Sequencing and Cost Reduction using a Knapsack formulation and Dynamic Programming

by

Albert Eicker
24071600

Submitted in partial fulfilment of the requirements for the degree of

BACHELORS OF INDUSTRIAL ENGINEERING

in the

Faculty of Engineering, Built Environment, and Information Technology University of Pretoria

Pretoria, South Africa

October 2009

Executive Summary

The problem of cost minimisation through sequencing different nested patterns according to priority and requirements are addressed in this project.

The feasibility of scheduling these nest's cutting order as to minimise the number of sheets required as well as to reduce the total production cost incurred, is to be determined and compared to current sequencing methods as are currently employed at Aerosud.

The method that is implemented to address this problem is a dynamic programming model that incorporates a knapsack formulation at its core.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter the background information about the company and the problem is presented and is discussed.

1.1 Background on the Company

Aerosud is an aviation company that was formed in the early 90's by designers of the Rooivalk helicopter and leaders of the Product Support team of the Cheetah fighter aircraft.

Aerosud currently supplies aircraft components to both commercial and military institutions, some of whom include Airbus, Boeing, BAE Systems, Agusta Westland Helicopters and Spirit AeroSystems. These components are manufactured in-house and shipped to the customers assembly line.

Aerosud is a respected name in aviation industry, locally as well as internationally.

1.2 Background on the Problem

After doing some initial inspection and analysis, a bottleneck was identified at one of the initial processes that most of the production lines pass through.

A direct result of this bottleneck is a major delay in the downstream processes that is reflected in the delivery dates of the final products which in turn has a substantial financial implication.

Figure 1.1: Aerosud Facility

The identified cell, the router cell, cuts the needed parts from blank sheets of aluminium. Off cuts cannot be stored because the burrs caused by the cutting process cause surface damage to the neighbouring sheets, therefore the nesting of the parts is done in such a way that the sheets are utilised fully in one go, and batches of parts are produced.

Since the nests, which are of different size, are cut from different types and thickness of aluminium, the batching results in the overproduction of some parts, while the production of other parts are kept on hold.

It is this delay in production of certain parts that often results in excessive idle time for operators involved in downstream processes and ultimately these delays are reflected in the overshooting of the delivery dates.

The correct sequencing is thus a major oversight in the current production method, and forms the baseline of the problem addressed in this project.

Chapter 2

Project Objectives

2.1 Project Aim

The main objective is to determine an improved cutting sequence and to better group the different nests together as to reduce the overall costs incurred by the production of the different parts.

The secondary objective of the project is to locally improve the throughput of the identified problem cell and to align it with the the rest of the production line.

The third objective will be to generate some guidelines for the router's operator to better determine the sequence of cutting and job selection.

2.2 Project Scope

Due to the complexity and extensive range of variables that factor into the actual cutting process, some aspects of the process need to be limited.

There will be no detail modelling of the parts as they are mostly odd- or irregular in shape, they will be approximated using rectangles.

There is a wide range of aluminium types used for the fabrication of different parts, there will only be focussed on the most commonly used type.

Only two routers will be available to do the cutting of the parts, each having certain specified cutting bits available. The machines will work in

parallel with each other.

We will assume that no job needs to be processed by both machines.

The part range that will be focused on will also be limited to only one product family using a single thickness of material, this would give sufficient representation of different size and types of parts produced.

All cost values will be expressed in terms of Rand, all time values are expressed in minutes and all measures of length are given in millimetres.

Due to some confidentiality clauses, some of the data had to be adjusted.

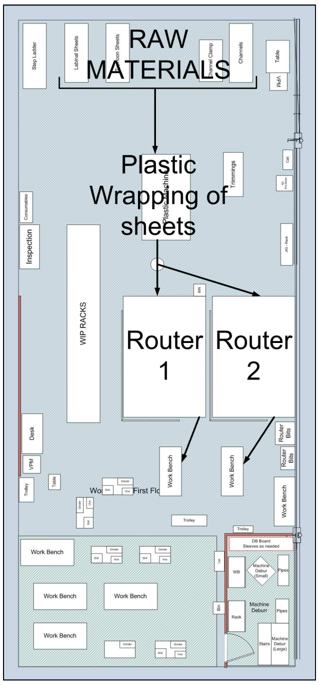

Figure 2.1: Layout of Facility with Routers in Parallel

Chapter 3

Project Planning

In this chapter, different methods and techniques of achieving the goals as set out in the project aim are discussed.

3.1 Analysis of Relevant Literature

3.1.1 Basic Concepts

Overall Throughput Improvement

"Optimising bits and pieces, does not optimise the entire system" [Degarmo,2003]

Some conditions need to be satisfied prior to any improvement attempts on the assembly line or production process. The first and probably most important is that the process that is under review needs to be stable, and controllable. Production control can be defined as the methods by which the handling of materials, sub-assemblies and parts are governed and implemented in the work environment. [Tompkins,2003]

Since the critical controls of different types of manufacturing systems vary with respect to certain key aspects of production, it is crucial to identify the type of system that is applicable to the environment under study.

Different problems and applicable methods where examined in order to ascertain the required mathematical formulation and structure to address the nest-scheduling problem.

Wastes in Manufacturing

The waste that occur in the manufacturing environment can be classified as any activity that is non-value adding [Taylor,2001] and can be classified as in table 3.1. The one we will address is the overproduction.

Table 3.1: Wastes in Manufacturing

Waste	Description
Transportation	Moving of parts from station to station
Inventory	Tied up resources and material between stations
Motion	Excessive operator or machine movement
Waiting	Idle time of operator while waiting for process to complete
Overproduction	Produce more than is needed
Over processing	Unnecessary work done on parts
Defects	Producing non conforming products

Scheduling

Scheduling is a task that links a time requirement to a certain operation. [Chase,2004] Scheduling is used to plan production and helps in managing the manufacturing environment.

There are many different scheduling techniques. The one we will focus on is Material Requirement Planning (MRP).

MRP is an easy and adequate method of planning. The method takes into account the material and components needed to produce a completed product and then computes the schedule specifics of when to order the material and when to manufacture the components.

When compiling the schedule, certain rules need to be applied to determine the correct order in which tasks need to be completed. The most commonly used rules are listed in Table 3.2 as identified by Chase et al. [Chase,2004]

Nesting

Nesting is a process of grouping parts or orders together to be cut from a sheet of material. This grouping is mathematically based and aims to max-

Table 3.2: Scheduling Rules

Id	Description
1.	First in - first out (FIFO)
2.	Last in - first out (LIFO)
3.	Last in - last out (LILO)
4.	Earliest due date (EDD)
5.	Random order (RO)

imise either the use of material or minimise the waste or scrap that remains.

Using nesting could greatly reduce machine setup time, batch sizes and reduce material cost. [Herrman,2001]

It is however important to calculate the nest using the correct information surrounding the release procedures of the parts produced and the downstream requirements. If done incorrectly this would reduce the machines overall throughput. [Fu,1997]

There are many different approaches in computing nesting algorithms. Since the complexity of nesting irregular shaped parts increase the computational burden, it is difficult to apply a deterministic algorithm. To obtain an approximate solution some heuristic and meta-heuristic algorithms were developed. The works of Hopper and Turton [Hopper,2001] gave an in-depth overview of 2D irregular shaped nesting.

Other authors such as Jacobs [Jacobs,1996] used order based generic algorithms to approximate the nesting of polygons. Since the parts that will be nested at Aerosud will be approximated by rectangles as in figure 3.1 these algorithms mentioned above are mathematically too advanced.

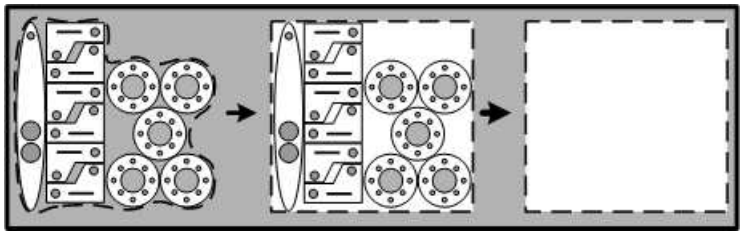

Figure 3.1: Parts Approximated by Rectangles

Standard Operating Procedures

If each worker did the same job in his own way the variability between them would be substantial. [Chase,2004]

The implementation of standard operating procedures (SOP') would set a standard and also give sufficient guidance to complete the job. It also makes the operators interchangeable, one can continue where the other left off and the transition would be smooth. This is due the fact that both work in the same manner.

3.1.2 Bin Packing

A common scenario that occurs in practise is to complete a task or sequence of tasks using as little resources as possible in an attempt to reduce costs and improve efficiency.

