$$\sum_{j=1}^{t} w_j > \sum_{i=1}^{m} c_i$$

3.5.4 Surrogate Relaxation

The *Surrogate Relaxation* of MKP, relative to a non-negative vector *(n)*, is defined as

$$\text{maximize} \quad z_\pi = \sum_{i=1}^{m} \sum_{i}^{n} p_j x_{ij}$$

$$\text{subject to} \quad \sum_{i=1}^{m} \pi_i \sum_{j=1}^{n} w_j x_{ij} \leq \sum_{i=1}^{m} \pi_i c_i$$

$$\sum_{i=1}^{m} x_{i,j} \leq 1 \quad (\ i = 1,\ldots,m;\ j = 1,\ldots,n)$$

$$x_{i,j} = 0 \text{ or } 1 \quad (\ i = 1,\ldots,m;\ j = 1,\ldots,n)$$

In this case too, Hung and Fisk [34] have set the n_i's equal to the optimal dual multipliers of constraints (3.5) in the continuous relaxation, that is, $\pi_i = p_t\ /\ w_t$ for all i. Martello and Toth [17] have proved that setting $\pi_i = k$ (k any positive constant) for all i's leads t o the minimum value of the optimal solution of the surrogate relaxation, that is, to the tightest upper bound this relaxation can give for MKP.

If we set $y_{i=} \sum_{i=1}^{m} x_{i,j}$ $\ (\ j = 1, \ldots, n\ )$, the surrogate relaxation of MKP can thus be expressed as the 0- 1 single knapsack problem

$$\text{maximize} \quad z_\pi = \sum_{j}^{n} p_j y_j \qquad (3.9)$$

subject to $\sum_{j=1}^{n} w_j y_j \leq \sum_{i=1}^{m} c_i$ (3.10)

$y_j = 0$ or 1 $(j = 1, \ldots, n)$ (3.11)

Both the Lagrangean and the surrogate relaxation are NP-hard problems, since they require the solution of 0-1 single knapsack problems. However, we can compute upper bounds on the value of z_λ or z_π , and hence upper bounds for MKP in polynomial time. Neither relaxation dominates the other. In general, it can be expected that z_π **,** gives tighter upper bounds when *m* is small or the ratio *n/m* is large, since the surrogate problem allows one to split items between two knapsacks and the number of split items is in this case comparatively small.

3.5.5 Reduction

The size of a 0-1 multiple knapsack problem can be reduced, in a way similar to that presented for KP in Section (3.3), so as t o determine two sets:

$$J1 = \{ \boldsymbol{j} : \sum_{i=1}^{m} x_{i,j} = 1 \text{ in an optimal solution to MKP} \};$$

$$J0 = \{ j : \sum_{i=1}^{m} x_{i,j} = 0 \text{ in an optimal solution to MKP} \};$$

In this case, however, only *JO* allows one to reduce the size of the problem by eliminating the corresponding items, while J1 gives only information useful in reducing the number of nodes of a branch-decision tree, since it cannot specify in which knapsack the items must be inserted.

3.5.6 Exact algorithms

Algorithms for MKP are generally oriented either to the case of low values of the ratio *n / m* or to the case of high values of this ratio. Algorithms for the first class, which has applications, for example, when m liquids, that cannot be mixed, have to be loaded into *n* tanks, have been presented by Christofides, Carpaneto, Mingozzi and Toth [37] and by Neebe and Dannenbring

[38]. In the following we will review algorithms for the second class, which has been more intensively studied. Hung and Fisk [16] have proposed a depth-first branch-and-bound algorithm, where successively higher levels of the decision-tree are constructed by selecting an item and assigning it to knapsacks in decreasing order of their capacities; when all the knapsacks have been considered, the item is assigned to a dummy knapsack, implying its exclusion from the solution: so, each node of the decision-tree generates (m + 1) descendent nodes. The upper bound associated with each node can be computed either as the solution of the Lagrangean relaxation, or the surrogate relaxation, of the current problem, or as the smaller of the two. As was pointed out in Section (3.4.2), vectors (λ) and (π) have been set respectively equal to the optimal dual multipliers of constraints (3.6) and (3.5) in the continuous relaxation of MKP.

Martello and Toth [39] have obtained better computational results through a different branching scheme, which computes at each node a Lagrangean relaxation with $\lambda_j = 0$ for all j (it can easily be verified that this choice is not dominated and does not dominate the Hung -Fisk choice). In the resulting problem (given by equations (3.4), (3.5) and (3.7)), each knapsack can be solved independently of the others. If no item appears in two or more knapsacks, a feasible solution to the current problem has been found and a backtracking can be performed. Otherwise, an item which appears in m′ ($2 \leq m' \leq m$) knapsacks is selected and m' nodes are generated (m′ - 1) by assigning the item to the first (m′ - 1) knapsacks where it appears, the m′-th one by excluding it from them). The m' upper bounds are computed by solving only m' single knapsacks and utilizing part of the solutions previously found for the ascendant nodes. Each bound can be improved by assuming the smaller between it on the one hand and the solution of the corresponding surrogate relaxation on the other. The experimental results given in [35] indicate the clear superiority of the Martello-Toth algorithm over the Hung-Fisk one, both with and without previous application of the reduction algorithm of Section (3.4.5).

A further improvement has been obtained by Martello and Toth [17] through a so-called bound-and-bound algorithm. This modification of a branch-and-bound approach, based on the computation at each decision-node of both an upper bound and a lower bound of the current problem, can be used to solve any 0-1 linear programming problems. A complete description of the general approach can be found in [17].

