Investigating some heuristic solutions for the two-dimensional cutting stock problem

S.M. MANYATSI Hons. BSc

Dissertation submitted in partial fulfillment of the requirements for the degree *Magister Scientiae* in Computer Science at the Potchefstroom campus of the North-West University

Supervisor : Prof T Steyn
Co-supervisor: Prof J.M Hattingh

DECEMBER 2010
POTCHEFSTROOM CAMPUS

Abstract [English]

In this study, the two-dimensional cutting stock problem (2DCSP) is considered. This is a problem that occurs in the cutting of a number of smaller rectangular pieces or items from a set of large stock rectangles. It is assumed that the set of large objects is sufficient to accommodate all the small items. A heuristic procedure is developed to solve the two-dimensional single stock-size cutting stock problem (2DSSSCSP). This is the special case where the large rectangles are all of the same size. The major objective is to minimize waste and the number of stock sheets utilized.

The heuristic procedures developed to solve the 2DSSSCSP are based on the generation of cutting pattern. The Wang algorithm and a specific commercial software package are made use of to generate these patterns. The commercial software was chosen from a set of commercial software packages available in the market. A combinatoric process is applied to generate sets of cutting patterns using the Wang algorithm and the commercial software. The generated cutting patterns are used to formulate an integer linear programming model which is solved using an optimization solver.

Empirical experimentation is carried out to test the heuristic procedures using data obtained from both small and real world application problem instances. The results obtained shows that the heuristic procedures developed produce good quality results for both small and real life problem instances. It is quite clear that the heuristic procedure developed to solve the 2DSSSCSP produces cutting patterns which are acceptable in terms of waste generated and may offer useful alternatives to approaches currently available.

Broadly stated, this study involves investigating available software (commercial) in order to assess, formulate and investigate methods to attempt to benchmark software systems and algorithms and to employ ways to enhance solutions obtained by using these software systems.

Keywords

Cutting stock problem, heuristics, trim loss problem

OPSOMMING

Die twee-dimensionele materiaalsnitprobleem (2DMSP) word in hierdie studie ondersoek. Dit is ŉ probleem wat te make het met ŉ aantal kleiner reghoekige stukke of items wat uit ŉ versameling voorraad- (of groter reghoekige) stukke gesny word. Dit word aanvaar dat die versameling van voorraadstukke voldoende is om al die kleiner stukke te sny. ŉ Heuristiese prosedure is ontwikkel om die 2DMSP te hanteer waar die voorraadstukke van dieselfde grootte is. Dit is dus die spesiale geval waar al die voorraadstukke van dieselfde grootte is of ook die 2D_DGVMSP genoem word. Die hoofdoel is om die aantal voorraadstukke wat gebruik word en dus die afval, te minimeer

Die heuristiese prosedures wat ontwikkel is om die 2D_DGVMSP op te los, is gebaseer op die generering van snypatrone. Implementerings van Wang se algoritme en ŉ kommersiële sagteware pakket is gebruik om die snypatrone te genereer. Die kommersiële sagteware pakket is na eksperimentering gekies uit ŉ aantal pakkette wat in die industrie beskikbaar is. ŉ Kombinatoriese proses is ook gevolg om versamelings van snypatrone met behulp van Wang se algoritme en die kommersiële sagteware te genereer. Hierdie snypatrone is gebruik om heeltallige lineêre programmeringsmodelle te formuleer en die modelle is dan met behulp van ŉ optimeringspakket opgelos.

Empiriese eksperimentering is uitgevoer om die heuristiese prosedures te toets deur van kleiner datastelle vanuit die literatuur en ook datastelle uit die industrie gebruik te maak. Die resultate toon dat die heuristiese prosedures in beide gevalle goed vaar. Die heuristiese prosedures lewer resultate wat aanvaarbaar is in terme van die hoeveelheid afval wat gegenereer is en mag as bruikbare alternatiewe dien tot benaderings wat tans gevolg word.

Breedweg gestel, hierdie studie behels die ondersoek van (kommersiële) sagteware om metodes te evalueer, te formuleer en te ondersoek sodat sagteware stelsels en algoritmes in die literatuur geëvalueer kan word. Verder stel dit ook riglyne waarvolgens oplossings verbeter kan word.

Acknowledgements

This piece of work will never have been accomplished without God the Almighty with His blessings and His power that work within me and people behind my life who inspired, guided and accompanied me through thick and thin. Therefore I would like to convey my special thanks to the following persons:

- My supervisor, Prof T Steyn: For his enthusiasm, his inspiration and great effort to explain things clearly and simple. He made this thesis enjoyable. Throughout my thesis-writing period, he provided encouragement and lots of good ideas even during difficult times. I would have been lost without him.
- My co-supervisor, Prof J.M Hattingh: For his wisdom, wide knowledge and logical way of thinking have been of great value. Despite his health challenges and other commitments, he was always there to offer support and sound advice and this inspired and motivated me to finish this work.
- Department of Computer Science: For their continued financial support. Without them I would not have finished this work.
- My friends and fellow colleagues: For their general input regarding the research and spiritual support during difficult time.
- My family: For giving me the opportunity to do this work, providing a loving environment and their continued support. My special gratitude goes to my brothers (Muzi & Mthokozisi), sisters (Gugulethu & Dumsile) and my entire extended family. Lastly and most importantly, I wish to thank my mother for being there with me both emotionally and spiritually for the entire duration of my studies and to her I dedicate this work.

Table of Contents

Table of Figures

Table of Tables

Table of Graphs

Chapter 1: Introduction

1.1 Introduction

The cutting stock problem (CSP) is an optimization problem that has been extensively studied during the past 50 years. This problem involves identifying an efficient pattern when cutting small items from large ones in order to minimize waste or maximize profits. Such a problem mostly manifests itself in many mass production industries where large stock sheets or reels must be cut into smaller pieces; the stock material may be pulp and paper, steel, glass, or wood. This issue also emerges in other applications involving both cutting operations and situations unrelated to cutting, such as the multiprocessing of a batch of programs (Albano & Orsini, 1978:338). This problem is of great importance to the industry because of the economic incentive.

The CSP is considered an NP-hard problem in the sense that mathematically solving it to optimality is very difficult; thus any small improvement to the existing algorithms and heuristic solutions goes a long way towards solving the said problem (Bischoff & Wäscher, 1995:503).

1.2 Background, notation and models

The CSP represents a wide range of problems that can be classified by using the classification scheme which was originally developed by Dyckhoff (Dyckhoff, 1990:145). However his typology was not accepted as widely as desirable due to the fact that he provided a coding scheme that was not self explanatory from the view point of an international community of researchers. Wäscher (Wäscher et al., 2007:1109) later improved the original classification scheme. Their classification of the CSP divided it according to five characteristics. The characteristics outlined below will be explained in detail in the later paragraphs.

1. Dimensionality
    - One-dimensional (1D)
    - Two-dimensional (2D)

- Three-dimensional (3D)

2. Kind of assignment
   - Output (value) maximization
   - Input (value) minimization
3. Assortment of small items
   - Identical small items
   - Weakly heterogeneous assortment of small items
   - Strongly heterogeneous assortment of small items
4. Assortment of large items
   - One large object
   - Several large objects
5. Shape of small items
   - Regular items
   - Irregular items

The first characteristic is the dimensionality that coincides with that of the stock or large objects. W*ä*scher (Wäscher et al., 2007:1109), lists the types of dimensionality as being one, two, and three dimensional. An example of the 1D CSP is the cutting of cables, the 2D CSP is the cutting of glass and the 3D CSP is bin packing.

The second characteristic affecting the complexity of the problem refers to the kind of assignment. Two categories are used in this classification. The output (value) maximization is where a set of small items has to be assigned to a given set of large objects. In this instance, the set of large objects is not sufficient to accommodate all the small items and all the large objects are to be used. The other category is the input (value) minimization where a given set of small items is to be assigned to a set of large objects. In this case the set of large objects must be sufficient to accommodate all small items and all small items are to be assigned to a selection of large objects.

The third characteristic affecting the complexity of the problem is the assortment of small items. Three categories are used in this classification, the first being the case where all the small items are of the same shape and size, thus identical. The second category describes a weakly heterogeneous assortment where many small items are almost identical with respect to shape and size. The third category is that of strongly heterogeneous assortment items where few small items are identical with respect to shape and size.

The fourth characteristic affecting the complexity of the problem is the assortment of large objects. Two categories are used in this classification. The first category is the case of a single large object type. The second category is that of several types of large objects. Examples for this characteristic are the single stock-size cutting stock problem (SSSCSP) and the multiple stock-size cutting stock problems (MSSCSP). According to the complexity of the solution process, solution processes normally attend to the problem in trying to optimize over just one large object. This constitutes the so-called trim-loss problem which is based on the output (value) maximization characteristic listed above. In order to satisfy a customer's order in total, it is in general necessary that more than one large object be used and thus the optimization process spans more than one large object and the set of large objects may be according to characteristic four as described above, i.e. single stock-size or multiple stock-size. This would be a case of the input (value) minimization characteristic listed above.

The fifth characteristic that affects the complexity of the problem is the shape of the small items. In the case of the two and three-dimensional problems, analysts may distinguish between regular small items and irregular small items.

These characteristics can be associated in different combinations to characterize a given problem as it may suit the researcher or application. The characteristics considered in this research concentrate on the two-dimensional, input (value) minimization, weakly heterogeneous items, large single stock-size objects and regular shaped small items. This problem type is also known as the two-dimensional single stock-size cutting stock problem (2DSSSCSP). Using the term trim-loss (as mentioned above) will imply the special case when optimization involves just one large object based on the output (value) maximization characteristic.

Due to the complexity of this problem, many different optimization formulations and solution approaches are treated in the literature, depending on their dimensions, application field, special constraints and requirements (Macedo et al., 2010:991). Some of these will be discussed later on.

1.3 Problem Statement

The ever increasing costs of raw material and the need to reduce production costs while at the same time maximizing profits make research into the CSP, and in the case of this study, the 2DSSSCSP of great relevance. Broadly stated, this study involves investigating available software (commercial or otherwise) in order to assess, formulate and investigate ways to try and benchmark software systems and algorithms and to employ ways to enhance solutions obtained by using these software systems.

1.4 Research methodology and objectives

Different types of research methods are followed and different objectives are strived for to address the set problem statement. These can be stated as follows:

- Study the two-dimensional single stock-size cutting stock problem (2DSSSCSP) and the associated trim-loss problem by reading articles and books in the field of cutting stock problems and concentrating on algorithms and heuristics for the 2DSSSCSP problem;
- Study the Wang algorithm and employ an implementation thereof to solve the trim-loss problem and also solve instances of the 2DSSSCSP;
- Investigate and assess some commercial software packages available to solve the 2DSSSCSP;
- Develop heuristic procedures based on integer linear programming models that can solve the 2DSSSCSP;
- Report on and interpret the findings from the selected software when used to conduct empirical experimentation in order to solve certain problems drawn from the literature and industry.

1.5 Organization of the dissertation

In this section, a description of the purpose of each chapter and its structure is furnished.

Chapter 1: Introduction

The first chapter discusses the problem statement, objectives of the study, methodology and the organization of the dissertation.

Chapter 2: Background and literature survey

This chapter gives an overview of the cutting stock problem and the 2DSSSCSP as well as of relevant literature.

Chapter 3: Overview of software packages and the Wang algorithm

This chapter provides an overview of some of the two-dimensional cutting stock commercial software systems available on the market as well as a discussion of the Wang algorithm. Empirical experimentation was undertaken to evaluate the software.

Chapter 4: Heuristic procedure

This chapter offers a detailed explanation of the heuristic procedures proposed for solving the 2DSSSCSP; the generation of cutting patterns in order to assist in solving the problem will be discussed.

Chapter 5: Integer linear programming approach and model illustration

In this chapter, the sets of cutting patterns generated in Chapter 4 will be used to formulate an integer linear programming model to solve the 2DSSSCSP.

Chapter 6: Empirical results

This chapter discusses the results obtained from the empirical experimentation performed in Chapter 5 by applying the heuristic procedure to both small and some real world application problems.

Chapter 7: Conclusion and further research

This chapter discusses the results obtained in terms of the aims and objectives of the study.

Chapter 2: Background and literature review

2.1 Introduction

This chapter gives an overview of existing literature considering the general cutting stock problem (CSP) introduced in Chapter 1. The purpose of this overview is to gain an understanding of the CSP and the importance of its applicability in the industry. The two-dimensional cutting stock problem (2DCSP) as a specialization of the CSP and the two-dimensional single stock-size cutting stock problem (2DSSSCSP) as a further specialization of the 2DCSP will also be discussed. Some other related fields of study within the CSP environment like the two-dimensional assortment cutting stock problem (2DACSP) are also discussed and some of the previous research done in these fields is given. The CSP is discussed in paragraph 2.2; the 2DCSP is discussed in paragraph 2.3; the 2DSSSCSP is discussed in paragraph 2.4 and the 2DACSP is discussed in paragraph 2.5.

2.2 Cutting stock problem (CSP)

The CSP is a problem that involves determining a set of cutting patterns which minimize or maximize an objective function (Dyson & Gregory, 1974:41-53) and at the same time satisfies a customer order. A saving on stock material cost is achieved when less stock material is utilized. The CSP is a hard problem to solve exactly and most of the procedures proposed in the literature are heuristic in nature (Yanasse et al., 1991:673). There is no common approach to solve different configurations (classes) of the problem (Elmaghraby et al., 2000).

When small items are being cut from large stock objects, the trim loss problem (paragraph 1.2) occurs which deals with the issue of how to cut a set of small items from a single large stock object in such a way that stock material wastage is minimized. During the cutting process, the stock material can rarely be exploited perfectly but some residual pieces or trim loss will be produced (Eroglu & Noche, 2004). These pieces or items are seen as useless for practical purposes because of their small size (Oberholzer, 2003:1) and they are discarded. In most cases many companies lose a lot of money from this loss since they paid a considerable amount of money to acquire the stock.

A classification according to W$\ddot{a}$scher (Wäscher et al., 2007:1109) was given in Chapter 1 whereby the CSP can be classified into different classes based on a set of five characteristics each supported by a number of categories. These categories constitute a certain way of specialization of the CSP and will be addressed as is necessary. The main specialization for this study is the 2DSSSCSP as mentioned in Chapter 1.

2.2.1 Solution method approaches for the cutting stock problem

Since most of the standard problems in this area are known to be NP-hard, there exist different classes of methods used in the literature for solving the CSP. One class is the exact algorithmic method and this type of method guarantees that the optimal solution for the problem is obtained. The major disadvantage of this method is the computational complexity which requires immense computational resources, especially with large problems and thus the development of new algorithms using these methods has been limited.

The most common class of solution methods is heuristic in nature. This type of method rarely finds the guaranteed optimal solution, but usually generates an acceptable solution faster. A heuristic solution is usually considered acceptable if the solution is close enough to the known or assumed optimal solution (Karelahti, 2002:8).

The other class of solution methods is the meta-heuristic methods. This type of method has a capability of not being trapped into local optima as might happen with other heuristic approaches and the solution process is often guided by some lower level heuristic (Karelahti, 2002:8).

The heuristic and meta-heuristic methods are discussed as given by Eroglu and Noche (Eroglu & Noche, 2004) and Karelahti (Karelahti, 2002:8) in paragraph 2.2.1.1 and paragraph 2.2.1.2 since they are the most commonly considered methods**.**

2.2.1.1. Heuristic methods

There are at least three heuristic methods for solving the CSP that has been identified in the literature (Eroglu & Noche, 2004).

- **Delayed column generation technique: DCG**

  This technique splits the problem into two problems which is the primary and the secondary problem. The secondary problem is a subproblem depending on the dual values of the constraints of the primary problem in linear programming model solution processes. The idea is to find the solution with the minimum reduced cost in the secondary problem to generate new columns for the primary problem. This leads to a much smaller primary problem. This technique is very fast and generates only cutting patterns that lead to the optimal solution. The disadvantage with this technique is that it produces fractional solutions and the only possible objective function is the trim loss minimization.

- **Sequential heuristic procedure: SHP**

  This is a three-stage sequential heuristic. In the first stage, a width-cutting pattern is determined. Determining the stock size and the associated layout of the piece sizes to produce a good cutting pattern is the second stage. In the final stage, the number of times in which the generated cutting pattern will be used is determined. This technique is very fast and produces integer solutions and also allows multiple objectives to be considered. The disadvantage with this technique is that it may result in high trim loss values, especially in large problems.

- **Hybrid solution procedures.**

  The hybrid solution procedure is a combination of the linear programming based procedure, the SHP and other procedures. This technique produces integer solutions and combines the best parts of pure heuristics. The disadvantage with this technique is that it is much slower than pure heuristics.

2.2.1.2. Meta-heuristic methods

There are a number of available meta-heuristic methods for solving the CSP that have been identified in the literature. Some of them which are listed by Eroglu and Noche (Eroglu & Noche, 2004) and Karelahti (Karelahti, 2002:8) are discussed below:

- **Tabu search**

  In this method, the iteration always proceeds to the best solution in the neighbourhood even though it could be worse than the current one. This reduces the likelihood of being trapped in local optima. This method produces high quality solutions but it is slower than the greedy randomized adaptive search procedure (GRASP) discussed next.

- **Greedy randomized adaptive search procedure (GRASP)**

  In this method, the algorithm has multi-starts and each GRASP iteration consists of two phases, a construction phase and a local search phase. A feasible solution is produced in the construction phase and a local optimum in the neighbourhood of the constructed solution is sought in the local search phase. This algorithm restarts searching from another better region of the search space as soon as a local optimum is found in order to escape the local minima. This method is very fast and produces reliable solutions.

- **Evolutionary approach (EA)**

  This type of algorithms does not depend on constructive or local search heuristics. Instead, this algorithm works by updating a set P of original solutions randomly generated. The solutions are represented by a vector of elements that resemble the genetic structure of a chromosome. The evolutional approach consists of two types, the genetic algorithm and evolutionary programming. The difference between the two is that the genetic algorithm uses crossover as a primary operator and mutation as a secondary operator whereas evolutionary programming uses only mutation.

In an attempt to solve the different variations of CSPs, many solution approaches have been tried and tested in the literature. According to Elmaghraby (Elmaghraby et al., 2000) some of the approaches include the following.

- Linear programming approaches.
- Dynamic programming approaches.
- Heuristic approaches.
- Genetic algorithm approaches.
- Expert systems approach.

Elmaghraby (Elmaghraby et al., 2000) introduced a new intelligent approach to solving the CSP using a hierarchical architecture and their overall architecture consists of four layers of intelligent tools and techniques. Figure 2.1 displays the hierarchical architecture adapted from Elmaghraby (Elmaghraby et al., 2000) and a discussion of each layer will follow.

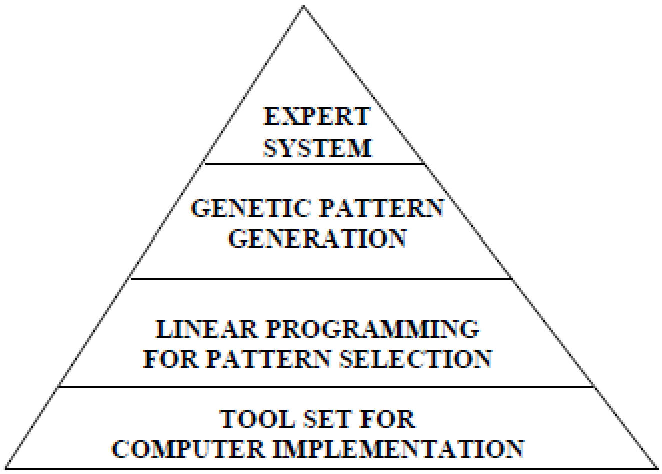


Figure 2. 1: Hierarchical architecture (Adapted from Elmaghraby et al. (2000))

- **Expert system**

  Expert systems solve difficult problems that require the use of expertise and experience. This is achieved by applying knowledge of the techniques, information, heuristics and problem solving techniques that human experts employ to solve such problems.

- **Genetic pattern generation**

  The generation of a cutting pattern depends on the order of handling the items, and the way of fitting these items into the sheet with respect to the sheet sizes and to other fitted items.

- **Linear programming for pattern selection**

  The solving of a CSP is done by first constructing suitable cutting patterns by enumeration. The best cutting patterns to be realized is determined using linear programming. This approach is followed to some extent in this study and will be discussed later on.

- **Tool set for computer implementation**

  This is the computer implementation of the developed algorithms. This is done by importing designs using DXF files generated from AutoCAD, Corel Draw, or other CAD programs. This is then handled through a user interface.

2.3 Two-dimensional cutting stock problem (2DCSP)

A very important variant of the CSP is the two-dimensional cutting stock problem (2DCSP). Based on the characteristics discussed in Chapter 1, the 2DCSP is a specialization of the CSP, but can also be specialized further when necessary. The 2DCSP deals with a set of small items that need to be cut from a large supply of rectangular stock sheets of given sizes. The major objective of this cutting is to minimize the amount of waste generated. The 2DCSP is a well studied optimization problem which occurs in many real world applications of business and industry. A lot of the articles available in the literature were published before the classification structure which was published by Wäscher (Wäscher et al., 2007:1109). This problem occurs mostly in the cutting of glass and paper.

2.3.1 Classifications of the 2DCSP

The 2DCSP can be classified according to the following diagram. Figure 2.2 illustrates a classification of the 2DCSP adapted from Suliman (Suliman, 2006:177).