Bin packing is a common problem of assigning or packing items of varying sizes into bins, where each bin has a fixed capacity, this is considered to be a NP-hard combinatorial problem. [Bansal,2005]

Since this problem extends across multiple fields of application it has been well researched and many different solving approaches and approximations have been developed. [Coffman,1996]

The bins need not be physical bins, but could also be seen as machines having limited processing capacity that need to be allocated to an array of jobs.

Mathematically the goal is *to select and place items from set* $i_1, i_2, i_3, ..., i_n \in (0, 1]$ *into the fewest number of bins, each bin with a size of* 1.

We can assume that the bins have a size of 1 and still be able to apply the equation to a wide range of applications since up scaling both the bins and the items would amount an equivalent result.

3.1.3 Glass Cutting

The cutting of glass sheets into smaller rectangle shapes as per customer order is a real life problem that occurs in the glass manufacturing environment.

The main objective is to determine the best grouping of jobs as to reduce the overall waste that is produced as well as to reduce the number of setups that is required.

There is however a major constraint in the way that the cuts are made on the blank sheets of glass. The cutting needs to be done in a guillotineable manner and the patterns need to be arranged accordingly. A major result of this is that the parts are only allowed to be rotated at an angle of 90 degrees when placed into the cutting pattern. [Kroger,1995] *Appendix A* discusses this in more detail.

Only straight line cuts can be made on the sheets of glass as is illustrated in figure 3.2. After the initial cuts are made (A and B), the smaller sections are then again loaded into the cutting machine and cut horizontally into the desired parts.

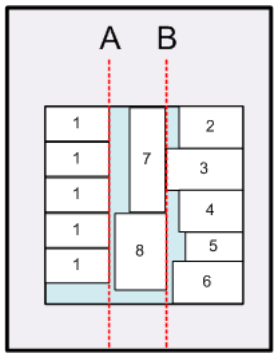

Figure 3.2: Guillotine Constraint of Glass Cutting

A direct mathematical constraint that can be derived from this guillotineable cutting method is that no overlapping of parts are allowed. A vertical asymptote needs to be developed and included in the constraints of the models formulation. The cuts will typically be made on the asymptotes.

3.1.4 Carpet Assignment

The weaving industry has the problem of scheduling different orders of carpets, each carpet having its own width, length and quantity to produce.

Since carpets are woven on machines with fixed widths, there will always be "wasted" areas between the different carpet sections that cannot be used.

The unusable area is a direct, but unavoidable waste and we can only try to minimise overall cost incurred, not eliminate it. [Demir,2005]

A unique difference between this carpet assignment and a regular two dimensional stock cutting problem is that material that the shapes are cut from is only confined by the width of the machine and not the length of the input material since the carpet is woven off rolls of yarn.

The other characteristic that that defines the carpet assignment is that unlike a two dimensional bin-packing problem, no overlapping of parts or segments are allowed. [Lodi,2002]

Considering the above motioned, there are strong correlations in the formulation of both glass cutting and the carpet assignment problems.

3.1.5 The Knapsack Problem

The Knapsack Problem is an optimisation problem that entails the selection of items to maximise overall value or benefit given a certain capacity constraint. [Gilmore,1967] It is generally regarded as being a one dimensional binpacking problem.

The most commonly suggested methods of solving knapsack problems are Dynamic Programming or Tree search techniques such as the branch and bound method. [Greenberg,1970]

Dynamic programming (DP) is a technique that breaks a problem down into a series of smaller more manageable optimisation problems, each having the same structure as the original one. [Winston,2003]

The solving method works in a backwards manner, calculating from the end of the problem back to the beginning.

Each of the smaller problems is assigned as stages, and each stage can be in a certain state. The state of the stage is basically the information to make an optimal decision in moving to the next stage. The principle of optimality requires that the decisions to move from the current stage to the next not be influenced by the previous decisions or movements made.

If we define the problem as having T different stages, the recursion that relates the movement and the benefit obtained during the stages

$t, t+1, t+2, \ldots T$ can be formulated as:

$$f_t(x) = min/max(c_{ij} + f_{t+1}(j)) \tag{3.1}$$

Where j must be the value at stage $t+1$ and c_{ij} the relevant optimal for the previous stages.

3.1.6 Sequencing and Resource Allocation

Similar to the knapsack problem, resource allocation and sequencing can be formulated as a function that tries to find the critical path or either minimizing waste, or maximising production.[Lawler,1969]

One such problem that has been studied extensively is the travelling salesman problem. The objective is to determine a sequence or route to travel that will minimise travelling time or travelling expense.

If we define x_{ij} as the distance between cities i and j, and

$$c_{ij} \triangleq \begin{cases} 1 & : \text{if traveling from } i \text{ to } j \quad i, j \in \{1, 2..., 8\} \\ 0 & : \text{otherwise} \qquad\qquad\quad j \neq i \end{cases}$$

Then the solution can be found by solving:

$$min(Z) = \sum_i \sum_j (c_{ij} x_{ij}) \tag{3.2}$$

$$s.t. \tag{3.3}$$

$$\sum_{i=1}^{i=N} c_{ij} = 1 \qquad j = 1, 2, ...N \tag{3.4}$$

$$\sum_{j=1}^{j=N} c_{ij} = 1 \qquad i = 1, 2, ...N \tag{3.5}$$

$$i \neq j; \tag{3.6}$$

$$i = 1, 2..., N \tag{3.7}$$

$$u_i - u_j + N x_{ij} \leq N - 1 \qquad j = 2, 3, ..., N \tag{3.8}$$

$$x_{ij} = 0 \text{ or } 1; \; u_j \geq 0 \tag{3.9}$$

The constraints of [3.6]-[3.9] are very important, since they ensure a feasible route.

Chapter 4

Define and Design

This chapter discusses the defining and collection of the required data surrounding the input variables for the problem's formulation.

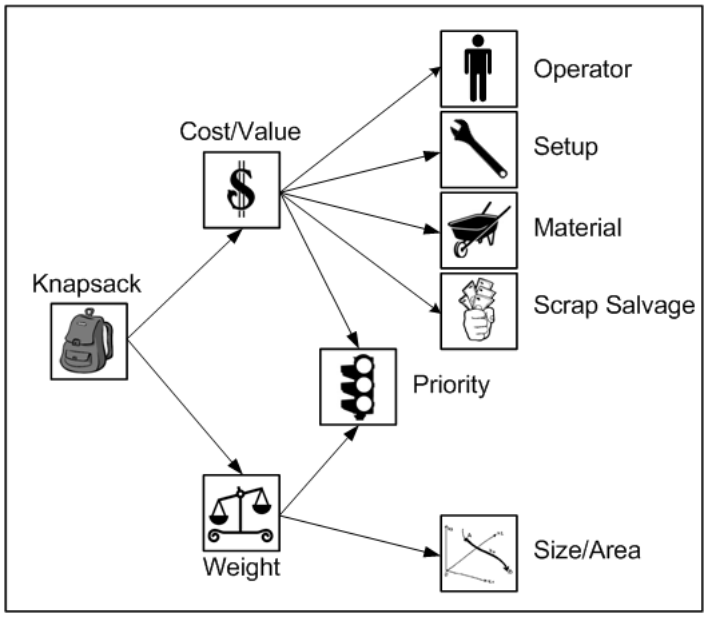

Figure 4.1: Breakdown Diagram of Model Outline

The data and information surrounding these components are required to formulate the model. This data needs to be coherent with the requirements of a knapsack as defined in the previous chapter.

4.1 Defining Parameters

These are only the main parameters that arise from analysing the cutting process.

4.1.1 The Nests

Since the nests are developed by grouping several irregular shaped parts, the overall shape of the nests is irregular more often than not. The nesting programme Sigma-Nest strives to nest the parts as close to a rectangle shape as far as possible. We will represent these developed nests by defining rectangles for use in our calculations.

