Solving the Quadratic Knapsack Problem with a New Heuristic

Ylaney Ramlall (ylaney@aims.ac.za)
African Institute for Mathematical Sciences (AIMS)

Supervised by: Professor Montaz Ali
University of the Witwatersrand, Johannesburg South Africa

12 October 2018

Submitted in partial fulfillment of a structured masters degree at AIMS South Africa

Abstract

The quadratic knapsack problem (QKP) is a problem that has been extensively studied in the field of optimization. The problem emerges when given the choice to make an optimal decision from a set of items that are pairwise-wise related. The quadratic knapsack problem is a modification of the well known knapsack problem which is a decision problem that consists of deciding which choice of items to make in the case where there are no items that depend on one another. The quadratic knapsack problem consists of maximizing a quadratic profit function with binary variables subject to a single knapsack constraint. This combinatorial optimization problem, has diverse applications to important real world decision making, for example in the location of satellites, airports, railway stations and freight terminals. The QKP has been shown to be NP-hard in the strong sense and there does exist exact solution techniques such as the well known branch-and-cut algorithm but it is often more practical to apply intuitive heuristic methods since many industrial applications of QKP are satisfied with approximate solutions. The objective of this paper is to solve the quadratic knapsack problem by using new heuristic. The newly suggested heuristic will be tested and applied to a practical application from finance.

Declaration

I, the undersigned, hereby declare that the work contained in this research project is my original work, and that any work done by others or by myself previously has been acknowledged and referenced accordingly.

Ylaney Ramlall, 12 October 2018

Contents

1. Introduction

Decisions making is a vital part of life and can range from the trivial to highly complex. The complex nature of problems today have many competing factors and many manual methods can regularly generate bad decisions that could have negative consequences. Consequently there is a need for concise methodology for good decision making. The field of Optimisation was developed as a branch of Operational Research. Optimisation contributes significantly to decision making as it provide a technical framework for optimal decisions.

The optimisation model is the result of optimising some objective function that satisfy conditions of the decision variables.The performance of the system is measured and used to identify the objectives which are dependent on decision variables i.e. the characteristics of the system. These features are optimised and are formulated as a mathematical function of the decision variables.

This mathematical function is know as the objective function and is the various measure of performance of the mathematical model of the system. In order to optimise the objective function the function is minimised or maximised on a set of constraints as required by the problem.

An optimisation problem is defined as:

$$\min_{x \in S} f(x), \tag{1.0.1}$$

where x stands for the decision variables, S is the feasible set where:

$$f : S \subseteq R^n \to R.$$

The feasible set is defined by:

$$S = \{x \in R^n : g(x) \geq 0, h(x) = 0\},$$

where $g(x), h(x)$ have real component functions, example $g(x)^T = \{g_1(x), g_2(x), \ldots, g_m(x)\}$.

Optimisation has two main classes which are continuous optimisation and discrete optimisation. When dealing with continuous optimisation the decision variables assume real values which where investigated by Nocedal and Wright (1999), Griva et al. (2009). With discrete optimisation the discrete values are confined to integer values. There exists a special category of optimisation studied which merge both continuous and integer decision variables called mixed-integer programming (Yajima and Fujie (1998)).

Discrete optimisation is referred to as Integer programming when the decision variables are constrained and take on only integer values.The study of best selection with respect to some suitable objective function is known as combinatorial optimisation. Combinatorial optimisation is a special class of discrete optimisation which maximises or minimises an objective function over a finite collection of subsets over the set. Combinatorial problems are characterized by an input which is a general description of conditions and parameters defining the properties of a solution. They involve finding a grouping, ordering, or assignment of a finite set of objects that satisfy the given conditions. The resulting candidate solutions are combinations of objects that do not necessarily satisfy all given conditions but solutions that are taken from the candidate solutions will satisfy all given conditions.

1.1 Knapsack Problem

Suppose an AIMS student was to set out on a planned hiking trip up Table Mountain but could only carry a single knapsack with limited capacity. The student has N different items that she can chose from to be place in the knapsack. Each item has two characteristics; the weight and attributed value that the student needs to consider when making her decision and when all N items are considered, the knapsack's capacity is exceeded, see Figure 1.1. The student now has to figure out how to combine the different items to yield the best worth from the knapsack. The predicament described by the AIMS's student situation is an illustration of what can be called the *knapsack problem*. Simply put the knapsack problem is the problem of deciding on a combination of items to be chosen from larger set under weight restrictions while ensuring an optimal profit.

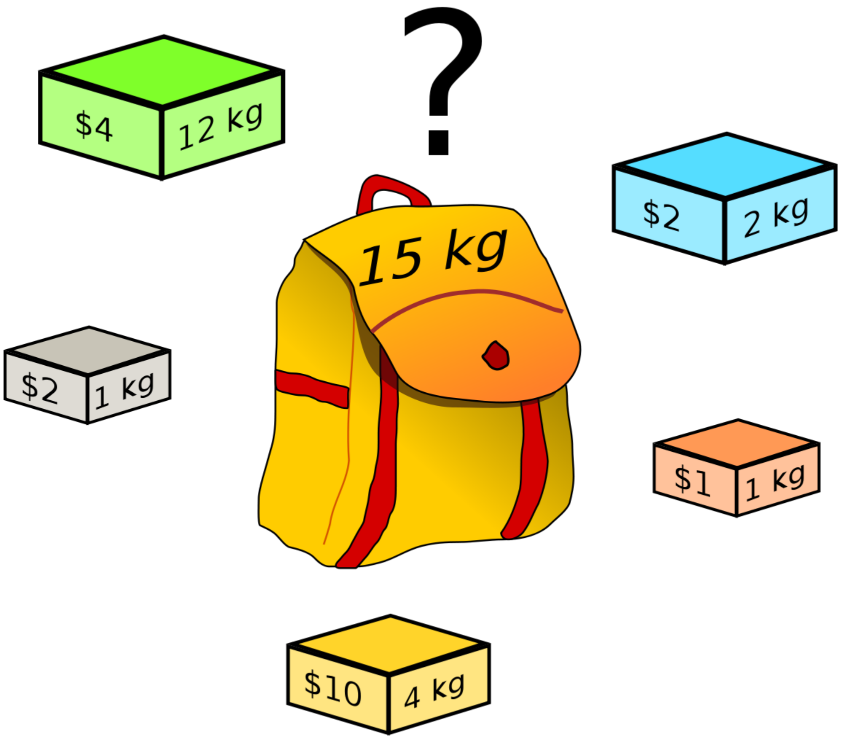

Figure 1.1: Illustration of knapsack problem WikiKP (2018)

The knapsack problem is a widely recognised combinatorial optimisation problem that has gained a reputation for being one of the simplest non-trivial integer programming models applied to binary variables with only a single constraint with positive coefficients. The study of the knapsack problem is also important from a theoretical point of view as it regularly occurs as a sub-problem for more complicated combinatorial optimisation problems.

When large scale industrial data can be modelled by the knapsack problem viable methods for finding solutions emerge that result in a variety of applications. A well studied application of the knapsack problem is in the transport industry, where the problem has been able to outline the logistics of cargo air-planes and cargo ships. Other areas of interest in study of the knapsack problem has been in telecommunication, cutting and packing,reliability, budget allocation, advertisement, production management.and investment.