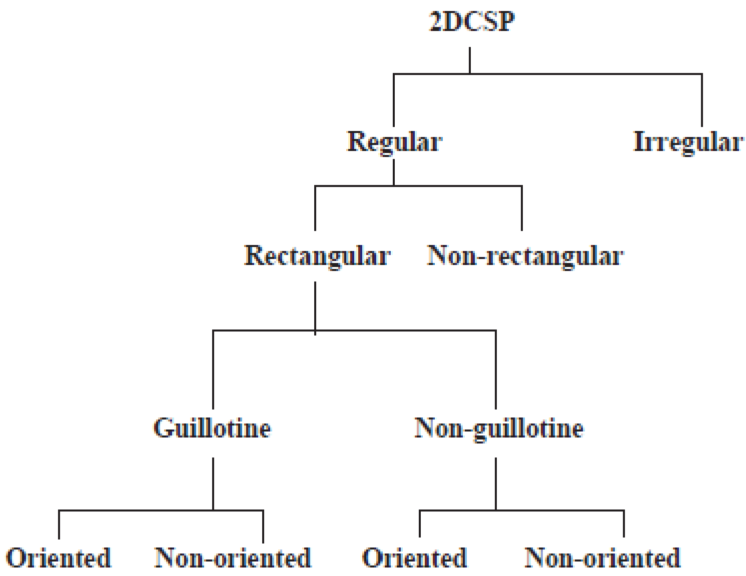


Figure 2. 2: Classifications of 2DCSP (Adapted from Suliman, (2006))

The 2DCSP can be classified into the cutting of regular or irregular shapes (Suliman, 2006:177). In the irregular 2DCSP, the small items to be cut may take any shape whereas in the regular 2DCSP, the small items are of rectangular shapes. Only regular type of cutting will be considered in this research study.

The 2DCSP can take a guillotine cut type or a non-guillotine cut type. A guillotine cut type is one that requires that the rectangle be cut from one edge of the rectangle to the opposite edge. The resulting small items may then be treated separately and cut again, but again each is cut all the way through (Gilmore & Gomory, 1965:94-120). A non-guillotine cut type is one that does not need edge to edge cutting and there is no required order or sequence to cut them.

The guillotine cuts can also be made in successive stages. The cuts of the same stage are in the same direction. If an upper bound is introduced on the number of stages, then the cutting pattern is called staged or otherwise non-staged. The direction of cuts may either be horizontal or vertical. A vertical cut is one that is parallel to the width of the stock sheet whereas a horizontal cut is one that is parallel to the length of the stock sheet. The cut directions of any two adjacent stages must be perpendicular to each other (Macedo et al., 2010:991).

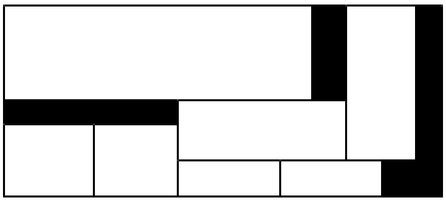


Figure 2. 3: Non-guillotine cut

Figure 2.3 displays an example of a possible non-guillotine cut type and the black blocks represent the waste generated by that cutting pattern. The non-guillotine cut type is mostly applicable to the cutting of paper.

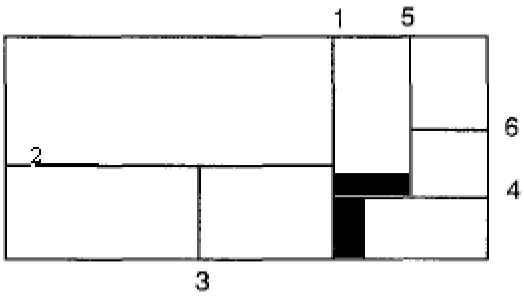


Figure 2. 4: Guillotine cut

Figure 2.4 displays an example of a possible guillotine cut type where the cuts are numbered in the order in which they could be made, although other sequences are obviously possible. The black blocks represent the waste generated by that cutting pattern. The guillotine cut type is mostly applied in the solving of 2DCSPs. Beasley (Beasley, 1985a:253) reports that one of the reasons for using this approach is that the vast majority of 2DCSPs met with in practice are guillotine cutting stock problem. This is due to the nature of the cutting machines employed.

The rectangular cutting can be further classified into oriented cutting, where the lengths of rectangles are aligned parallel to the length of the stock sheet, or non-oriented (Suliman, 2006:177) where the lengths of the rectangles can be aligned to the lengths or width of the stock sheet.

In many applications, non-oriented cutting is not allowed because of the constraints such as the patterns of the cloth or of the grain of the wood (Kenyon & Rémila, 2000:645). In some instances, the oriented type of cutting allows no rotation of items and non-oriented type of cutting allows rotations.

The 2DCSP appears in a constrained and unconstrained form. In the constrained form, the most usual constraint is the one that restrict the maximum number of pieces or small items of each type to be cut. In the unconstrained form, there is no restriction on the maximum number of pieces of small items to be cut.

From these classification characteristics, problems can also be classified for instance a constrained 2DCSP would mean that there is a restriction on the number of pieces to be cut. Other classifications are also possible as can be seen in the following paragraph on research work reported.

2.3.2 Previous research on the 2DCSP

The 2DCSP usually appears in the process of cutting steel or glass plate into required small item sizes to minimize waste. Christofides and Withlock (Christofides & Whitlock, 1977:30) proposed a depth-first branch and bound algorithm for the solution of the two-dimensional constrained cutting stock problem. Their algorithm described limits the magnitude of the tree-search by deriving and imposing necessary conditions for the cutting pattern to be optimal and by using a dynamic programming procedure during the solution process. Their algorithm only considered guillotine cuts.

The guillotine cutting problem mostly occurs in the cutting of glass into required sizes and in the cutting of wood sheets to make furniture. Beasley (Beasley, 1985a:253) presented a number of algorithms, both heuristic and exact, based upon dynamic programming for unconstrained two-dimensional guillotine cutting. He used two types of restrictions and the first one was that it should only consider guillotine cuts and the second type was to limit the cutting that occurs to a number of stages. All the algorithms and empirical work presented showed that they were capable of dealing effectively with large problems.

Viswanathan and Bagchi (Viswanathan & Bagchi, 1993:768) presented a best-first search method for the constrained 2DCSP. The algorithm was used to cut only guillotine cuts by seeking an upper bound on the value of the best guillotine cutting pattern on the stock rectangle that was constrained to include the required piece. Their algorithm guaranteed finding an optimal solution and the running time was very reasonable.

However Hifi (Hifi, 1997:727) proposed an improvement to the algorithm presented by Viswanathan and Bagchi (Viswanathan & Bagchi, 1993:768). His algorithm was able to develop better lower bounds and upper bounds by using the one-dimensional bounded knapsack problems and dynamic programming techniques. The results obtained from the algorithm were better than the results obtained from (Viswanathan & Bagchi, 1993:768).

Christofides and Hadjiconstantinou (Christofides & Hadjiconstantinou, 1993:21) presented a new tree-search algorithm for the solution of the 2DCSP and it can also be thought of as 2DCGCSP for the constrained guillotine cutting problem. Their algorithm limited the size of the tree search by using a tighter upper bound derived from a state space relaxation (SSR) of a dynamic programming formulation of the original problem. Their algorithm was tested on a number of standard problems from literature and some newly generated problems. The results obtained from that algorithm showed that the SSR performs reasonable well for 2DCGCSP of medium size.

Wang proposed an algorithm to handle the trim loss problem as mentioned previously and can be seen as a specialization of the 2DCSP. Since Wang's approach will be used explicitly in this study, a discussion thereof follows later on. Daza (Daza et al., 1995:633) proposed an improvement for the Wang algorithm. The improvement was achieved by representing the problem through a graph which characterizes the whole solution domain. Due to the strong combinatorial component in the problem, the graph's implicit generation was achieved by means of an implicit methodology.

Riehme (Riehme et al., 1996:543) proposed an algorithm for the two-stage guillotine cutting stock problem having extremely varying order demands. Their new approach considered a new relaxation problem which was used to build up two-stage guillotine cutting patterns which fulfil the most of the order demands. Then the remaining problem with the unfulfilled order demands was handled like an order with small demands.

Klempous (Klempous et al., 1996:323) presented a basic formulation and solution procedure for solving a particular large-scale 2DCSP at a furniture factory. Their formulation only concentrated on cases in which both stock and ordered sizes were rectangular. Their mathematical model for optimization relied on the Gilmore and Gomory (Gilmore & Gomory, 1965:94-120) seminal approach. However that approach led to a large-scale, time consuming linear programming model that was extremely complex. To reduce the total time, they proposed an interactive procedure based on the forecasting of the criterion values. The user of the program can stop the calculation on the basis of the forecasted optimal value.

Hifi and Zissimopoulos (Hifi & Zissimopoulos, 1996:553) presented a recursive exact algorithm for the weighted 2DCSP. Their algorithm used dynamic programming procedures and one-dimensional knapsack problems to obtain efficient lower and upper bounds and important optimality criteria which permitted a significant branching cut in a recursive tree-search procedure. Their algorithm performed very well, especially for the large instance size for the weighted problem.

According to Fayard (Fayard et al., 1998:1270), the two-dimensional guillotine problem cannot be solved within a reasonable amount of time, and consequently heuristics must be used to solve large scale real world problems. They presented an efficient approach for large-scale two-dimensional guillotine cutting stock. In their approach, they included two extra restrictions and the first restriction was that all the items should have a fixed orientation, i.e. a piece of length l and a width w should be different from a piece of length w and width l (when $l \neq w$) and all applied cuts should be of guillotine cut.

Burke (Burke et al., 2003:655) presented a best-fit heuristic for the two-dimensional rectangular cutting stock problem. A placement algorithm was used that took a list of shapes that were sorted by increasing height or decreasing area and then applied a placement rule to each of these in turn. Their algorithm allowed for both non-guillotine and guillotine cuts and the items were allowed rotations of 90 degrees.

Hadjiconstantinou and Iori ((Hadjiconstantinou & Iori, 2007:1150) proposed a hybrid genetic algorithm for the two-dimensional single large object placement problem. Their heuristic approach combined the use of greedy algorithms with genetic algorithms. The greedy algorithms were used to obtain initial solutions and the genetic algorithms were used for elitist theory, immigration and performing different crossover operators. Only non-guillotine cuts were considered in his approach.

Tiwari and Chakraborti (Tiwari & Chakraborti, 2006:384) presented a multi-objective optimization of a 2DCSP using genetic algorithms. The objective function in this problem was to minimize both the trim loss and the number of cuts to achieve the cutting process. They used a tree encoded multi-objective genetic algorithm to study both guillotine and non-guillotine cutting cases using a binary representation of the variables. That approach showed that globally optimum solutions were obtained.

Cui (Cui, 2007:943) proposed a simple block pattern for the unconstrained two-dimensional cutting problem. He used a recursive algorithm to generate the optimal simple block pattern. Only orthogonal solutions were considered and the small items were not rotated. His algorithm yielded a better material utilization than most of the common algorithms and computational time was very reasonable.

2.4 Two-dimensional single stock-size cutting stock problem (2DSSSCSP)

The two-dimensional single stock-size cutting stock problem (2DSSSCSP) is a special type of 2DCSP that involves assigning a given set of weakly heterogeneous small items to a set of same size large objects. The set of large objects are sufficient to accommodate all the small items. The major objective of this problem is to utilize the least number of large objects.

The 2DSSSCSP is classified under the single stock-size cutting stock problems as was classified in the classification scheme developed by W*ä*scher (Wäscher et al., 2007). The 2DSSSCSP used in this research is defined in the following paragraph:

- Dimensionality
  The proposed heuristic procedure is for solving the problem where only two-dimensional problems are considered.
- Kind of assignment
  The input (value) minimization where a given set of small items is to be assigned to a set of large objects will be used. In this case, the set of large objects is sufficient to accommodate all small items and all the small items are to be assigned.

- Assortment of small items

  The weakly heterogeneous assortment case is where many small items are almost identical with respect to shape and size. The demand for each item type is relatively large.
- Assortment of large objects

  The set of large objects considered consists of identical elements and only objects of rectangular shape are considered.
- Shape of small items

  Regular rectangular items are considered in this study.

The 2DSSSCSP has been studied less compared to other 2DCSPs. The 2DSSSCSP was earlier solved by Gilmore and Gomory (Gilmore & Gomory, 1965:94-120) when they proposed a linear programming approach to solve the 2DCSP. The problem was formulated and solved as a linear programming problem, where the columns of the constraint matrix represented the different cutting patterns. They used a column generation approach to solve the problem since it was impossible to enumerate all the cutting patterns and the column generation approach was able to avoid generating all the cutting patterns. Their method is very efficient, but it is difficult to get the optimal solution due to the need of rounding the relaxed fractional solutions to integer solutions.

Most of the approaches proposed for solving the 2DCSPs require immense computational resources and He (He et al., 2009:116) proposed a Hybrid Meta-Heuristic (HMH) algorithm that utilizes little computational resources. Their algorithm combines a greedy algorithm with genetic algorithms to solve the one and 2DSSSSCSP. Their mixed heuristic algorithm was implemented using the C language and the final solution achieved a utilization ratio of more than 97%.

Macedo (Macedo et al., 2010:991) proposed a new model to solve exactly the 2DCSP with two stages and the guillotine cutting constraint. Reduction criteria were applied to reduce the size and symmetry of the model and a new family of cutting planes was derived together with a new lower bound. Some variants of the problem such as the rotation of items and the possibility of considering the horizontal or vertical orientation for the first cut were explored. A commercial software package IBM ILOG CPLEX was used to solve the problem.

2.5 Two-dimensional assortment cutting stock problem (2DACSP)

The two-dimensional assortment cutting stock problem (2DACSP) is another special type of the 2DCSP. Regardless of the type of material or materials stocked, virtually all manufacturing operations need to determine the amount and size of material to stock (Vasko & Wolf, 1994:281). Pentico (Pentico, 2008:295) defined the assortment or catalog problem as a scenario in which a given set of sizes or qualities of some product and their known or expected demands. The major problem is to define the subset of sizes to stock in order to minimize the storage costs and to enable good cutting patterns so as to minimize the total waste.

Because of storage or manufacturing limitations, economies of scale in production or storage, or the cost associated with holding different sizes in stock, a subset of sizes will be stocked and the demand for an un-stocked size are filled from a stocked size with an associated substitution cost (Pentico, 1988:324). The assortment problem arises in various industries such as the cutting of steel, paper, textile and transportation industries as they face the problem of choosing the best stock material to cut in order to meet customer requirements.

According to Beasley (Beasley, 1985b:297), given the sizes of the various types of stock rectangles that are available and details of the requirement for the small rectangular items, the problem is one of deciding the following;

- The appropriate type of stock rectangles to use (and how many of each type) and,
- The two-dimensional cutting pattern for each stock rectangle that is cut into smaller rectangular items.

According to Lin (Lin, 2006:175), the assortment problem has not been considered extensively in the literature and Baker (Baker, 1999:83) reports that most of the various assortment problems that appear in the literature have been solved using dynamic programming.

2.5.1 Previous research on 2DACSP

A number of approaches have been suggested to solve the 2DACSP and the techniques range from dynamic programming and linear programming to recursive procedures, artificial intelligence and heuristic solutions.

The early work done on the assortment problem assumed that the demands and proportional substitution cost function for a single problem are known in advance. Pentico (Pentico, 1974:286) introduced a model in which the demands were not known but which were subject to known independent probability distributions. Since he was dealing with probabilistic rather than known demand, there was a possibility that there will be some inventory at the end of time which will incur a holding charge per unit and the possibility that demand might exceed the available stock which in turn incurs a cost of each unit not supplied. He developed a model which simplified the problem by assuming that the demand for any size will be supplied only from the smallest possible stocked size and not from any larger available stocked size. Dynamic programming was used to solve this problem.

The major problem of the assortment problem was to determine the subset of sizes to stock and the substitution rules to follow to minimize all the relevant costs and thus Pentico (Pentico, 1988:324) addressed the discrete two-dimensional assortment problem by considering situations with concave production-inventory cost functions and substitution cost. His research was only limited to a consideration of problems for which the variable stocking cost were linear. He developed two simple heuristics and the first heuristic procedure involved two stages; (1) determining the size to stock and (2) determining the substitution to use. The second heuristic was developed to modify the solution obtained by the first heuristic. He used dynamic programming to solve the problem and his procedure did not perform well cost wise on problems with fixed charge cost function.

Yanasse (Yanasse et al., 1991:673) presented an algorithm for the 2DACSP in the wood industry where requirements have to be exactly met and the cuts are of guillotine type. Their heuristic procedure identified possible combinations of the board sizes from which all required panels could be cut and an enumeration scheme was used to generate the combinations in a non-decreasing order of area. The algorithm was tested with problems of varying sizes and the performance of the algorithm was good in terms of the quality of the solution but it was not so good with computational times.

Since most of the methods for solving the assortment problem were either unable to find an optimal solution or were being computationally inefficient for reaching an optimal solution, Li and Tsai (Li & Tsai, 2001:1245) proposed a new method which finds the optimal solution of the assortment problem by solving a few linear mixed 0-1 programs. Using fewer variables improved the computational efficiency. The numerical examples done in the model indicated that the method was able to find an optimal solution using less CPU time.

Alves and Valerio de Carvalho (Alves & Valerio de Carvalho, 2007:1) proposed an exact branch-and-price-and-cut algorithm to solve the assortment and trim loss minimization problem in a single stage. They used an integrated integer linear programming model to compute lower bounds at the nodes of the branch-and-bound tree and derived a robust branching scheme to find the integer optimal solution. A set of binary variables were used to determine whether a particular stock length is used or not. They were able to solve the problem efficiently and in a reasonable amount of time.

2.6 Summary

This chapter has given a short overview of the existing literature of the CSP and the 2DCSP. The 2DSSSCSP was discussed as well as the 2DACSP. Solution approaches that are available in the literature for solving the CSP were discussed. The next chapter gives an overview of some of the two-dimensional cutting stock software packages available in the market.

Chapter 3: Overview of commercial software packages and the Wang algorithm

3.1 Introduction

This chapter gives an overview of some of the two-dimensional cutting stock software packages available in the market. This will help in highlighting the differences among industry-oriented software packages presently available. Empirical experimentation will be conducted to test some of the commercial software packages using data from previous research papers. The results will be compared, and the package that produces better results will be used further in this research to generate cutting patterns. A discussion of the Wang algorithm and the greedy approach will be done.

These software packages and the implementation of the Wang algorithm will be used to gain experience in solving 2DSSSCSP instances. Additionally, the chosen software package and the Wang algorithm implementation will be used to generate different pools of cutting patterns. The cutting patterns in these pools can be employed in the formulation of integer linear programming models to solve the 2DSSSCSP. This will enable the evaluation of software systems in terms of their differences.

The overview of optimization software packages is done in paragraph 3.2 while the results of the empirical experimentation are given in paragraph 3.3. The analyses of results obtained from the experimentation are discussed in paragraph 3.4. The CUTLOGIC 2D software and Wang algorithm are discussed in paragraphs 3.5 and 3.7 respectively. The greedy approach is discussed in paragraph 3.6.

3.2 Two-dimensional cutting stock optimization software overview

Optimization software packages for 2DCSPs are tools used to produce the most efficient cutting patterns on several sheet material boards, taking into consideration technological and organizational parameters of production (Macedo et al., 2008). Optimized cutting is very fundamental for economic production of cut component pieces. This is very important since a small enhancement in the cutting patterns can result in major saving in material and considerable diminution in production costs. It is important to note that there are multiple commercial software packages available, many of which are not mentioned in this survey.

Macedo (Macedo et al., 2008) presented a detailed survey of software packages for the 2DCSP. Their comparison between the relevant software packages was based on some of the package main features. Table 3.1 displays the general information about the software packages they presented. It shows the commercial name of each software package, company that developed the product, company's head office country, year of products commercialization and the website for the software.

Note that the software packages that were used in this research are the ones in bold letters. These packages were chosen because they were available to use for a certain period of time.