To calculate the particulars of the rectangles (nests) we need to first define its dimensions (Supposing we have n - nests):

$$\begin{array}{lcll} l_i & \triangleq & \text{the length of nest } i & i \in \{1, 2..., n\} \\ w_i & \triangleq & \text{the width of nest } i & i \in \{1, 2..., n\} \\ A_i & \triangleq & \text{the Area needed for nest } i & i \in \{1, 2..., n\} \end{array}$$

It would then be of substantial importance to associate a priority parameter with each nest, and we will define p_i as:

$$p_i \triangleq \begin{cases} 0 & : \text{Nest Cancelled} \\ 1 & : \text{Normal Priority} \\ 2 & : \text{High Priority} \end{cases} \qquad \text{where } i \in \{1, 2..., n\}$$

The different nests will have to be cut each requiring two uniquely specified cutting bits. We thus need to define a variable that associates the required bits to each nest. If we assume that there are only 10 cutting bits, we can define R_{ikb} as:

$$R_{ikb} \triangleq \begin{cases} 1 & : \text{Use bit } b \text{ for sheet } i \text{ in position } k \\ 0 & : \text{Otherwise} \\ & i \in \{1, 2..., n\} \qquad k \in \{1, 2\} \qquad b \in \{1, 2..., 10\} \end{cases}$$

Each nest also has a setup time to program the machine, this will be represented by:

$s_i \triangleq$ total programme loading time for nest i, $i\{1, 2..., n\}$

4.1.2 The Operators

We will need to know the remuneration costs involved for employing the operator that needs to operate and oversee the cutting process.
We will define :

$O \triangleq$ Defines the per hour rate of employing the operator

4.1.3 The Machines

The setup of the machine prior to the cutting operation will also need to be taken into account. We will define S_j as :

$S_j \triangleq$ total setup time required for sheet j, $j \in \{1, 2..., X\}$ where X is the number of sheets required

We also need to define a 0/1 variable to indicate which nest is placed on which sheet, which in turn will determine which router will be used to cut the sheet.

$$y_{ij} \triangleq \begin{cases} 1 & : \text{if nest } i \text{ is nested on sheet } j \quad i \in \{1, 2..., n\} \\ 0 & : \text{otherwise} \qquad\qquad\qquad\quad j \in \{1, 2..., X\} \end{cases}$$

$$S_j = \sum_{i=1}^{i=n} s_i y_{ij} \qquad i = \{1, 2, ...n\};\ j = \{1, 2, ...X\} \tag{4.1}$$

$$\tag{4.2}$$

Since the actual cutting time will not be affected by the changes made to the sequence, it will not be taken into account.

4.1.4 The Material

There are allot of input variables to take into account when looking at the material used in the cutting process. We define the following :

c_j	$\triangleq$	the cost of sheet j
r_j	$\triangleq$	the scrap value per square unit of the material remaining on sheet j
t_j	$\triangleq$	loading time for sheet j
L_j	$\triangleq$	length of sheet j
W_j	$\triangleq$	width of a sheet j
M_j	$\triangleq$	Area of sheet j
X	$\triangleq$	number of sheets required
		all of the above are subject to $j \in \{1, 2, ...X\}$

4.2 Model Formulation

4.2.1 Cost/Value

Upon completion of the grouping of the nests, there could be some unused material that could be salvaged at a rate r_j, we will define the total salvageable value of sheet j as v_j:

$$v_j = r_j(M_j - \sum_{i=1}^{n} A_i y_{ij}) \quad \forall j,\ i \in \{1, 2, ..., n\} \tag{4.3}$$

Let T_j be the total setup and material *costs* for sheet j then:

$$T_j = (O(t_j + S_j) + c_j) \quad \forall j, j \in \{1..X\} \tag{4.4}$$

4.2.2 Weight

The weight of each nest as with respect to conforming to the capacity constraint of the knapsack would mostly consist of A_i, the area required by it.

$$A_i = l_i w_i \, i \in \{1, 2..., n\} \tag{4.5}$$

We must also make provision for the priority constraint p_i that we defined. A high priority item will be forced to fit, regardless of the cost implications.

The inclusion of the setup time and cost that is incurred by each nest can also be included in the weight formulation. Using this parameter in both the *cost/value* and the *weight* sections of the item does not cause any concern since the *weight* is only used as a decision tool in the *weight* section and does not again get added to the cost of the nest.

4.2.3 Capacity

The capacity of each sheet would simply be defined as M_j:

$$M_j = L_j W_j \ , \ j \in \{1, 2..., X\}$$

In the calculation of the capacity constraint we will not be able to only use the area (A_j) of the sheets, but we will also have to look at the respected lengths (L_j) and widths (W_j). This is better explained in *Appendix B*

4.2.4 Objective function

The goal of the model as discussed in section 2.1 is to reduce the cost of the cutting operation and to correctly sequence the nests. This would be accomplished by the following equation:

$$min(Z) = (O(t_j + S_j) + c_j) - r_j(M_j - \sum_{i=1}^{n} A_i y_{ij}) \tag{4.6}$$

subject to

$$\sum_{i=1}^{n} l_i y_{ij} \leq L_j \qquad \forall j,\ i \in \{1, 2, ..., n\} \tag{4.7}$$

$$\sum_{i=1}^{n} w_i y_{ij} \leq W_j \qquad \forall j,\ i \in \{1, 2, ..., n\} \tag{4.8}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{X} y_{ij} = 1 \qquad j \in \{1, 2, ..., X\}; i \in \{1, 2, ..., n\} \tag{4.9}$$

$$\tag{4.10}$$

[4.7;4.8] Sets the restriction that any sheet's capacity cannot be exceeded. [4.9] Limits the decision parameter to only assume {0,1} and ensures that all nests are cut.

This model does not take into account the forced insertion of a high priority nest, this will be coded into the the model by using a simple *if*-statement.

The nest will be divided into 3 groups. Nests that can only by processed on a specific router due to the cutting tools that it requires (group 1 & 2), and nests that can be cut on either of the two machines (group 3).

Each machine will then be assigned its own "knapsack". The selection of items from group 3, will be done on a first in, first out basis, this means that the first machine that can use a specific nest from group 3, will do so, this is illustrated in figure 4.2.

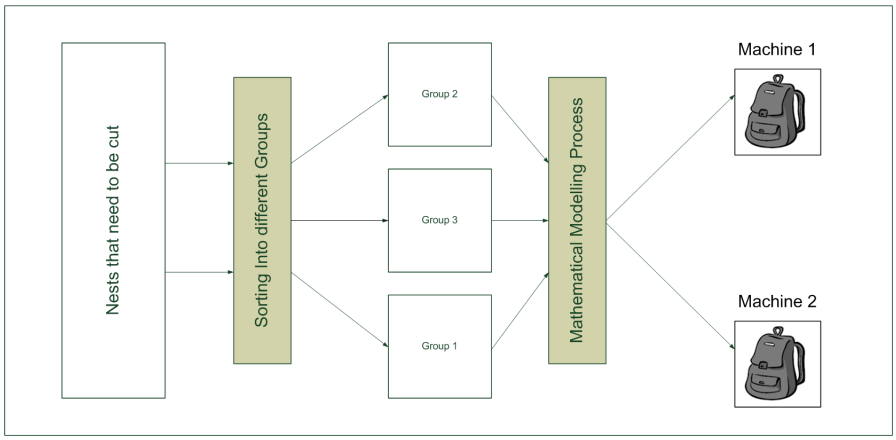

Figure 4.2: Outline of Parts to Machines

4.3 Dynamic Programming

If we formulate the model as shown in the previous section as a dynamic programming problem we obtain the following result:
We first need to additionally define :

$f_t(M_j) \triangleq$ minimum cost incurred on sheet j with capacity M_j, that is filled with nests $t, t+1, ..., n$ where n is the available nests.

$$f_t(M_j) = f_{t-1}(M_j) \quad if\ A_t \geq M_j \quad (4.11)$$
$$f_t(M_j) = min\{f_{t-1}M_j, O(s_t + t_t) + f_{t-1}(M_j - A_t)\} \quad if\ A_t \leq M_j \quad (4.12)$$
$$A_t y_{tj} \leq M_j \quad (4.13)$$
$$A_t, Os_t \geq 0 \quad (4.14)$$

[4.11] If nest is bigger than available sheet area.
[4.12] Nest is smaller than available area.
[4.13,4.14] Limits the control variables and nest sizes to be larger that 0.

Chapter 5

Model Implementation

5.1 Data

This section links the variables as defined in the previous chapter with the data as collected. Due to some disclosures and confidential information policies surrounding the material types and costs involved, the data had to be scaled and adjusted. The functionality and output results of the adjusted data is accurate enough to illustrate the models operation.

5.1.1 Nests

The nest's data that was collected was tabulated as illustrated in table 5.1. The nests were numbered and all relevant attributes and requirements were documented.

For the purpose of doing relevant comparisons and analysis, groups of 100 nests were used.

5.1.2 Routers

As discussed in the previous chapters, each router only has a limited cutting tool carrying capacity, thus each router only has a limited range of tools available. The tool bits each router contains is tabulated in table 5.2.

