

1-1-2011

# Minimizing makespan of a flexible machine under tooling constraints

Latiful Kabir  
*Ryerson University*

Follow this and additional works at: <http://digitalcommons.ryerson.ca/dissertations>

 Part of the [Mechanical Engineering Commons](#)

---

## Recommended Citation

Kabir, Latiful, "Minimizing makespan of a flexible machine under tooling constraints" (2011). *Theses and dissertations*. Paper 744.

This Thesis is brought to you for free and open access by Digital Commons @ Ryerson. It has been accepted for inclusion in Theses and dissertations by an authorized administrator of Digital Commons @ Ryerson. For more information, please contact [bcameron@ryerson.ca](mailto:bcameron@ryerson.ca).

# MINIMIZING MAKESPAN OF A FLEXIBLE MACHINE UNDER TOOLING CONSTRAINTS

by

Latiful Kabir

Bachelor of Science in Mechanical Engineering  
Rajshahi University of Engineering & Technology (former BIT),  
Bangladesh, 1991

A thesis

Presented to Ryerson University

in partial fulfillment of the  
requirements for the degree of  
Master of Applied Science  
in the program of  
Mechanical Engineering

Toronto, Ontario, Canada, 2011

©Latiful Kabir 2011

## **Author's Declaration**

I hereby declare that I am the sole author of this thesis.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

---

Latiful Kabir

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

---

Latiful Kabir

## **Borrower's Page**

Ryerson University requires the signature of all using or photocopying this thesis. Please sign below, and give address and date.

## **Abstract**

# MINIMIZING MAKESPAN OF A FLEXIBLE MACHINE UNDER TOOLING CONSTRAINTS

A. S. Latiful Kabir

Master of Applied Science, Mechanical Engineering, 2011

Ryerson University

Computer Numerical Control (CNC) milling and lathe machines are widely used in manufacturing due to their flexibility in producing parts with a wide variety of geometries. Each flexible machine has a tool magazine capable of holding a set of tools. As machining requirements for each job change, tools can be removed and different ones can be inserted so that the next job can be processed. The existing literature on the job scheduling and the tool loading can be divided into four main areas. The first area is the tool loading for a pre-specified job sequence where the objective is to determine the optimal tool loading by minimizing the number of tool switching. In addition to tool loading, the second area also focuses on sequencing the jobs too; however, the objective is same as the first one. Rather than to minimizing the number of tool switching, the focal point of the third area has been shifted to minimizing the makespan in presence of multiple process plans. However, the main assumption is that the magazine can hold all tools needed to process all the jobs and tool switching is not required. The fourth area considers the geometric and mechanical properties of the tool, assuming a tool switching may be required due to tool life. The job scheduling and the tool loading literatures do not consider multiple process plans or tool life into their problem. Therefore, the first part of this thesis provides a Dynamic Programming method to determine the optimal makespan for a pre-specified sequence of jobs, assuming tool switching may be required due to multiple process plans, the capacity of the tool magazine and due to the tool life. In the second part, the assumption for fixed job sequence is relaxed and a heuristic approach is used to first sequence the jobs and then Dynamic Programming is applied to find the optimal makespan for that particular job sequence.

## **Acknowledgements**

I am heartily thankful to my co-supervisor, Dr. Vinh Quan, whose encouragement, guidance and support from the initial stages to the final level enabled me to develop a deep understanding of the subject. At the same time, I am deeply indebted to my co-supervisor, Dr. Saeed Zolfaghari, not only for his valued and generous guidance and encouragement in making this thesis happen, but also for his presence in my academic life as a pole star.

Lastly, I offer my regards to all of my friends, family members, librarians and other staff of Ryerson University who extended their great support in all respect during the completion of the thesis.

## Table of Contents

Author's Declaration.....	ii
Borrower's Page.....	iii
Abstract.....	iv
Acknowledgements.....	v
List of Tables .....	ix
List of Figures.....	x
Nomenclature.....	xi
Chapter 1. Introduction and Literature Review .....	1
1.1 Introduction .....	1
1.2 Literature Review.....	2
1.3 Summary .....	6
Chapter 2. Problem Definition and Assumptions.....	8
2.1 Introduction & Problem Definition.....	8
2.2 Model Development and Assumptions .....	8
2.2.1 Tool Requirement by Each Job.....	9
2.2.2 Tool Life .....	10
2.2.3 Tool Setup.....	10
2.2.4 Uniform vs. Non-uniform Tool Setup Time.....	11
2.2.5 Tool Combinations.....	11
2.2.6 Job Processing Time and Process Plan.....	12
2.2.7 Commonality Factor .....	12
Chapter 3. Dynamic Programming Approach .....	14
3.1 Introduction .....	14
3.2 Formulation of DP for Optimal Tool Loading.....	14