1.2 The Quadratic Knapsack Problem

The knapsack problem has evolved into many variations over time due to the problem being studied intensively and being modified according to specified needs. This intensive research by Pisinger (2007) and Martinelli and Contardo (2013) is a testament to the practical relevance of the knapsack problem and will continue to develop in the future as new data and applications emerge. However the focus of this essay will be the quadratic knapsack problem which is a modifications of the knapsack problem. The QKP uses a quadratic objective function instead of the classical linear version.

When evaluating the knapsack problem, the profit of an individual item is usually kept independent of other items chosen with it. But in most real world applications the choice of an item and its corresponding probability is interdependent on how well this choice fits together with other items chosen. Gallo et al. (1980) introduced the first formalization of the binary quadratic knapsack problem which established a possible method to claim profit from the item itself as well as the profit contributed by the selection of other items with it, if selected together.

Like the knapsack problem, the quadratic knapsack problem is a combinatorial optimisation technique that can be applied to a variety of large scale industrial problems, from applications initially investigated by Gallo et al. (1980) to Witzgall (1975) who developed deep insight to the applications of the quadratic knapsack problem in the telecommunications sector and Rhys (1970). Johnson et al. (1993) who divulged deeply into the compiler design problem. And at its root the quadratic knapsack problem is the knapsack problem which has allowed for the utilization of this newer technique in the deeper study in models arising from problems in the location of airports, railway stations or freight handling terminals which has always been a well known application of the knapsack problem. The area of application of the quadratic knapsack problem that has received the widest attention has been in the selection of a portfolio optimization in the finance area, where one wants to invest a limited amount of money in the financial market by buying share's of companies or other financial assets where any set of items are correlated in some manner.

1.3 Heuristic

A heuristic is a technique for finding solutions to problems by using an intuitive strategy where a solution is achieved through insightful interpretation of the problem at hand so that a solution or the close approximation to an exact solution can be found.

Practical real world decision making is often complicated due to the available data being embedded with restrictive and often unforeseen errors. Simple heuristic methods were adopt at obtaining approximate solutions especially when managing large scale data sets that contain unexpected errors.

The solutions generated by heuristics, in general are a logical solution to practical problems where in some cases, an optimal solution is at best just a guide and would not provide an exact solution due to the known errors. Another positive attribute of heuristics is that they provide an option to solving large scale problems without the need for expensive computer hardware and complex software that would be required by sophisticated algorithm finding exact solutions.

Heuristics can also be combined with optimization algorithms to improve their efficiency i.e heuristics are often used to generate high quality seed values. As a result of heuristic's variability and practicality of nature heuristics are chosen regularly as an application to handling industrial size combinatorial optimisation problems.

A heuristic algorithm that performs wells should have the following attributes:

- Require reasonable computational effort.
- Generate solutions close to optimal solution.
- Must not generate "poor" solutions and must be easily adaptable.
- Be simple to explain and implement.

Heuristics have wide spread applications but for large scale problems which generally in industrial applications and given time complexity of the exact solution, heuristic has a role to play.

1.4 Complexity Classes

$\mathcal{P}$ is the set of decision problems that can be solved completely in polynomial time. These can be translated as the collection of problems that can be solved swiftly. $\mathcal{NP}$ is the non-deterministic polynomial set of decision problems that given a positive answer i.e. a YES result will have efficiently verifiable proofs, these proofs would be validated by deterministic computations conducted in polynomial time. This means given a solution to a problem in the $\mathcal{NP}$ class and it is verifiable to be a solution in polynomial time. $\mathcal{NP}$-hard is the non-deterministic polynomial acceptable problems which is the set of decision problems that simply put are "at least as hard as the hardest problems in $\mathcal{NP}$". $\mathcal{NP}$-complete is the set of decision problems that belong to both the complexity sets $\mathcal{NP}$ and the $\mathcal{NP}$-hard.

The most difficult decision problem to solve in the $\mathcal{NP}$ complexity classes are the $\mathcal{NP}$-complete problems Cook (1971). Once a problem is established to be an $\mathcal{NP}$-complete problem and this problem can be solved by a polynomial-time algorithm, then it follows that all the problems in $\mathcal{NP}$ can be solved by a polynomial-time algorithm.

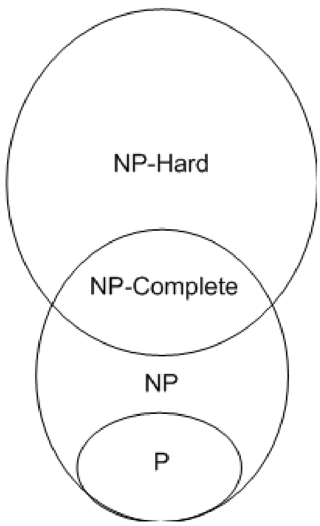

Figure 1.2: Diagram for the various complexity groups StackOverflow (2018)

The knapsack problem was shown to be $\mathcal{NP}$-complete by Garey and Johnson (2002). As such a solution for the problem can be found in polynomial time but no known algorithm can identify a solution efficiently. Karp (1972) proved that the knapsack problem is $\mathcal{NP}$-hard, by illustrating that its associated decision problem are $\mathcal{NP}$-complete. But the knapsack problem was also be shown to be solvable in pseudo-polynomial $\mathcal{O}(nc)$ time by Bellman (1957) who verified his finding by using dynamic programming. This then results in the knapsack problem being $\mathcal{NP}$-hard in the weak sense only.

With current research establishing that the knapsack problem is $\mathcal{NP}$-hardness, heuristics present the only viable option for a variety of difficult optimization problems that need to be frequently solved in real-world applications. An interesting and vital field of mathematics and computer science occurred as a result the knapsack problem being $\mathcal{NP}$-hard which is known as the knapsack cryptosystem. Public-key cryptosystem is based on a special case the knapsack cryptosystem and was developed by Merkle (1980)and Hellman (1979). This special application of the knapsack problem is also similar to the subset sum-problem, where the solution is time-consuming to compute as it belongs in the complexity class of NP.

1.5 Structure of the Essay

The aim of this essay is to apply a newly suggested heuristic algorithm for the quadratic knapsack problem and used the new heuristic method to solve a practical application of portfolio selection in finance.

The outline of this paper is structured as follows :

- An overview and formulation of the quadratic knapsack problem, which is a modification of the knapsack problem using quadratic objective functions will be introduced and discussed in **Chapter 2**.
- In this paper the quadratic knapsack problem is investigated with the use of an adoption of the newly suggested heuristic.This approach is considered in two steps.
  - The heuristic is first applied to test examples to investigate the algorithm for solutions for the quadratic knapsack problem, this will be covered in **Chapter 3** .
  - The second approach is to model the quadratic knapsack problem and solve for solutions using the new heuristic. In doing so a practical case of selecting real South African stocks into a portfolio where by the selection of stocks will be done by using new heuristic that will be tackled in **Chapter 4**.

2. Knapsack Problem

Making decisions have varied forms of difficulty but one of the easiest possible form of a decision is a choice between just two outcomes either a 'YES' or a 'NO' a single boolean value. A problem of this type in optimisation is called the *decision problem*. This binary decision is formulated in a quantitative model as a binary variable $x \in \{0, 1\}$ with $x = 1$ when the first alternative is chosen and $x = 0$ otherwise which indicates the selection of the second option or the rejection of the first option.

In the simplest version of a linear decision model the resulting solution of the complete decision process is evaluated by a linear combination of the values associated with every binary decision made. When evaluating the pairwise interdependencies between decisions, a quadratic function will be considered to represent the outcome of a decision process which is possible since binary variables are used such that $x \in \{0, 1\}$ with $x^2 = 1^2 = 1$ when the first alternative is chosen and $x^2 = 0^2 = 0$ otherwise which indicates the selection. The feasibility of a particular class of alternatives may be very complex to demonstrate in practice because the binary decisions may influence or even contradict each other in real world instances, this is the case when observing the selecting of stocks or assets for a financial portfolio that could result in shares lowering another share's value by association in the portfolio.