The variable R_{ikb} of the nests describes the corresponding tool that is required to perform the cut on it. Ex. R_{125} means that nest 1, requires for its secondary (2) cutting bit, cutting tool 5. This is also used in the assignment

Table 5.1: Extract From Nests Data list

Nest	Length	Width	Area	Priority	Cutting Tools		Time
Number	l_i	w_i	A_i	p_i	R_{i1b}	R_{i2b}	s_i
1	800	400	320000	1	4	5	6
2	660	220	145200	2	4	8	3
3	500	1200	600000	1	7	2	9
4	700	550	385000	1	10	4	7
5	1200	350	420000	1	8	2	3
6	990	660	653400	1	8	7	7
7	400	330	132000	1	9	3	3
8	500	550	275000	1	6	2	5
9	300	660	198000	0	6	7	2
10	400	1350	540000	1	4	5	8
11	350	880	308000	1	4	5	6
12	450	710	319500	1	3	1	4
13	780	330	257400	1	7	6	8
14	340	800	272000	1	6	3	9
15	350	1050	367500	2	9	3	11
16	1200	600	720000	1	3	4	4
17	550	600	330000	1	4	5	3
18	1100	740	814000	2	9	10	6
19	350	940	329000	1	3	9	17
20	1000	670	670000	1	2	2	3
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

of the nests to a specific router.

5.1.3 Sheets

The properties of the aluminium sheets that are used, is tabulated in table 5.3. There are two different sizes of sheets that are used, the smaller of the two is used on router 2.

5.1.4 Operator

The cost of employing an operator (O) for one hours work was calculated at $R100$.

Table 5.2: Cutting Tools of the different machines

Machine	Cutting Bits									
Router 1	1	2	3	-	-	6	7	8	-	-
Router 2	-	-	3	4	5	-	-	8	9	10

Table 5.3: Sheet Information

Sheet Type	Cost (R)	Scrap Value	Loading Time	Length	Width	Area
	c_j	r_j	t_j	L_j	W_j	M_j
AluARC	4000	0.12	10	2000	2500	5000000
AluARC	5000	0.13	15	2000	3000	6000000

5.2 Output

The output of the model groups the different nests and allocates these groups to specific sheets. The prioritised nests are placed within the first two sheets that are cut, depending on where they fit best. Tables 5.4, 5.5 tabulates the allocations made for the first 100 nests in the data set using the created model.

Table 5.4: Nesting Sequence 1

Nesting Sequence:Router 1				
Sheet 1	Sheet 2	Sheet 3	Sheet 4	Sheet 5
82	89	16	72	61
18	95	65	57	54
93	91	69	52	79
88	87	73	30	21
67	100	25	81	23
92	75	71	42	83
51	78	17	10	15
37	90	66	41	1
27	60	19	44	80
	2	4	68	29
	50	38	11	33
	35		85	36
			7	98
				34

Table 5.5: Nesting Sequence 2

Nesting Sequence:Router 2			
Sheet 1	Sheet 2	Sheet 3	Sheet 4
94	56	5	12
64	62	99	39
77	76	53	3
84	70	24	45
47	63	22	28
46	74	20	32
86	14	40	49
58	13	43	97
		96	6
		8	48
		26	55
		31	

A graphical example of these nest allocations can be seen in figure 5.1 where the first 3 sheets of Router 1 are illustrated.

The relevant sheet data is also tabulated in table 5.6. The table contains the utilisation and costs as calculated by the model.

The models results was compared to 3 different nesting strategies, Biggest First, Fastest First and using the sequential numbering as the nests were launched from the designers. The last method mentioned is the one that is currently employed at Aerosud. These method's sheet information can be seen in Appendix C.

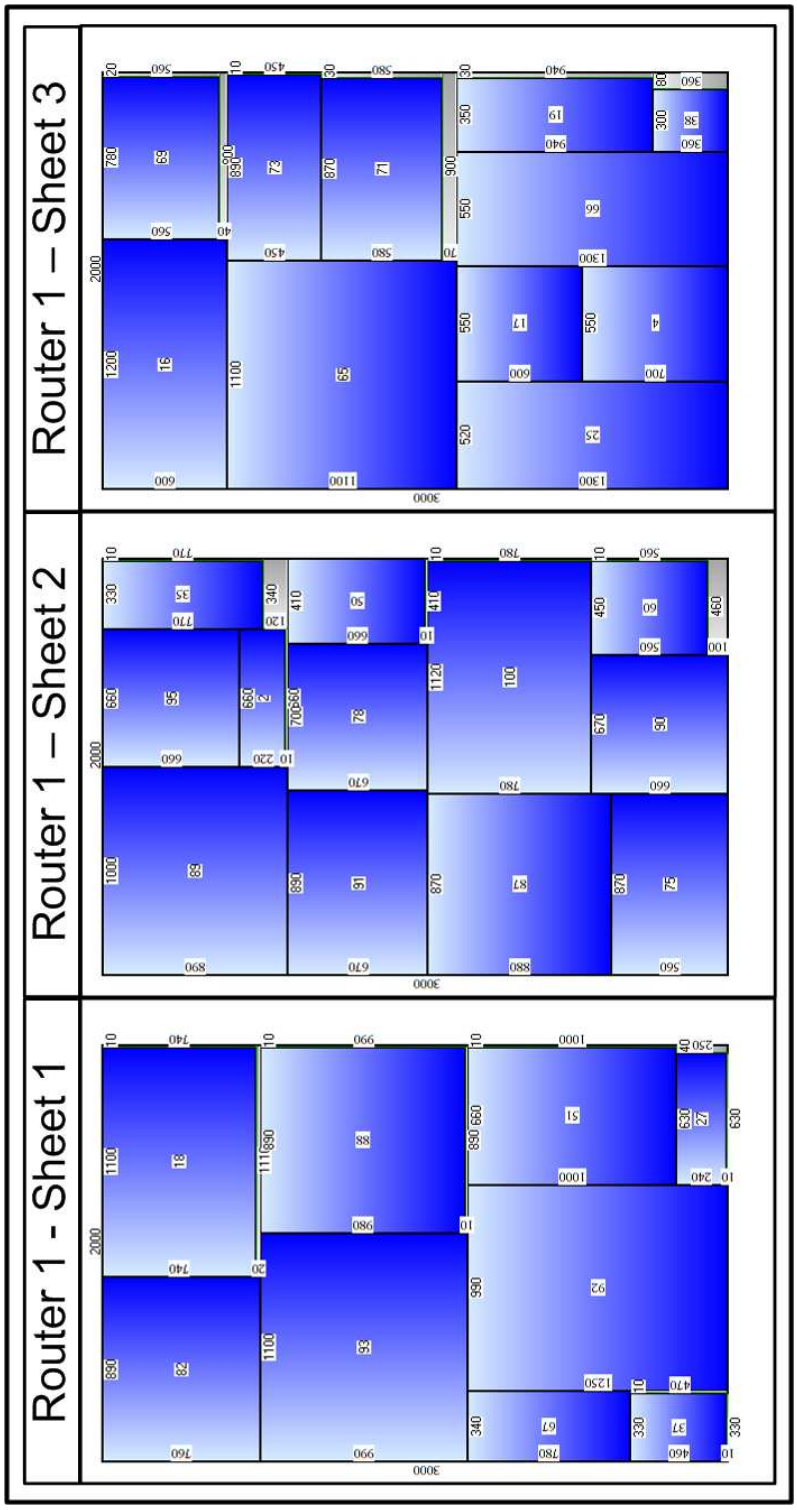

Figure 5.1: Nests Packed on Sheets

Table 5.6: Sheet Results From Model

Router 1					
Sheet Number	1	2	3	4	5
Area Available	6000000	6000000	6000000	6000000	6000000
Sheet Loading Time	15	15	15	15	15
Area Used	5917300	5881400	5814900	5741900	5294000
Utilisation	0.9862	0.9802	0.9692	0.9570	0.8823
Scrap Area	82700	118600	185100	258100	706000
Nest Programming Time	45.0	82.0	78.0	73.0	82.0
Scrap Value	76.92	76.46	75.59	74.64	68.82
Sheet Cost	5023.08	5085.21	5079.41	5072.02	5092.84
Router 2					
Sheet Number	1	2	3	4	5
Area Available	5000000	5000000	5000000	5000000	-
Sheet Loading Time	10	10	10	10	-
Area Used	4851600	4803300	4800600	3914700	-
Utilisation	0.9703	0.9607	0.9601	0.7829	-
Scrap Area	148400	196700	199400	1085300	-
Nest Programming Time	44.0	51.0	80.0	70.0	-
Scrap Value	58.22	57.64	57.61	46.98	-
Sheet Cost	4031.78	4044.03	4092.39	4086.36	-

Chapter 6

Results

The main idea of implementing the model was to try and achieve the results as were defined in the project objectives.

The comparison between the developed model and the three other methods were measured by the number of sheets required (the material costs), the operator costs and the combination of these costs.

Looking at the data given in tables 5.6, C.1, C.2, C.3, we can summarise the key results of the different methods in Table 6.1.