For MKP, the resulting algorithm consists of an enumerative scheme where each node of the decision-tree generates two branches either by assigning an item j to a knapsack i ($x_{i,j} = 1$) or by excluding j from i ($x_{i,j} = 0$). Stack S_i ($i = 1, . . . , m$) contains those items that are currently assigned to knapsack i or excluded from it. The *current problem* corresponding to (S) is the original problem with the additional constraints given by fixing the items in S_i ($i = 1, . . . , m$) . Let *upper* (S) and *lower* (S) be respectively an upper and a lower bound to this problem. *upper*(S) is computed by procedure SIGMA which solves a surrogate relaxation of the current problem. *lower* (S) is computed by a heuristic procedure, PI, which finds an optimal solution for the first knapsack, then excludes the items inserted in it and finds an optimal solution for the second knapsack, and so on.

The heuristic solution found is stored in $\dot{x}_{i,j}$ ($i = 1, . . . , m$; $j = 1, . . . , n$). At each iteration, the algorithm selects as branching variable the next $x_{i,j}$ such that $\dot{x}_{i,j} = 1$, according to increasing values of i. It follows that, given the current value of i , knapsacks 1, . . . , i - 1 are «completely loaded» (i.e. no further item can be inserted in them), knapsack i is «partially loaded», and knapsacks i + 1,. . . , m are «empty». The main conceptual difference between this approach and a standard depth-first branch-and-bound one is that the branching phase is here performed by updating the partial solution through the solution obtained from the computation of a lower bound. This gives two important advantages:

(a) For all S for which *lower* (S) =*upper* (S), ($\dot{x}$) is obviously an optimal solution to the corresponding current problem, so it is possible to avoid exploration of part of the tree.

(b) For all *S* for which *lower* (S) < *upper* (S) , S is updated through the heuristic solution previously found by procedure PI, so the resulting partial solution can generally be expected to be better than that which would be obtained by a series of forward steps, each fixing a variable independently of the following ones.

A general description of the algorithm follows. It is assumed that items and knapsacks are sorted no that $p_i / w_i \geq p_{i+1} / w_{j+1}$ for $j = 1, . . . , n - 1$ and $c_i \leq c_{i+1}$
for $i = 1 , . . . , m - 1$.

3.5.7 ALGORITHM MKP

1. *[initialization*]

  for i = 1 to m do set $S_i = \emptyset$;

  set $z^* = o$, $x_{i,j} = 0$ for all i and j, i = 1;

  apply procedure SIGMA (i) and let u = upper (S) ***;***

2 **.** *[heuristic]*

  apply procedure PI (i) and let l = *lower (S)*, $(\hat{x})$ = heuristic solution found;

  if $l > z^*$ then set $z^* = l$, $(x^*) = (\hat{x})$;

    if $1 = u$ then go to step 4**;**

3 **.** *[updating]*

  let $j = \min\{v : v \notin S_i$ and $\hat{x}_{i,v} = 1\}$ ($j = 0$ if no such v exists);

  if $j = 0$ then if $i < m - 1$ then set $i = i + 1$ and repeat step 3 ;

      else go to step 4;

    else set $S_t = S_t \cup \{ j \}$ $x_{i,j} = 1$;

      apply procedure SIGMA *(i)* and let u = *upper (S);*

      if $u > z^*$ then repeat step 3;

4. *[backtracking]*

  let j be the last item inserted in S_i such that $x_{i,j} = 1$ ($j = 0$ if no such j exists);

  if j = 0 then if i = 1 then stop;

      else set $S_i = \emptyset$, i = i - 1 and repeat step 4;

    else set $x_{i,j} = 0$, $S_i = S_i - \{v \in S_i : v$ was inserted in S_i after $j\}$;

      apply procedure SIGMA (i) and let u = *upper (S);*

      if $u \leq z^*$ then repeat step 4;

        else go to step *2* **.**

Procedure SIGMA (i)

let $c = (c_i - \sum_{j \in s_i} w_j x_{i,j}) + \sum_{r=i+1}^{m} c_r$, Q={j: $x_{r,i} = 0$ for all r} solve the single knapsack problem defined by the items in Q and by capacity c and let $\overline{z}$ be the solution value;

set $upper\ (s) = \overline{z} + \sum_{r=1}^{i} + \sum_{j \in s_r} p_j x_{r,j}$

Procedure PI *(i)*

let $lower\ (s) = \sum_{r=1}^{i} + \sum_{j \in s_r} p_j x_{r,j}, Q = \{ j : x_{r,j} = o \forall r \}$

set $c = c_i - \sum_{j \in s_i} w_j x_{i,j}, \overline{Q} = Q - S_i, r = i;$

repeat

solve the single knapsack problem defined by the items in $\overline{Q}$ and by capacity c *;*

let $\hat{z}$ be the solution value and store the solution vector in the r-th row of ($\hat{x}$);

set lower (S) = lower(S) + $\hat{z}$;

set Q = Q – {j: $\hat{x}_{r,j} = 1$}; $\overline{Q}$ = Q, r = r + 1;

if $r \leq m$ then set $c = c_r$;

until r > m

for r = 1 to i – 1 do for j = 1 to n do set $\hat{x}_{r,j} = x_{r,j}$;

for each $j \in s_i$ do if $x_{i,j} = 1$ then $\hat{x}_{r,j} = 1$.

3.5.8 Heuristic algorithms

Fisk and Hung [39] have presented a heuristic approach which is based on the exact solution of the surrogate relaxation of MKP and hence requires, in the worst case, a non-polynomial running time. Martello and Toth [40] have presented heuristic procedures, all polynomial in the problem size $m + n$, which can be combined in different ways according to the size and the difficulty of the problems to be solved, to produce various approximate algorithms for MKP. The basic approach can be outlined as follows (assume that items and knapsacks are arranged so that

$p_j / w_j \geq p_{i+1} / w_{j+1}$

for $j = 1, \ldots, n - 1$ and $c_i \leq c_{i+1}$ for $i = 1, \ldots, m - 1$).