2.1 Linear Knapsack Problem

The knapsack problem which has been extensively studied is a well established combinatorial optimisation problem that aims to provide the best selection and configuration of a collections of items that keeps in mind the objective function. The study of the linear knapsack problem has been carried out extensively in discrete programming problems which has contributed to these three most important reasons;

(i) The problem can be generalised to a simple integer linear programming problem. Integer linear programming is an area of optimization in which the variables are restricted to be integer values, where the objective function and the constraints (other than the integer constraints) are all linear.

(ii) The linear knapsack problem is often embedded as a sub problem in more complex problems.

(iii) The linear knapsack problem has many applications for real world problems and can be applied in a practical and feasible way to solve these problems.

The mathematician (Dantzig (1957)) was the first to mathematically formulate the knapsack problem.

2.1.1 Definition. : The knapsack problem is defined as follows: from the collection of $N = \{1, ..., n\}$ are all possible items that can be contained in the knapsack. Given n-tuples of positive numbers $(v_1, v_2, ..., v_n)$, $(w_1, w_2, ..., w_n)$ and $W > 0$

- $w_j =$ the weight of item j, for $j \in \{1, 2, .., n\}$
- $v_j =$ the value associated with item j, for $j \in \{1, 2, .., n\}$
- W limited weight capacity of the knapsack

$$\text{Maximize} \quad \sum_{j=1}^{n} v_j x_j, \quad x_j \in \{0,1\}, \quad x_j \quad \text{is the decision variable} \tag{2.1.1}$$

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

$$\text{where} \quad W < \sum_{j=1}^{n} w_j.$$

Given n-tuples of positive numbers $(v_1, v_2, ..., v_n)$, $(w_1, w_2, ..., w_n)$ and $W > 0$, the objective of the linear knapsack problem is to find a suitable subset of $S \subseteq N$ of items that each have a weight w_i and a value v_i, (Montaz Ali (2018)).

2.1.2 Assumptions made when using knapsack problem. There are some assumptions to made when apply the knapsack problem to avoid tedious and trivial situations of needless sub cases (Kellerer et al. (2004a)).These assumptions are made without a loss of generality and may be applied to other variations of the knapsack problem that will not be discussed in this research.

(i) Only problems with two or more items will be considered, i.e. we assume $n \geq 2$. The assumption is made since when the case $n = 1$ is considered, the trivial single binary decision of whether to included the item or not, is the only consideration made in the whole problem.

(ii) Given a single item j:

$$\text{Assume} \quad w_j \leq W, \quad j = \{1, ..., n\}. \tag{2.1.2}$$

Otherwise, any binary variable corresponding to an item j that violates (2.1.2) is set to 0. Hence j is immediately rejected and will naturally not be included in the knapsack where W is the total capacity of the knapsack.

(iii) Given a situation where all n items would fit into the knapsack and the problem would be solved trivially by packing them all into the knapsack.

$$\text{Assume} \quad W \geq \sum_{j=1}^{n} w_j, \quad j = \{1, ..., n\}. \tag{2.1.3}$$

Otherwise, any binary variable corresponding to an item j that violates (2.1.3) is set $x_j = 1$ for $j = \{1, ..., n\}$

(iv)

$$\text{Assume} \quad p_j > 0, \quad w_j > 0, \quad j = i, ..., n. \tag{2.1.4}$$

Without loss of generality and assumption that all the profits and weights will only take on positive values. The are ways in which cases that violate (2.1.4) can be transformed to satisfy this assumption but since we will not be dealing with these instances those transformations will not discussed.

The versatility of nature of the knapsack results in many variations of the problem and depending on the need of the problem being tackled identify these variants prove crucial in model the individual classes of the knapsack problem.

The simple case of deciding whether each individual item is included or not from the knapsack is called the $0-1$ knapsack problem. The bounded knapsack problem, appears when given multiple copies of each object in a subset $S \subseteq N$ where N is the total number of items to choose from and up to $c \in \mathbb{Z}$ copies of each object may be included in the knapsack. And similar to the bounded knapsack problem exists the unbounded knapsack problem, where once more copies for each object occur, but in this unbounded case there are no restrictions on the number of copies that may be placed in the knapsack. And one of the most complicated variant of the knapsack problem is the multiple knapsack problem, where by there exists m distinct knapsacks will be filled by elements of S in order to optimise the profit. The following definitions elaborate these knapsack problem variations.

2.1.3 Definition. (0-1 Knapsack Problem, (Kellerer et al. (2004a)))

$$\text{Maximize} \quad \sum_{j=1}^{n} v_j x_j \tag{2.1.5}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} w_j x_j \leq W, \quad x_j \in \{0,1\} \tag{2.1.6}$$

2.1.4 Definition. (Bounded Knapsack Problem (BKP)(Kellerer et al. (2004a))

$$\text{Maximize} \quad \sum_{j=1}^{n} v_j x_j \tag{2.1.7}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} w_j x_j \leq W, \quad 0 \leq x_j \leq c \tag{2.1.8}$$

2.1.5 Definition. (Unbounded Knapsack Problem (UKP)(Kellerer et al. (2004a))

$$\text{Maximize} \quad \sum_{j=1}^{n} v_j x_j \tag{2.1.9}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} w_j x_j \leq W, \quad x_j \geq 0 \tag{2.1.10}$$

2.1.6 Definition. (Multiply-constrained Knapsack Problem(MKP)(Kellerer et al. (2004a)) The Multiply-constrained Knapsack Problem is modelled if there exists more than one constraint.

$$\text{Maximize} \quad \sum_{j=1}^{n} v_j x_j \tag{2.1.11}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} w_{ij} x_j \leq W_i, \qquad \text{for all} \quad 1 \leq i \leq m \tag{2.1.12}$$

$$x_j \geq 0 \quad x_j \quad \text{integer for all} \quad 1 \leq j \leq n \tag{2.1.13}$$

2.2 The Quadratic Knapsack Problem(QKP)

The quadratic knapsack problem (QKP) is a non-linear combinatorial optimization problem that is an extension of the classical knapsack problem with many applications in multiple discipline areas. Given a knapsack with limited capacity and a set of items that a selection can be made from, where each item has a positive weight and produces a profit if selected. The selection of items placed in the knapsack is performed on the bases that the items selected maximize the overall profit without exceeding the capacity of the knapsack. The quadratic knapsack problem varies from the classical knapsack problem in that the QKP takes into consideration the interdependency of items that are selected to be placed in the knapsack so that the pairwise profit is maximised and contributes positively to the overall value of the knapsack.

The quadratic knapsack problem was introduced by Gallo et al. (1980) and like in the knapsack problem if given an item j selected from n items there exits a corresponding weight w_j where the total positive integer capacity of the knapsack is represented by W. By introducing the variable x_j which has the binary values 1 if item j is selected and 0 if item j is not selected thus the problem is formulated as follows:

$$\text{Maximize} \quad \sum_{j=1}^{n} c_j x_j + \sum_{j=1}^{n-1} \sum_{i=j+1}^{n} d_{ij} x_i x_j, \tag{2.2.1}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} w_j x_j \leq W, \quad j = \{1, 2, .., n\}, \tag{2.2.2}$$

$$\text{where} \quad x_j \in \{0, 1\}, \quad \text{Max} \quad w_j \leq W < \sum_{j=1}^{n} w_j.$$

The objective of the quadratic knapsack problem is to ensure that subsets of items are selected for the knapsack whose combined weights do not exceed the capacity of the knapsack. In doing so the model ensures the maximisation of the overall profit. The profit gained from choosing a specific pair or combination of items is expressed by a supplementary term in the objective function.