3.2.1	Construction of the Network Diagram.....	16
3.2.2	Example of Dynamic Programming .....	17
	Solution .....	18
3.3	Summary .....	26
Chapter 4.	Heuristics for Job Sequencing without Tool Life.....	27
4.1	Introduction .....	27
4.2	Sequencing the Jobs without Tool Life Consideration .....	27
4.2.1	Heuristic 1: Tool Set Common among All Jobs .....	27
4.2.2	Heuristic 2: Tool Set Common in Two or More Jobs but not in All Jobs .....	28
4.2.3	Heuristic 3: No Tool Set Common among the Jobs .....	30
4.3	Example of Job Sequencing without Tool Life Consideration .....	32
	Solutions.....	33
4.3.1	Sequencing the Jobs.....	34
4.3.2	Tool Loading.....	36
4.4	Analysis.....	39
Chapter 5.	Heuristic for Job Sequencing with Tool Life Consideration .....	40
5.1	Introduction .....	40
5.2	Heuristic 4: Sequence the Jobs Based On Total Tool Life for Each Tool Set.....	40
5.3	Example of Job Sequencing with Tool Life Consideration .....	42
	Solutions.....	43
5.3.1	Sequencing the Jobs.....	43
5.3.2	Tool Loading.....	45
5.3.3	Analysis.....	47
Chapter 6.	Concluding Remarks and Future Work .....	48
6.1	Conclusion.....	48

6.2	Contributions.....	48
6.3	Direction for Future Work.....	49
	References.....	50
	Appendix-A.....	53

## List of Tables

Table 2. 1 Tool Switching Time .....	11
Table 3.1 Job Processing & Tool Requirement Data.....	18
Table 3.2 Switching Time between the Tools .....	18
Table 3.3 Tool Requirement Table Modified .....	18
Table 3.4 Calculation of Tool Loading for Job Sequence 1-2-3-4-5 .....	20
Table 3.5 Tools Loading for Job Sequence 1-2-3-4-5 .....	21
Table 3.6 Job Processing & Tool Requirement Data.....	22
Table 3.7 Switching Time between the Tools .....	22
Table 3.8 Tool Life for Each Tool .....	22
Table 3.9 Calculation for Tool Loading with Tool Life .....	24
Table 3.10 Tool Loading for Job Sequence 1-2-3-4-5 with Tool Life .....	26
Table 4.1 Tool Requirements for Each Job .....	31
Table 4.2 Tool Switching Time between Two Tools .....	31
Table 4.3 Requirement of Tool Switch for Each Job.....	32
Table 4.4 Job Processing & Tool Requirement Data.....	32
Table 4.5 Switching Time between the Tools .....	33
Table 4.6 Tool Requirement Table Modified .....	33
Table 4.7 Tool Set to Process All Jobs .....	34
Table 4.8 Usage of Tool Set in Previous Jobs .....	35
Table 4.9 Usage of Tool Set in Previous Jobs .....	36
Table 4.10 Calculation for Tool Loading for Job Sequence 3-1-4-5-2.....	38
Table 4.11 Optimal Tool Loading for Job Sequence 3-1-4-5-2.....	39
Table 5.1 Job Processing & Tool Requirement Data.....	42
Table 5.2 Switching Time between the Tools .....	43
Table 5.3 Tool Life Data.....	43
Table 5.4 Calculation for Tool Loading for Job Sequence 3-1-2-4-5 with Tool Life .....	45
Table 5.5 Tool Loading for Job Sequence 3-1-2-4-5.....	47

## List of Figures

Figure 3.1 Basic Network Diagram .....	16
Figure 3.2 Network Diagram for Job Sequence 1-2-3-4-5 .....	19
Figure 3.3 Network Diagram for Job Sequence 1-2-3-4-5 with Tool Life.....	23
Figure 4.1 Network Diagram for Job Sequence 3-1-4-5-2 .....	37