Table 6.1: Output Comparison Summary

	Method			
	Developed Model	Biggest Size First	Fastest Time First	Generated Number
Cost of Sheets	41000	50000	46000	46000
Cost of Operator	1200	1241.67	1225	1225
Salvaged Material	592.88	1751.14	1151.14	1151.14
Total Cost Incurred	41607.12	49490.53	46073.86	46073.86

6.1 Operator Costs

If we look at the cumulative operator costs incurred by the different methods as the sequencing of the nests progressed and the number of sheets increased, we can clearly see that the methods deliver varying results. Figure 6.1 illustrates this variation in operator costs.

The operator costs increased drastically as the number of sheets required also increased. This is because each sheet needs to be loaded and setup on the machines prior to the commencing of the cutting process.

The additional loading times of the sheets was the only cause of variation in operator costs in this instance. The reason for this is that the programming of the nests into the machines collectively equate to the same amount of time in each method. This is illustrated graphically in Figure 6.2. The setup times however are uniquely distributed over the sheets in each case.

The results achieved by the developed model, outperformed the other methods and resulted in the reduction of the operator expenses. The greedy approach of constantly trying to group the nests to achieve the fastest programming time for each consecutive sheet, parallelled the results achieved by using the nesting sequence as generated by the designers that do the launching of the parts. The least favourable results was achieved by sequencing the parts using the "Biggest Parts First" method. The cost are tabulated in Table 6.2.

Table 6.2: Operator Comparison Summary

	Method			
	Developed Model	Biggest Size First	Fastest Time First	Generated Number
Cost of Operator	1200	1241.67	1225	1225

6.2 Material Utilisation

The material utilisation factor gives an idea of how efficiently the nests where scheduled and placed onto the sheets. A high utilisation factor indicates that there is a minimal amount of scrap material after all the parts have been cut. The scrap however can be salvaged at a fixed rate per square unit of material.

The utilisation factors of the different methods are given in Table C.4, and is graphically illustrated in Figure 6.3.

The anomaly that occurs on sheet number six and eleven on the figure is due to the severe under filling of the specific sheets by the three comparative methods.

Sheet six is the last sheet of Router 1 that is used by these alternate methods and the material required is far less than what is available on the sheet, thus the poor utilisation. If the test sets were increased, the anomaly would still occur on the last sheet of a router set sequence, this due to the fact that all of the parts need to be cut, even if this requires an additional, under utilised, sheet. Looking at the utilisation factors (Table tab:utilsummary) this holds true for all of the different methods.

The developed model however succeeded in using only 5 sheets for Router 1, and 4 sheets for Router 2 - utilising the material far better that the other methods, this is also clearly illustrated in the figure.

6.3 Overall Cost

The most significant result that was obtained from running the developed model was the reduction of overall costs involved in the process. If we look at Figure 6.4 it is clear that the developed model is far less costly to employ than any of the comparative methods.

The reduction of the number of sheets required to produce the nests, not only reduces the cost, but it also increases the throughput of the process. By reducing the overall setup times, more parts can be produced and this in turn increases the productivity.