This additional term that is being maximised is a n x n non-negative integer profit matrix $P = (p_{ij})$ such that (2.2.1) can be written as:

$$\text{Maximize} \quad \sum_{i \in N} \sum_{j \in N} p_{ij} x_i x_j \quad \text{where} \quad N := \{1, ..., n\}, \tag{2.2.3}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} w_j x_j \leq W, \quad j = \{1, 2, .., n\},$$

$$\text{where} \quad x_j \in \{0, 1\}, j \in N \quad \text{Max} \quad w_j \leq W < \sum_{j=1}^{n} w_j,$$

If item j is selected then corresponding profit p_{jj} is achieved and if both items i and j are selected as a pair, the profit earned is $p_{ij} + p_{ji}$. The profit matrix is symmetric i.e.,

$$d_{ij} = p_{ij} + p_{ji} \quad \text{for all} \quad i, j \in N \tag{2.2.4}$$

Notice, that if negative weights $w_j < 0$ are present, we may flip variable x_j to $1 - x_j$. If $w_j > W$ we may fix $x_j = 0$, and if $w_j = W$ then we may decompose the problem. Hence,normally $0 \leq w_j < W$.

The quadratic knapsack problem has a wide range of industrial and technical problems that it can be applied to. These applications include project selection, location of airports, freight, railways stations, handling terminals and portfolio optimisation which is the application that will be concentrated on in this essay. And inherited from the knapsack problem characteristic's the QKP is also intensively studied for theoretical interests as it regularly occurs as a sub-problem in other optimisation problems. But a big challenge for the quadratic knapsack problem is present by research like Alberto Caprara (1998) who has shown that the quadratic knapsack problem is known to be strongly $\mathcal{NP}$-hard. This conclusion on the computational complexity of the problem poses some complications in that it is unlikely that a pseudo-polynomial time algorithm exists. Hence it is highly practical to employ the use of well performing heuristic methods for solving the quadratic knapsack problem.

3. Solving the Quadratic Knapsack Problem

When studying the knapsack problem the most crucial part of the problem is choosing what strategy will be adopted when solving the problem. Over the last two decade exact methods have been heavily investigated by Bellman (1957) who proposed Dynamic programming to Kolesar (1967) who suggested the Branch and Bound method which are well established approaches to the knapsack problem and more recently Kellerer et al. (2004b) has studied various methods to solving the knapsack problem for larger data.

The study of algorithms provide the cornerstone to the development of solving the knapsack problem as the algorithms are used as an intrinsic tool in the deep understanding of the problem. There are two basic types of algorithms that are utilised for calculating optimal solutions are exact methods and approximation algorithms or heuristics and these methods logically will not perform equivalently when computing the optimum solution. As a result it is expected that an approximation algorithm should theoretically be able to perform better computationally when finding the optimal solution than an exact method. In general the computation performance of an algorithm is measured by the running time and the amount of computer memory space required in order to complete a solution.

When solving and implementing an algorithm for the knapsack problem the following must be considered:

- An efficient algorithm usually requires high performance computing and advance quality software but will if implemented correctly result is a better optimal solution.
- The complexity of computation of an optimisation problem might result in there being an extensively long time in solving the problem and in some cases no efficient algorithm might exist at all to solve the problem for large instances and this is where heuristics provide a good alternative.
- If after much analysis of the feasibility of the problem there is no plausible practicality in the algorithm design the algorithm should be modified and the process repeated in order for the solution to be obtained.
- It is the decision maker's responsibility to use insightful information of the problem at hand to reconstruct the scheme to find the optimum.

Computational complexity of knapsack problems has also played an important role in choosing a suitable solution method. With the knapsack problem being $\mathcal{NP}$-hard and similarly Alberto Caprara (1998) showed that the quadratic knapsack problem is strongly $\mathcal{NP}$-hard. This has lead to the popular belief that there might not be an effective exact algorithm which perform satisfactory in smaller scale data sets but would not be suitable to solve big data versions of this problem, given the acceptable computer space and time. One of the only practical alternatives that prove adapt at handling large scale data with multiple variants is a well consider heuristic method. Since heuristic methods provide a good approximate solution many practitioners find this resort highly realistic, since quite often an approximate of the optimal solution is all that tends to be required.

The heuristic methods also provide the added benefit of being easier to implement when considering the complexity of the problem than an exact approximation and in many cases come close to the optimal solution if not finding the optimum outright as well as also allowing for the internal placement of another heuristic search algorithm to improved the overall performance of the method which was illustrated by Al-Iedani et al. (2017).

Heuristic approaches where also studied by Sahni (1977) intensive interest for the earliest heuristic methods being the polynomial time approximation scheme developed for the knapsack problem. The development of these types of heuristics schemes where done around the guaranteed of the quality of the solution at the inception of the problem which established that work was done in accordance to the qualify of solution that was required.

3.1 The Greedy Algorithm

The greedy algorithm is a well known and highly implemented heuristic and is one of the most important constructive methods amongst single sweep algorithms. It is also one of the most popular heuristics used for solving the knapsack problems. The greedy algorithm prioritises the density of an item as a main criteria in the selection process for items to be chosen for the knapsack, hence the algorithm behaves in a "greedy" manner towards the class of items selecting only the best items until the knapsack has reached its optimal capacity.

Listed below are some of the most important features of the greedy approximation algorithm is the following:

- **Incremental:** The manner in which the greedy algorithm handles the problem generates solutions that are vied as a sequencing set of elements. This results in the solution being created a single element at a time and terminates itself when a the first solution is found in a single sweep of the data.
- **No-backtracking:** If an item is selected for the solution this item is not removed or replace by another item which also enables the decision made by greedy algorithm to not be revisable later.
- **Greedy Selection:** The method of sequentially selecting each additional item for the solution set using the greedy scheme does so in a process that causes the overall profitability of the knapsack to be maximised.
- **Myopic feature:** Decisions are naively made by the greedy method since decisions are based on the situation at that moment and do not consider any other steps. The algorithm only makes selections for the solution set based on the effect an item has on maximising the overall profitability at that stage of selection and does not consider the consequences this item might have at later stages.

3.2 Methods of Solving the Knapsack Problems

The main consideration when solving the knapsack problem is to find an effective algorithm which is practical to apply without the use of costly computation considerations. The greedy algorithm provides a logical technique to tackling the knapsack problem and all the associated applications. A new heuristic technique will be applied to the quadratic knapsack problem in a similar approach to the greedy algorithm.

The greedy approximation algorithm was initially proposed for solving the knapsack problem by Dantzig (1957). This early version established a sorting method based on the density of an item $\frac{v_i}{w_i}$ which is also know as the efficiency function.

To successfully implement the greedy algorithm with n available items, with the associated weights w_i, values v_i for a given item i and the maximum capacity of the bag must be identified.

A score or efficiency function is calculated for each item which represents the profit of the weight ratio $\frac{v_i}{w_i}$,i.e. the items are sorted in a non-increasing order according to the score function:

$$\frac{v_1}{w_1} \geq ... \geq \frac{v_i}{w_i} \geq ... \geq \frac{v_n}{w_n}.$$

Thus the items with the highest efficiency function is added in the order 1,2,...,n into the knapsack with consideration to accumulated weights until the capacity of the knapsack is met. Finally the algorithm will list the items that satisfy the weight limit and returns the maximum profit.