1. Determination of an *initial feasible solution.* The simplest way to do this is to apply the greedy algorithm *m* times: to the fist knapsack, then to the second one by using only the remaining items, and so on. Other approaches are also proposed.

2. Rearrangement of the feasible solution found. The purpose of this step is to exchange items between the knapsacks so that each contains items of dissimilar profit per unit weight. In fact, it has been experimentally verified that in this way the subsequent improving algorithms tend to have better performance.

3. *Improvement* of the rearranged feasible solution. This step is performed by applying two procedures. The first considers all pairs of items in the current solution and, if possible, interchanges them between knapsacks should the insertion of a new item into the solution be allowed. The second procedure tries to exclude in turn each item currently in the solution and to replace it with one or more items not in the solution so that the total profit is increased.

The overall complexity of this approach is $O(mn + n^3)$. A Fortran implementation of the method has been given in [40].

3.6 Other single knapsack problems

In this section we review some problems strictly connected with the 0-1 knapsack problem, in the sense that they can be either transformed into a 0-1 knapsack problem or solved through techniques similar to those described in Section 2. We will not examine other knapsack-type problems which conceptually differ from those considered in this thesis, such as, for example, the Multiple Choice Knapsack Problem, the Fractional Knapsack Problem, and the Quadratic Knapsack Problem and so on.

3.6.1. Unbounded knapsack problem

When, for each j $(j = 1, \ldots, n)$, an infinite number of items of profit p_j and weight w_i is available, we have the so-called *Unbounded Knapsack Problem:*

$$\text{maximize} \quad z = \sum_{j=1}^{n} p_i x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} w_j x_j \leq c \tag{3.12}$$

$$x_j \geq o \text{ and integer} \quad (j = 1,\ldots,n)$$

The most efficient dynamic programming algorithm for exact solution of the problem is that of Gilmore and Gomory [26] as improved by Hu [30]. Enumeration methods have been proposed by Gilmore and Gomory [24], Cabot [41] and Martello and Toth [42]. This last method experimentally turned out to be the fastest (Martello and Toth [43,42]). In fact, it can exactly solve problems with up to 10000 variables (with p_i, w_j uniformly random in 1 - 1000 and $c = 0.5\sum_{j=1}^{n} w_j$ in average time of 3.1 seconds, sorting time included.

The unbounded knapsack problem can also be transformed into an equivalent 0-1 problem (and hence solved with one of the algorithms of Section 3) as follows:

Set k = 0;

For j = 1 to n do

set d = [c/w_j], e = 1 ;

while e ≤ d do set k = k + 1, $\bar{p}_k$ = ep_j, $\bar{w}_k$ = ew_j, e = 2e.

The 0- 1 knapsack problem to be solved is

$$\text{maximize} \quad z = \sum_{j=1}^{k} \bar{p}_j \bar{x}_j$$

$$\text{subject to} \quad \sum_{j=1}^{k} \bar{w}_j \bar{x}_j \leq c\,,$$

$$\bar{x}_j = 0 \text{ or } 1 \qquad (j = 1,\ldots,k)$$

where $k = \sum_{j=1}^{n} [\log_2 (c / w_j)]$ (the solution vector can easily be obtained from $(\bar{x})$)

3.6.2 Bounded knapsack problem

When, for each j (j = 1, . . . , n), b_j items of profit p_i and weight w_j are available, we have so-called Bounded Knapsack Problem:

maximize $z = \sum_{j=1}^{n} p_j x_j$

subject to $\sum_{j=1}^{n} w_j x_j \leq c$,

$0 \leq x_j \leq b_j$ and integer (j = 1, . . . , n) ,

where it is usually assumed, without loss of generality, that $\sum_{j=1}^{n} b_j w_j > c$ and $b_j\, w_j \leq c$

for j = 1, . . . , n.

The most efficient method for the exact solution of this problem is Martello and Toth's branch-and-bound algorithm [42].

3.6.3. Change- making problems

Consider an unbounded knapsack problem where $p_j = -1$ for $j = 1, \ldots, n$ and where, in condition (3.12), the equality constraint is imposed. The resulting problem can be expressed as

$$\text{minimize} \quad z = \sum_{j=1}^{n} x_j \tag{3.13}$$

$$\text{subject to} \quad \sum_{j=1}^{n} w_j x_j = c \ , \tag{3.14}$$

x_j and integer (j = 1, . . . , n)

and is generally called the *Unbounded Change-Making Problem.* It can be viewed, in fact, as the problem of assembling a given change, c, using the least number of coins of specified values w_j (j = 1, . . . , n) in the case where, for each value, an infinite number of coins is available.

A recursive algorithm for the exact solution of the problem in the case where one of the w_j's has value 1 (i.e. a feasible solution always exists) was presented by Chang and Gill [45]. The exact solution of the general case has been obtained by Wright [46] through dynamic programming and by Martello and Toth [45,46] through branch-and-bound.

The Martello-Toth algorithm [46] experimentally proved to be very much faster than all the other methods, solving problems up to 1000 variables (with w_j uniformly random in the range 1 - 2000 and $c = \sum_{j=1}^{n} w_j$) in average time of 0.3

seconds, sorting time included, on a CDC-Cyber 730.