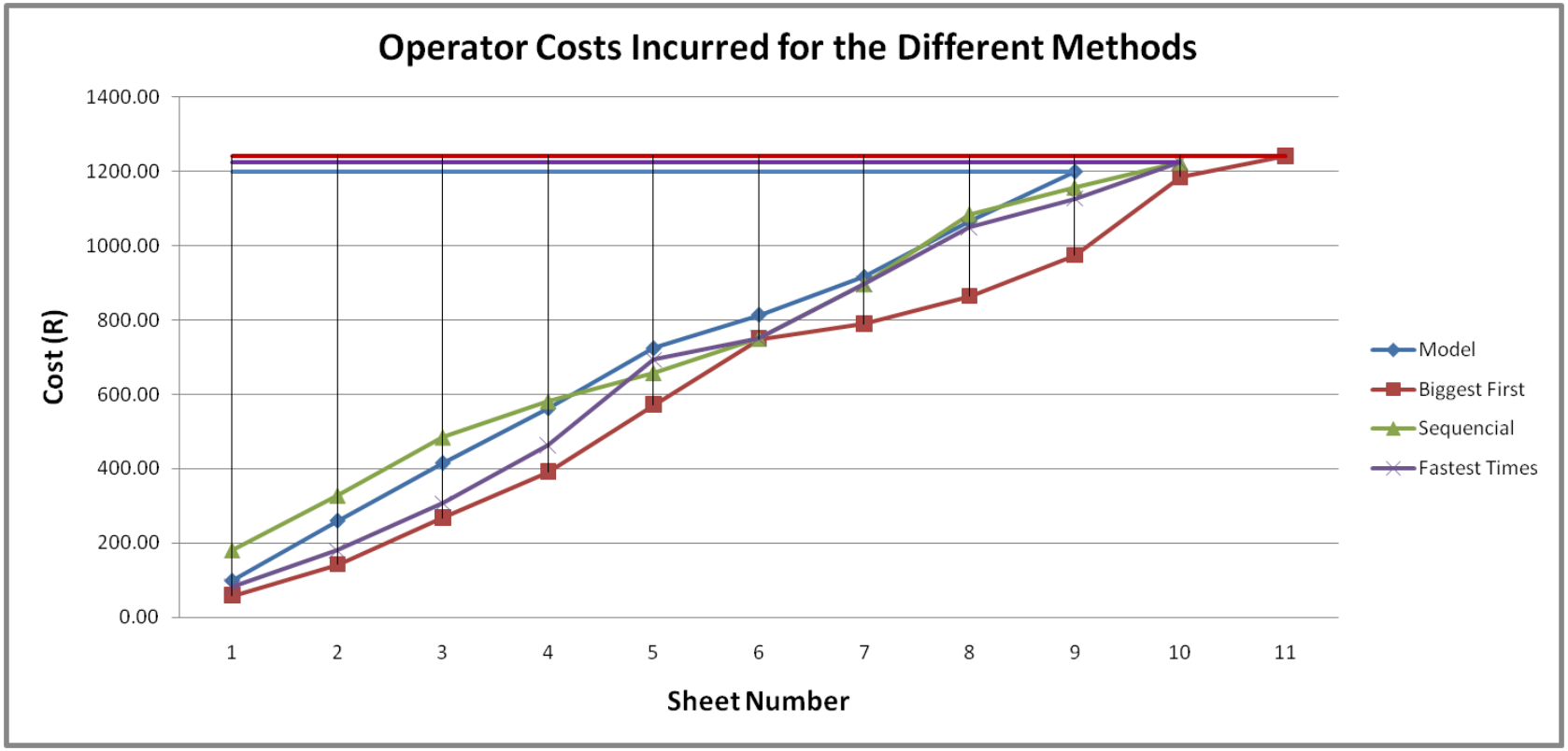

Figure 6.1: Comparison of Operator Costs using the different Methods

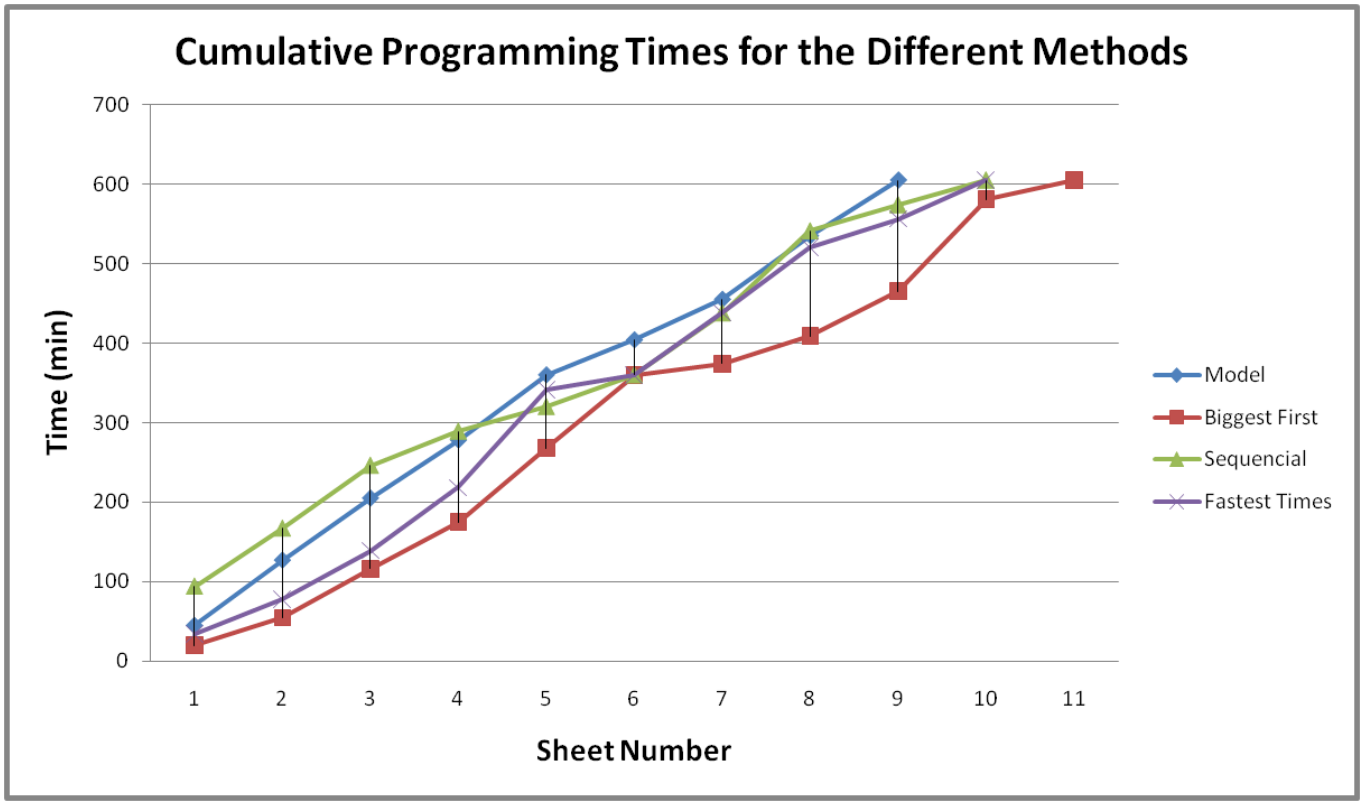

Figure 6.2: Comparison of Total Programming Times for the different sheets

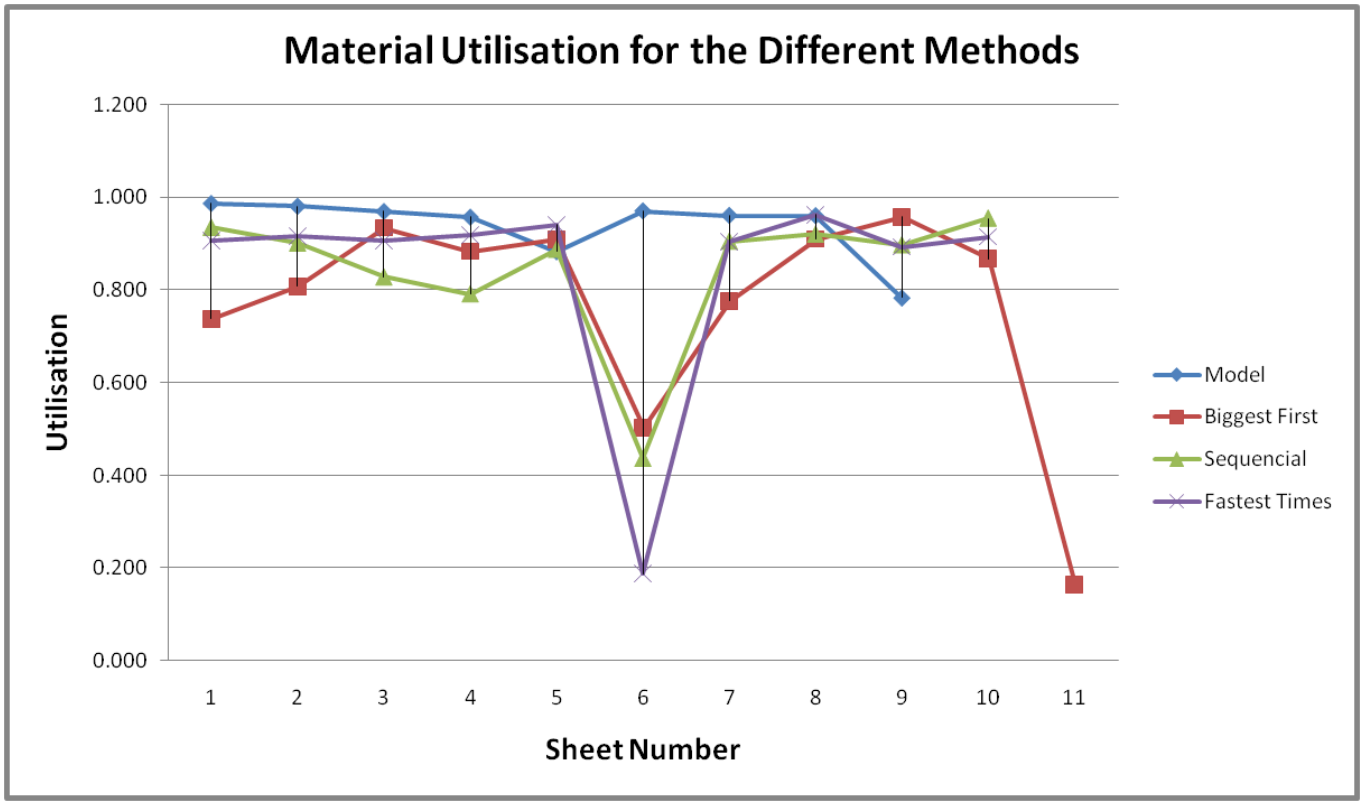

Figure 6.3: Comparison of different Method's Utilisation of sheets

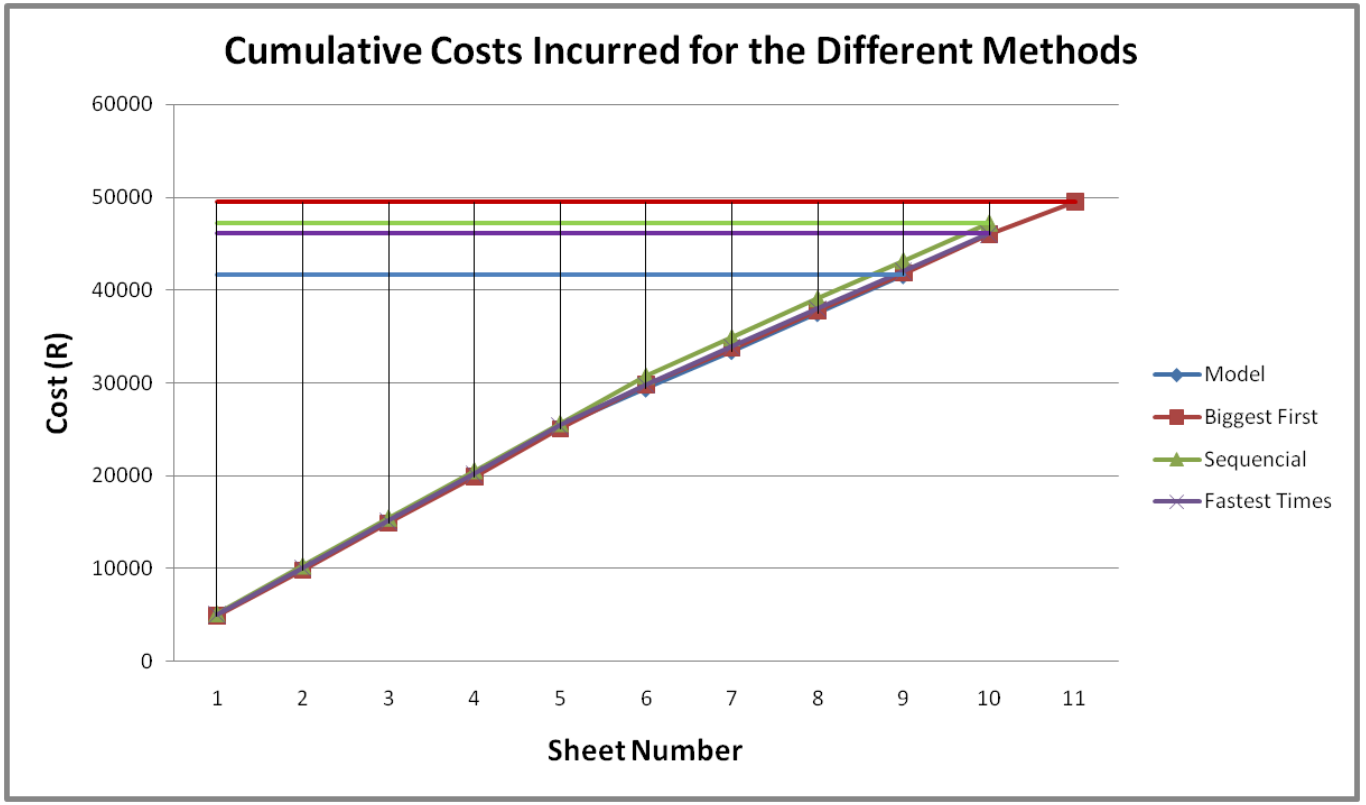

Figure 6.4: Comparison of Total Cost for different Methods

Chapter 7

Conclusion

Correctly sequencing and grouping nests of parts together for cutting has a great cost saving potential in sheet-metal manufacturing. It does not only reduce the material required, but also improves the setup times, which in turn reduces the operation costs.