## Nomenclature

$i, k$	Index for tools, $i, k = 1, 2, \dots, M$
$j, p$	Index for jobs, $j, p = 1, 2, \dots, N$
$C$	Capacity of the tool magazine
$K$	Unused capacity of the tool magazine
$t_l$	Tool loadings to process a single job. This is a set of tools that includes the tools not taking part in processing the job but may be required to fill out the unused capacity of the tool magazine
$t_k$	A specific tool loading to process a particular job and $t_k \in t_l$
$C_j$	Set of tool combinations that can process job $j$ , this is also referred to as multiple process plans and $C_j = \{t_l^j \mid l = 1, 2, \dots, n_j\}$
$C^T$	Total number of possible tool sets and $C^T = \left\{ t_l \mid l = 1, 2, \dots, \binom{M}{C} \right\}$
$t_l^j$	All tool loadings to process $j^{\text{th}}$ job
$t_k^j$	A specific tool loading to process $j^{\text{th}}$ job
$M[t_k^j]$	The optimal makespan up to and including job $j$ for a specific tool loading $t_k$
$p[t_k^j]$	The time to process job $j$ by a specific tool loading $t_k$
$S[t_l^{j-1}, t_k^j]$	Total tool switching time from all tool loadings corresponding to job $j-1$ to the specific tool loading corresponding to job $j$
$X_i$	Tool life, i.e., number of job(s) tool $i$ can process before it wears out
$y_l$	Maximum number of jobs tool loading $t_l$ can process before making a tool switch

due to tool life

$Y_j$	Maximum among all $y_l$ from corresponding tool loadings present in $C_j$
$P_{ij}$	A tool requirement matrix where tool $i$ is required to process job $j$
$t_{ik}$	Tool switching time between two similar or dissimilar tools, this is also referred to as tool setup time
$F_c$	Commonality factors, i.e., tool set common in different tool sets
$x_{jp}$	Set of tools common between two tool sets corresponding to two jobs $j$ and $p$
$N[x_{jp}]$	Number of items contained in the set of $x_{jp}$

# Chapter 1. Introduction and Literature Review

## 1.1 Introduction

Flexible Manufacturing System (FMS), as the name implies, is flexible enough to accommodate quick changes efficiently and effectively. A typical FMS primarily consists of two elements – flexible machine and flexible routing. Under the machine flexibility, a machine is capable of being changed to make new product types and to accommodate new process plans. Due to rapid changes in the market, demand for part-mix has been increasing which has widely been addressed through a flexible manufacturing system in recent time. Although efficient, establishing a FMS is expensive.

For a flexible manufacturing machine, minimizing makespan plays a significant role in improving the productivity and reducing the cost of production. Job scheduling, job processing and tool loading are crucial elements of a makespan, and a proper sequence of jobs along with an intelligent tool loading procedure not only minimizes the makespan, but improves the productivity as well. In a mass production system where a part may be produced in volume, a machine can be designed such that it only requires a tool replacement due to wear out (tool life). In a lean manufacturing system that produces a wide variety of part-mix, where successive parts produced are different, tool replacement may be required due to the capacity constraint of the tool magazine and/or tool life and/or process plan.

It is often assumed that the tool magazine cannot hold all the tools that are needed to process the all the jobs in the makespan and therefore, tool switching must be made. This is called capacity constraint of the tool magazine. It is also assumed that some tools can perform more jobs than others. For example, tool *A* can be used for processing 5 different jobs but tool *B* can only be used for 3. Some tools can process jobs faster than others but have a more limited usage. For example, tool *A* can be used to process jobs 1, 2 and 3 or alternatively, tool *B* can be used. Tool *B* processes jobs 1 and 2 faster than tool *A* but it cannot be used to process job 3 and therefore, a tool switching must be made if tool *B* is used. The idea of having more than one tool e.g. *A* or *B* that can process a given job is referred to as different process plans. In trying to minimize the total time to process all the jobs, one needs to consider the trade off between loading tool *B* to process jobs 1, 2 more quickly and the setup time to make more tool switches.

The total time to process all the jobs that includes the job processing time and tool switching time is called the makespan. For the case where different process plans are not considered, minimizing the number of tool switching will minimize the makespan. Therefore, in much of the existing literature, the objective is to minimize the total number of tool switching. However, if different process plans are considered, then minimizing number of tool switching may not necessarily minimize the makespan. In this case, the processing time for all of the jobs and the tool switching time is to be minimized. Therefore, the tool loading problem may not be solved optimally without seeing it from makespan point of view. In this thesis, the approach is to solve the job scheduling and the tool loading problem focusing on the minimization of the makespan rather than on the minimization of the number of tool switches.