Product Name	Company	Country	Year Introduced	URL
ACE Cutting Optimizer	Adelaide Computer Energy	Australia	2002	http://www.acecut.com/default.htm
Astra R-Nesting	Technos Ltd	Russia	1999	http://www.techno-sys.com/
Corte Certo	Dimensions softwares	Brazil	1997	http://www.cortecerto.com/english/main.htm
Cutlist Plus	Woodworking Software	USA	1999	http://cutlistplus.com/default.aspx
Merick Calc 3.15	Soft Consult	Czech Republic	1995	http://www.softconsult.cz/us/index.asp
Optimik	RK Software	Slovakia	1999	http://www.rksoft.sk/
Plan IQ	MagicLogic Cutting Software	Canada	1998	http://www.cuttingstock.com/home.cfm
Real Cut 2D	Optimal Programs	USA, UK	2004	http://www.optimalprograms.com/
Sheet Cutting Suit	**XY Soft**	**USA**	**1999**	**http://www.optimizecutter.com/**
SmartCUT	**Rasterweq Software**	**USA**	**2003**	**http://www.rasterweq.com/index.php**
The Itemizer	**R&R DRUMMOND**	**USA**	**2002**	**http://www.rrdrummond.com/**
2D Load Packer	Asrokette Algorithms	USA	2000	http://www.astrokettle.com/index.html
Best Cut	VRSoft Ltd	Belarus	2001	http://vrsoft.msk.ru/index.htm
Cut Master 2D	Cutting Optimizer	Serbian	1999	http://www.cutmaster2d.com/index.html
Cutlight	Joiners Placet	Austria	2003	http://www.itmanagement.at/joinersplanet/english/index.html
CUTLOGIC 2D	**TMachines, s.r.o.**	**Slovakia**	**2003**	**http://tmachines.com/CUTLOGIC 2D-2d.htm**
Cutting 3	Cutting Home	Russia	2007	http://www.cuttinghome.com/index.html
GNcutter 32	Optimalon	Canada	2004	http://www.optimalon.com/default.htm

	Softwares			
Opticut	Boole & Partners	France	1996	http://boole.club.fr/english/index_eng.php
Panel Optimizer	Small BITS	South Africa	2005	http://www.smallbits.co.za/index.htm
Plus 2D for Woodworking	Nirvana Technologies Private Limited	India	1999	http://www.nirvanatec.com/
Sheet Layout 9	Productivity Software	USA	1999	http://www.sheetlayout.com/
Cutting Optimizer Pro	**Gunsh Software Solution**	**Serbia**	**1997**	**http://gunsh.com/**

Table 3. 1: General information (Adapted from Macedo et al. (2008))

The most important feature in the software package is the type of optimization. The optimization type can either be minimize waste or minimize cost. Table 3.2 displays the type of optimization for each software package. It shows whether the software package minimizes waste only or minimizes cost. It also shows if the software package minimizes the number of different layouts and whether the package provides the best sheet size calculation.

Product Name	Minimize Waste	Minimize Cost	Minimize Number of different layouts	Best Sheet Size Calculation
ACE Cutting Optimizer	Yes	No	No	No
Astra R-Nesting	Yes	No	No	No
Corte Certo	Yes	No	Yes	Yes
Cutlist Plus	Yes	Yes	No	No
Merick Calc 3.15	Yes	Yes	No	No
Optimik	Yes	No	No	No
Plan IQ	Yes	No	Yes	No
Real Cut 2D	Yes	No	No	No
Sheet Cutting Suit	**Yes**	**Yes**	**No**	**No**
SmartCUT	**Yes**	**No**	**No**	**No**
The Itemizer	**Yes**	**No**	**No**	**No**
2D Load Packer	Yes	No	No	No
Best Cut	Yes	No	No	No
Cut Master 2D	Yes	No	Yes	No
Cutlight	Yes	No	No	No
CUTLOGIC 2D	**Yes**	**Yes**	**Yes**	**Yes**
Cutting 3	Yes	No	No	No
GNcutter 32	Yes	No	Yes	-
Opticut	Yes	No	Yes	No

Panel Optimizer	Yes	No	No	No
Plus 2D for Woodworking	Yes	Yes	-	Yes
Sheet Layout 9	Yes	No	No	No
Cutting Optimizer Pro	**Yes**	**Yes**	**-**	**-**

Table 3. 2: Optimization of each package (Adapted from Macedo et al. (2008))

3.3 Empirical experimentation: Industry-oriented packages

The approach used to calculate the waste generated will be based on the best utilization of the stock sheets. This will be represented by *N-1* with *N* being the number of stock sheets necessary to meet the customer order. This approach was chosen with the idea that if the utilization of *N-1* stock sheets is optimal or as good as possible, then the remnant of the last stock sheet will be as large as possible and can be stored as a new input stock sheet. Thus, the concept behind this approach is to utilize the least number of stock sheets so that an overall good or even optimal solution could be established.

Empirical experimentation was carried out on the chosen software packages with data from the literature and the following results were obtained.

3.3.1 Problem instance 1

The data used for problem instance 1 is obtained from Cung (Cung et al., 2000:185) in their paper entitled “Constrained two-dimensional cutting stock problems, a best-first branch-and-bound algorithm”. They developed a new version of the algorithm proposed by Hifi (Hifi, 1997) for solving exactly some variants of un-weighted constrained two-dimensional cutting stock problems.

Problem	Stock sheet length (L) and width (W)	Demand rectangle's length (l), width (w) and upper bound (b) where b specifies the number of times a demand rectangle must be cut.
1	(132,100)	(18,39,4);(13,49,2);(27,51,1);(65,31,3);(45,27,2);(69,21,4);(21,63,2);(54,41,2);(41,31,1);(37,22,1);(29,54,1);(47,31,2);(18;31,3);(21,17,1);(19,53,1);(13,41,4);(37,12,5);(19,29,2);(67,17,1);(49,31,3)

Table 3. 3: Problem instance 1

The data of problem instance 1 from Table 3.3 was used for each of the chosen software packages to solve the 2DSSSCSP.

Product Name	Number of Sheets Used (N)	N-1	Waste on N-1 Sheets	Total Waste on N Sheets	Waste Percentage on N-1 Sheets	Total Waste Percentage on N Sheets	Computational Time
CUTLOGIC 2D	4	3	846	5243	2.14	9.93	00:04:14
Cutting Optimizer Pro	4	3	1464	5243	3.70	9.93	00:00:02
Sheet Cutting Suite	4	3	1122	5243	2.83	9.93	00:06:16
The Itemizer	5	4	5121	18442	12.93	27.94	00:00:02
Smart Cut	5	4	10100	18442	19.13	27.94	00:00:02

Table 3. 4: Computational results for problem 1

Table 3.4 illustrates that Sheet Cutting Suite takes the most computational time with CUTLOGIC 2D being second most, while the other software packages take less time. The waste of 846 units generated by the CUTLOGIC 2D system on *N-1* sheets is better than the waste generated by the other packages on *N-1* sheets.

3.3.2 Problem instance 2

The data used for problem instance 2 is obtained from Christofides and Whitlock (Christofides & Whitlock, 1977:30) in their paper entitled "An algorithm for two-dimensional cutting problems". They proposed a depth-first branch and bound algorithm for the solution of the two-dimensional constrained cutting stock problem.

Problem	Stock sheet length (L) and width (W)	Demand rectangle's length (l), width (w) and upper bound (b) where b specifies the number of times a demand rectangle must be cut.
2	(40,70)	(21,22,1);(31,13,1);(9,35,3);(9,24,3);(30,7,2);(11,13,3);(10,14,1);(14,8,3); (12,8,3);(13,7,3);(31,43,4);(25,35,3);(22,32,3)

Table 3. 5: Problem instance 2

The data of problem instance 2 from Table 3.5 was used for each of the chosen software packages to solve the 2DSSSCSP.

Product Name	Number of Sheets Used (N)	N-1	Waste on N-1 Sheets	Total Waste on N Sheets	Waste Percentage on N-1 Sheets	Total Waste Percentage on N Sheets	Computational Time
CUTLOGIC 2D	6	5	753	2387	5.38	14.21	00:00:23
Cutting Optimizer Pro	6	5	1170	2387	8.36	14.21	00:00:04
Sheet Cutting Suite	6	5	1796	2387	12.8	14.21	00:00:05
The Itemizer	9	8	8763	10684	39.12	42.40	00:00:03
Smart Cut	8	7	5654	7989	28.84	35.67	00:00:03

Table 3. 6: Computational results for problem 2

Table 3.6 illustrates that the CUTLOGIC 2D system takes the most computational time to optimize the problem while the other software packages take less time. The waste of 753 units generated by the CUTLOGIC 2D system on *N-1* sheets is better than the waste generated by the other software packages on *N-1* sheets.

3.3.3 Problem instance 3

The data used for problem instance 3 is obtained from Tschöke and Holthöfer (Tschöke & Holthöfer, 1995:285) in their paper entitled "A new parallel approach to the constrained two-dimensional cutting stock problem". They presented a best-first branch and bound algorithm to solve the constrained two-dimensional cutting problem.

Problem	Stock sheet length (L) and width (W)	Demand rectangle's length (l), width (w) and upper bound (b) where b specifies the number of times a demand rectangle must be cut.
3	(99,99)	(14,16,1);(14,31,2);(15,17,3);(17,19,3);(18,19,2);(20,34,2);(20,39,3);(21,29,2);(25,25,2);(27,27,3);(29,34,2);(30,25,2);(30,27,3);(33,30,2);(35,30,2);(37,27,4);(38,49,2);(40,25,2);(43,28,4);(44,34,4)

Table 3. 7: Problem instance 3

The data of problem instance 3 from Table 3.7 was used for each of the chosen software packages to solve the 2DSSSCSP.

Product Name	Number of Sheets Used (N)	N-1	Waste on N-1 Sheets	Total Waste on N Sheets	Waste Percentage on N-1 Sheets	Total Waste Percentage On N Sheets	Computational Time
CUTLOGIC 2D	5	4	669	6638	1.71	13.54	00:01:56
Cutting Optimizer Pro	5	4	1409	6638	3.59	13.54	00:00:04
Sheet Cutting Suite	5	4	1677	6638	4.28	13.54	00:00:03
The Itemizer	6	5	8007	16442	16.34	27.96	00:00:03
Smart Cut	6	5	10021	16442	20.45	27.96	00:00:03

Table 3. 8: Computational results for problem 3

Table 3.8 illustrates that the CUTLOGIC 2D system takes the most computational time to produce a solution while the other software packages take less time. The waste of 669 units generated by the CUTLOGIC 2D system on *N-1* sheets is better than the waste generated by the other software packages on *N-1* sheets.

3.3.4 Problem instance 4

The data used for problem instance 4 is from Hifi (Hifi, 1997:727) in his paper entitled "An improvement of Viswanathan and Bagchi's exact algorithm for constrained two-dimensional cutting stock".

Problem	Stock sheet length (L) and width (W)	Demand rectangle's length (l), width (w) and upper bound (b) where b specifies the number of times a demand rectangle must be cut.
4	(70,80)	(29,41,1);(31,43,2);(32,39,1);(27,38,4);(28,37,3);(29,36,2);(35,35,3);(26,34,4);(31,24,2);(22,33,2);(31,22,3);(29,21,4);(19,27,2);(18,24,2);(17,25,2);(17,24,2);(25,17,1);(21,16,2);(19,15,3);(15,14,1);

Table 3. 9: Problem instance 4

The data of problem instance 4 from Table 3.9 was used with each of the chosen software packages to solve the 2DSSSCSP.

Product Name	Number of Sheets Used (N)	N-1	Waste on N-1 Sheets	Total Waste on N Sheets	Waste Percentage on N-1 Sheets	Total Waste Percentage On N Sheets	Computational Time
CUTLOGIC 2D	7	6	673	4446	2.00	11.34	00:02:36
Cutting Optimizer Pro	7	6	1792	4446	5.33	11.34	00:00:04
Sheet Cutting Suite	7	6	1968	4446	5.86	11.34	00:00:03
The Itemizer	8	7	7638	10049	19.48	22.43	00:00:03
Smart Cut 2D	8	7	8475	10049	21.61	22.43	00:00:03

Table 3. 10: Computational results for problem 4

Table 3.10 illustrates that the CUTLOGIC 2D system takes the most computational time to produce a solution while the other software packages take less time. The waste of 673 units generated by the CUTLOGIC 2D system on *N-1* sheets is better than the waste generated by the other software packages on *N-1* sheets.

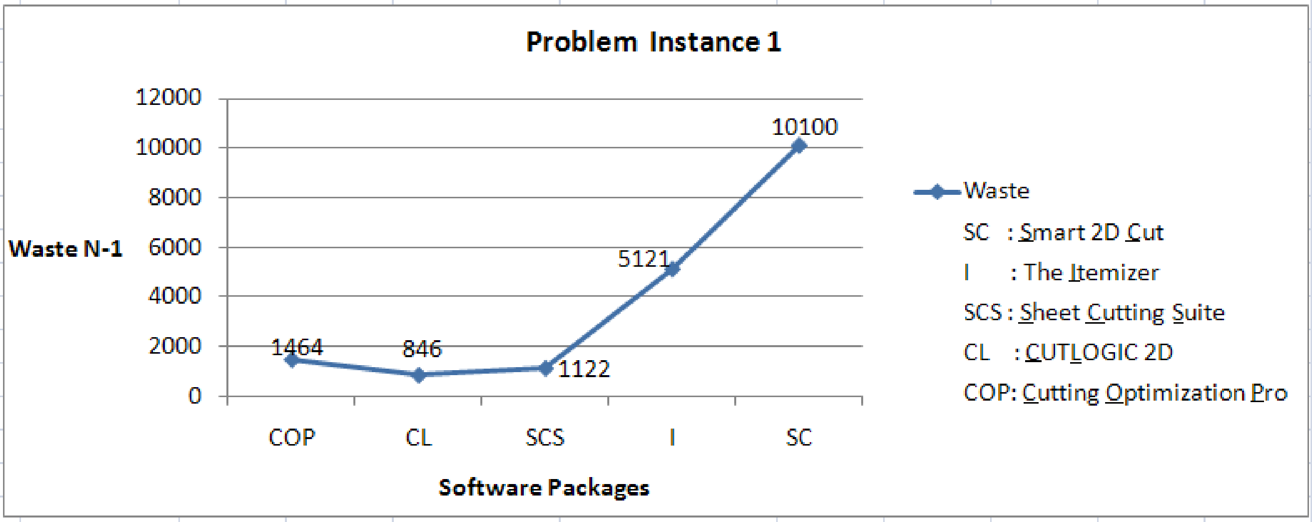
The Itemizer and Smart 2D Cut produce more waste than the other software packages since they use more stock sheets.

3.4 Analysis of results

The results obtained from the different commercial software packages will be analyzed using graphs for each problem instance. The graphs represent the waste generated on *N-1* sheets by each software package. These analyses will assist in choosing the best software package that will be used further in the research to solve the 2DSSSCSP.

3.4.1 Problem instance 1

The numbers on the Y-axis in Graph 3.1 represent waste generated by the different chosen software packages on *N-1* sheets.

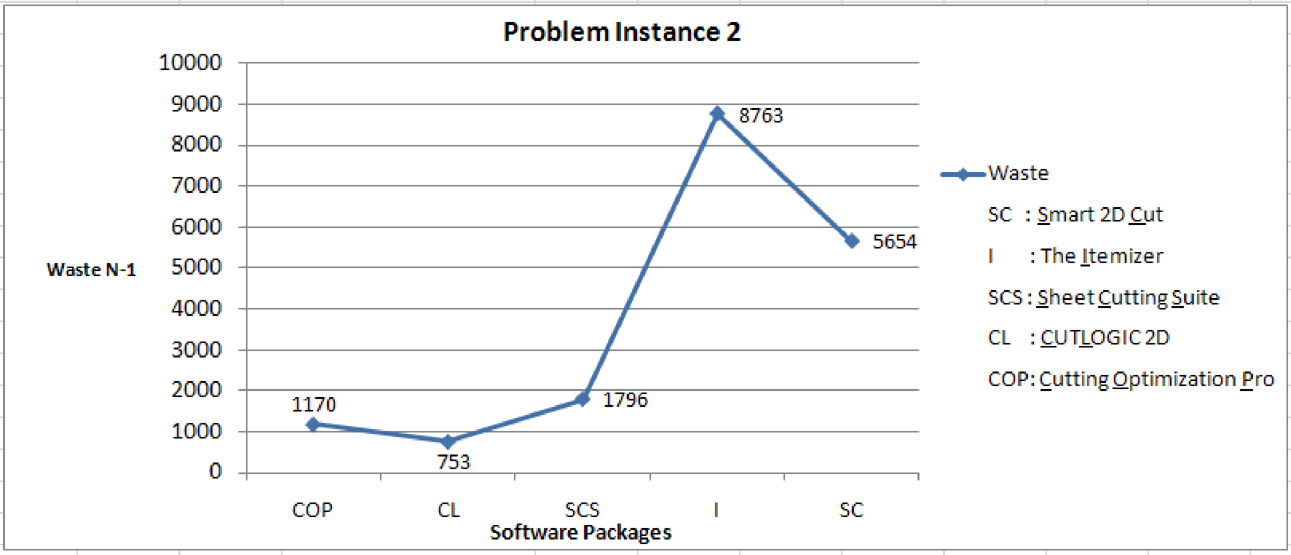


Graph 3. 1: Comparison of waste on N-1 sheets for problem instance 1

From Graph 3.1 above, the CUTLOGIC 2D system generates less waste than the other software packages whereas the Smart 2D Cut system generates more waste than the others.

3.4.2 Problem instance 2

The numbers on the Y-axis in Graph 3.2 represent waste generated by the different chosen software packages on *N-1* sheets.

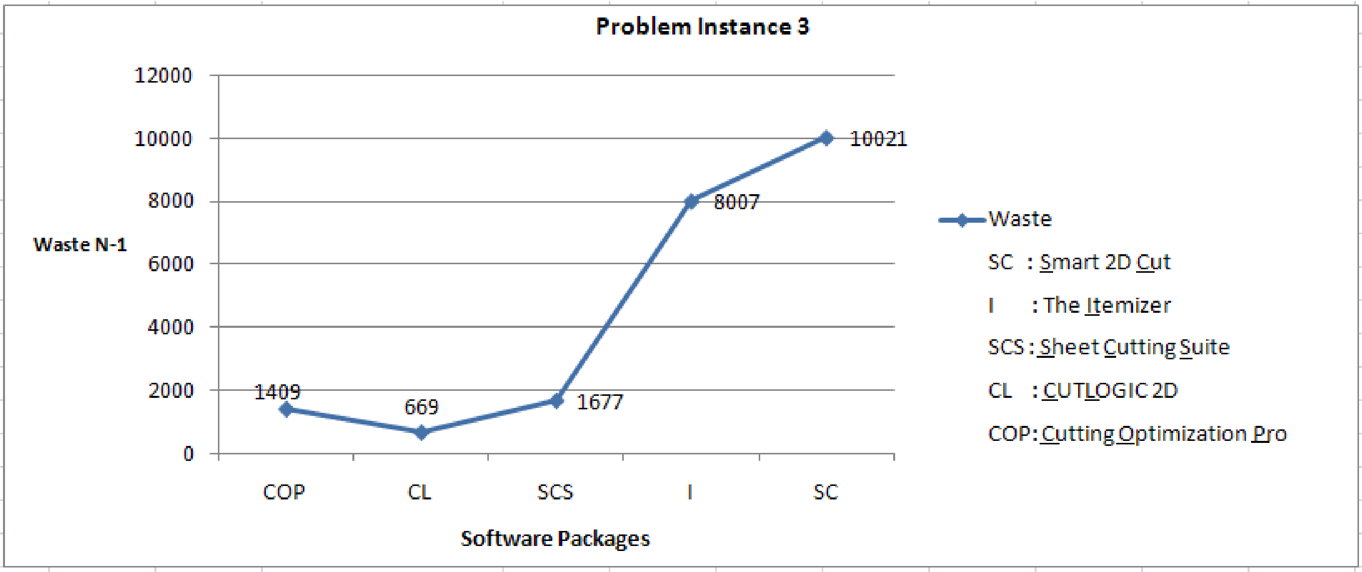


Graph 3. 2: Comparison of waste on N-1 sheets for problem instance 2

From Graph 3.2 above, the CUTLOGIC 2D system generates less waste than the other software packages whereas The Itemizer system generates more waste than the others.

3.4.3 Problem instance 3

The numbers on the Y-axis in Graph 3.3 represent waste generated by the different chosen software packages on *N-1* sheets.

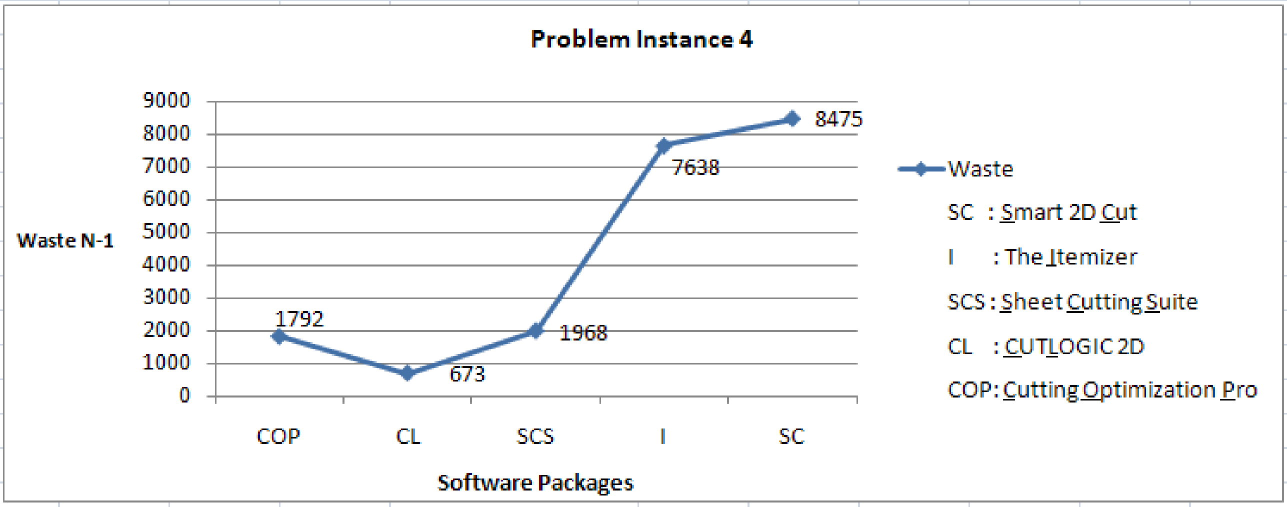


Graph 3. 3: Comparison of waste on N-1 sheets for Problem instance 3

From Graph 3.3 above, the CUTLOGIC 2D system generates less waste than the other software packages while the Smart 2D Cut software generates more waste than the others.