We looked at environments with similar nesting and sequencing aspects through a substantial literature review, and then combined some of their attributes and constructed an algorithm based on a knapsack problem using dynamic programming.

The model that we developed in Matlab was used to schedule a number of parts cutting orders and the surrounding data was compared to three different methods of scheduling as applied in areas of industry.

The developed model outperformed the comparative methods in all aspects of the process, namely: the reduction in setup times, the reduction in material usage, the reduction in operator expense and the reduction of the overall costs of executing the process.

The model developed has many advantages, it generates a sequence for cutting the different parts while attempting to reduce the overall costs incurred by the process. Simulation with different sets of test data has proven the model's effectiveness in reducing setup times and material usage, directly reducing the overall costs incurred in the process.

Bibliography

[Taylor,2001] D. Taylor, D. B. (2001). Manufacturing Operations and Supply Chain Management. London: Thompson Learning.

[Kelton,2007] David Kelton, R. P. (2007). Simulation with Arena. New York: McGraw Hill.

[Hopper,2001] E. Hopper, B. T. (2001). A Review of the application of meta-heuristic algorithms to 2D strip nesting problems. Artificial Intelligence Review Vol16 , 257-300.

[Degarmo,2003] E. P. Degarmo, J. B. (2003). Materials and Processes in Manufacturing. John Wiley and Sons.

[Giordano,2003] Frank R. Giordano, M. D. (2003). A First Course in Mathematical Modeling. Thompson Books.

[Geng,2004] H. Geng, (2004). Manufacturing Engineering Handbook. McGraw Hill.

[Greenberg,1970] H. Greenberg, R. L. (1970). A Branch Search Algorithm for the Knapsack Problem. Management Science 16 , 327-332.

[Kroger,1995] B. Kroger, (1995) Guiloteanable bin packing:A generic approach. European journal of Operations Research 84:545-661.

[Jacobs,1996] Jacobs, S. (1996). On generic algorithms for the nesting of polygons. European Journal of Operations Research 88 , 161-181.

[Tompkins,2003] James Tompkins, J. W. (2003). Facilities Planning. John Wiey and Sons.

[Herrman,2001] Jeffrey W. Herrman, D. R. (2001). Algorithms for Sheet Metal Nesting. IEEE Transactions in robotics and automation XX .

[Bansal,2005] Bansal N.,Lodi A.,Sviridenko M. (2005) A Tale of Two Dimensional Bin Packing, In FOCS, 657-666.

[Demir,2005] Mert. C Demir, Ozgur Kulak, An enumeration-ilp based approach to the carpet assignment problem, 35th International Conference on Computers and Industrial Engineering Journal, 501-506

[Lodi,2002] Lodi Andre, Martello Silvano,Monaci Midhele (2002) Two-dimensional packing problems: A survey. European Journal of operations Research, 141,241-252

[Lawler,1969] E.L. Lawler, J.M. Moore, (1969) A functional equation and its application to resource allocation and sequencing problems. Management Science, 16,77-84

[Coffman,1996] Coffman E.G. Jr, Garey M.R., Johnson D.S. (1996) Approximation algorithms for bin packing:A survey. In Hochbaum, D, (ed) Approximation Algorithms for NP hard Problems, 46-93

[Fu,1997] M.C. Fu, J. H. (1997). Setting thresholds for periodic order release. Journal of intellegent Manufacturing Vol8 , 369-383.

[Gilmore,1967] P. C. Gilmore, R. E. (1967). The Theory and Computation of Knapsack Functions. Operations Research 15 , 1045-1075.

[Chase,2004] R.B Chase, F. J. (2004). Operations Management for Competitive Advantage. New York: McGraw-Hill.

[Sule,1994] Sule, D. R. (1994). Manufacturing Facilities - Location, Planning, and Design. Louisiana: PWS Publishing Company.

[Winston,2003] Wayne L. Winston, M. V. (2003). Introduction to mathematical programming. Pacific Grove: Thomson Books.

Appendix A

Allowing for Rotation

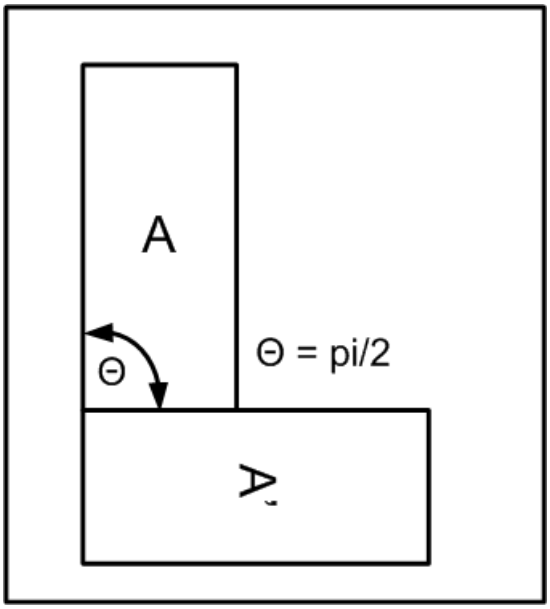

Figure A.1: Rotation of nests

To allow for rotation we will need to define a second set of attributes to each nest. This would be nothing more than inverting the length and width of the nest.

$$l'_i = w_i$$
$$w'_i = l_i$$

We can then use a *if* statement coupled with a $\{0, 1\}$ decision variable to ensure that only one set of values gets used, either $\{l'_i, w'_i\}$ or $\{l_i, w_i\}$.

Appendix B

The Area

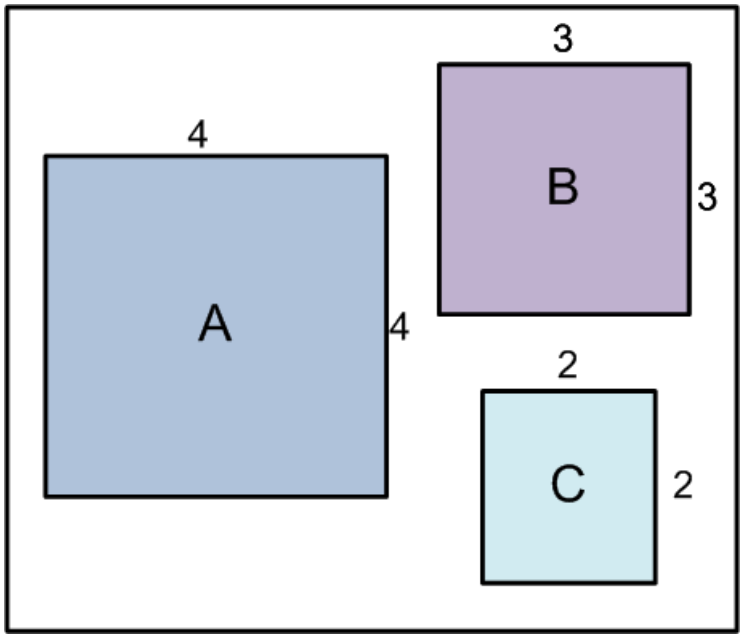

Figure B.1: Sheet and Nest dimensions

We need to be careful when formulation the model as to ensure that the nests actually fit onto the sheets.
If we look at image B.1 we see three squares. If A represents the sheet with area $A_a = 16$, and B and C represents nests with area $A_b = 9$ and $A_c = 4$ respectively, the following situation could arise:

Since $A_b + A_c \leq A_a$, simply imposing the condition that the combined area of the nests needs to be smaller than that of the sheet would not be sufficient.