Algorithm 1: Linear greedy algorithm

1. Identify the available items with their weights and values and take note of the maximum capacity of the bag.
2. Use of a score or efficiency function, i.e. the profit to weight ratio: $\frac{v_i}{w_i}$
3. Sort the items non-increasingly according to the efficiency function.
4. Add into knapsack the items with the highest score, taking note of their accumulative weights until no item can be added.
5. Return the set of items that satisfies the weight limit and yields maximum profit.

3.3 The Suggested Heuristic for Quadratic Knapsack Problem

In the linear greedy algorithm for the linear knapsack problem the selection of items are once-off once the scores of all items are calculated. This cannot be said for the quadratic knapsack problem due to the relationship between any pair of items. Hence we have suggested a modification. The resulting algorithm is a heuristic for QKP.

When applying the greedy algorithm to the quadratic knapsack problem a sample of k items from a set of n items are selected such that, $1 \leq k \leq n$. From k items all possible pairs are then selected and these pairs are sorted in a non-increasing order according to the efficiency function:

$$S = \frac{d_{ij} + v_i + v_j}{w_i + w_j} \tag{3.3.1}$$

The pair of items with the highest scores calculated with the efficiency function is added into the knapsack if the maximum capacity of the knapsack is not exceed by the cumulative weight of the pair and thus the process is repeated for each subsequent pair of items until no more pairs can be added to the knapsack and the addition of any successive pair would result in the combined weight of the items in the knapsack exceeding the maximum capacity of the knapsack.

If there is a remaining capacity in the knapsack, items will be added individually according to the linear greedy algorithm until the maximum capacity of the knapsack is met. The procedure is presented in Algorithm 2.

Algorithm 2: Heuristic for Quadratic Knapsack Problem

1. Sample k (say $k = 7$) items from the set of N items (say $N = 15$.
2. Obtain a set of all pairs from the k items.
3. Sort the items non-increasingly according to the efficiency function $\frac{d_{ij}+v_i+v_j}{w_i+w_j}$
4. Add into knapsack the pair of items with the highest score, ensuring that the accumulated weight does not exceed the maximum capacity.
5. Repeat steps 1 through 4 until pairs can no longer be added.
6. Fill remaining capacity with singleton items, using the linear greedy approach.

3.4 Test Examples

Two problems are considered: Linear knapsack problem and Quadratic knapsack problem. In linear knapsack the objective function and constraint(s) are linear and when considering a quadratic knapsack problem the quadratic objective function (2.2.1) is utilised.

3.4.1 Linear Knapsack Problem. Consider the following set of items which are represented by Table 3.1 we use the linear knapsack model see (2.1.1) such that:

- Item i of a total of $N = 28$ items.
- Where v_i is the value of item i.
- And w_i is the weight of item i.
- With the preselected knapsack capacity is W_{max}

Table 3.1: List of items with their value and weight

Index i	Value v_i	Weight w_i
1	2	7
2	6	3
3	8	3
4	7	5
5	3	4
6	4	7
7	6	5
8	5	4
9	10	15
10	9	10
11	8	17
12	11	3
13	12	6
14	15	11
15	6	6
16	8	14
17	13	4
18	14	8
19	15	9
20	16	10
21	13	14
22	14	17
23	15	9
24	26	24
25	13	11
26	9	17
27	25	12
28	26	14

Table 3.2: **Test Results:** List of items selected by Algorithm 1 as well as the total value and weight obtained for each case executed

Knapsack capacity W_{max}	**Selected Items**	**Knapsack Value**	**Knapsack Weight**
Case 1: $W_{max} = 3$	[11 3 12]	11	3
Case 2: $W_{max} = 30$	[11 3 12] [13 4 17] [8 3 3] [25 12 27] [6 3 2] [7 5 4]	70	30
Case 3: $W_{max} = 300$	[11 3 11] [13 4 17] [8 3 3] [25 12 27] [6 3 2] [12 6 13] [26 14 28] [14 8 18] [15 9 23] [15 9 19] [16 10 20] [7 5 4] [15 11 14] [5 4 8] [6 5 7] [13 11 25] [26 24 24] [6 6 15] [13 14 21] [9 10 8] [14 17 22] [3 4 5] [10 15 9] [8 14 16] [4 75 6] [9 17 26] [8 17 11] [2 7 1]	319	269

3.4.2 Remarks. The Table 3.2 shows that for a given knapsack capacity, items are chosen in a manner that allows for the largest possible combined knapsack value and weight using the linear knapsack problem.

Table 3.2 illustrates that if we set the $W_{max} = 3$ the item 11 is chosen with the knapsack value and weight being 11 and 3 respectively. But if $W_{max} = 30$ is set than items 12,17,3,27,2 and 4 are selected with a combined knapsack value and weight being 70 and 30 respectively. It should be noted the Algorithm 1 performs effectively in both these instances since for a relatively small W_{max} the algorithm selects items who's combined weight w match the exact overall capacity of the knapsack and when a $W_{max} = 300$ the algorithm comes sufficiently close to the knapsack's capacity.

The result of Algorithm 1 when performed on the given data performed as expected with the greedy algorithm selecting the items with the highest efficiency. It should also be noted that when $W_{max} = 0$, the algorithm made no selection since no item had a 0 value and when W_{max} was a large as the combined weight of all item the greedy algorithm did select all 28 item.

3.4.3 Quadratic Knapsack Problem. Consider the following set of items which are represented by Table 3.1 we use the quadratic knapsack model see (2.2.3) such that:

- Item i of a total of $N = 15$ items.
- Where $v_i \in v$ is the value of item i.
- And $w_i \in w$ is the weight of item i.
- If item j is selected the corresponding profit $p_{ij} \in P$ is achieved and if both items i
- With the preselected knapsack capacity is W_{max}

$$
\begin{aligned}
N =&15\\
v =&\{7, 6, 13, 16, 5, 10, 9, 23, 18, 12, 9, 22, 17, 32, 8\}\\
w =&\{13, 14, 14, 15, 15, 9, 26, 24, 13, 11, 9, 12, 25, 12, 26\}\\
P =&\{12, 7, 6, 13, 8, 11, 7, 15, 23, 14, 15, 17, 9, 15, 15,\\
&13, 10, 15, 9, 10, 8, 17, 11, 13, 12, 16, 15, 11, 16,\\
&6, 8, 14, 13, 4, 14, 8, 15, 9, 16, 10, 13, 14, 14, 17,\\
&15, 14, 6, 24, 13, 4, 9, 7, 25, 12, 6, 6, 16, 10, 15, 14,\\
&2, 13, 12, 16, 9, 11, 23, 10, 21, 8, 18, 4, 13, 14, 14, 17,\\
&15, 9, 16, 12, 3, 14, 14, 27, 15, 16, 13, 14, 7, 17,\\
&28, 5, 19, 6, 18, 13, 4, 13, 16, 11, 19, 13, 15, 12, 16)\}
\end{aligned}
$$

Note the profit terms $p_{ij} = p_{ji}$ see (2.2.4). The method chosen to solve the quadratic knapsack problem tests the newly suggested heuristic as shown in Algorithm 2.