The greedy algorithm for the unbounded change-making problem consists in sorting the items according to decreasing weights and then, for j = 1, . . . , *n* , in inserting in the solution as many items of the j-th type as possible. Chang and Korsh [47] , starting from the results obtained by Magazine, Nemhauser and Trotter [48] on the greedy solutions of knapsack problems, gave necessary and sufficient conditions for deciding whether a given instance of the problem is exactly solved by the greedy algorithm. The problem of maximum percentage error when the greedy algorithm does not work has been studied by Tien and Hu [49]. The experimental performance of the greedy algorithm has been analyzed by Martello and Toth [46].

The *Bounded Change-Making Problem* is defined by (3.12), (3.13) and by

$$0 \le x_j \le b_j \text{ , and integer (j = 1, . . . , } n).$$

The only exact algorithm for this problem is the Martello and Toth's branch-and-bound approach [45]**,** which solves randomly generated problems up to 1000 variables (with w_j in the range 1 - 2000, b_j in 1 - 5 and c = 0.5, $\sum_{j=1}^{n} b_j w_j$)in average time of 0.3 seconds, sorting time included, on a CDC-Cyber 730.

3.7 Simulated Annealing

According to Amponsah and Darkwah [80], the concept of Simulated Annealing is derived from Statistical mechanics in the area of natural science. Apiece of regular metal in its natural state has the magnetic directions of its molecules aligned in a uniform direction. In the preparation of alloys the metals are heated to a very high temperature where the molecules acquire higher energy state. The basic structure of the metallic bond breaks down and the magnetic directions of the molecules are oriented randomly. Annealing is the slow cooling of the metallic material so that at the natural temperature conditions the metal will achieve regularity if alignment of the magnetic direction so as to make the metal stable for use. Hasty cooling of solids results in defective crystal. In 1953 Metropolis and others recognized the use of Boltzmain Law to stimulate the efficient equilibrium conditions of a collection of molecules at a given temperature and thus facilitated annealing. When the metal is heated to higher temperature with higher energy state and it is being cooled slowly it is assumed that for a finite drop in temperature the system state change in the sense that the molecules assume new configuration of arrangement. The configuration depends on parameters like temperature, the energy of the system and others. Combining the parameters wee obtained an energy function from which the configuration can be obtained.

In 1993 Kirk Patrick showed how Simulated Annealing of Metropolis could be adopted to solve problems in Combinational Optimization.

The following analogy was made

1) (a) Annealing looks for system state at a given temperature and energy

(b) Optimization looks for feasible solution of the combinatorial problems

2) (a) Cooling of the metal is to move from one system state to another

(b) Search procedure (algorithm scheme) tries one solution after another in order to find the optimal solution.

3) (a) Energy function is used to determine the system state and energy

(b) Objective (cost) function is used to determine a solution and the objective function value.

4) (a) Energy results in evaluation of energy function and the lowest energy state corresponds to stable state.

(b) Cost results in evaluation of objective function and the lowest objective function value corresponds optimal solution.

5) (a) Temperature controls the system state and the energy

(b) A control parameter is used to control the solution generation and the objective function value.

Given an optimization problem, we put it in the form min $f(x)$ such that, $x \in S$

S = feasible solution

The basic SA algorithm is detailed below with the following parameters identification

$\chi^{(i)}$ = Solution (system state); $F(x) =$ Objective function (Energy function);

k = Iteration number (time check in cooling process);

$\delta = f(x^1) - f(x^0)$ (Energy change between states x^1 and x^0);

T = control parameter (Temperature of system);

$g(T) =$ control parameter function (Temperature function);

$e^{-(\delta/T)} =$ choice probability function (Boltzmann probability function);

It provides the condition under which a non-improvement solution is not discarded.

Step 1

(i) Selected initial solution $x^{(0)}$ assign $x^{(b)} = x^{(0)}$

(ii) Set $k = 0$. Selected an initial temperature (control parameter)

$T_k = T_0$ for $k = 0$ assign $T_b = T_0$

(iii) Selected a temperature function $g(T_k)$

Step 2

Choose a solution $x^{(1)}$ in $\mathrm{N}(x^{(0)})$ and compute $\delta = f(x^{(1)}) - f(x^{(0)})$

Step 3

(i) If $\delta = 0$ or, $[\delta > 0$ and $e^{-(\delta/T_k)} \geq \theta : \theta \leftarrow \bigcup = (0,1)]$, accept the new solution $x^{(1)}$. Assign $x^{(0)} \leftarrow x^{(1)}$ and keep the new $x^{(0)}$ such that $x^{(0)} = x^{(b)}$ set $T_b = T_k$

Step 4 If some stopping criteria are satisfied; stop.

Step 5 Update the temperature $T_{k+1} = g(T_K) \leq T_k$ and set $k = k+1$.

3.8 Using Heuristic Scheme in Solving Knapsack Problem

According to Amponsah and Darkwa [60], Heuristic scheme may be used to solve the knapsack problem instead of the method of branch and bound using the steps below

Step 1: Input the vector of weight $\{w_j\}$ and items value $\{c_j\}$

Step 2: Input random initial solution $S_o \in \{0,1\}^n$ and check for feasibility of S_o by using the equation $\sum_{j=1}^{n} w_j x_j \leq b$. If S_o is not feasible discard it and choose another S_o.

Step 3: Find a feasible solution and compute the objective function value $f(S_o)$ by using the objective function $Max \sum_{j=1}^{n} c_j x_j$

Step 4: Obtain a new solution $S_1 \in \{0,1\}^n$ from S_0 by flip operation and check for feasibility, continue flip operations until the solution S_1 so obtained is feasible. Compute the objective function value $f(s_1)$. If $f(S_1) > f(S_0)$ then put $S_0 = S_1$ else retain S_0 and descend S_1 .