3.4.4 Problem instance 4

The numbers on the Y-axis in Graph 3.4 represent waste generated by the different chosen software packages on *N-1* sheets. Graph 3.4 shows that CUTLOGIC 2D software generates less waste than the other software packages and the Smart 2D Cut software generates more waste.



Graph 3. 4: Comparison of waste on N-1 sheets for problem instance 4

From the information presented above, the CUTLOGIC 2D software performs better than the other packages in terms of minimizing waste. This software utilizes fewer stock sheets, which is the main aim of solving the 2DSSSCSP. Therefore, the CUTLOGIC 2D software package will be used further in the research to generate some cutting patterns for the problem instances considered.

3.5 CUTLOGIC 2D version 3.30 - Panel cutting optimization software

The CUTLOGIC 2D software is one of the commercial optimization software systems which are used to solve the 2DSSSCSP. It was chosen after conducting empirical experiments involving some of the available optimization software packages. It was chosen because it produced better results based on the empirical experimentation carried out in paragraph 3.3 for the different problem instances.

The CUTLOGIC 2D software will be used to generate cutting patterns as discussed later in this dissertation.

3.5.1 What is CUTLOGIC 2D?

CUTLOGIC 2D is an optimizer for cutting rectangular material in industries such as woodworking, furniture, sheet metal, glass cutting (TMACHINES, 2010). It minimizes waste by calculating good or near-optimal cutting layouts. It is a good product in terms of quality of optimization and it offers extensive cutting options which are often not available in similar cutting tools.

3.5.2 Why CUTLOGIC 2D

The solution process used by the software system is based on genetic algorithms which provide high-quality, reliable and stable optimization for any cutting case (TMACHINES, 2010). User interfaces in the software are designed with the accent on easy usage, minimal data entry time and maximal ergonomics. The software can be customized to meet customer needs.

The CUTLOGIC 2D contains a number of main features and some of the features include the following.

- Guillotine (edge to edge) and non-guillotine cutting.
- Part rotation allowance.
- Cost driven optimization – prioritization based on defined price of used sources.

3.6 The greedy approach

A greedy algorithm is a method that makes a decision which seems the best at that moment (BOWDOIN, 2010). The decision taken is based on the basis of the information available at that time without worrying about the effects of that decision later on. The Wang algorithm used in this research was employed in a greedy way in order to solve the 2DSSSCSP. The results obtained from this approach will be compared against the results obtained from the chosen commercial software packages. The greedy approach will also be used to generate a set of cutting patterns which will be used to formulate an integer linear programming model which is discussed in Chapter 5.

The 2DSSSCSP requires that a single size stock sheet be used to cut the set of required rectangular items. The typical trim loss problem will be employed in a greedy way. For the greedy approach, the 2DSSSCSP attempts to cut the single stock sheet as good as possible with a minimum waste. If the demand of the items is not met from the single stock sheet, another stock sheet with the same dimensions will be used. This is done by deducting the demand items that have been cut from the list of demand items. The remaining set of demand items will be cut from the next stock sheet. This procedure is repeated until all the demand items are cut. A guillotine cutting pattern with a minimum total trim loss will be produced by the Wang algorithm for each stock sheet. The total trim loss for each stock sheet will be combined to calculate the total waste produced for the problem or customer order. This will typically not be optimal for the order.

To meet the demand for the required rectangles, there will be a large supply of the rectangular stock sheets S of the same size. The idea behind this approach is to utilize the least number of rectangular stock sheets. To summarize, for the greedy approach, the best cutting pattern is produced on a single stock sheet. This procedure is repeated until the customer order or demand items are met utilizing a number of stock sheets.

In this research, the Wang algorithm that will be discussed in the next paragraph will be used to solve the trim loss problem and to employ the greedy approach to solve the 2DSSSCSP.

3.7 Wang algorithm

The Wang algorithm, which was developed by Wang (Wang, 1983) is an algorithm that is used to solve the constrained two-dimensional cutting stock problem over a single large object, also called the trim loss problem as indicated in Chapter 1. In this research study, the Wang algorithm will be employed in two ways. Firstly, since it is basically an algorithm to solve the trim loss problem (thus one large object at a time), it will be used to solve the 2DSSSCSP by means of a greedy approach as discussed in paragraph 3.6. Secondly, it will be adapted to generate cutting patterns to be used when formulating an integer linear programming model to solve the 2DSSSCSP as discussed in Chapter 5.

The Wang algorithm was presented by Wang (Wang, 1983:573) who proposed an inventive way of building larger rectangles by joining smaller ones. Rectangles are steadily generated taking into consideration the original demand for rectangles. Rotation of each item is allowed in each state of the algorithm. Every newly generated build may or may not contain trim loss. The building of rectangles results in guillotine builds being formed. The problem can be defined as:

Let $L \times W$ be a rectangular stock sheet S having length L and width W, and let R be a set of demand rectangles of type R_i (1, 2, 3,……., n). Each type has dimensions of length l_i and width w_i and a demand constraint of b_i. A guillotine cutting pattern with a minimum trim loss is determined that uses no more than b_i replicates of rectangle R_i (1, 2, 3,……, n). This typical trim loss problem can be stated in the form:

$$Maximize \sum_{i}^{n} x_i l_i w_i$$

subject to $0 \leq x_i \leq b_i$ $\quad (i = 1, 2 \ldots\ldots, n)$ and

Guillotine cutting patterns applied to S.

x_i integer $\quad (i = 1, 2 \ldots\ldots, n)$

In the formulation, x_i is an integer indicating the number of times the rectangle R_i appears in a guillotine cutting pattern. The idea is that the area of the stock sheet being used is maximized thus minimizing the waste.

Wang (Wang, 1983) states that for each of the newly generated rectangles, three main constraints are considered.

- Demand rectangle's dimensions ($l_i \times w_i$) must be less than or equal to the stock sheet dimensions $L \times W$.

- Internal trim loss must be less than or equal to a certain predetermined percentage (indicated by β)to the stock sheet.
- The number of demand rectangles of a specific type i cut from the stock sheet must be less than or equal to its upper bound b_i.

3.7.1 The bottom-up approach

The bottom-up approach uses a list of demand rectangles which can be rotated or not during the formation of builds. Figure 3.1 displays two rectangles that need to be cut whereas Figure 3.2 displays the rotated versions of the rectangles.

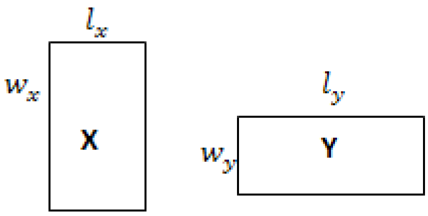


Figure 3. 1: Rectangles

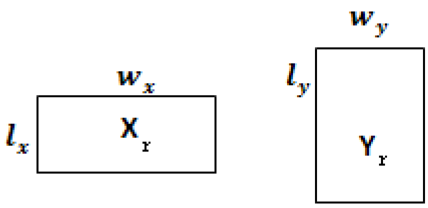


Figure 3. 2: Rotated rectangles

The bottom-up building approach is one that is based on using horizontal and vertical builds on rectangles. All the cuts are of a guillotine type.

- **Horizontal build**

A horizontal build of two rectangles X = $(l_x \times w_x)$ and Y = $(l_y \times w_y)$ is a rectangle S_h having dimensions $(l_x + l_y) \times \max(w_x, w_y)$ and containing X and Y.

- **Vertical build**

A vertical build of two rectangles X = $(l_x \times w_x)$ and Y = $(l_y \times w_y)$ is a rectangle S_v having dimensions $\max(l_x, l_y) \times (w_x + w_y)$ that contain X and Y.

Figures 3.3 and 3.4 illustrates how a vertical and a horizontal build is constructed by using the given demand rectangles.

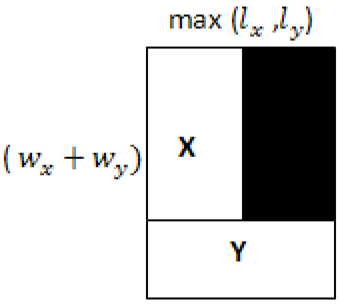


Figure 3. 3: Vertical build (S_v)

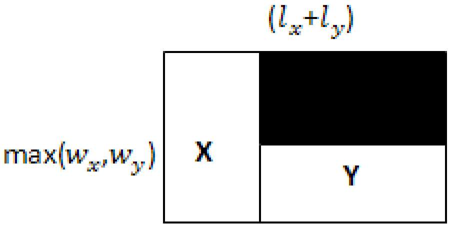


Figure 3. 4: Horizontal build (S_h)

3.7.2 Trim loss

During the cutting process, the stock sheet can rarely be utilized fully (100%). The remaining residual pieces or unused area result in trim loss. Trim loss is the amount of waste generated during the cutting process of a single sheet. This waste usually results from a mismatch in the relevant dimensions of the items/rectangular pieces being combined. Three types of trim loss may occur, namely internal trim loss, external trim loss and total trim loss. These types of trim loss are explained next.

- **Internal trim loss**

  Internal trim loss is the waste generated by combining two rectangular pieces. These pieces can either be combined vertically or horizontally. The shaded areas in Figure 3.5 display internal trim loss generated from vertical and horizontal builds.

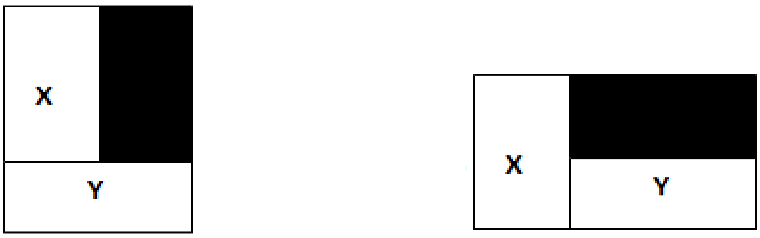


Figure 3. 5: Internal trim loss from a vertical and horizontal build

The internal trim loss in a vertical build can be computed as follows. Let TL_V represent the internal trim loss for a vertical build and TL_H represent the internal trim loss for a horizontal build. Let $l \times w$ represent the area of the new rectangle formed during the combination of rectangles. l represents the length of the new rectangle and w represents the width of the new rectangle.

$TL_V = (w_x + w_y) \times \max(l_x, l_y) - ((w_x \times l_x) + (w_y \times l_y))$

$TL_H = \max(w_x, w_y) \times (l_x + l_y) - ((w_x \times l_x) + (w_y \times l_y))$

- **External trim loss**

External trim loss refers to the amount of waste that is generated when a build is placed onto the stock sheet. In this instance, rectangles X and Y are combined to form rectangle Z with dimensions $l \times w$ and rectangle Z is placed onto the stock sheet T with dimensions $L \times W$. Figure 3.6 illustrates how external trim loss is generated.

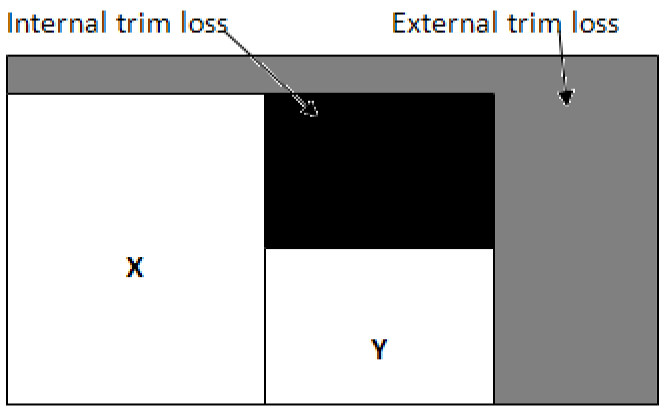


Figure 3. 6: External trim loss

The external trim loss can be calculated as follows. Let TL_E represent external trim loss.

$TL_E = L \times W - l \times w$

- **Total trim loss**

Total trim loss is the waste generated by adding the internal trim loss and the external trim loss.

3.7.3 Waste percentages (β)

Wang proposed two algorithms that are based on the method of generating rectangles. Both algorithms use percentage wastage β as a measurement to determine acceptable internal trim loss. The first algorithm employs β_1 which is calculated with respect to the area of the stock sheet $L \times W$. The second algorithm employs β_2 which is calculated with respect to the area of the rectangle that was generated by means of a horizontal or vertical build. β is chosen to be between 0 and 1 where 0 implies a percentage of 0% and 1 imply a percentage of 100%.

3.7.4 Wang algorithm constraints

As defined in paragraph 3.7, the Wang algorithm starts an iterative process of combining each rectangle in turn with itself and then with all the other rectangles to form vertical and horizontal builds. This building process forms new rectangles that may lead to the generation of internal trim loss. The process is repeated so that each successive horizontal or vertical build forms a larger guillotine rectangle from two smaller guillotine rectangles. The generation of all the possible rectangular items is subject to the following constraints.

- **Acceptable wastage (β)**

  The generated rectangles will be measured against the acceptable percentage wastage β with respect to the area of the stock sheet. Generated rectangles that contain internal trim loss of less than or equal to $\beta \times L \times W$ will be stored for further consideration while the rest will be discarded.

- **Rotation of rectangles**

  The rotation of the rectangles also plays a vital role in finding all possible combinations of builds. Each build or rectangle is considered in its normal orientation as well as rotated orientation.

- **Dimensions**

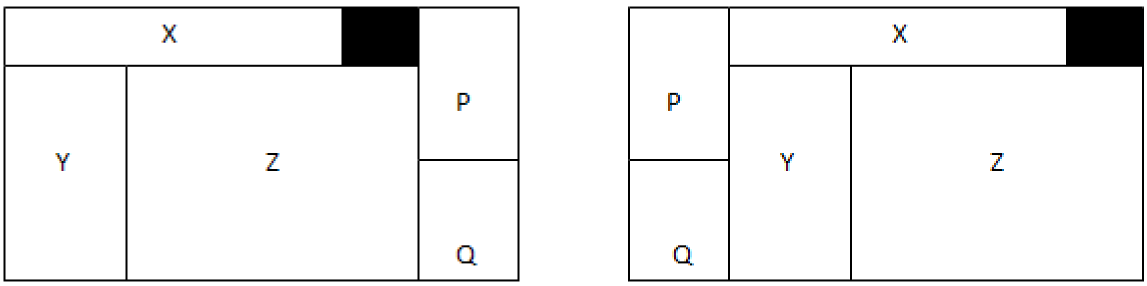
If a build or rectangle exceeds the size of the stock sheet, that build is discarded and only those that are within the dimensions of the stock sheet are stored for further consideration.

- **The upper bound**

Guillotine rectangles (builds) that contain more than b_i replicates of a rectangle R_i are also eliminated from further consideration.

- **Duplicates and equivalents**

The formation of duplicates can also be detected and discarded. Figure 3.7 illustrates some equivalent rectangles that can occur. Pattern 2 is a spatial variation of pattern 1 and thus pattern 2 will be discarded.



1 **2**

Figure 3. 7: Types of equivalents

The building process of rectangles continues until no new rectangles can be generated by means of horizontal and vertical builds. If one or more allowable builds exist, the cutting pattern that produces the least total waste for the predetermined β value is chosen. This cutting pattern yields the optimal pattern for the trim loss problem. The two algorithms developed by Wang (Wang, 1983) are given below.

3.7.5 Wang's two algorithms

In order to solve the trim loss problem, Wang (Wang, 1983) proposed two methods that generate constrained cutting patterns by successive horizontal and vertical builds of rectangles. Each of the algorithms uses a parameter to limit the maximum acceptable percentages of waste that they create. The first algorithm measures β_1 with respect to the area of the stock sheet $L \times W$ while the second measures β_2 with respect to the area of T or A (T) where T is a rectangle that is formed by a horizontal or vertical build. Algorithm one thus functions in terms of the total waste (total trim loss) whereas algorithm two operates in terms of the internal waste (internal trim loss). Algorithm one provides a better measure of the optimality of the best solution than algorithm two. However, the second algorithm usually finds an optimal solution faster than the first algorithm.

1. Algorithm one

A description of the first algorithm adapted from Wang (Wang, 1983) is given below.

Step 1:

a) Choose a value for β_1, $0 \leq \beta_1 \leq 1$.
b) Define $L^{(0)} = F^{(0)} = \{R_1, R_2, \ldots\ldots, R_n\}$, and set $k = 1$.

Step 2:

a) Compute $F^{(k)}$ which is the set of all rectangles T satisfying
   i. T is formed by a horizontal or vertical build of two rectangles from $L^{(k-1)}$. Rotation is allowed.
   ii. The amount of internal trim loss in T does not exceed $\beta_1 \times L \times W$ and
   iii. The number of rectangles R_i appearing in T does not violate the bound constraints $b_1, b_2, \ldots, b_n$.
b) Set $L^k = L^{(k-1)} \cup F^{(k)}$. Remove equivalent rectangle patterns from L^k.

Step 3:

If $F^{(k)}$ is nonempty, set $k \leftarrow k + 1$ and go to Step 2 otherwise, go to step 4

Step 4:

a. Set $M = k - 1$.

b. Choose the rectangle of $L^{(M)}$ that has the smallest total trim loss when placed onto the stock sheet $L \times W$.

2. Algorithm two

A description of the second algorithm adapted from Wang (Wang, 1983) is given below.

Step 1:

a) Choose a value for β_2, $0 \leq \beta_2 \leq 1$.
b) Define $L^{(0)} = F^{(0)} = \{ R_1, R_2, \ldots\ldots, R_n\}$, and set $k = 1$.

Step 2:

a. Compute $F^{(k)}$ which is the set of all rectangles T satisfying
   i. T is formed by a horizontal or vertical build of two rectangles from $L^{(k-1)}$. Rotation is allowed.
   ii. The amount of internal trim loss in T does not exceed $\beta_2 \times A$ (T) and
   iii. The number of rectangles R_i appearing in T does not violate the bound constraints $b_1, b_2, \ldots, b_n$.
b. Set $L^k = L^{(k-1)} \cup F^{(k)}$. Remove equivalent rectangle patterns from L^k.

Step 3:

If $F^{(k)}$ is nonempty, set $k \leftarrow k + 1$ and go to Step 2 otherwise, go to step 4

Step 4:

a. Set $M = k - 1$.
b. Choose the rectangle of $L^{(M)}$ that has the smallest total trim loss when placed onto the stock sheet $L \times W$.

The difference between the two algorithms is the replacement of β_1 by β_2 in step 1 (a) of the second algorithm while step 2 (a) (ii) of algorithm one is replaced by the amount of trim loss in T does not exceed $\beta_2 \times A$ (T). The two algorithms show an iterative process of building rectangles using Wang's algorithm.

3.8 Summary

This chapter gave an overview of some available software packages for solving the 2DSSSCSP. The overview highlighted the differences among a number of industry-oriented software packages. Empirical experimentation was conducted in order to test some of these software packages using data from previous research papers.

The results obtained from those packages were compared with each other which led to the CUTLOGIC 2D software being chosen as the best package currently available. It will be used further in the research to generate further cutting patterns that will be combined with patterns generated from the Wang algorithm.

A discussion of the Wang algorithm was done to gain an understanding of the algorithm. The Wang algorithm will be used further in the research to generate more cutting patterns that will be combined with patterns generated from the CUTLOGIC 2D software.

Chapter 4: Heuristic procedure

4.1 Introduction

This chapter gives a detailed explanation of the heuristic procedure proposed for this study as regards solving the 2DSSSCSP. Two heuristic procedures are developed. The integer linear programming model formulated to solve the 2DSSSCSP will be solved using an optimization solver called IBM ILOG CPLEX. The two heuristic procedures are outlined in paragraph 4.2 and then discussed in paragraph 4.3.

4.2 Algorithm outline

The two heuristic procedures developed to solve the 2DSSSCSP are based on cutting pattern generation. These cutting patterns are generated by means of the chosen software package which is the CUTLOGIC 2D software and the Wang algorithm implementation. The fundamental concept is to utilize the available systems to generate a set of cutting patterns. This includes the solutions by the systems for the whole order and also alternative patterns that may be used in order to improve the solutions.

4.2.1 Cutting pattern selection

For a given set of stock rectangles and an order of demand items, a selection of cutting patterns that utilize the least number of stock sheets and generates the least waste is required.

4.2.2 Algorithm details

The description of the heuristic procedure requires the following definitions.

Let

K be the set of stock sheets of same sizes available for solving the 2DSSSCSP.

m be the number of different types (sizes) of demand items to be cut from the stock sheets.

W be the width of the stock sheet.

L be the length of the stock sheet.

w_j be the width of the j^{th} demand item.

l_j be the length of the j^{th} demand item.

- **Cutting pattern generation**

  For the original demand items $j(j = 1, \ldots \ldots, m)$ and the stock sheets from K, solve the 2DSSSCSP using either the Wang algorithm implemented in a greedy way or the CUTLOGIC 2D software resulting in cutting patterns. This is performed repeatedly with different parameters or starting points as explained below until an extended set of cutting patterns is obtained from the Wang algorithm and the CUTLOGIC 2D software. These sets will be used for further processing.

4.2.3 The Wang algorithm program

The Wang algorithm system used in this research study for solving the 2DSSSCSP was programmed by Oberholzer (Oberholzer, 2003). In his thesis, he developed a system that implemented artificial intelligence search methods which generate a guillotine build or pattern for a single stock sheet. This system will be employed in this research study to generate cutting patterns. These cutting patterns will be combined with cutting patterns generated from the chosen commercial software package. The combined cutting patterns will be used to formulate an integer linear programming model. This integer linear programming model will be solved using an optimization solver which will determine the best cutting patterns required to solve the 2DSSSCSP.