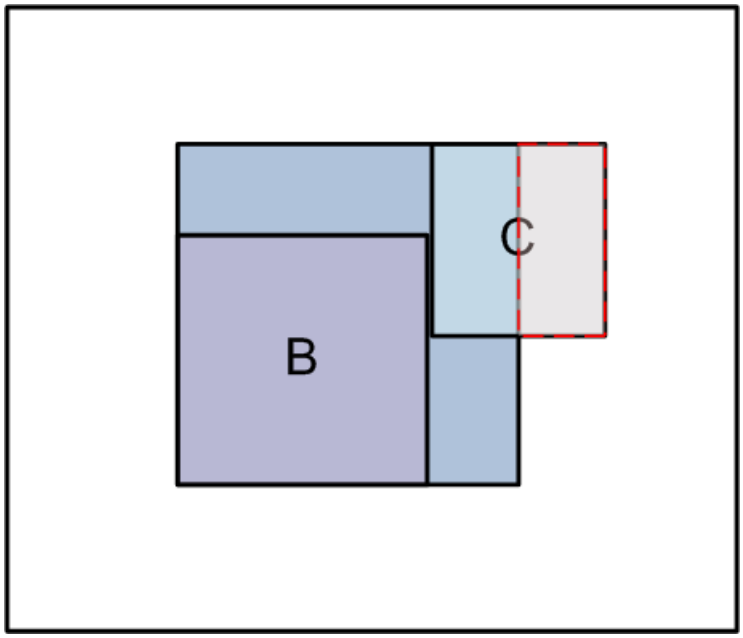

Figure B.2: Nesting Problem

We will need to specifically impose the condition that the combined width as well as the combined length of the nests needs to stay within the available length $\{L_j\}$ and width $\{W_j\}$ of the sheet, working solely with the areas would be problematic.

Appendix C

Output Data

Table C.1: Nesting Using Biggest First

Router 1						
Sheet Number	1	2	3	4	5	6
Sheet Size	6000000	6000000	6000000	6000000	6000000	6000000
Sheet Loading Time	15	15	15	15	15	15
Total Area	4426500	4846400	5600300	5301200	5453900	3021200
Utilisation	0.7378	0.8077	0.9334	0.8835	0.9090	0.5035
Scrap	1573500	1153600	399700	698800	546100	2978800
Programming Time	20	35	61	59	93	92
Scrap Value	204.555	149.968	51.961	90.844	70.993	387.244
Cost	4853.78	4933.37	5074.71	5032.49	5109.01	4791.09
Router 2						
Sheet Number	1	2	3	4	5	6
Sheet Size	5000000	5000000	5000000	5000000	5000000	-
Sheet Loading Time	10	10	10	10	10	-
Total Area	3878300	4547900	4786400	4335000	822600	-
Utilisation	0.7757	0.9096	0.9573	0.8670	0.1645	-
Scrap	1121700	452100	213600	665000	4177400	-
Programming Time	14	35	56	116	24	-
Scrap Value	134.604	54.252	25.632	79.8	501.288	-
Cost	3905.40	4020.75	4084.37	4130.20	3555.38	-

Table C.2: Nesting Using Sequential Order

Router 1						
Sheet Number	1	2	3	4	5	6
Sheet Size	6000000	6000000	6000000	6000000	6000000	6000000
Sheet Loading Time	15	15	15	15	15	15
Total Area	5604900	5401000	4962300	4743600	5320200	2617500
Utilisation	0.9342	0.9002	0.8271	0.7906	0.8867	0.4363
Scrap	395100	599000	1037700	1256400	679800	3382500
Programming Time	94	73	79	43	31	40
Scrap Value	51.363	77.87	134.901	163.332	88.374	439.725
Cost	5181.66	5146.66	5156.66	5096.66	5076.67	5091.66
Router 2						
Sheet Number	1	2	3	4	5	6
Sheet Size	5000000	5000000	5000000	5000000	-	-
Sheet Loading Time	10	10	10	10		
Total Area	4521300	4597600	4481700	4769600	-	-
Utilisation	0.9043	0.9195	0.8963	0.9539	-	-
Scrap	478700	402400	518300	230400	-	-
Programming Time	78	103	33	31	-	-
Scrap Value	57.44	48.28	62.19	27.64	-	-
Cost	4146.66	4188.33	4071.66	4068.33	-	-

Table C.3: Nesting Using Fastest First

Router 1						
Sheet Number	1	2	3	4	5	6
Sheet Size	6000000	6000000	6000000	6000000	6000000	6000000
Sheet Loading Time	15	15	15	15	15	15
Total Area	5437400	5497700	5434200	5512000	5640500	1127700
Utilisation	0.9062	0.9163	0.9057	0.9187	0.9401	0.1880
Scrap	562600	502300	565800	488000	359500	4872300
Programming Time	34	44	61	61	79	123
Scrap Value	73.14	65.30	73.55	63.44	46.74	633.40
Cost	5008.53	5033.03	5053.11	5063.23	5109.93	4596.60
Router 2						
Sheet Number	1	2	3	4	5	6
Sheet Size	5000000	5000000	5000000	5000000	-	-
Sheet Loading Time	10	10	10	10	-	-
Total Area	4521300	4811000	4463300	4574600	-	-
Utilisation	0.9043	0.9622	0.8927	0.9149	-	-
Scrap	478700	189000	536700	425400	-	-
Programming Time	19	78	36	49	-	-
Scrap Value	57.44	22.68	64.40	51.05	-	-
Cost	3990.89	4123.99	4012.26	4047.29	-	-

Table C.4: Utilisation Comparison Summary

	Sheet Utilisation										
Method	1	2	3	4	5	6	7	8	9	10	11
Model	0.986	0.980	0.969	0.957	0.882	0.970	0.961	0.960	0.783	-	-
Biggest Size First	0.738	0.808	0.933	0.884	0.909	0.504	0.776	0.910	0.957	0.867	0.165
Fastest Time First	0.9062	0.9163	0.9057	0.9187	0.9401	0.1880	0.9043	0.9622	0.8927	0.9149	-
Generated Number	0.9342	0.9002	0.8271	0.7906	0.8867	0.4363	0.9043	0.9195	0.8963	0.9539	-

Appendix D

Basic Model Functioning

D.1 Model Development

The initial development of the model in Matlab was only to obtain a simple working Knapsack solver. This solver only requires the value and weight of an item. It then takes the specified capacity and applies the dynamic programming formulation to equate the best selection of items to maximise overall value whilst keeping within the capacity limit.

D.1.1 A Simple Example

The Inputs

Table D.1: Input Values

Item #	Size	Value
1	4	2083
2	8	2842
3	3	1315
4	4	438
5	5	789
6	6	672
7	3	307
8	1	292
9	2	92

Table D.1 specifies the input items used in the example. These items was fed into the model with a specified *capacity of 20.*

The Calculation

The model basically applies the dynamic programming recursion to compute the best value at each stage given the remaining items and limiting capacity. It then writes these values into a matrix. A search function is then called to trace and select the items that would maximise the overall value of the knapsack.

Table D.2: Computational Matrix

Capacity	1	2	3	4	5	6	7	8	9
1.	0	0	0	0	0	0	0	292	292
2.	0	0	0	0	0	0	0	292	292
3.	0	0	1315	1315	1315	1315	1315	1315	1315
4.	2083	2083	2083	2083	2083	2083	2083	2083	2083
5.	2083	2083	2083	2083	2083	2083	2083	2375	2375
6.	2083	2083	2083	2083	2083	2083	2083	2375	2375
7.	2083	2083	3398	3398	3398	3398	3398	3398	3398
8.	2083	2842	3398	3398	3398	3398	3398	3690	3690
9.	2083	2842	3398	3398	3398	3398	3398	3690	3690
10.	2083	2842	3398	3398	3398	3398	3705	3705	3782
11.	2083	2842	4157	4157	4157	4157	4157	4157	4157
12.	2083	4925	4925	4925	4925	4925	4925	4925	4925
13.	2083	4925	4925	4925	4925	4925	4925	5217	5217
14.	2083	4925	4925	4925	4925	4925	4925	5217	5217
15.	2083	4925	6240	6240	6240	6240	6240	6240	6240
16.	2083	4925	6240	6240	6240	6240	6240	6532	6532
17.	2083	4925	6240	6240	6240	6240	6240	6532	6532
18.	2083	4925	6240	6240	6240	6240	6547	6547	6624
19.	2083	4925	6240	6678	6678	6678	6678	6839	6839
20.	2083	4925	6240	6678	7029	7029	7029	7029	7029

The way the search function works is, it starts by looking at value in the bottom right hand corner 7029. This value represents the maximum obtainable value using the input items and specified capacity.

The function then traces this maximum value from right to left in the bottom row until there is a decrease in value, once this happens it looks at

the top of that column and includes the item with corresponding number 5 in the knapsack.

It then deducts the chosen item's *weight* from the total remaining capacity $20 - 5 = 15$ and then repeats this process, only starting from the row representing the remaining capacity 15.

This is repeated until the knapsack's capacity is reached, or the remaining items are larger than the available capacity. The results are displayed in table D.3.

Results

These are the items that would, if selected, maximise the overall value of the knapsack. Their respected contributions with regards to weight and value are shown.

Table D.3: Output Values

Item #	Size	Value
1	4	2083
2	8	2842
3	3	1315
5	5	789
	20	7029