Step 5: Repeat step 3 for all feasible S_i

The example below explain these steps

Assuming a traveler has a traveling bag (knapsack) that takes a maximum of b items. The traveler has n items (1,2,3,…,n). the items weigh w_j and are of value c_j to the traveler. How many of items should be placed in the knapsack in order to maximize the total value to the traveler while not exceeding b using the table bellow as an example and given b=10kg.

Item Type	No. of Items	Weight of Items (w_j)	Value of Items (c_j)
1	4	1	2
2	3	3	8
3	2	4	11
4	2	7	20

The knapsack problem can be modeled as follows

Let $x_j = 1$ if the item x_j is include and

$x_j = 0$ if the item x_j is not included

Then we have

Maximize $c_1x_1 + c_2x_2 + ... + c_nx_n \qquad \sum_{j=1}^{n} c_j x_j$

Subject to $w_1x_1 + w_2x_2 + ... + w_nx_n \le c \qquad \sum_{j=1}^{n} w_j x_j \le b$

Substituting the values into the model, we have

$$\text{Max } \{2(x_1 + x_2 + x_3 + x_4) + 8(x_5 + x_6 + x_7) + 11(x_8 + x_9) + 20(x_{10} + x_{11})\} \tag{3.14}$$

$$\text{Subject to } [1(x_1 + x_2 + x_3 + x_4) + 3(x_5 + x_6 + x_7) + 4(x_8 + x_9) + 7(x_{10} + x_{11})] \le 10 \tag{3.15}$$

Note: $x_j = 1$ if the item x_j is include and $x_j = 0$ if the item x_j is not included

Data structure $x = [\{0,1\},\{0,1\},\{0,1\}...\{0,1\}] = \{0,1\}^n$

Assuming $s_1 = \{0,1,1,0,1,0,0,0,1,0,1\}$ is a solution, we check if it is feasible. By substituting into equation 3.15, we have $[1(0+1+1+0)+3(1+0+0)+4(0+1)+7(0+1)] \le 10$

$$2 \quad + \quad 3 \quad + \quad 7 \quad \le \quad 10$$

$16 \le 10$ is false, hence S_1 infeasible.

We define a simple flip operation to be changing a zero to one and vice versa. If we flip the last digit we get $s_2 = \{0,1,1,0,1,0,0,0,1,0,0\}$. By checking for feasibility we obtain

$[1(0+1+1+0)+3(1+0+0)+4(0+1)+7(0+0)] \le 10$

$$2 \quad + \quad 3 \quad + \quad 4 \quad \le \quad 10$$

$9 \le 10$ is true, thus S_2 is feasible.

The objective function value corresponding to S_2 is

$f(S_2) = 2(0+1+1+0)+8(1+0+0)+11(0+1)+20(0+0) = 23$

Hence we find other feasible solutions S_3, S_4… and pick the one with the highest objective function value as our optimal solution.

CHAPTER FOUR

4.1 DISTRIBUTION OF COMMERCIALS

The pile-up of commercials at the FM-station at any point in time is a typical case of a knapsack problem. The FM-station has three categories of time for commercials:

- Prime time (6am, 1pm, 6pm news)
- Adjacencies (five minutes before and after news Prime time)
- Break in programmes.

Each of these categories have different rate (charges) and for that matter income to the radio station. The rate also depends on the duration of the advert.

This practical problem can be represented by a set of entities (types of radio advert), each having an associated value (cost of advert), from which one or more subsets has to be selected in such a way that the sum of the values of the selected entities is maximized, and some predefined conditions are respected. The most common condition is obtained by also associating a size to each entity and establishing that the sum of the entity size in each subset does not exceed some prefixed bound. These problems are generally called knapsack problems, since they recall the situation of a hitch-hiker having to fill up his knapsack by selecting from among various possible objects those which will give him the maximum comfort. In this project we will adopt the following terminologies.

4.2 KNAPSACK ALGORITM

4.2.1 NOTATONS

n_i ; the total number of commercials to be packed

x_i ; the entities(commercials/adverts)

c_i ; the value of commercials

w_i ; weight or the duration it takes to play the commercials

Let us consider a typical prime time (say 6am to 6:30am major news); the total time for adverts to be slotted during the news period is 360 seconds (6 minutes). Most companies will want their commercials to be played during this period. The question is how many pieces of commercials should be selected from the pile of commercials to be played on air (placed in the knapsack) in order to maximize the total value of air time of the radio station while not exceeding b = 360sec.

4.3 SOFTWARE DEVELOPED FOR DATA ANALYSIS(Resource Optimizer)

Visual Basic dot Net was used to develop the software using Simple Heuristic Scheme algorithm. The main features of the software are:-

- ✓ Data Entry :- Data can be entered manually, load from previously saved file or generated
- ✓ Data Verification:- Data entered, retrieved from storage media or generated using the software can be edited and stored or processed.
- ✓ Specify extra options for the algorithm
- ✓ Generation of initial solution and also fliping to generate other solutions to obtain an optimal solution
- ✓ Allows any number of Iterations
- ✓ Report generated can be viewed or saved for future use.

Appendix E is the Visual Basic dot Net codes for Resource Optimizer.

Fig 4.1 Bellow is the initial screen (Form) when one runs the software. The form describes briefly what the software can do. This contains the file menu where all activities start.

Fig 4.2 is the form that contains the buttons for data input, modify an existing data, selecting the number of iterations and process data.

Fig 4.3 Form used to modify data and also to save data.