4.3 Two Heuristic procedures

The two heuristic procedures developed to solve the 2DSSSCSP are discussed next.

4.3.1 Heuristic procedure 1

The notion underlying this heuristic procedure is to generate many cutting patterns that will be used to formulate an integer linear programming model. These cutting patterns are generated by using a combinatoric process. Short descriptions of the main ideas are given below. These are followed by a more formal presentation.

Firstly, the software is applied to the original demand items to yield a sequence of (original) cutting patterns that can be used to cut the customer order. The first pattern in this sequence is then considered. A demand item type occurring in this original first pattern is removed from the customer order and CUTLOGIC 2D or Wang is applied to the remaining customer order. This results in a new or another sequence of cutting patterns. The demand items in the first cutting pattern obtained are then removed from the original customer order, and the demand item type removed from the original first pattern is put back. This resulting customer order is given to CUTLOGIC 2D or Wang to produce the remaining sequence of patterns that will generate patterns for the rest of the customer order. This will result in a sequence of patterns that differs from the original sequence.

In order to generate even more cutting patterns, the demand item types from the original first pattern are removed one demand item type after the other in turn resulting in different demand lists. The problem is also handled again in the same way as described above by removing a combination of two, three, four and five demand item types. After each removal, the software is applied to the new set of demand items in order to obtain a new first pattern. In doing this, different cutting patterns are obtained. In each case, after the establishment of the new first pattern, the removed demand item types are put back on the list of uncut demand items and the problem is solved again until the order is met.

Two sets of sequences of cutting patterns will be obtained from the Wang algorithm and the CUTLOGIC 2D software. The Wang algorithm implementation is used to generate the first set of cutting patterns in a greedy way while the CUTLOGIC 2D software is used to generate the second set of cutting patterns. The cutting patterns will be used to formulate an integer linear programming model. The model will be solved to try and improve the solution obtained from the commercial software.

1. Pattern generation with the greedy Wang algorithm

The concept behind the greedy approach in this study is to generate cutting patterns using available software. A more formal description of this process is given below.

- **Initial cutting pattern generation**

  The purpose of the initial cutting pattern is to obtain a starting point for the generation of cutting patterns. Firstly apply the Wang algorithm to the original set of demand items $j(j = 1, \dots \dots, m)$ and the given stock sheets to produce the original first cutting pattern.

  Keep track of the demand items delivered in this first cutting pattern.

- **Further cutting pattern generation**

  The aim of this phase is to use the above mentioned initial cutting pattern as the basis to implement different starting points in order to generate different cutting patterns. Depending on the number of demand items obtained on the first stock sheet above, with reference to Figure 4.1 the following combinatoric process is employed:

  Let

  $P1$ be the first cutting pattern generated by the Wang algorithm on the original data.

  d be the number of different types (sizes) of demand items cut in $P1$.

From the list of demand item types cut in $P1$, remove one demand item type j $(j = 1, \dots \dots, d)$ at a time in order to create different lists of demand items ($L*_{11}$, $L*_{21}$, $L*_{31}$,………., $L*_{d1}$). Apply the Wang algorithm repeatedly to one stock sheet and each of the lists of demand items to obtain d new cutting patterns ($P*_{11}$, $P*_{21}$, $P*_{31}$,………., $P*_{d1}$).

Removing the demand items used for each of the cutting patterns ($P*_{11}$, $P*_{21}$, $P*_{31}$, …….., $P*_{d1}$) from the original demand item list L, results in d demand item lists $i.e.$ ($L*_{12}$, $L*_{22}$, $L*_{32}$,…, $L*_{d2}$). This implies the reduction of each of the demand lists by means of the demand items used on the relevant first pattern $P*_{j1}$ and also entails replacing the demand item type that was removed from the relevant list $L*_{j1}$ resulting in $L*_{j2}$.

Use each of these demand item lists $L*_{j2}$ $(j = 1,2, \dots \dots \dots, d)$ as the starting point to apply the Wang algorithm in a greedy way to solve the 2DSSSCSP. For each of these lists, a series of cutting patterns is thus obtained. The relevant initial or starting cutting pattern and the d

different series of cutting patterns are combined. This results in d different solutions to the original cutting stock problem. The solutions can be denoted as follows:

$S_1 = P*_{11}, P*_{12}, \ldots\ldots\ldots, P*_{1k}$

$S_2 = P*_{21}, P*_{22}, \ldots\ldots\ldots, P*_{2k}$

$S_3 = P*_{31}, P*_{32}, \ldots\ldots\ldots, P*_{3k}$

$\vdots$

$S_d = P*_{d1}, P*_{d2}, \ldots\ldots\ldots, P*_{dk}$

The same procedure is applied for the generation of other cutting patterns through the combination of two, three, four and five demand items using the demand item types obtained in $P1$. In this manner a number of cutting patterns are generated, with the waste associated with each also being available.

In summary, to illustrate the first heuristic procedure graphically, see Figure 4.1. The Wang algorithm implementation is applied in Block A whereas the Wang greedy procedure is applied in Block B.

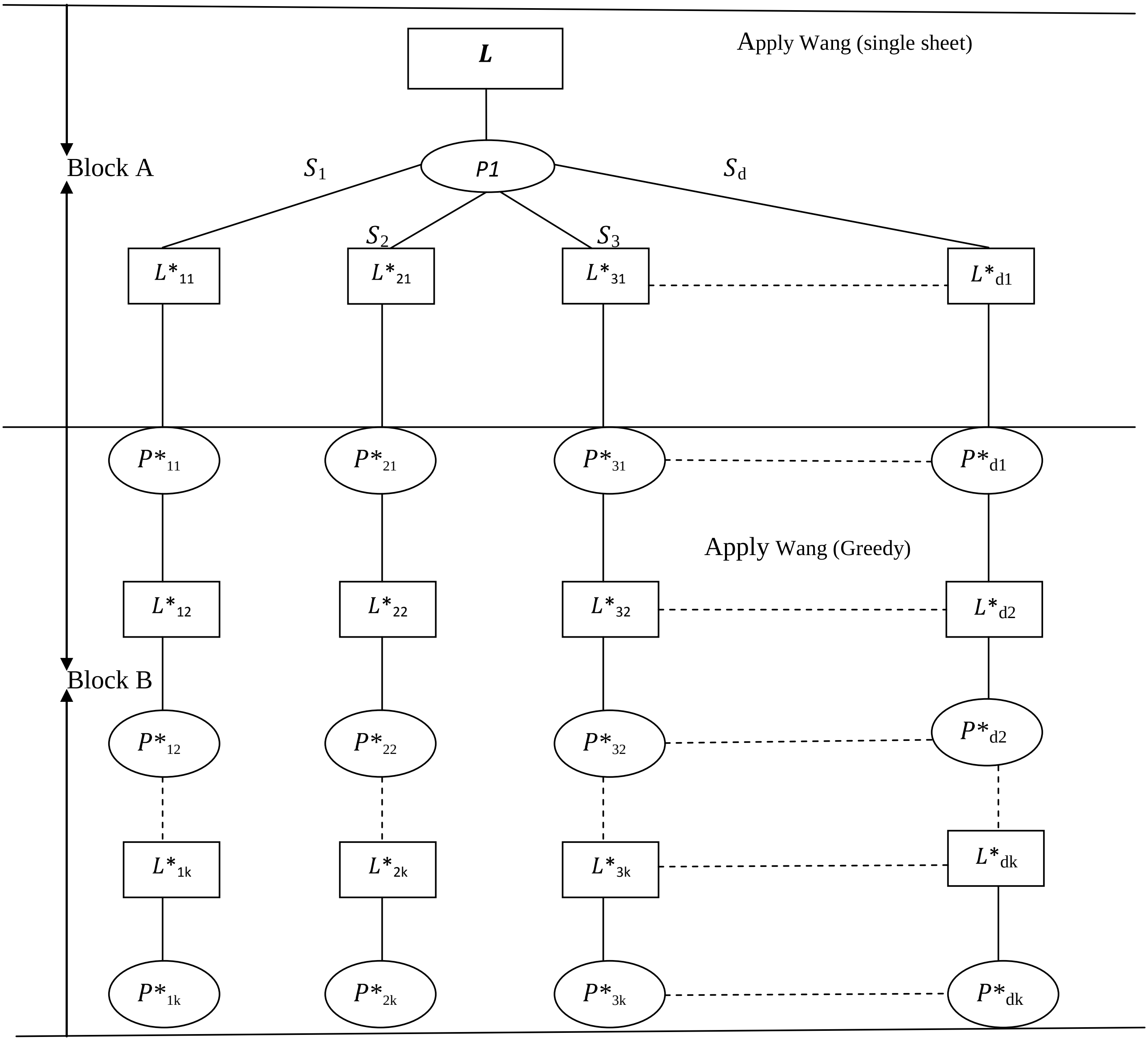


Figure 4. 1: General description of the heuristic procedure

❖ **Example**

Assume L = {1, 2, 3, 4, 5} represents the original list of demand items and $P1$ = {1, 4} represents the list of demand items cut in $P1$. Let b = [1, 1, 2, 2, 1] represent the quantity required for each demand item.

A starting list with the removed demand items is obtained.

$L^*_{11} = L - \{1\}$ =≫ {2, 3, 4, 5}: removing demand item 1 as from $P1$ = {1, 4}.

$L^*_{21} = L$ - {4} =≫ {1, 2, 3, 5}: removing demand item 4 as from $P1$ = {1, 4}.

Apply the Wang algorithm to obtain the two starting lists. Assume the following cutting patterns are obtained.

P^*_{11} =≫ {4, 5} from list L^*_{11} and

P^*_{21} =≫ {3, 4} from list L^*_{21}

Demand items 3 and 4 require that they be cut twice according to the values in b above. Therefore the first instance of item 3 is obtained in P^*_{21} above and the second instance is obtained in P^*_{23} below. The first instance of item 4 is obtained in P^*_{11} above and the second instance is obtained in P^*_{12} below. New lists with demand items used and the initially removed demand items being put back are obtained. See lists L^*_{12} and L^*_{22} below.

$L^*_{12} = L - P^*_{11}$ =≫ {1, 2, 3, 4}: demand items used in P^*_{11} reduced and demand item 1 is replaced. One instance of item 4 is still needed.

$L^*_{22} = L - P^*_{21}$ =≫ {1, 2, 3, 4, 5}: demand items used in P^*_{21} reduced and demand item 4 is replaced. One instance of items 3 and 4 is still needed.

Apply the Wang algorithm on lists L^*_{12} and L^*_{22} in a greedy way to obtain a new cutting pattern. Assume the following cutting patterns are obtained.

P^*_{12} =≫ {3, 4} from list L^*_{12} and

P^*_{22} =≫ {1, 5} from list L^*_{22}

The following lists of demand items are obtained.

$L^*_{13} = L^*_{12} - P^*_{12}$ =≫ {1, 2, 3}

$L^*_{23} = L^*_{22} - P^*_{22}$ =≫ {2, 3, 4}

After applying the Wang algorithm again to the lists L^*_{13} and L^*_{23}, let us assume that the following cutting patterns are obtained

P^*_{13} =≫ {1, 2, 3} from list L^*_{13} and

P^*_{23} =⋙ {2, 3, 4} from list L^*_{23}.

Suppose P^*_{13} met the requirement and $L^*_{33} = L^*_{23} - P^*_{23}$ =⋙ {2} gives final pattern P^*_{24}. When all the requirements of the demand items are met, two series of cutting patterns are generated to form S_1 and S_2 respectively.

$S_1 = P^*_{11}, P^*_{12}, P^*_{13}$ and

$S_2 = P^*_{21}, P^*_{22}, P^*_{23}$

The same procedure is repeated for the combination of two demand items {1, 4}. In this case, two demand item types are removed from $P1$. If $P1$ contains more than two demand item types, a combination of three, four and five is removed at a time.

2. The CUTLOGIC 2D software

A combinatoric process of generating cutting patterns equivalent to the one described above for the Wang algorithm applied in a greedy way is used to generate the second set of cutting patterns using the CUTLOGIC 2D software. The two sets of cutting patterns generated can be combined to form a set of cutting patterns or can be used separately.

The set of cutting patterns generated as described above can be used to formulate a large integer linear programming model which will be solved with an optimization solver.

4.3.2 Heuristic procedure 2

The second heuristic procedure is an extension of the Wang algorithm. This heuristic is developed to handle problems where the combinatoric process described above did not generate enough new cutting patterns. This means there are no or few differences in patterns and thus a small number of different cutting patterns are used. The heuristic procedure described here extends the Wang algorithm by using a range of specified β-values in order to generate more cutting patterns. Three sets of cutting patterns are generated from this heuristic procedure. The first two sets are described in the previous paragraphs, that is, by means of the Wang procedure and the CUTLOGIC 2D software. The third set is obtained from generating cutting patterns using a specified β.

- **Cutting pattern generation using β**

  The original Wang algorithm utilizes a single sheet optimally and if a limited number of cutting patterns are generated, the Wang algorithm has to be extended. The program was altered to generate a number of cutting patterns using a certain range of β. During the building procedure of cutting patterns, all patterns are removed where the total waste exceeds a percentage β of the stock sheet size. Thus, only patterns with total waste less or equal to a given percentage β of the stock sheet size are kept. In doing so, the computational time is reduced. Thus by employing β a number of cutting patterns can be generated. The method used is to start the program with a small $\beta, i.e$ 0.01 and increase β by 0.01 until a sufficient number of cutting patterns is generated or a time constraint forces the procedure to stop. Depending on the β being chosen, the process may be time consuming and thus a small β is recommended. If a larger β is used, more cutting patterns are generated and consequently a better solution is possible using the integer linear programming model.

  For demand items of type j and the stock sheet size, generate all the possible cutting patterns using a specified range of β. Solve the trim-loss problem using the Wang algorithm. The program will generate all cutting patterns that produce a waste less than or equal to a specified percentage of the stock sheet. This yields a set of cutting patterns that produces relatively low waste.

This set can be combined with the other two sets obtained with the heuristic procedures based on the CUTLOGIC 2D and the greedy Wang procedure to produce a larger set of cutting patterns.

4.4 Summary

This chapter describes heuristic procedures developed for solving the 2DSSSCSP. The fundamental concept pursued in this chapter is the design of heuristic procedures that can be used to generate cutting patterns and that may be used in an integer linear programming model for the 2DSSSCSP. In many complex problems, the sets of all possible guillotine cutting patterns are generally too large to consider. The work in this chapter is an effort to reduce such sets to smaller and hopefully promising subsets.

Chapter 5: Integer linear programming approach and model

5.1 Introduction

In this chapter, the sets of cutting patterns generated in Chapter 4 are used to formulate an integer linear programming model to solve the 2DSSSCSP. Empirical experimentation based on the Wang algorithm (greedy) and CUTLOGIC 2D will be carried out to compare the two software packages and also to test the two heuristic procedures introduced in paragraph 4.3. The first empirical experimentation is discussed in paragraphs 5.2 and 5.3 respectively and the results are analyzed in paragraph 5.4. The integer linear programming model is discussed in paragraph 5.5 while the mathematical formulation is undertaken in paragraph 5.6. Empirical experiments to test the two heuristic procedures are carried out in paragraph 5.7.

5.2 Empirical experimentation: Wang algorithm (greedy approach)

During the empirical experimentation carried out in paragraph 3.3 in order to evaluate and compare a number of available commercial software systems to solve the 2DSSSSCSP, the CUTLOGIC 2D software package performed better than the other software packages. Therefore it was chosen as the software package to be used further in this study. Since an implementation of the Wang algorithm to solve trim loss problems was also available, it was also made use of in this regard.

The empirical experimentation carried out on the 2DSSSCSP instances is reported for the Wang algorithm (greedy) in this paragraph and for the CUTLOGIC 2D software in paragraph 5.3. The results are analyzed and compared in paragraph 5.4.

The same problem instances used in paragraph 3.3 were made use of in this stage.

Problem Number	Number of Sheets Used N	Number of N-1 Sheets	Waste on N-1 Sheet	Total Waste on N Sheets	Waste Percentage on N-1 Sheets	Total Waste Percentage on N Sheets	Computational Time
1	4	3	411	5243	1.04	9.93	01:32:53
2	6	5	1525	2387	10.89	14.21	00:03:53
3	5	4	902	6638	2.30	13.54	00:12:34
4	7	6	1289	4446	3.84	11.34	00:00:16

Table 5. 1: Computational results (Wang- greedy approach)

The above results obtained for the Wang algorithm (greedy) will be discussed and compared to those obtained for the CUTLOGIC 2D in paragraph 5.3.

5.3 Empirical experimentation: CUTLOGIC 2D

The results obtained from the empirical experimentation for the CUTLOGIC 2D software (paragraph 3.5) are recorded in Table 5.2 with respect to each of the problem instances.

Problem Number	Number of Sheets Used N	Number of N-1 Sheets	Waste on N-1 Sheet	Total Waste on N Sheets	Waste Percentage on N-1 Sheets	Total Waste Percentage on N Sheets	Computational Time
1	4	3	846	5243	2.14	9.93	00:04:14
2	6	5	753	2387	5.38	14.21	00:00:23
3	5	4	669	6638	1.71	13.54	00:01:56
4	7	6	673	4446	2.00	11.34	00:02:36

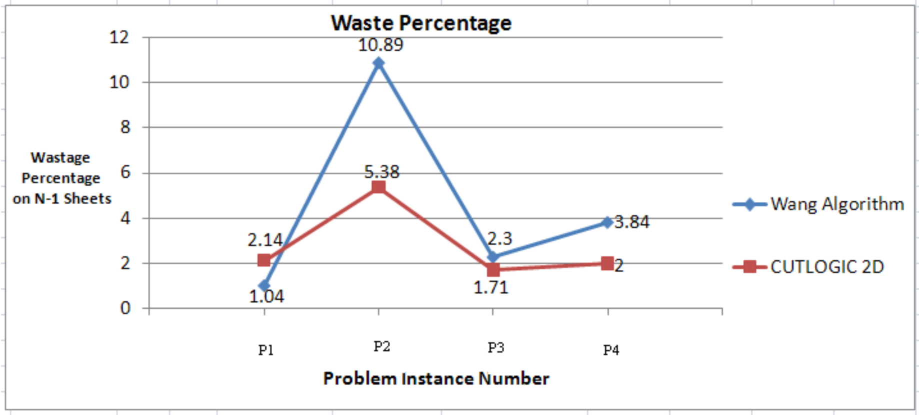
Table 5. 2: Computational results (CUTLOGIC 2D)

The CUTLOGIC 2D software performs better than the Wang algorithm (greedy) in terms of computational time in almost all of the problem instances and also outperforms the Wang algorithm (greedy) in terms of the waste percentage on *N-1* sheets.

5.4 Analysis of software results

The results obtained from the two different software packages will be analyzed using a graph. The graph represents the waste percentage generated on *N-1* sheets by the CUTLOGIC 2D software package and the Wang algorithm (greedy) on each problem instance.

The numbers on the Y-axis in Graph 5.1 represent the waste percentage generated by the Wang algorithm and the CUTLOGIC 2D software package on *N-1* sheets.



Graph 5. 1: Waste generated

Graph 5.1 illustrates the waste generated by the CUTLOGIC 2D software package and the Wang algorithm (greedy) in percentages. In problem instance 1 (P1), the Wang algorithm (greedy) generates a waste which is almost half that generated by the CUTLOGIC 2D software. In problem instance 2 (P2), the Wang algorithm (greedy) generates waste which is almost twice that generated by the CUTLOGIC 2D software. In problem instance 3 (P3) and 4 (P4), the Wang algorithm also generates more waste than the CUTLOGIC 2D software. The Wang algorithm (greedy) and the CUTLOGIC 2D software are therefore used further in the research to generate more cutting patterns.

5.5 Integer programming problem

An integer linear programming problem is a linear program in which some or all of the variables are restricted to be integers. Integer linear programming has become an important specialized area of optimization modelling (Moore & Weatherford, 2001). Integer programming occurs frequently because many decisions are essentially discrete in that one or more options must be chosen from a finite set of alternatives. The integer linear programming model is formulated as:

$$Minimize \qquad \sum_{j=1}^{n} c_j x_j$$

$$subject\ to \qquad \sum_{j=1}^{n} a_{ij} x_j = b_i \qquad (i = 1,2,3, \dots \dots \dots m),$$

$$x_j \geq 0 \qquad (j = 1,2,3, \dots \dots \dots n),$$

$$x_j \ \ integer \qquad (j = 1,2,3, \dots \dots \dots n).$$

Where A= $\begin{bmatrix} a_{11} & a_{12} & \cdots a_{1n} \\ a_{21} & a_{22} & \cdots a_{2m} \\ \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots a_{mn} \end{bmatrix}$ is known as the constraint matrix.

5.6 Problem formulation

In order to formulate the 2DSSSCSP, a large number of factors need to be defined which relate to the following:

- The stock rectangle;
- The items to be cut;
- The cutting patterns generated.

1. Stock rectangle

   Let

   L be the length of the stock sheet,

   W be the width of the stock sheet.

2. Items

   Let

   m be the number of different types (sizes) of items to be cut from the stock sheets,

   l_i be the length of an item of type i $(i = 1, \dots \dots, m)$,

   w_i be the width of an item of type i $(i = 1, \dots \dots, m)$.