Table 3.3: **Test Results:** List of items selected for the knapsack by Algorithm 2 as well as the total value and weight obtained for each case executed with $N = 15$

Knapsack capacity W_{max}	**Selected Items**	**Knapsack Value**	**Knapsack Weight**
$W_{max} = 10$	$[6]$	10	9
$W_{max} = 50$	$[12 \quad 14 \quad 1 \quad 9]$	184	50
$W_{max} = 200$	$[12 \quad 14 \quad 1 \quad 9 \quad 2 \quad 6 \quad 10 \quad 11 \quad 5 \quad 8 \quad 4 \quad 13 \quad 3]$	1263	186

3.4.4 Remarks. The Table 3.3 shows that for a given knapsack capacity, items are chosen in a manner that allows for the largest possible combined knapsack value and weight using the quadratic knapsack problem.

Observing Table 3.3 we see that if W_{max} set to $W_{max} = 10$ the item 6 is chosen with the knapsack value and weight being 10 and 9 respectively. But if $W_{max} = 50$ is set than items 12,14,1 and 9 are selected with a combined knapsack value and weight being 184 and 500 respectively. It should be noted the Algorithm 2 performs effectively in both these instances since for a relatively small W_{max} the algorithm selects items who's combined weight w almost match the exact overall capacity of the knapsack and when a $W_{max} = 200$ the algorithm comes sufficiently close to the knapsack's capacity.

The newly suggested algorithm performed well with this test example as it uses a stronger efficiency function in Algorithm 2. It should also be noted that when $W_{max} = 0$, the Algorithm 2 made no selection since no item had a 0 value.

4. Applying New Heuristic to Portfolio Selection

In Chapter 3, we tested the newly suggested heuristic on a text example that modelled the quadratic knapsack problem and found the results to be satisfactory. In this chapter we will focus on applying this new heuristic on a practical problems from finance.

The stock market acts a very important role in economic development using the tools like pricing, reducing the risk, resource mobilization and optimal allocation of capital as explained by Ghodrati and Zahiri (2014). When investing in the stock market one of the most important decisions an investor will face is portfolio selection and selecting the optimal portfolio is the crucial task. Simply put a portfolio is a collection of assets such as stocks, bonds, commodities, currencies and cash equivalents.

The main purpose for modelling the quadratic knapsack problem is to determine an optimal portfolio with the use of the newly suggested algorithm. The share prices of companies, which are active in Pharmaceutical, Health Care and Mining Sector of Johannesburg Stock Exchange(JSE) were utilised. For achieving this purpose, Historical Data from the daily adjusted closing prices of companies' stocks spanning from 01/01/2014 – 29/12/2017 have been used. The data was imported via CSV files from Yahoo Finance (2018).

One of the most prominent application of the quadratic knapsack problem is the selection of a portfolio optimization in finance, where one wants to invest a limited amount of money in a given financial market.The Greedy Algorithm discussed in Chapter 3 will be applied to the data of the JSE listed companies with the aim of increasing the total return, in order to form a financial portfolio of optimal gain.

4.1 Motivation

Table 4.1: List of Company in Pharmaceutical, Health Care and Mining Sector of Johannesburg Stock Exchange (JSE)

Trading Name Company Listed on JSE	**Sector**	**JSE Code**
ADCOCK INGRAM HOLDINGS LIMITED	Pharmaceutical	AIP
ASCENDIS HEALTH LIMITED	Pharmaceutical	ASC
LIFE HEALTHCARE GROUP HOLDINGS LIMITED	Health Care	LHC
NETCARE LIMITED	Health Care	NTC
GLENCORE PLC	Mining	GLN
SASOL LIMITED	Mining	SOL
ANGLO AMERICAN PLATINUM LIMITED	Mining	AMS

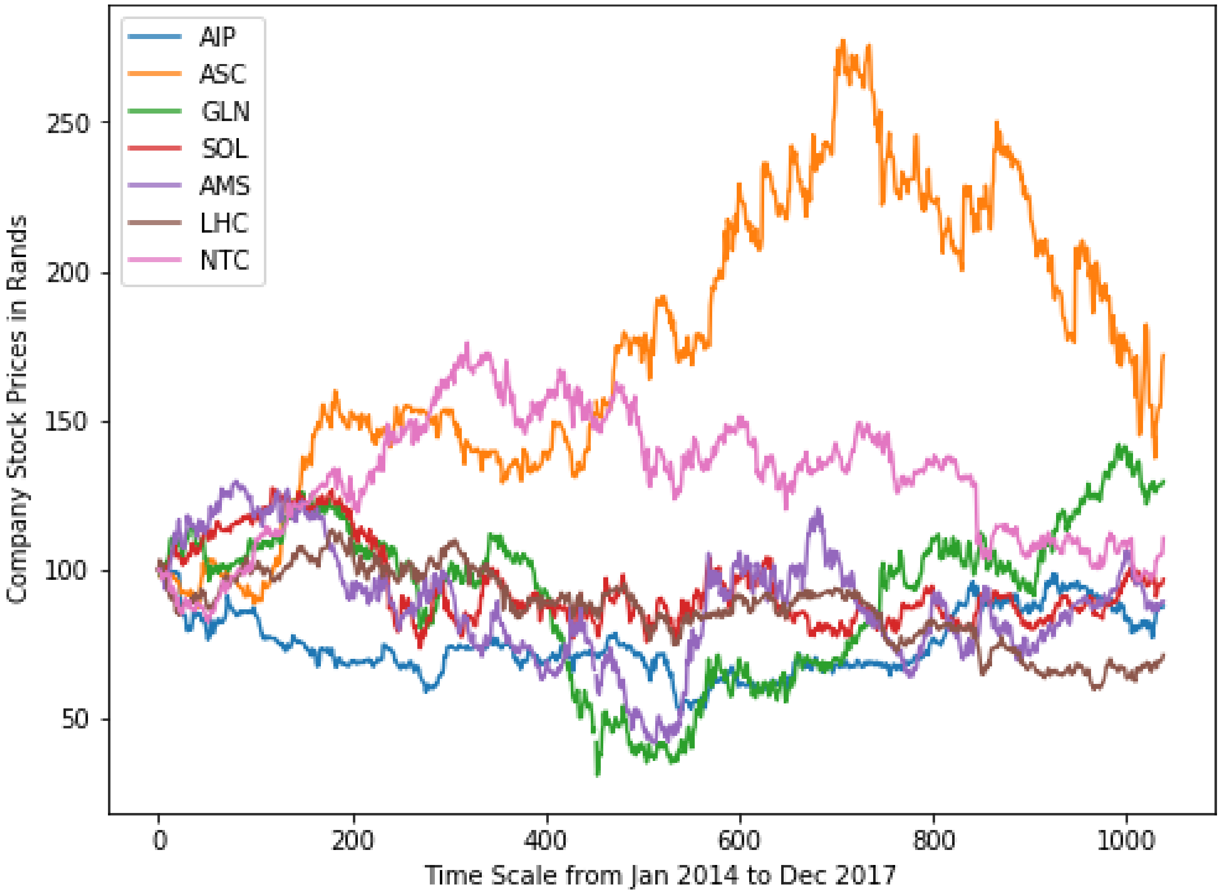

Figure 4.1: Time progression of Share Prices

Figure 4.1 is an graph depicting the share price progression of the JSE listed company's listed in Table 4.1 during the time period from 1 January 2014 to 29 December 2017 in real time. This graph was constructed from historical data that were downloaded from Yahoo Finance (2018).

The purpose of this figure is to see the physical progression of the share prices of the chosen companies so that a possible inference can be made with regards which company share prices would be most likely to chosen. Using Figure 4.1 ASC, Ascendis Health Limited would logically be chosen for the optimal portfolio since it has the highest share price with the fasted share growth progression.