Fig 4.1

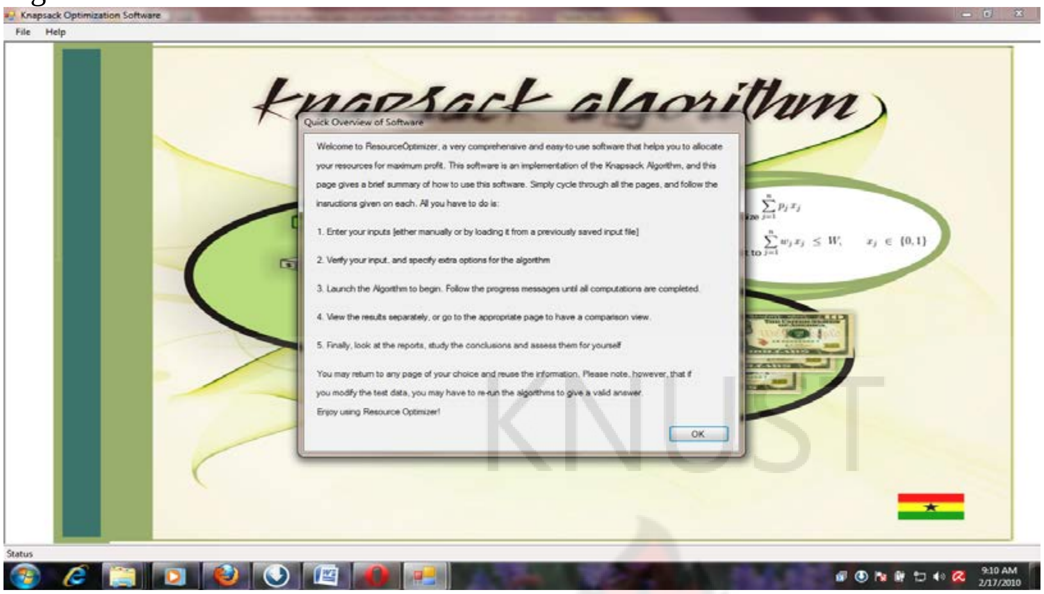

Knapsack Optimization Software
File
Help
Knapsack algorithm
Quick Overview of Software
Welcome to ResourceOptimizer, a very comprehensive and easy-to-use software that helps you to allocate your resources for maximum profit. This software is an implementation of the Knapsack Algorithm, and this page gives a brief summary of how to use this software. Simply cycle through all the pages, and follow the insructions given on each. All you have to do is:
1. Enter your inputs [either manually or by loading it from a previously saved input file]
2. Verify your input, and specify extra options for the algorithm
3. Launch the Algorithm to begin. Follow the progress messages until all computations are completed.
4. View the results separately, or go to the appropriate page to have a comparison view.
5. Finally, look at the reports, study the conclusions and assess them for yourself
You may return to any page of your choice and reuse the information. Please note, however, that if you modify the test data, you may have to re-run the algorithms to give a valid answer.
Enjoy using Resource Optimizer!
OK
Status
9:10 AM
2/17/2010

Fig 4.2

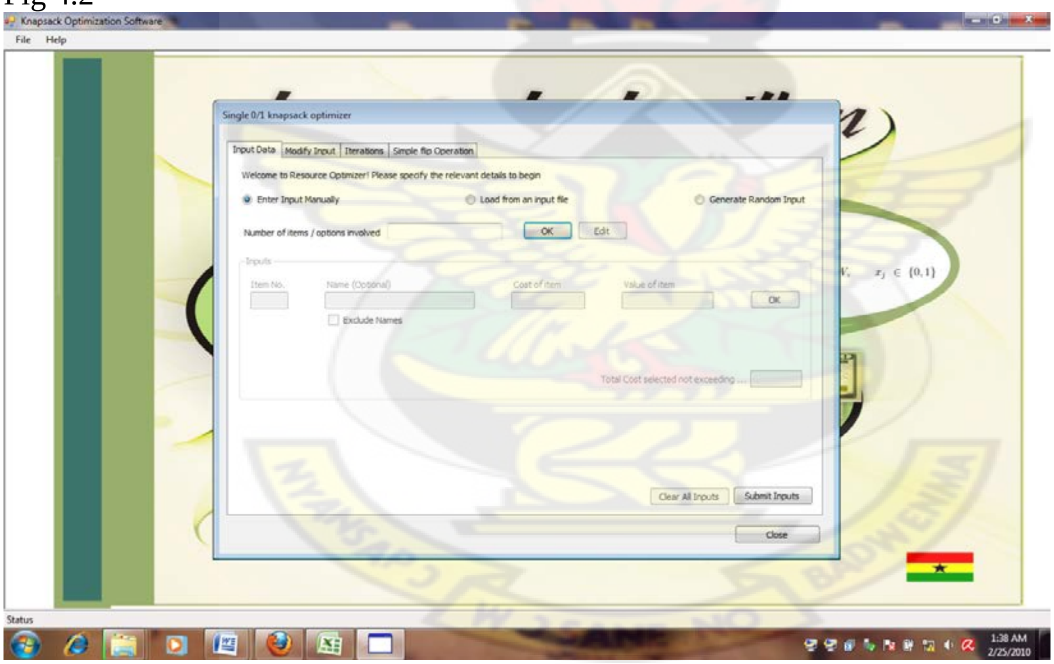
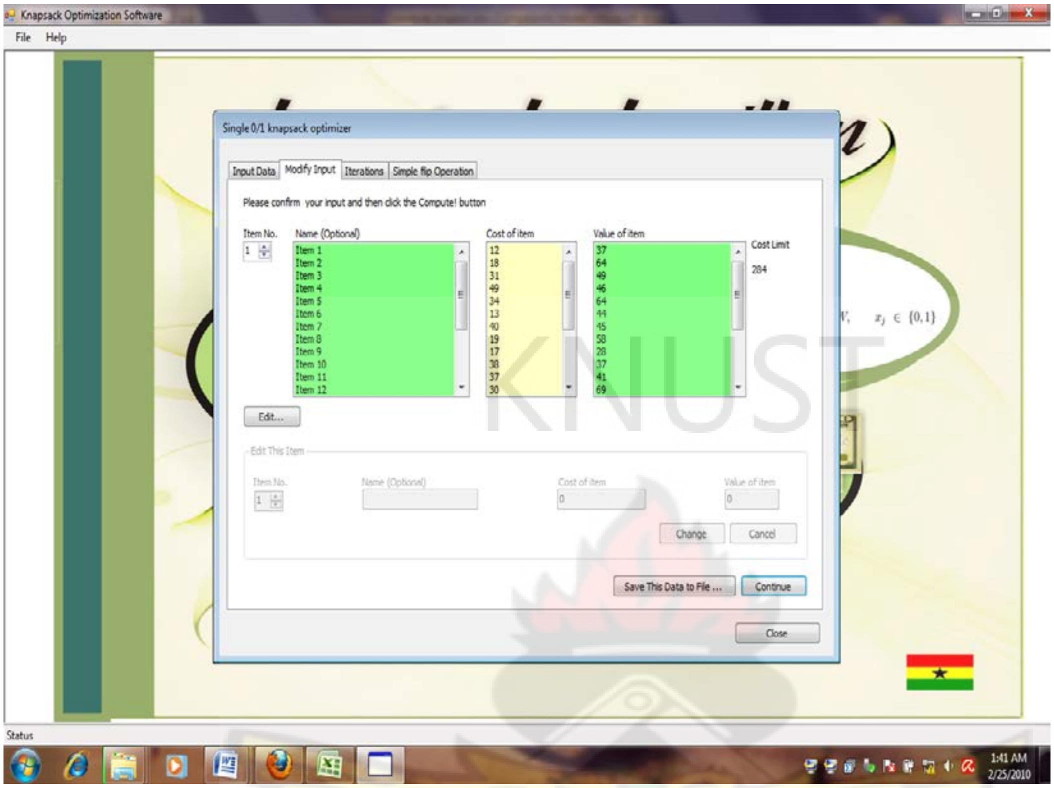