3. Cutting patterns

   Let

   T be the total number of cutting patterns generated,

   P be a set of two-dimensional cutting patterns generated,

   d_{ij} be the number of items of type i $(i = 1, \dots \dots, m)$ that occur in cutting pattern j where pattern $j \in P$.

The 2DSSSCSP can now be formulated as an integer linear program.

Let

F_j be the amount of waste generated by cutting pattern j where $(j = 1, \dots \dots, T)$,

x_j to be computed as the number of times a cutting pattern is used where

$(j = 1, \dots \dots, T)$,

N be the total number of stock sheets utilized,

b_i be the number of items of type i $(i = 1, \dots \dots, m)$ required to be cut in order to meet the demand,

G_i be the amount of penalty for overcutting the item of type i $(i = 1, \dots \dots, m)$, (Overcutting is cutting more than the required demand)

s_{iv} to be computed as the number of times the item of type i is undercut where

$(i = 1, \dots \dots, m)$,

s_{iw} to be computed as the number of times the item of type i is overcut where $(i = 1, \dots \dots, m)$.

The objective function of the integer linear model seeks to minimize the amount of waste generated when solving the 2DSSSCSP.

Hence the integer linear program is:

$$\text{minimize} \sum_{j=1}^{T} F_j x_j + \sum_{i=1}^{m} G_i s_{iw} \quad (1)$$

Subject to

$$\sum_{j=1}^{T} d_{ij} x_j + s_{iv} - s_{iw} = b_i, \quad i = 1, \dots \dots, m, \quad (2)$$

$$\sum_{j=1}^{T} x_j = N - 1, \quad (3)$$

$$x_j \geq 0, integer\ j = 1, \dots \dots, T,$$

$$s_{iv} \geq 0, integer\ i = 1, \dots \dots, m, \quad (4)$$

$$s_{iw} \geq 0, integer\ i = 1, \dots \dots, m. \quad (5)$$

Objective function (1)

This is the objective function where F_j represents the amount of waste generated by the cutting pattern j. G_i represents the amount of penalty enforced for overcutting item i and is chosen as the area used by the item.

Constraint 2

s_{iv} represents the number of times the item i is undercut while s_{iw} represents the number of times item i is overcut and must be computed. These constraints ensure that if the cutting patterns produce more items than required, then the objective function will ensure that such items are counted as waste.

Constraint 3

This constraint ensures that a certain number of stock sheets are utilized. N is specified by the user.

Constraints (4) and (5) are the integrality constraints.

5.7 Empirical experimentations

To illustrate and test the heuristic procedure (paragraph 4.3), small problem instances gleaned from previous research papers and real world application problem instances from a large corporation called PG Glass Pty.Ltd located in South Africa will be used.

5.7.1 Small problem instances

The first heuristic procedure (paragraph 4.3.1) was employed to solve the set of small problem instances in order to generate sets of cutting patterns for these problem instances. In the end these cutting patterns were used to formulate an integer linear programming model to be solved. The results are compared.

1. Problem instance 1

Table 5.3 contains the problem instance from Cung (Cung et al., 2000:185) used in Chapter 3.

Problem	**Stock Sheet Length(L) and Width(W)**	**Demand items length (l), width (w) and upper bound (b) where b specifies the number of times a demand rectangle must be cut with each type being denoted by (l, w, b).**
1	(132,100)	(18,39,4);(13,49,2);(27,51,1);(65,31,3);(45,27,2);(69,21,4);(21,63,2); (54,41,2);(41,31,1);(37,22,1);(29,54,1);(47,31,2);(18;31,3); (21,17,1);(19,53,1);(13,41,4);(37,12,5);(19,29,2);(67,17,1);(49,31,3)

Table 5. 3: Problem instance 1

P1: Wang algorithm solution (greedy approach)

The initial step in this heuristic procedure (paragraph 4.3.1) is to generate the first set (A) of cutting patterns using the Wang algorithm in a greedy way. The second set (B) of cutting patterns is obtained by employing the combinatoric process using the Wang algorithm (paragraph 3.7). The two sets (A and B) of cutting patterns are combined to formulate an integer linear programming model and CPLEX is used to solve the model. Table 5.4 reports the waste generated from *N-1* sheets where *N* is the total number of sheets required for the problem and the results obtained from CPLEX. The total waste is the area not used over *N* sheets.

The effectiveness of the different processes is based upon the waste generated from *N-1* sheets. Since the idea is to find a good solution from *N-1* sheets, the last sheet (*N*) will be as large as possible and may be useful in future optimization activities. Numbers in the column "Demand item removed" in all the tables represent the item type or types removed during the first process of the combinatoric process.

Solution procedure	N-1 Waste	Total Waste	Demand item removed	Set
Wang greedy	411	5243		A
Combinatoric Process				B
R1_1	478	5243	1	
R1_2	618	5243	3	
R1_3	684	5243	4	
R1_4	524	5243	10	
R1_5	431	5243	13	
R1_6	478	5243	16	
R1_7	684	5243	18	
R2_1	343	5243	16,18	
. . .	. . .	. . .	. . .	
R5_1	411	5243	3,8,10,13,18	
R5_2	478	5243	1,3,8,10,16	
R5_3	666	5243	3,4,13,16,18	
COMBINED CPLEX SOLUTION	343	5243		A ∪ B

Table 5. 4: Wang algorithm results (Problem instance 1)

The combinatoric process of the Wang algorithm (set B) produces some good cutting patterns which are better than the Wang algorithm (greedy solution) (set A). The combined CPLEX solution (set A∪B) for *N-1* sheets is better than the solutions obtained by both the Wang algorithm (greedy) and the combinatoric process of the Wang algorithm.

P1: CUTLOGIC 2D solution

The second step in this heuristic procedure (paragraph 4.3.1) is to generate a third set (C) of cutting patterns using the CUTLOGIC 2D software. The fourth set (D) of cutting patterns is obtained by employing the combinatoric process using the CUTLOGIC software. The two sets (C and D) of cutting patterns are combined to formulate an integer linear programming model and CPLEX is used to solve the model. Table 5.5 reports the waste generated from *N-1* sheets where *N* is the total number of sheets required to solve the problem and the results obtained from CPLEX.

Solution procedure	N-1 Waste	Total Waste	Demand item removed	Set
CUTLOGIC 2D Original	844	5243		C
Combinatoric Process				D
R1_1	688	5243	4	
R1_2	691	5243	6	
R1_3	769	5243	13	
R1_4	503	5243	19	
R2_1	530	5243	4,6	
R2_2	623	5243	7,12	
R2_3	579	5243	12,13	
R2_4	780	5243	19,20	
. . .	. . .	. . .	. . .	
R5_1	772	5243	4,6,7,12,13	
R5_2	614	5243	7,12,13,19,20	
R5_3	640	5243	4,6,12,19,20	
COMBINED CPLEX SOLUTION	450	5243		C ∪ D

Table 5. 5: CUTLOGIC 2D results (Problem instance 1)

The combinatoric process of the CUTLOGIC 2D software (set D) produces good cutting patterns which are better than the original solution of the CUTLOGIC 2D software (set C). The combined CPLEX solution (set C∪D) is better than the solutions obtained by both the original CUTLOGIC 2D software and the combinatoric process of the CUTLOGIC 2D.

P1: Combined CPLEX solution

The third step in this heuristic procedure (paragraph 4.3.1) is to combine all the sets (A, B, C and D) of cutting patterns generated in Tables 5.4 and 5.5. An integer linear programming model is formulated to solve the problem using CPLEX. Table 5.6 furnishes the results for the final solution obtained when all the cutting patterns are combined.

	N-1 Waste	Total Waste	Set
COMBINED CPLEX SOLUTION	**334**	**5243**	**A,B,C,D**

Table 5. 6: Combined CPLEX results (Problem instance 1)

The CPLEX solution over *N-1* sheets for the corresponding set (A, B, C, D) is better than the solutions obtained from the applied solution procedures separately.

2. Problem instance 2

Table 5.7 indicates the problem instance from Christofides and Whitlock (Christofides & Whitlock, 1977:30) used in Chapter 3.

Problem	Stock Sheet Length(L) and Width(W)	Demand items length (l), width (w) and upper bound (b) where b specifies the number of times a demand rectangle must be cut with each type being denoted by (l, w, b).
2	(40,70)	(21,22,1);(31,13,1);(9,35,3);(9,24,3);(30,7,2);(11,13,3);(10,14,1); (14,8,3);(12,8,3);(13,7,3);(31,43,4);(25,35,3);(22,32,3)

Table 5. 7: Problem instance 2

P2: Wang algorithm solution (greedy approach)

The initial step in this heuristic procedure (paragraph 4.3.1) is to generate the first set (A) of cutting patterns using the Wang algorithm in a greedy way. The second set (B) of cutting patterns is obtained by employing the combinatoric process using the Wang algorithm (paragraph 3.7). The two sets (A and B) of cutting patterns are combined to formulate an integer linear programming model and CPLEX is used to solve the model. Table 5.8 indicates the waste generated from *N-1* sheets where *N* is the total number of sheets required to solve that problem and the results obtained from CPLEX.

Solution procedure	N-1 Waste	Total Waste	Demand item removed	Set
Wang greedy	1525	2387		A
Combinatoric Process				B
R1_1	1166	2387	2	
R1_2	1449	2387	3	
R1_3	1522	2387	7	
R1_4	1525	2387	11	
R2_1	1449	2387	2,3	
R2_2	1859	2387	7,8	
R2_3	1624	2387	10,11	
R3_1	1449	2387	2,3,7	
. . .	. . .	. . .	. . .	
R5_1	924	2387	2,3,7,8,10	
R5_2	1760	2387	2,7,8,10,11	
R5_3	1166	2387	3,7,8,10,11	
COMBINED CPLEX SOLUTION	844	2387		A∪B

Table 5. 8: Wang algorithm results (Problem instance 2)

The combinatoric process of the Wang algorithm (set B) produces good cutting patterns which are better than the Wang algorithm (greedy solution) (set A). The combined CPLEX solution (A∪B) is better than the solutions obtained by both the Wang algorithm (greedy) and the combinatoric process of the Wang algorithm.

P2: CUTLOGIC 2D solution

The second step in this heuristic procedure (paragraph 4.3.1) is to generate a third set (C) of cutting patterns using the CUTLOGIC 2D software. The fourth set (D) of cutting patterns is obtained by employing the combinatoric process using the CUTLOGIC software. The two sets (C and D) of cutting patterns are combined to formulate an integer linear programming model, which is solved using CPLEX. Table 5.9 shows the waste generated from *N-1* sheets where *N* is the total number of sheets required to solve the problem and the results obtained from CPLEX.

Solution procedure	N-1 Waste	Total Waste	Demand item removed	Set
CUTLOGIC 2D Original	753	2387		C
Combinatoric Process				D
R1_1	745	2387	2	
R1_2	873	2387	3	
R1_3	920	2387	5	
R2_1	815	2387	2,3	
R2_2	896	2387	5,11	
R2_3	893	2387	2,5	
R2_4	920	2387	2,11	
R2_5	836	2387	3,5	
. . .	. . .	. . .	. . .	
R3_2	1016	2387	3,5,11	
R3_3	1225	2387	2,5,11	
R3_4	920	2387	2,3,11	
COMBINED CPLEX SOLUTION	745	2387		C∪D

Table 5. 9: CUTLOGIC 2D results (Problem instance 2)

The combinatoric process of the CUTLOGIC 2D software (set D) produces some good cutting patterns which are better than the original solution of the CUTLOGIC 2D software (set C). The combined CPLEX solution (C∪D) is better than the solutions obtained by both the original CUTLOGIC 2D software and the combinatoric process of the CUTLOGIC 2D software.

P2: Combined CPLEX solution

The third step in this heuristic procedure (paragraph 4.3.1) is to combine all the sets (A, B, C and D) of cutting patterns generated in Table 5.8 and Table 5.9. An integer linear programming model is formulated to solve the problem. This model is solved using CPLEX. Table 5.10 contains the results for the final solution obtained when all the cutting patterns are combined.

	N-1 Waste	Total Waste	Set
COMBINED CPLEX SOLUTION	**745**	**2387**	**A,B,C,D**

Table 5. 10: Combined CPLEX results (Problem instance 2)

The CPLEX solution over *N-1* sheets for the corresponding set (A, B, C, D) is the same as the solution obtained from CUTLOGIC 2D and is better than the solution obtained from the Wang algorithm (greedy solution).

3. Problem instance 3

Table 5.11 indicates the problem instance from Tschöke and Holthöfer (Tschöke & Holthöfer, 1995:285) used in Chapter 3.

Problem	Stock Sheet Length(L) and Width(W)	Demand items length (l), width (w) and upper bound (b) where b specifies the number of times a demand rectangle must be cut with each type being denoted by (l, w, b).
3	(99,99)	(14,16,1);(14,31,2);(15,17,3);(17,19,3);(18,19,2);(20,34,2);(20,39,3); (21,29,2);(25,25,2);(27,27,3);(29,34,2);(30,25,2);(30,27,3);(33,30,2); (35,30,2);(37,27,4);(38,49,2);(40,25,2);(43,28,4);(44,34,4)

Table 5. 11: Problem instance 3

P3: Wang algorithm solution (greedy approach)

The initial step in this heuristic procedure (paragraph 4.3.1) is to generate the first set (A) of cutting patterns using the Wang algorithm in a greedy way. The second set (B) of cutting patterns is obtained by employing the combinatoric process using the Wang algorithm (paragraph 3.7). The two sets (A and B) of cutting patterns are combined to formulate an integer linear programming model which is solved using CPLEX. Table 5.12 indicates the waste generated from *N-1* sheets where *N* is the total number of sheets required to solve the problem and the results obtained from CPLEX.

Solution procedure	N-1 Waste	Total Waste	Demand item removed	Set
Wang greedy	902	6638		A
Combinatoric Process				
R1_1	815	6638	1	
R1_2	683	6638	2	
R1_3	874	6638	9	
R2_1	750	6638	1,2	
R2_2	1103	6638	3,4	
R2_3	784	6638	9,10	
R3_1	870	6638	1,2,3	B
R3_2	754	6638	4,9,12	
. . .	. . .	. . .	. . .	
R5_1	768	6638	1,2,3,4,9	
R5_2	719	6638	4,9,12,17,20	
R5_3	715	6638	1,3,9,17,20	
COMBINED CPLEX SOLUTION	498	6638		A∪B

Table 5. 12: Wang algorithm results (Problem instance 3)

The combinatoric process of the Wang algorithm (set B) produces good cutting patterns which are better than the Wang algorithm (greedy solution) (set A). The combined CPLEX solution (A∪B) is better than the solutions obtained by both the Wang algorithm (greedy) and the combinatoric process of the Wang algorithm.

P3: CUTLOGIC 2D solution

The second step in this heuristic procedure (paragraph 4.3.1) is to generate a third set (C) of cutting patterns using the CUTLOGIC 2D software. The fourth set (D) of cutting patterns is obtained by employing the combinatoric process using the CUTLOGIC software. The two sets (C and D) of cutting patterns are combined to formulate an integer linear programming model which is solved using CPLEX. Table 5.13 indicates the waste generated from *N-1* sheets where *N* is the total number of sheets required to solve the problem and the results obtained from CPLEX.

Solution procedure	N-1 Waste	Total Waste	Demand item removed	Set
CUTLOGIC 2D Original	669	6638		C
Combinatoric Process				D
R1_1	667	6638	1	
R1_2	608	6638	5	
R1_3	631	6638	8	
R2_1	567	6638	1,2	
R2_2	646	6638	3,4	
R2_3	603	6638	5,8	
R3_1	559	6638	1,2,3	
R3_2	750	6638	4,5,6	
. . .	. . .	. . .	. . .	
R5_1	646	6638	1,2,3,4,5	
R5_2	654	6638	6,8,10,16,19	
R5_3	523	6638	1,5,8,19,20	
COMBINED CPLEX SOLUTION	523	6638		C∪D

Table 5. 13: CUTLOGIC 2D results (Problem instance 3)

The combinatoric process of the CUTLOGIC 2D software (set D) produces good cutting patterns which are better than the original solution of the CUTLOGIC 2D software (set C). The combined CPLEX solution (C∪D) is better than the solutions obtained by both the CUTLOGIC 2D software and the combinatoric process of the CUTLOGIC 2D software.

P3: Combined CPLEX solution

The third step in this heuristic procedure (paragraph 4.3.1) is to combine all the sets (A, B, C and D) of cutting patterns generated in Table 5.12 and Table 5.13. An integer linear programming model is formulated to solve the problem which is solved using CPLEX. Table 5.14 furnishes the results for the final solution obtained when all the cutting patterns are combined.

	N-1 Waste	Total Waste	Set
COMBINED CPLEX SOLUTION	**471**	**6638**	**A,B,C,D**

Table 5. 14: Combined CPLEX results (Problem instance 3)

The CPLEX solution over *N-1* sheets for the corresponding set (A, B, C, D) is better than the solutions obtained from the applied solution procedures separately.

4. Problem instance 4

Table 5.15 indicates the problem instance from Hifi (Hifi, 1997:727) used in Chapter 3.

Problem	Stock Sheet Length(L) and Width(W)	Demand items length (l), width (w) and upper bound (b) where b specifies the number of times a demand rectangle must be cut with each type being denoted by (l, w, b).
4	(70,80)	(29,41,1);(31,43,2);(32,39,1);(27,38,4);(28,37,3);(29,36,2);(35,35,3); (26,34,4);(31,24,2);(22,33,2);(31,22,3);(29,21,4);(19,27,2);(18,24,2); (17,25,2);(17,24,2);(25,17,1);(21,16,2);(19,15,3);(15,14,1)

Table 5. 15: Problem instance 4

P4: Wang algorithm solution (greedy approach)

The initial step in this heuristic procedure (paragraph 4.3.1) is to generate the first set (A) of cutting patterns using the Wang algorithm in a greedy way. The second set (B) of cutting patterns is obtained by employing the combinatoric process using the Wang algorithm (paragraph 3.7). The two sets (A and B) of cutting patterns are combined to form an integer linear programming model which is solved using CPLEX. Table 5.16 indicates the waste generated from *N-1* sheets where *N* is the total number of sheets required to solve the problem and the results obtained from CPLEX.

Solution procedure	N-1 Waste	Total Waste	Demand item removed	Set
Wang greedy	1289	4446		A
Combinatoric Process				
R1_1	1774	4446	5	
R1_2	1097	4446	10	
R1_3	1564	4446	20	
R2_1	1306	4446	5,19	
R2_2	1260	4446	10,18	
R2_3	1456	4446	18,19	
R3_1	1564	4446	5,14,19	B
R3_2	1289	4446	10,14,20	
. . .	. . .	. . .	. . .	
R5_1	1411	4446	5,10,14,19,20	
R5_2	1523	4446	10,14,18,19,20	
R5_3	1235	4446	5,10,14,18,19	
COMBINED CPLEX SOLUTION	1097	4446		A∪B

Table 5. 16: Wang algorithm results (Problem instance 4)

The combinatoric process of the Wang algorithm (set B) produces some good cutting patterns which are better than the Wang algorithm (greedy solution) (set A). The combined CPLEX solution (A∪B) is better than the solutions obtained by both the Wang algorithm (greedy) and the combinatoric process of the Wang algorithm.

P4: CUTLOGIC 2D solution

The second step in this heuristic procedure (paragraph 4.3.1) is to generate a third set (C) of cutting patterns using the CUTLOGIC 2D software. The fourth set (D) of cutting patterns is obtained by employing the combinatoric process using the CUTLOGIC software. The two sets (C and D) of cutting patterns are combined to formulate an integer linear programming model which is solved using CPLEX. Table 5.17 records the waste generated from *N-1* sheets where *N* is the total number of sheets required to solve the problem and the results obtained from CPLEX.

.

Solution procedure	N-1 Waste	Total Waste	Demand item removed	Set
CUTLOGIC 2D Original	673	4446		C
Combinatoric Process				D
R1_1	746	4446	6	
R1_2	746	4446	8	
R1_3	747	4446	15	
R1_4	685	4446	16	
R1_5	766	4446	17	
R2_1	788	4446	6,8	
R2_2	872	4446	15,16	
R2_3	769	4446	16,17	
. . .	. . .	. . .	. . .	
R4_2	756	4446	8,15,16,17	
R4_3	680	4446	6,15,16,17	
R4_4	826	4446	6,8,16,17	
COMBINED CPLEX SOLUTION	564	4446		C∪D

Table 5. 17: CUTLOGIC 2D results (Problem instance 4)

The combinatoric process of the CUTLOGIC 2D software (set D) produces other cutting patterns to obtain set (C). The combined CPLEX solution (C∪D) is better than the solutions obtained by both the original CUTLOGIC 2D software and the combinatoric process of the CUTLOGIC 2D.

P4: Combined CPLEX solution

The third step in this heuristic procedure (paragraph 4.3.1) is to combine the sets (A, B, C and D) of cutting patterns generated in Table 5.16 and Table 5.17. An integer linear programming model is formulated to solve the problem using CPLEX. Table 5.18 shows the results for the final solution obtained when all the cutting patterns are combined.

	N-1 Waste	Total Waste	Set
COMBINED CPLEX SOLUTION	**531**	**4446**	**A,B,C,D**

Table 5. 18: Combined CPLEX results (Problem instance 4)

The CPLEX solution over *N-1* sheets for the corresponding set (A, B, C, D) is better than the solutions obtained from the applied solution procedures separately.