4.2 Optimal Portfolio Problem Formulation and Analysis

Using the data that was extracted from Yahoo Finance (2018) we can now formulate the quadratic knapsack problem so that the newly suggested algorithm can be applied to find the optimal portfolio.

We start by using the information from the JSE and formulate the quadratic knapsack problem as follows:

- Itemise the JSE listed companies such that $i = \{1, ..., 7\}$ where $N = 7$

Table 4.2: Label of company with index as used in the code compiled in Python for future reference.

Company code	**Item Index**
AIP	1
ASC	2
LHC	3
NTC	4
GLN	5
SOL	6
AMS	7

- Find correlation see (4.2.1)
- Calculate profit matrix p from c_{ij}

Such that

- Item i of a total of $N = 7$ items.
- Where $v_i \in v$ is the value of item i.
- And $w_i \in w$ is the weight of item i.
- If item j is selected the corresponding profit $p_{ij} \in P$ is achieved and if both items i
- With the preselected knapsack capacity is W_{max}

$$
\begin{aligned}
N =& 7 \\
v =& \{20, 13, 20, 15, 26, 16, 14\} \\
w =& \{49, 18, 46, 403, 343, 34, 29\} \\
P =& \{18, 34, 14, 12, 18, 35, 17, 26, 18, 16, 14, 14, 11, 15, 18, 28, 11, 15, 14, 16, 12\}
\end{aligned}
$$

Note: The item values v_i are the values of the diagonal from the profit matrix and the weights of items w_i are the mean value in Rands per Share of each companies adjusted closing price. The p_{ij} values are the upper triangular values of the profit matrix since $p_{ij} = p_{ji}$

4.2.1 Definition 4.1. In this essay we will be using c_{ij} as the correlation term between items i and j from a set N items. The method of calculating the correlation is based on the following formula:

$$c = \frac{n \sum xy - \sum x \sum y}{n \sum x^2 - (\sum y)^2} \tag{4.2.1}$$

The Figure 4.2 the correlation matrix, with an in built function that utilises (4.2.1), was computed with Python 3 .

	AIP	ASC	GLN	SOL	AMS	LHC	NTC
AIP	1.0	-0.1727	0.5975	0.1109	0.1683	-0.4073	-0.6192
ASC	-0.1727	1.0	-0.2685	-0.5885	-0.3194	-0.4553	0.1603
GLN	0.5975	-0.2685	1.0	0.4122	0.4611	-0.1024	-0.4727
SOL	0.1109	-0.5885	0.4122	1.0	0.671	0.4775	-0.392
AMS	0.1683	-0.3194	0.4611	0.671	1.0	0.3685	-0.4642
LHC	-0.4073	-0.4553	-0.1024	0.4775	0.3685	1.0	0.403
NTC	-0.6192	0.1603	-0.4727	-0.392	-0.4642	0.403	1.0

Figure 4.2: Company correlation c_{ij} matrix calculated in Python

The profit matrix was calculated with the use of randomisation based on the following:

If $-1 < c_{ij} \leq -0.5$ then $p_{ij} \in [20, 40]$ such that $p_{ij} = 20 + random(40 - 20)$
If $-0.5 < c_{ij} \leq 0$ then $p_{ij} \in [15, 20]$ such that $p_{ij} = 15 + random(20 - 15)$
If $0 < c_{ij} \leq 0.5$ then $p_{ij} \in [25, 35]$ such that $p_{ij} = 25 + random(35 - 25)$
If $0.5 < c_{ij} \leq 1$ then $p_{ij} \in [10, 18]$ such that $p_{ij} = 10 + random(18 - 10)$

The result of this application is shown in Figure 4.3.

	AIP	ASC	GLN	SOL	AMS	LHC	NTC
AIP	20.0	18.0	34.0	14.0	12.0	18.0	35.0
ASC	18.0	13.0	17.0	26.0	18.0	16.0	14.0
GLN	34.0	17.0	20.0	14.0	11.0	15.0	18.0
SOL	14.0	26.0	14.0	15.0	28.0	11.0	15.0
AMS	12.0	18.0	11.0	28.0	26.0	14.0	16.0
LHC	18.0	16.0	15.0	11.0	14.0	16.0	12.0
NTC	35.0	14.0	18.0	15.0	16.0	12.0	14.0

Figure 4.3: Company profit matrix p calculated in Python

4.2.2 Solving the optimal portfolio using the Linear Greedy Method.

Table 4.3: **Optimal portfolio Results:** List of items selected by Algorithm 1 as well as the total value and weight obtained for each case executed with $N = 7$

Knapsack capacity W_{max}	**Indices Selected**	**Value of solution**	**Weight of solution**
$W_{max} = 10$	[] return empty	0	0
$W_{max} = 50$	$[2 \quad 7]$	27	47
$W_{max} = 500$	$[2 \quad 7 \quad 6 \quad 3 \quad 1]$	83	176
$W_{max} = 550$	$[2 \quad 7 \quad 6 \quad 3 \quad 1 \quad 5]$	103	519

4.2.3 Solving the optimal portfolio using the New Heuristic Method.

Table 4.4: **Optimal portfolio Results:** List of items selected by Algorithm 2 as well as the total value and weight obtained for each case executed with $N = 7$ and $k = 3$

Knapsack capacity W_{max}	**Selected Items**	**Knapsack Value**	**Knapsack Weight**
$W_{max} = 10$	[] return empty	0	0
$W_{max} = 50$	$[2 \quad 7]$	41	47
$W_{max} = 500$	$[3 \quad 7 \quad 2 \quad 6 \quad 1]$	264	176
$W_{max} = 550$	$[3 \quad 7 \quad 2 \quad 6 \quad 1 \quad 5]$	359	519

4.2.4 Remarks. The Table 4.2 and 4.3 shows that for a given knapsack capacity, items are chosen in a manner that allows for the largest possible combined knapsack value and weight using the linear knapsack and quadratic knapsack problem respectively. When looking at the results of Table 4.2 we see that given a $W_{max} = 50$ the item 2,7 is chosen with the knapsack value and weight being 27 and 47 respectively these values differ from the new heuristic approach used for the quadratic knapsack problem due to the different efficiency functions used in Algorithm 1 and 2.

Observing Table 4.3 we see that if W_{max} is smaller that the that the item with the least weight no item is chosen but if we set $W_{max} = 50$ the item 2,7 is chosen with the knapsack value and weight being 41 and 49 respectively. But if $W_{max} = 500$ is set than items 3,7,2,6 and 1 are selected with a combined knapsack value and weight being 264 and 176 respectively. It should be noted the Algorithm 2 performs satisfactorily for when $W_{max} = 50$ as it's combined weight of the items come close to the maximum capacity. But when we observe $W_{max} = 500$ Algorithm 2 has selected items with value significantly higher than the smaller knapsacks but raising the knapsack by just 50 we see that the value and weight drastically increase. This could be as a result of the weights being uneven when considering the items individual weight.

The newly suggested algorithm performed well with this financial application and was able to find the solution set in good running time. What is also interesting to note was that the ASC stocks observed in Figure 4.1 earlier on in Chapter 4 performed well which validated initial exceptions and was selected in all three cases, which again highlight this new heuristics capabilities of performance.

5. Conclusion

The objective of this essay was to establish the validity of a newly proposed heuristic as a method to solve a practical application of the quadratic knapsack problem. In order to demonstrate this new heuristic method we first introduced the concept optimisation and combinatorial optimisation which leads directly to the quadratic knapsack problem.