Knapsack Optimization Software
File
Help
Single 0/1 knapsack optimizer
Input Data
Modify Input
Iterations
Simple flip Operation
Welcome to Resource Optimizer! Please specify the relevant details to begin
Enter Input Manually
Load from an input file
Generate Random Input
Number of items / options involved
OK
Edit
Inputs
Item No.
Name (Optional)
Cost of item
Value of item
OK
Exclude Names
Total Cost selected not exceeding ...
Clear All Inputs
Submit Inputs
Close
Status
1:38 AM
2/25/2010

Fig 4.3



4.3 DATA COLLECTION

GCR is a state owned Radio station which depends on government subvention. GCR is however mandated to generate revenue to supplement the government subvention. To this end GCR has various ways of generating additional income. These include sponsorship of programmes, social and funeral announcements, advertisements among others. In this research, we shall focus on advertisements which are slotted in the programmes, before programmes and after programmes. Appendix A is the table for the list of prices for commercials taken from the Garden City Radio. Appendix B, C and D are lists of commercials for Prime Time, Agencies and Break in programmes for a day

4.4 DATA ANALYSIS

Appendix B and C can be reorganized to include the cost for analysis as follows:

Table 4.3b List of Commercials for Prime Time for a day

Company Name (x_i)	**Duration of Advert in seconds** (w_i)	**Value/Cost per advert** (c_i)
A001	30	9.00
A001	30	9.00
A002	60	17.00
A003	25	9.00
A003	25	9.00
A003	25	9.00
A004	30	9.00
A004	30	9.00
A004	30	9.00
A004	30	9.00
A004	30	9.00
A005	29	9.00
A005	29	9.00
A005	29	9.00
A006	45	14.00

A006	45	14.00
A007	57	17.00
A008	30	9.00
A008	30	9.00
A008	30	9.00
A008	30	9.00
A010	30	9.00
A010	30	9.00
A010	30	9.00
A010	30	9.00
A011	45	14.00
A011	45	14.00
A011	45	14.00
A012	40	14.00
A012	40	14.00
A012	40	14.00
A013	30	9.00
A013	30	9.00
A013	30	9.00
A014	30	9.00

A014	30	9.00
A015	30	9.00
A015	30	9.00

Fig 4.4b List of Commercials for Agencies for a day

Company Name **(x_i)**	**Duration of Advert in seconds** (w_i)	**Value/Cost per advert** **(** c_i)
B001	56	11.00
B001	56	11.00
B002	60	11.00
B002	60	11.00
B002	60	11.00
B003	45	9.00
B003	45	9.00
B003	45	9.00
B003	45	9.00
B003	45	9.00
B004	27	7.00
B004	27	7.00
B004	27	7.00

B004	27	7.00
B004	27	7.00
B004	27	7.00
B005	30	7.00
B005	30	7.00
B005	30	7.00
B006	30	7.00
B006	30	7.00
B006	30	7.00
B007	30	7.00
B007	30	7.00
B007	30	7.00
B007	30	7.00
B007	30	7.00
B001	45	9.00
B001	45	9.00
B001	45	9.00
B002	30	7.00
B002	30	7.00
B008	30	7.00

B008	30	7.00
B008	30	7.00
B009	50	11.00
B009	50	11.00
B009	50	11.00
B009	50	11.00
B010	60	11.00
B010	60	11.00
B010	60	11.00
B010	60	11.00
B011	45	9.00
B011	45	9.00
B011	45	9.00
B012	30	7.00
B012	30	7.00
B012	30	7.00
B013	30	7.00
B013	30	7.00
B014	30	7.00
B014	30	7.00

B014	30	7.00
B015	29	7.00
B015	29	7.00
B015	29	7.00
B015	29	7.00
B015	29	7.00
B016	30	7.00
B016	30	7.00
B017	45	9.00
B017	45	9.00
B018	30	7.00
B018	30	7.00
B019	57	11.00
B020	60	11.00
B020	60	11.00
B020	60	11.00

Results for analysis of data in Appendix B using the software (Resource Optimizer) is shown below.