5.7.2 Real world application problem instances

The first heuristic procedure (paragraph 4.3.1) functioned well for small problems since it was able to produce better results. These small problems however do not indicate whether the heuristic procedure scales well when used to solve large industry-sized problem instances. This section consequently provides the results obtained from solving two larger problem instances using the second heuristic procedure (paragraph 4.3.2). The problem instances were obtained from PG Glass Pty.Ltd which deals with solving two-dimensional cutting stock problems and selling glass sheets.

The application of the heuristic procedure (paragraph 4.3.1) by implementing the Wang algorithm in a greedy way and also the CUTLOGIC 2D software to the real world problem instances results only in a few new cutting patterns being generated. Therefore the heuristic procedure (paragraph 4.3.2) was employed in order to try and generate additional cutting patterns that can be used in the integer linear programming model.

1. PG Problem instance 1

Table 5.19 indicates the problem instance PG 1 from PG Glass Company.

Problem	**Stock Sheet Length(L) and Width(W)**	**Demand items length (l), width (w) and upper bound (b) where b specifies the number of times a demand rectangle must be cut with each type being denoted by (l, w, b).**
PG1	(1500,2125)	(290,1440,3);(585,955,1);(925,560,17);(950,290,12);(956,1195,2);(1440,1195,5);(1490,1140,1)

Table 5. 19: Problem instance PG 1

PG1: Wang algorithm solution (greedy approach)

The initial step in this heuristic procedure (paragraph 4.3.2) is to generate the first set (A) of cutting patterns using the Wang algorithm in a greedy way. Table 5.20 reports the number of sheets (*N*) utilized when solving the problem and the amount of waste generated by each sheet. The *N-1* waste is calculated by adding the waste generated in each of the first *N-1* sheets. In this case *N* represents 9 sheets while *N-1* represents 8 sheets.

Sheet Number	Waste per Sheet	Set
1	70980	A
2	213900	
3	215580	
4	248325	
5	256000	
6	322000	
7	430700	
8	673200	
9	254100	
N-1 Waste	2430685	

Table 5. 20: Wang algorithm results (Problem instance PG1)

PG1: Generated Wang solution

The second step in this heuristic procedure (paragraph 4.3.2) is to generate further cutting patterns using a specified β to obtain the second set (B) of cutting patterns. Table 5.21 records the amount of waste generated in each cutting pattern involving β. The process concerns generating cutting patterns for each β. A smaller β contains less cutting patterns. A larger β contains patterns that appear with the smaller β and possible new ones. The patterns were compared and duplicates were removed.

β	Waste per Sheet	Set
0.03	**70980**	**B**
0.04	**103725**	
0.04	**111400**	
0.05	**136725**	
0.05	**144400**	
0.06	**177400**	
0.07	**213900**	
0.07	**215580**	
0.08	**236100**	
. . .	. . .	
0.21	**664900**	
0.22	**691880**	
0.22	**673200**	
0.22	**697900**	
0.22	**700325**	

Table 5. 21: Generated results (Problem instance PG1)

PG1: CUTLOGIC 2D solution

The third step of this heuristic procedure (paragraph 4.3.2) is to solve the problem using the CUTLOGIC 2D software to obtain the set (C) of cutting patterns. Table 5.22 reports the number of sheets (*N*) utilized when solving the problem and the amount of waste generated by each sheet. The *N-1* waste is calculated by adding the waste generated in each of the first *N-1* sheets. In this case *N* represents 10 sheets while *N-1* represents 9 sheets.

Sheet Number	Waste per Sheet	Set
1	**103912**	**C**
2	**214200**	
3	**215793**	
4	**215793**	
5	**256275**	
6	**397800**	
7	**430950**	
8	**430950**	
9	**430950**	
10	**2669500**	
N-1 Waste	**2696623**	

Table 5. 22: CUTLOGIC 2D solution (Problem instance PG1)

PG1: Combined CPLEX solution

The fourth step in this heuristic procedure (paragraph 4.3.2) is to combine all the sets (A, B, C) of cutting patterns generated in Table 5.20, 5.21 and 5.22. An integer linear programming model is formulated to solve the problem using CPLEX. Table 5.23 indicates the result for the final solution obtained when all the cutting patterns are combined. The *N-1* waste is calculated by adding the waste generated in each of the first *N-1* sheets. In this case *N* represents 9 sheets while *N-1* represents 8 sheets.

	N-1 Waste	Set
COMBINED CPLEX Solution	**1834985**	**A,B,C**

Table 5. 23: Combined CPLEX results (Problem instance PG1)

The CPLEX solution over *N-1* sheets is better than the solution obtained by both the CUTLOGIC 2D software and the Wang algorithm.

2. PG Problem Instance 2

Table 5.24 depicts the problem instance PG 2 from PG Glass Company.

Problem	Stock Sheet Length(L) and Width(W)	Demand items length (l), width (w) and upper bound (b) where b specifies the number of times a demand rectangle must be cut with each type being denoted by (l, w, b).
PG2	(1000,1500)	(290,129,20);(585,355,4);(925,650,17);(950,290,12);(555,395,2); (650,796,5);(200,324,10)

Table 5. 24: Problem instance PG 2

PG2: Wang algorithm solution (greedy approach)

The first step in this heuristic procedure (paragraph 4.3.2) is to generate the first set (A) of cutting patterns using the Wang algorithm in a greedy way. Table 5.25 indicates the number of sheets (*N*) utilized when solving the problem and the amount of waste generated by each sheet. The *N-1* waste is calculated by adding all the waste generated in each of the first *N-1* sheets. In this case *N* represents 14 sheets while *N-1* represents 13 sheets.

Sheet Number	Waste pre Sheet	Set
1	8610	A
2	23900	
3	59500	
4	59500	
5	86890	
6	122500	
7	244775	
8	244775	
9	464000	
10	464000	
11	464000	
12	464000	
13	464000	
14	465200	
N-1 Waste	2430685	

Table 5. 25: Wang algorithm results (Problem instance PG2)

PG2: Generated Wang solution

The second step in this heuristic procedure (paragraph 4.3.2) is to generate more cutting patterns using a specified β to obtain the second set (B) of cutting patterns. Table 5.26 records the amount of waste generated in each cutting pattern involving β.

β	Waste per Sheet	Set
0.01	11600	B
0.01	12450	
0.01	8610	
0.02	25980	
0.02	29235	
0.02	25725	
0.02	23900	
0.02	27865	
0.02	28880	
. . .	. . .	
0.04	53565	
0.04	6215	
0.04	58855	
0.04	49955	
0.04	51585	

Table 5. 26: Generated results (Problem instance PG2)

PG2: CUTLOGIC 2D solution

The third step of this heuristic procedure (paragraph 4.3.2) is to solve the problem using the CUTLOGIC 2D software to obtain the set (C) of cutting patterns. Table 5.27 reports the number of sheets (*N*) utilized when solving the problem and the amount of waste generated by each sheet. The *N-1* waste is calculated by adding the waste generated in each of the first *N-1* sheets. In this case *N* represents 13 sheets while *N-1* represents 12 sheets.

Sheet Number	Waste per Sheet	Set
1	59550	C
2	59550	
3	59550	
4	59550	
5	59550	
6	122550	
7	132600	
8	132600	
9	144150	
10	144150	
11	144150	
12	144150	
13	874200	
N-1 Waste	1262100	

Table 5. 27: CUTLOGIC 2D results (Problem instance PG2)

PG2: Combined CPLEX solution

The fourth step in this heuristic procedure (paragraph 4.3.2) is to combine the sets (A, B, and C) of cutting patterns generated in Tables 5.25, 5.26 and 5.27. An integer linear programming model is formulated to solve the problem using CPLEX. Table 5.28 indicates the result for the final solution obtained when all the cutting patterns are combined. The *N-1* waste is calculated by adding the waste generated in each of the first *N-1* sheets. In this case *N* represents 13 sheets while *N-1* represents 12 sheets.

	N-1 Waste	Set
COMBINED CPLEX Solution	1261800	A,B,C

Table 5. 28: Combined CPLEX results ((Problem instance PG2)

The CPLEX solution over *N-1* sheets is better than the solution obtained by both the CUTLOGIC 2D software and the Wang algorithm.

The heuristic procedure for solving the real world application problem instances worked well since better results were obtained.

5.5 Summary

The large integer linear programming model was formulated using the sets of generated cutting patterns and CPLEX was used to solve the model. The heuristic procedure was employed to solve both small problem instances and real world application problem instances. In both cases the heuristic solution was able to obtain better solutions than those currently obtained.

Chapter 6: Empirical results

6.1 Introduction

This chapter discusses the results obtained from the empirical experimentation carried out in Chapter 5 by applying the heuristic procedure to both small problems and real world application problems. The small problem instances were gleaned from previous research papers while the instances regarding real world application problems were obtained from PG Glass Pty.Lty. The results arising from small problem instances are discussed in paragraph 6.2 whereas those steaming from real world applications are discussed in paragraph 6.3.

6.2 Small problem instances

The first heuristic procedure (paragraph 4.3.1) was used to solve the small problem instances and the results obtained were satisfactory.

6.2.1 Problem instance 1

Table 6.1 displays the results for the experiment conducted on the problem instance drawn from Cung (Cung et al., 2000:185). The table lists the software that was employed, the number of sheets that were utilized and the amount of waste generated.

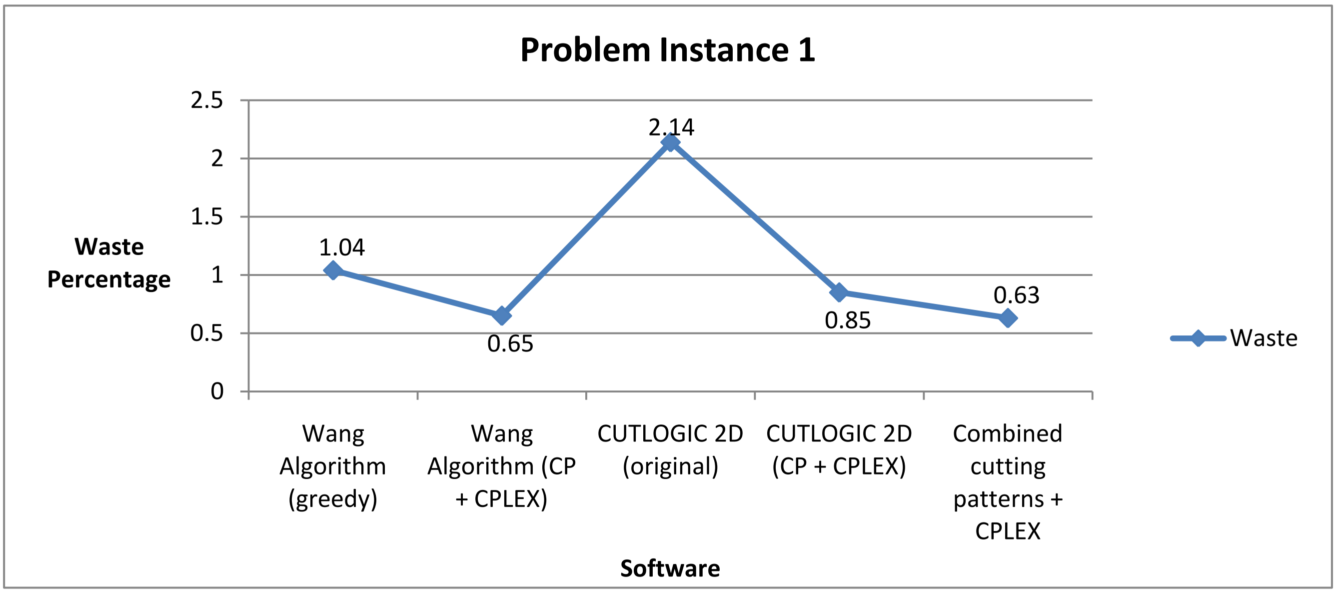
Software Name		N-1 Sheets	N-1 Waste	N-1 Waste Percentage	Set
Wang algorithm	Greedy algorithm	3	411	1.04	A
	Combinatoric Process + CPLEX	3	343	0.65	B
CUTLOGIC 2D	Original	3	846	2.14	C
	Combinatoric Process + CPLEX	3	450	0.85	D
Combined cutting patterns + CPLEX		3	334	0.63	A,B,C,D

Table 6. 1: Results for the first problem

All the software utilized the same number of sheets while the combined cutting patterns solved with CPLEX generated the least amount of waste over *N-1* sheets. This was followed by the Wang algorithm (greedy) and the original CUTLOGIC 2D system came last.

Graph 6.1 displays a graphic representation of the waste percentage. The numbers on the Y-axis represent the percentage of waste generated.

NB: CP: Combinatoric Process



Graph 6. 1: Graphic representation of the waste percentage

Looking at the waste percentage, it can be noted that the combined cutting patterns solved with CPLEX produce a relatively low amount of waste. This represents a saving of more than twice the waste generated by the original CUTLOGIC 2D system.

6.2.2 Problem instance 2

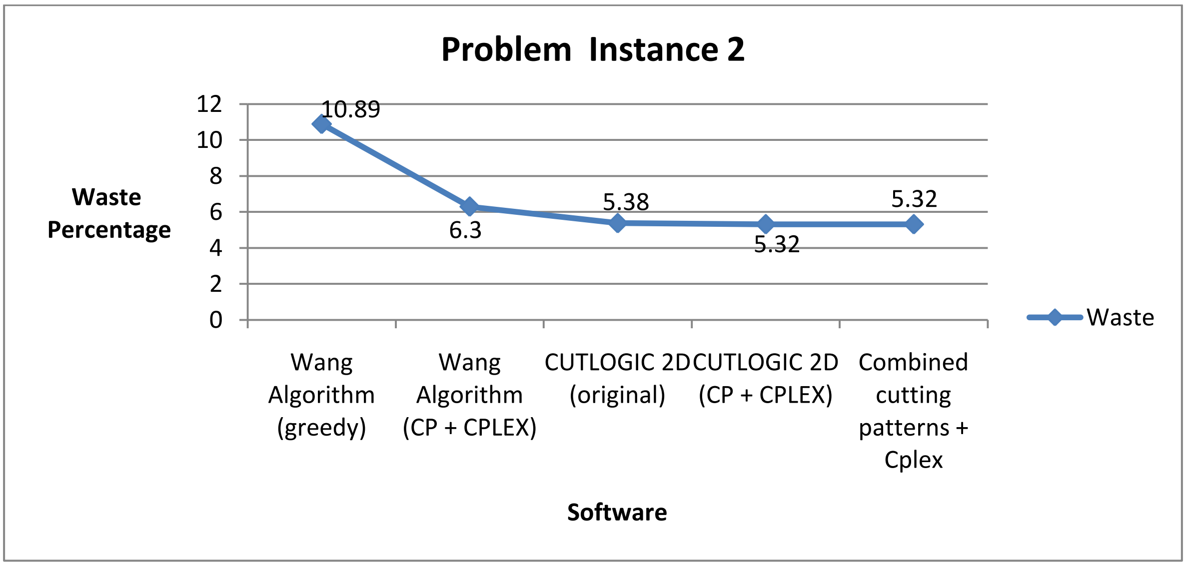
Table 6.2 reports the results for the experiment carried out with regard to the problem instance drawn from Christofides & Whitlock (Christofides & Whitlock, 1977:30). The table indicates the software that was employed, the number of sheets that were utilized and the amount of waste generated.

Software Name		N-1 Sheets	N-1 Waste	N-1 Waste Percentage	Set
Wang algorithm	Greedy algorithm	5	1525	10.89	A
	Combinatoric Process + CPLEX	5	844	6.30	B
CUTLOGIC 2D	Original	5	753	5.38	C
	Combinatoric Process + CPLEX	5	745	5.32	D
Combined cutting patterns + CPLEX		5	745	5.32	A,B,C,D

Table 6. 2: Results for the second problem

All the software utilized the same number of sheets while the combined cutting patterns solved with CPLEX and the combinatoric process of the CUTLOGIC 2D system generated the least amount of waste over *N-1* sheets. This was followed by the Wang algorithm (greedy).

Graph 6.2 displays a graphic representation of the waste percentage. The numbers on the Y-axis represent the percentage of waste generated.



Graph 6. 2: Graphic representation of the waste percentage

Considering the waste percentage in this problem it can be noted that the combined cutting patterns solved with CPLEX and the combinatoric process of the CUTLOGIC 2D software produce a relatively low amount of waste. This represents a saving of more than twice the waste generated by the Wang algorithm (greedy).

6.2.3 Problem instance 3

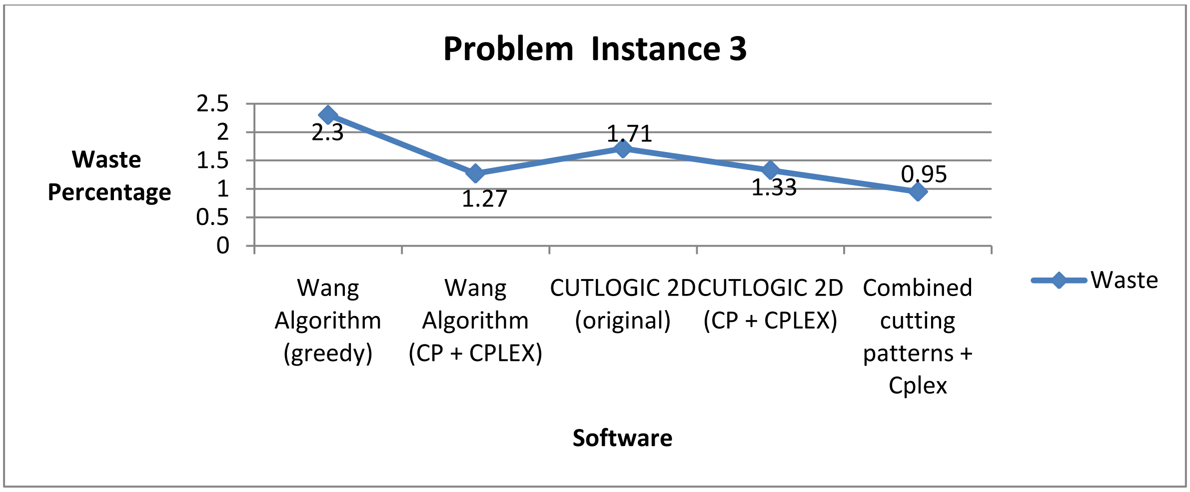
Table 6.3 displays the results for the experiment carried out with regards to problem instance drawn from Tschöke & Holthöfer (Tschöke & Holthöfer, 1995:285). The table indicates the software that was employed, the number of sheets that were utilized and the amount of waste generated.

Software Name		N-1 Sheets	N-1 Waste	N-1 Waste Percentage	Set
Wang algorithm	Greedy algorithm	4	902	2.30	A
	Combinatoric Process + CPLEX	4	498	1.27	B
CUTLOGIC 2D	Original	4	669	1.71	C
	Combinatoric Process + CPLEX	4	523	1.33	D
Combined cutting patterns + CPLEX		4	471	0.95	A,B,C,D

Table 6. 3: Results for the third problem

All the software utilized the same number of sheets while the combined cutting patterns solved with CPLEX generated the least amount of waste over *N-1* sheets. This was followed by the combinatoric process of the Wang algorithm while the Wang algorithm (greedy) came last.

Graph 6.3 displays a graphic representation of the waste percentage. The numbers on the Y-axis represent the percentage of waste generated.



Graph 6. 3: Graphic representation of the waste percentage

Considering the waste percentage for this problem it can be noted that the combined cutting patterns solved with CPLEX produce a relatively low amount of waste. This represents a saving of more than twice the waste generated by the Wang algorithm (greedy).

6.2.4 Problem instance 4

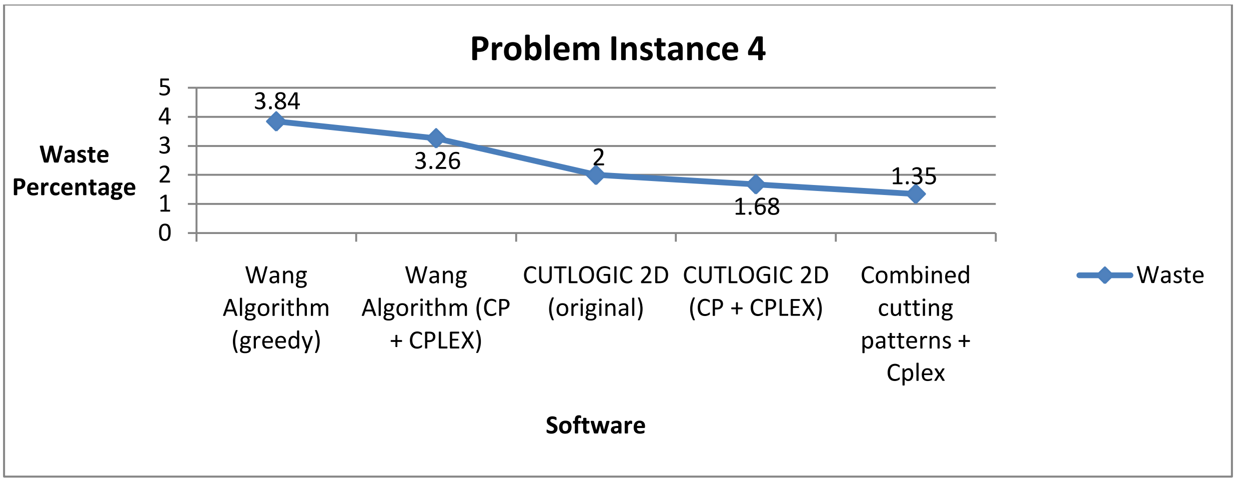
Table 6.4 displays the results for the experiment carried out with regards to the problem instance drawn from Hifi (Hifi, 1997:727). The table displays the software that was employed, the number of sheets that were utilized and the amount of waste generated.