With wide spread applications across multiple disciplines, the quadratic knapsack problem proves to be a highly interesting problem to model. As discussed in this essay the quadratic knapsack can be applied to problems in the location of airports, railway stations or freight handling terminals which has always been a well known application of the knapsack problem. The area of application that was concentrated in this essay was the quadratic knapsack problem applied to the selection of an optimal portfolio in the finance.

Heuristic methods for solving optimisation problems provide a good approximate solution and quite often an approximate of the optimal solution is all that tends to be required. In the case for optimal portfolio selection this proves to be accurate since finding an exact solution set of assets for an optimal portfolio does not always result in the best future returns.

The new heuristic introduced as a solution method for the quadratic knapsack problem is a modification of the well known greedy approximation algorithm. This new heuristic relies on a more efficient efficiency function that is vital for the algorithm's selection of solution pairs of items that are highly correlated. It is clear from the results in Table 3.3 which applies the new heuristic to test examples the this new method for solving the quadratic problem performs very well. The same can be said for the results shown on Table 4.3 where the newly suggested heuristic is applied to financial shares of companies listed in the Johannesburg Stock Exchange(JSE) modelled as a quadratic knapsack problem.

We observe that the optimal portfolio as chosen by the new heuristic for a knapsack the size of R500 consists of the following shares Life Healthcare Group Holdings, Anglo American Platinum Limited,Ascendis Health Limited, Sasol Limited and Adcock Ingram Holdings Limited. Ascendis Health Limited was the best performing company with positive growth in share price over the studied time period and had the largest final adjusted closing price as seen in Figure 4.1 which implies that logically it would be selected for any optimal portfolio. The new heuristic does in fact select the ASC share every time it was run which clearly illustrates the effectiveness of the algorithm for this particular problem. When applying the heuristic to the selection of portfolio(s) for an optimal result, it must also be considered that when the algorithm was applied it used the mean of each stock's adjusted closing price. This choice of what type of historical data used will definitely affect which shares where chosen for the knapsack and should not be a reflection of the performance of the newly suggested heuristic.

Acknowledgements

To my AIMS family from Barry, Jeff, the staff and students that I had the pleasure of meeting and working with this year, AIMS is but a building and it is only through the people that walk these corridors that a global force of higher learning continues to grow and shape the landscape of Africa and the World. I am so thrilled to have been given this opportunity and I can only say that you have all contributed exponentially to my growth this year.

Prof Montaz Ali, I would like to sincerely thank you for the kindness you have shown me. Working with you on this essay has been truly an amazing experience. You inspire me through you devotion to mathematics and enthusiasm that is wonderfully infectious. Your hard work has not gone unnoticed and I wish that you are continually blessed. I would also like to extend my gratitude to Dr Kenneth Dadedzi, who guided me calmly and competently through this essay process as my tutor.

And to my fantastic, wonderful, amazing, supportive... honestly there can never be enough adjectives to describe how lucky I am to have been blessed to be born into the Ramlall family. Mom, Love, Yraney and Adhir writing this acknowledgement fills me with emotions as I can never truly say how thankful I am for all that you'll done and are to me. There will never be a moment that I don't think about the sacrifices and love and care that I receive in droves from everyone of you'll. This will end with an infinite amount of "Thank you"s and a simple "I love you."

References

Al-Iedani, N., Hifi, M., and Saadi, T. A reactive search for the quadratic knapsack problem. In *Control, Decision and Information Technologies (CoDIT), 2017 4th International Conference on*, pages 0495–0499. IEEE, 2017.

Alberto Caprara, P. T., David Pisinger. *Exact Solution of the Quadratic Knapsack Problem*. Addison-Wesley, 1998.

Bellman, R. E. Dynamic programming, ser. *Cambridge Studies in Speech Science and Communication. Princeton University Press, Princeton*, 1957.

Cook, S. A. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.

Dantzig, G. B. Discrete-variable extremum problems. *Operations research*, 5(2):266–288, 1957.

Davey, M. *Error-correction using Low-Density Parity-Check Codes*. Phd, University of Cambridge, 1999.

Diffie, W. The first ten years of public-key cryptography. *Proceedings of the IEEE*, 76(5):560–577, 1988.

Diffie, W. and Hellman, M. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

G, P. R. and R.L, R. *Discrete Optimization*. Elsevier, 2014.

Gallo, G., Hammer, P. L., and Simeone, B. Quadratic knapsack problems. In *Combinatorial optimization*, pages 132–149. Springer, 1980.

Garey, M. R. and Johnson, D. S. *Computers and intractability*, volume 29. wh freeman New York, 2002.

Ghodrati, H. and Zahiri, Z. A monte carlo simulation technique to determine the optimal portfolio. *Management Science Letters*, 4(3):465–474, 2014.

Gonzalez, T. F. *Handbook of approximation algorithms and metaheuristics*. Chapman and Hall/CRC, 2007.

Griva, I., Nash, S., and Sofa, A. Linear and nonlinear optimization (2nd edn). *SIAM Publications*, 2009.

Hellman, M. E. I will be totally insecure within ten years'. *IEEE spectrum*, 16(7):32–40, 1979.

Johnson, E. L., Mehrotra, A., and Nemhauser, G. L. Min-cut clustering. *Mathematical programming*, 62(1-3):133–151, 1993.

Karp, R. M. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

Kellerer, H., Pferschy, U., and Pisinger, D. Knapsack problems, 2004a.

Kellerer, H., Pferschy, U., and Pisinger, D. Introduction to np-completeness of knapsack problems. In *Knapsack problems*, pages 483–493. Springer, 2004b.

Kolesar, P. J. A branch and bound algorithm for the knapsack problem. *Management science*, 13(9): 723–735, 1967.

Martinelli, R. and Contardo, C. *The quadratic capacitated vehicle routing problem*. Groupe d'études et de recherche en analyse des décisions, 2013.

Merkle, R. C. Protocols for public key cryptosystems. In *Security and Privacy, 1980 IEEE Symposium on*, pages 122–122. IEEE, 1980.

Montaz Ali. Linear,and quadratic knapsack optimisation problem, Graduate Modelling Camp-MISG2018, 2018.

Murty, K. G. et al. *Optimization for decision making*. Springer, 2009.

Nocedal, J. and Wright, S. Numerical optimization springer-verlag. *New York*, 1999.

Pisinger, D. The quadratic knapsack problem—a survey. *Discrete applied mathematics*, 155(5):623–648, 2007.

Rhys, J. A selection problem of shared fixed costs and network flows. *Management Science*, 17(3): 200–207, 1970.

Sahni, S. General techniques for combinatorial approximation. *Operations Research*, 25(6):920–936, 1977.

StackOverflow. Np-hard problems, , 2018. https://i.stack.imgur.com/NnsMO.png, Last accessed on October 2018.

WikiKP. Knapsack problem, Wikipedia, 2018. https://en.wikipedia.org/wiki_Knapsack_problem, Last accessed on October 2018.

Witzgall, C. Mathematical methods of site selection for electronic message systems (ems). *NASA STI/Recon Technical Report N*, 76, 1975.

Yahoo Finance. Historical data, Johannesburg Stock Exchange Listed Companies, 2018. https://finance.yahoo.com, Last accessed on 2018-10-05.

Yajima, Y. and Fujie, T. A polyhedral approach for nonconvex quadratic programming problems with box constraints. *Journal of Global Optimization*, 13(2):151–170, 1998.