4.5 RESULTS

The software generated an initial data structure of

S_0 = {0 0 0 1 0 0 0 1 0 0 0 1 0 1 1 0 1 0 1 0 1 0 0 0 0 0 1 0 1 1 1 0 0 0 1 1 1 0}

The total time for Prime commercials for a day is 900sec (15min).

The objective function value corresponding to S_1 is

$F(S_1)$ = GH₵ 168.00

Number of loops to obtain optimal solution = 115

Optimal Solution: $S_{115} =${0 0 0 1 0 1 0 1 1 1 1 1 1 0 1 1 1 1 0 1 0 1 0 1 0 1 1 1 1 1 1 0 0 1 1 1 1 1}

$F(S_{115})$ = GH ₵ 282.00

Prime Time adverts selected to be played on air are shown on the table bellow.

Table 4.3c Selected Adverts for PrimeTime News

Selected Adverts for PrimeTime News -06:00 -06:30 & 013:300 - 18:30 hours GMT Total time available is 900 secs**(15min)**				
Adverts	Time in sec	No of Adverts	Total time	Amount (GH ₵)
A003	25	2	50	18.00
A004	30	4	120	36.00
A005	29	2	58	18.00
A006	45	2	90	28.00
A007	57	1	57	17.00
A008	30	2	60	18.00

A010	30	2	60	18.00
A011	45	3	135	42.00
A012	40	3	120	42.00
A013	30	3	90	27.00
A015	25	2	50	18.00
Total		26	890	282.00

The total time for Adjacencies commercials for a day is 1800sec (30min).

Results for analysis of data in Appendix C using Resource Optimizer is shown below.

S_1= {0 1000011101100000110100100110110110101101000010110110101011101111000011}

The objective function value corresponding to S_1 is

$F(S_1)$ = GH₵ 294.00

Number of loops to obtain optimal solution = 332

Optimal Solution:

$S_{332} =${00011001111110111O1101111111101110011001110101101110111100110010010111}

$F(S_{332})$ = GH ₵ 382.00

Adverts selected to be played on air are shown on the table below.

Table 4.4c Selected Adverts for News Adjacencies

Selected Adverts for News Adjacencies 1800 sec(30min)				
Advert	Time in sec	No of Adverts	Total time	Amount (GH ₵)
B001	45	2	90	18
B002	30	2	60	14
B007	30	2	60	14
B008	30	2	60	14
B009	50	3	150	33
B010	60	3	180	33
B011	45	3	135	27
B012	30	5	150	35
B013	30	2	60	14
B014	30	6	180	42
B015	29	5	145	35
B016	30	1	30	7
B017	45	3	135	27
B018	30	2	60	14
B020	60	5	300	55
Total		46	1795	382

4.6 Summary of Results

Prime Time Advertisement		
	Software	Manual
Total number of Adverts Selected	26	15
Total Time	890	900
Revenue from Adverts	282	255

Adverts for News Adjacencies		
	Software	Manual
Total number of Adverts Selected	46	30
Total Time	1795	1800
Revenue from Adverts	382	330

CHAPTER FIVE

5.1 Conclusion and future Work

The pile of commercials at most radio stations are a typical case of a Knapsack optimization problem. Given that a 0–1 knapsack optimization problem is NP-hard, we used two heuristic procedures to solve the problem of selecting which commercials to be played on air given a time limit. These are the simple heuristic flip method by Amponsah and Darkwah, [58] and the Genetic algorithm (GA).

Among the major areas of our research were the use of the Knapsack problem for selecting adverts in critical situations such as the prime time news and news adjacencies. Previously, adverts were selected from a pile randomly or based on their value but now an effective, efficient and more scientific means can be used. Piling up of adverts is reduced significantly since more of them are played within a given period of time.

Using Visual Basic dot Net programming language, we developed a software for simple flip operation, genetic and simulated annealing algorithm to select commercials from a pile of commercials given limited time constraint. However it can be applied to any situation where given a set of items, each with a weight and a value, and we are to determine the number of each item to include in a collection so that the total weight is less than a given limit and the total value is as large as possible. Examples of such situations are capital investment, cargo handling, banking among others.

Since Garden City Radio is owned by the state, in an event where management may have to include certain adverts for national interest, we suggest that total time for such advert(s) should be deducted from time available before selecting the others to compete for the remaining time.

The two heuristic procedures used are among the few heuristic procedures that can be used to solve the radio adverts selection problem. The Simulated Annealing (SA) procedure, which is also polynomial time heuristic procedures, was considered in the software development but was not used in our research work.

5.2 Recommendation

We recommend to GCR to use the software in selecting adverts to be played on air.

There are situations where too many adverts also spoil the beauty of a program and makes it boring. In this case, it is considered as more than one constraint (i.e. both adverts limit and time limit, where the adverts number and limited time are not related), we get the multiple-constrained knapsack problem. We recommend that in future such a situation should be considered.

TABLE OF CONTENT

LIST OF TABLES