Software Name		N-1 Sheets	N-1 Waste	N-1 Waste Percentage	Set
Wang algorithm	Greedy algorithm	6	1289	3.84	A
	Combinatoric Process + CPLEX	6	1097	3.26	B
CUTLOGIC 2D	Original	6	673	2.00	C
	Combinatoric Process + CPLEX	6	564	1.68	D
Combined cutting patterns + CPLEX		6	531	1.35	A,B,C,D

Table 6. 4: Results for the fourth problem

All the software utilized the same number of sheets whereas the combined cutting patterns solved with CPLEX generated the least amount of waste over *N-1* sheets. This was followed by the combinatoric process of the CUTLOGIC 2D system while then the original CUTLOGIC came last.

Graph 6.4 displays a graphic representation of the waste percentage. The numbers on the Y-axis represent the percentage of the waste generated.



Graph 6. 4: Graphic representation of the waste percentage

Considering the waste percentage for this problem it can be noted that the combined cutting patterns solved with CPLEX produce a relatively low amount of waste. This represents a saving of more than twice the waste generated by the Wang algorithm (greedy).

The heuristic procedure worked well in small problem instances since it produced better results than those currently produced by both the CUTLOGIC 2D system and the Wang algorithm.

6.3 Real world application problem instances

The second heuristic procedure (paragraph 4.3.2) was used to solve the real world application problem instances and the results obtained were satisfactory.

6.3.1 PG Problem instance 1

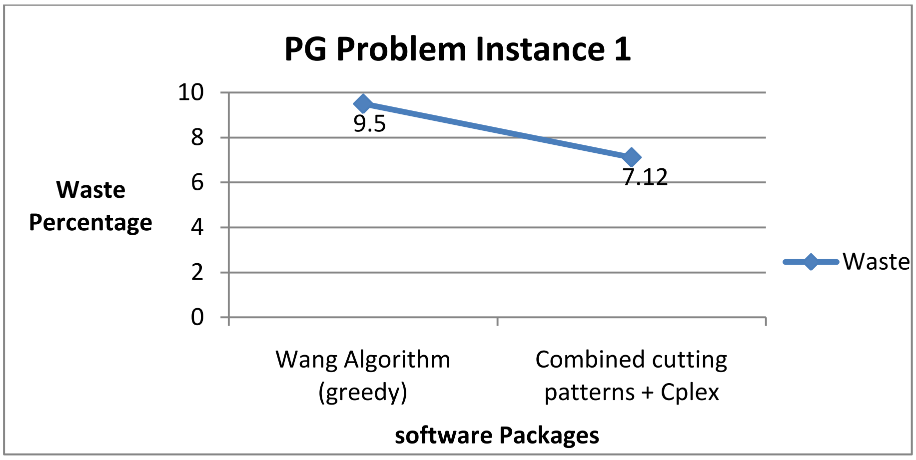
Table 6.5 displays the results for the experiment carried out with regards to the data from the first real world application problem instance. The table displays the software that was employed, the number of sheets that were utilized and the amount of waste generated.

Software Name	N-1 Sheets	N-1 Waste	N-1 Waste Percentage	Set
Wang algorithm	8	2430685	9.50	A
CUTLOGIC 2D	9	2696623	9.40	B
Combined cutting patterns + CPLEX	8	1834985	7.12	A,B

Table 6. 5: Results for PG1

The Wang algorithm and the combined cutting patterns solved with CPLEX utilized the same number of sheets whereas the CUTLOGIC 2D system utilized more sheets. The combined cutting patterns solved with CPLEX generate the least amount of waste over *N-1* sheets. This was followed by the Wang algorithm. The CUTLOGIC 2D system performed poorly and required an extra sheet.

Graph 6.5 displays a graphic representation of the waste percentage. The numbers on the Y-axis represent the percentage of waste generated.



Graph 6. 5: Graphic representation of the waste percentage

Considering the waste percentage for this problem it can be noted that the combined cutting patterns solved with CPLEX produces a relatively low amount of waste.

6.3.2 PG Problem instance 2

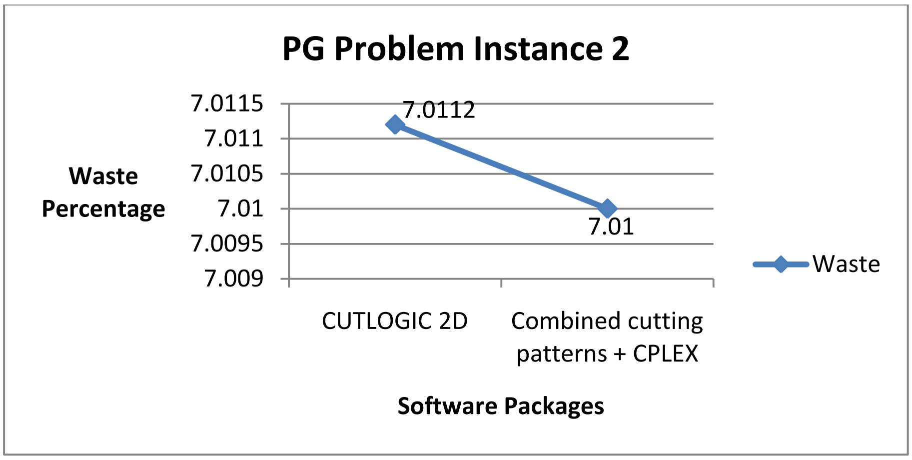
Table 6.6 displays the results for the experiment carried out on the data from the second real world application problem instance. The table shows the software used, the number of sheets utilized and the amount of waste generated.

Software Name	N-1 Sheets	N-1 Waste	Waste Percentage	Set
Wang algorithm	13	2430685	2.40	A
CUTLOGIC 2D	12	1262100	7.0112	B
Combined cutting patterns + CPLEX	12	1261800	7.01	A,B

Table 6. 6: Results for PG2

The CUTLOGIC 2D system and the combined cutting patterns solved with CPLEX utilized the same number of sheets whereas the Wang algorithm utilized more sheets. The combined cutting patterns solved with CPLEX generated the least amount of waste over *N-1* sheets. This was followed by the CUTLOGIC 2D system. The Wang algorithm performed poorly and required an extra sheet.

Graph 6.6 displays a graphic representation of the waste percentage. The numbers on the Y-axis represent the percentage of waste generated.



Graph 6. 6: Graphic representation of the waste percentage

Considering the waste percentage for this problem it can be noted that the combined cutting patterns solved with CPLEX produces a relatively low amount of waste.

Furthermore it is noted that (in general) the heuristic procedure produces a relatively low amount of waste. Although as is common with heuristic procedures, it cannot be guaranteed how far from optimal the results obtained are, since the optimal result is not known. It is quite clear that the heuristic procedure for the 2DSSSCSP produces better cutting patterns which are better than what is currently obtained.

6.4 Summary

This chapter discusses the results obtained from the empirical experimentation carried out in Chapter 5 by applying the heuristic procedure to both small and real world application problems. Graphic representations were used to illustrate the amount of waste percentage generated by the different methods.

Chapter 7: Conclusions and further research

7.1 Introduction

This chapter discusses the results obtained in terms of the aims and objectives of the study. Some challenges are considered that could help to improve the development of algorithms for solving the 2DSSSCSP in the future. The research problem in this study has been to investigate some heuristic procedures for solving the 2DSSSCSP. In the light of the importance of this type of problem and the fact that there is relatively few research articles on this specific problem, any significant improvement should be desirable.

The main aims and objectives of the study have been to:

- Study the 2DSSSCSP and the associated trim-loss problem by reading articles and books in the field of cutting stock problems and concentrating on algorithms and heuristics for the 2DSSSCSP (see paragraph 7.2);
- Study the Wang algorithm and employ an implementation thereof to solve the trim-loss problem and also solve instances of the 2DSSSCSP (see paragraph 7.3);
- Investigate and assess some commercial software packages available to solve the 2DSSSCSP (see paragraph 7.4);
- Develop heuristic procedures based on integer linear programming models that can solve the 2DSSSCSP (see paragraph 7.5);
- Report on and interpret the findings from the selected software when used to conduct empirical experimentation in order to solve certain problems drawn from the literature and industry (see paragraph 7.6).

The extent to which these aims and objectives of the study have been realized is discussed below.

7.2 A literature study on the 2DSSSCSP and the trim-loss problem

The second chapter of this dissertation gave an overview of the existing literature of the CSP and the 2DCSP. Descriptions of related fields of study within the CSP environment as well as previous research in this field were discussed. Since the 2DSSSCSP is a special type of the 2DCSP, much reading was carried out in order to understand the 2DCSP. Since the 2DSSSCSP has been studied less than the other CSPs, available research articles concerning the 2DSSSCSP were studied. Studying this material provided a better understanding of the 2DSSSCSP and the importance of its applicability in the industry.

7.3 The Wang algorithm and an implementation thereof

The third chapter of the study gave a discussion of the Wang algorithm that is used when solving the 2DSSSCSP. This algorithm was employed in two ways in this study. Firstly, since it is basically an algorithm to solve the trim loss problem (thus one large object at a time); it was used to solve the 2DSSSCSP by means of a greedy approach. Secondly it was adapted to generate cutting patterns to be used when formulating an integer linear programming model to solve the 2DSSSCSP, which was discussed in Chapter 5.

The trim loss problem was investigated in Chapter 3 since it forms part of the Wang algorithm. Empirical experimentation was carried out by means of the Wang algorithm using data from previous research papers.

7.4 Assessment of some commercial software packages to solve the 2DSSSCSP

The third chapter of the study also gave an overview of some of the two-dimensional cutting stock software packages available in the market. These packages are used to solve the 2DSSSCSP. The overview helped in highlighting the differences among the industry-oriented software packages presently available. Empirical experiments were carried out to test some of these packages using data from previous research papers. Through such experimentation, it was found that the CUTLOGIC 2D software is one of the best packages that are currently being used to solve the 2DSSSCSP. Therefore the CUTLOGIC 2D software was used further in the research study.

7.5 Integer linear programming models to solve the 2DSSSCSP

The fourth chapter gave an explanation of heuristic procedures for solving the 2DSSSCSP. This was achieved by developing two heuristic procedures that generate additional cutting patterns. The first was employed according to the following strategy.

- An initial generation of cutting patterns using the CUTLOGIC 2D software and the Wang algorithm in a greedy way. This was undertaken to obtain the first set of cutting patterns.
- Further generation of cutting patterns by employing a combinatoric process which ensures different starting points in order to generate additional cutting patterns. This method generated a second set of cutting patterns.
- The generated sets of cutting patterns were combined to form one set of cutting patterns to be used in an integer linear programming model to be solved by means of CPLEX.

The second heuristic procedure was used to also generate additional cutting patterns by using a range of specified β-values applied to the Wang algorithm. These cutting patterns were furthermore employed to formulate an integer linear programming model to be solved

7.6 Reporting on results

Different sets of cutting patterns were generated to formulate an integer linear programming model to solve the 2DSSSCSP. The layout of the heuristic procedure was given in Chapter 4 while the mathematical formulation of the problem was performed in Chapter 5.

To illustrate and test the heuristic procedure proposed, small problem instances collected from previous research papers as well as real world application problem instances from a large corporation called PG Glass Pty.Ltd were used. The integer linear programming model was solved using an optimization solver called IBM ILOG CPLEX which is a commercial optimization tool. The results were presented and discussed in Chapter 6.

The main result of the heuristic procedure was that it produced better results than what is currently being offered in the market. It is known that solving the 2DSSSCSP to optimality can be economically momentous to such an extent that even small improvements in the cutting layout can result in large savings of material. The proposed heuristic algorithm could therefore be of great value in the market.

7.7 Further research

It has been established through the empirical experimentation carried out in Chapter 5 that the heuristic procedure developed for solving the 2DSSSCSP produces better results. The major challenge in applying this heuristic procedure is the time it takes to generate cutting patterns. Therefore further research may result in faster algorithms.

It could be worthwhile to study further the possibility of reducing the number of infeasible cutting patterns generated when using β. In the present Wang algorithm, all possible cutting patterns are allowed to be generated, of which many are duplicates. The process of removing duplicates is time consuming. More efficient ways of data handling may improve the solution.

7.8 Conclusions

The work presented in this study described a new heuristic procedure that could be used by research organizations and large companies to solve the 2DSSSCSP. Small problem instances and real world application problem instances were made use of to test this new heuristic procedure, and the results obtained were satisfactory. Possible improvements can still be achieved since the method used in this study is a non-exact method and thus, the results obtained provide no guarantee regarding optimality.

Nonetheless, one of the benefits of the approaches used in this dissertation is that certain existing systems could be used to achieve a significant improvement in wastage reduction.

References

ALBANO, A. & ORSINI, R. 1978. A heuristic solution of the rectangular cutting stock problem. *The computer journal*, 23(4):338-343.

ALVES, C. & VALERIO DE CARVALHO, J.M. 2007. Solving the assortment and trim loss problem with branch-and-price-and-cut. ORP3 meeting, Guimares. September 12-15.5p. http://pessoais.dps.uminho.pt/vc/AlvesCarvalho_ORP3.pdf Date of access: 10 Oct. 2009.

BAKER, B.M. 1999. A spreadsheet modelling approach to the assortment problem. *European journal of operational research*, 114(1):83-92.

BEASLEY, J.E. 1985b. Algorithms for unconstrained two-dimensional guillotine cutting. *The Operational Research Society*, 36(4):297-306.

BEASLEY, J.E. 1985a. An algorithm for the two-dimensional assortment problem. *European journal of operational research*, 19(2):253-261.

BISCHOFF, E.E. & WÄSCHER, G. 1995. Cutting and packing. *European journal of operational research*, 84(3):503-505.

BOWDOIN.EDU. 2010. Greedy algorithms. http://www.bowdoin.edu/~ltoma/teaching/cs231/fall09/Lectures/12-dynamicAndGreedy/greedy.pdf Date of access: 06 Mar. 2010.

BURKE, E.K., KENDALL, G. & WHITWELL, G. 2003. A new placement heuristic for the orthogonal stock-cutting problem. *Operations research*, 52(4):655-671.

CHRISTOFIDES, N. & HADJICONSTANTINOU, E. 1993. An exact algorithm for orthogonal 2 D cutting problems using guillotine cuts. *European journal of operational research*, 83(1):21-38.

CHRISTOFIDES, N. & WHITLOCK, C. 1977. An algorithm for two-dimensional cutting problems. *Operations research*, 25(1):30-45.

CUI, Y. 2007. Simple block pattern for the two-dimensional cutting problem. *Mathematical and computer modelling*, 45(7-8):943-953.

CUNG, V.D., HIFI, M. & LE CUN, B. 2000. Constrained two-dimensional cutting stock problems: A best-first branch-and-bound algorithm. *International transactions in operational research*, 7(3):185-210.

DAZA, V.P., DE ALVARENGA, A.G. & DE DIEGO, J. 1995. Exact solutions for constrained two-dimensional cutting problems. *European journal of operational research*, 84(3):633-644.

DYCKHOFF, H. 1990. A typology of cutting and packing problems. *European journal of operational research,* 44(2):145-159.

DYSON, R.G. & GREGORY, A.S. 1974. The cutting stock problem in the flat glass industry. *Operational research quarterly,* 25(1):41-53.

ELMAGHRABY, A.S., ABDELHAFIZ, E. & HASSAN, M.F. 2000. An intelligent approach to stock cutting optimization. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.93.7710&rep=rep1&type=pdf Date of access: 15 Nov. 2009.

EROGLU, S. & NOCHE, B. 2004. Slitting optimisation problem. http://www.sim-serv.com/pdf/whitepapers/whitepaper_80.pdf Date of access: 20 Sept. 2009.

FAYARD, D., HIFI, M. & ZISSIMOPOULOS, V. 1998. An efficient approach for large-scale two-dimensional guillotine cutting stock problems. *The journal of the Operational Research Society,* 49(12):1270-1277.

GILMORE, P.C. & GOMORY, R.E. 1965. Multistage cutting stock problems of two and more dimensions. *Operations research,* 13(1):94-120.

HADJICONSTANTINOU, E. & IORI, M. 2007. A hybrid genetic algorithm for the two-dimensional single large object placement problem. *European journal of operational research,* 183(3):1150-1166.

HE, K.J., HUO, Y.Y. & ZHANG, R.G. 2009. A hybrid metaheuristic for the 2D orthogonal cutting stock problem. Proceedings of the eighth international conference on machine learning and cybernetics, 12-15 July, Baoding. 6p. http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=05212495 Date of access: 12 Mar. 2010.

HIFI, M. 1997. An improvement of Viswanathan and Bagchi's exact algorithm for constrained two-dimensional cutting stock. *Computers & operations research,* 24(8):727-736.

HIFI, M. & ZISSIMOPOULOS, V. 1996. A recursive exact algorithm for weighted two-dimensional cutting. *European journal of operational research,* 91(3):553-564.

KARELAHTI, J. 2002. Solving the cutting stock problem in the steel industry. Helsinki: Helsinki University of Technology. (Thesis - PhD.) 86 p.

KENYON, C. & RÉMILA, E. 2000. A near-optimal solution to a two-dimensional cutting stock problem. *Mathematics of operations research,* 25(4):645-656.

KLEMPOUS, R., KOTOWSKI, J. & SZLACHCIC, E. 1996. Interactive procedures in large-scale two-dimensional cutting stock problems. *Journal of computational and applied mathematics,* 66(1-2):323-331.

LI, H.L. & TSAI, J.F. 2001. A fast algorithm for assortment optimization problems. *Computers & operations research*, 28(12):1245-1252.

LIN, C.C. 2006. A genetic algorithm for solving the two-dimensional assortment problem. *Computers & industrial engineering*, 50(1-2):175-184.

MACEDO, R., SILVA, E., ALVES, C., ALVELOS, F.P., VALERIO DE CARVALHO, J.M.V., ARBIB, C., MARINELLI, F., PEZZELLA, F., DE GIOVANNID, L. & GAMBELLA, L. 2008. 2D cutting stock optimization software survey. *OR/MS today*. http://www.scoop-project.net/DOCUMENTI/File/MacedoEtAlii_080715.pdf
Date of access: 25 Aug. 2009.

MACEDO, R., ALVES, C. & VALÉRIO DE CARVALHO, J.M. 2010. Arc-flow model for the two-dimensional guillotine cutting stock problem. *Computers & operations research,* 37(6): 991-1001.

MOORE, J.H. & WEATHERFORD, L.R. 2001. Decision modeling with microsoft® excel. 6 th ed. Upper Saddle River ,N.J: Prentice Hall. 693p.

PG Glass, 1993. Data received from PG Glass on CD. (CD in possession of Prof T Steyn, School for Computer Science, NWU)

OBERHOLZER, J.A. 2003. Implementing artificial intelligence search methods to solve constrained two-dimensional guillotine-cut cutting stock problem. Potchefstroom: Potchefstroom University for CHE (Thesis - PhD.) 212 p.

PENTICO, D.W. 1974. The assortment problem with probabilistic demands. *INFORMS*, 21(3):286-290.

PENTICO, D.W. 1988. The discrete two dimensional assortment problem. *Operations research*, 36(2):324-332.

PENTICO, D.W. 2008. The assortment problem: a survey. *European journal of operational research*, 190(2):295-309.

RIEHME, J., SCHEITHAUER, G. & TERNO, J. 1996. The solution of two-stage guillotine cutting stock problems having extremely varying order demands. *European journal of operational research*, 91(3):543-552.

SULIMAN, S.M.A. 2006. A sequential heuristic procedure for the two-dimensional cutting-stock problem. *International journal of production economics*, 99(1-2):177-185.

TIWARI, S. & CHAKRABORTI, N. 2006. Multi objective optimization of a two-dimensional cutting problem using genetic algorithms. *Journal of materials processing technology*, 173(3):384-393.

TMACHINES.COM. 2010. Cutlogic 2D. http://www.tmachines.com/index.htm Date of access: 20 Feb. 2010.

TSCHÖKE, S. & HOLTHÖFER, N.1995. A new parallel approach to the constrained two-dimensional cutting stock problem. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.55.7755&rep=rep1&type=pdf Date of access: 07 Sept. 2009.

VASKO, F.J. & WOLF, F.E. 1994. A practical approach for determining rectangular stock sizes. *The journal of the Operational Research Society*, 45(3):281-286.

VISWANATHAN, K.V. & BAGCHI, A. 1993. Best-first search methods for constrained two-dimensional cutting stock problems. *Operations research,* 41(4):768-776.

WANG, P.Y. 1983. Two algorithms for constrained two-dimensional cutting stock problems. *Operations research,* 31(3):573-586.

WÄSCHER, G., HAUBNER, H. & SCHUMANN, H. 2007. An improved typology of cutting and packing problems. *European journal of operational research,* 183(3):1109-1130.

YANASSE, H.H., ZINOBER, A.S.I. & HARRIS, R.G. 1991. Two-dimensional cutting stock with multiple stock sizes. *The journal of the Operational Research Society,* 42(8):673-683.