The objective of this thesis is to minimize the makespan of a flexible CNC machine considering tool switches due to the magazine capacity, tool life and multiple process plans for jobs as well. There are two parts in this thesis. In the first part, a tool loading problem is solved to minimize the makespan given a fixed job sequence and the in second part, both the job scheduling and the tool loading problem is solved to minimize the makespan for variable job sequence.

## **1.2 Literature Review**

The job scheduling and the tool loading is an important issue in the manufacturing arena. A survey conducted by Gray *et al.* [1] indicates that the lack of tool management considerations often results in the poor performance. In another paper, Shirazi *et al.* [2] indicate, based on their surveys of seven medium to large manufacturing companies in UK, that none have adopted a formal method to minimize tool switching. Companies mostly rely on random selection of jobs and tools within a block period, although, they are aware of the excessive time spent in set-ups. The literature identifies three different areas where the tool loading problems are highlighted, these areas are:

### **i. Capacity of the tool magazine and its impact on tool loading**

Tool magazine is assumed to have a limited capacity. It cannot hold all the tools to process a number of jobs, thus tool switching occurs. Given a fixed job sequence, the problem is to

determine the optimal tool loading that minimizes the frequencies of tool switching. This loading problem has been investigated extensively in the literature.

## ii. Job scheduling and process planning and its impact on tool loading

Here, the job sequence is no longer assumed to be fixed and the problem is to determine both the sequence of jobs to be processed as well as the loading of tools to minimize the number of tool switch. However, none of the literature considers the impact of the tool loading due to multiple process plans.

## iii. Tool life and its impact on tool loading problem

The main focus has been on determining optimal tool replacement time based on both deterministic tool life and stochastic tool life, quality control of the jobs being made, cost of tool etc.

Much research has been done on the tool loading problem. The focus has been on determining how to load tools into a magazine that cannot hold the tools to process all jobs in the makespan and therefore, tool switches must be made. The time to switch from tool  $A$  to  $B$  may be different than that of the time to switch from  $B$  to  $A$ ;  $A \rightarrow B \neq B \rightarrow A$ . This is called non-uniform setup time. Crama *et al.* [3] have improvised the tool loading problem for a flexible manufacturing machine that has originally been formulated by Tang and Denardo [4]. However, both Tang and Crama have considered a uniform tool setup time, i.e.,  $A \rightarrow B = B \rightarrow A$ , and an indefinite tool life. They found that tool change may be necessary because of the capacity of the magazine. They suggested sequencing jobs in such way to minimize tool switching.

For a fixed job sequence with a uniform tool setup time and indefinite tool life, Tang and Denardo [3] have proposed a rule, known as Keep Tool Needed Soonest (KTNS) policy, to determine the optimal tool loading of a single machine when the capacity of the tool magazine is less than the number of tools needed to process all jobs. Moreover, Tang and Denardo [4] also have showed that job scheduling and tool switching problem is NP-hard<sup>1</sup> for magazine capacity greater than or equal to 2. In a broader sense, when it comes to the question of minimizing the

---

<sup>1</sup> According to computational complexity theory, NP-hard or Non-deterministic Polynomial-time hard is a class of problems that are at least as hard as the hardest problems in NP. For more information, visit <http://www.esi2.us.es/~mbilbao/complexi.htm>

makespan under these tooling constraints, Lawler *et al.* [5] have also showed that the problem is NP-hard too.

Privault *et al.* [6] have introduced a different model considering both uniform and non-uniform tool setup times while applying a network approach to solve the tool switching problem. According to their model, a flow of maximum value (number of tools equal to the capacity of the tool magazine) at a minimum cost (setup time due to tool switch) is moved in an acyclic graph. However, neither the KTNS policy nor the network model has considered multiple process plans and tool life as a reason for tool switching. The network approach, as has been proposed by Privault *et al.* [6], may be modified to accommodate tool life into the model, however, none of the models is capable of accommodating multiple process plans into the tool loading problem.

The combined job scheduling and the tool loading (switching) problem are commonly known as Sequencing and Switching Problem (SSP). Here, the sequence of jobs is not pre-specified (fixed) and therefore, the problem is to determine the sequence of jobs while loading the tools to minimize the number of tool switching. In order to solve the SSP, Laporte *et al.* [7] have proposed two integer linear programming formulations, one being a branch-and-cut and another being a branch-and-bound algorithm. Several heuristics exist in the literature to solve the SSP and a list can be found in Gianpaolo *et al.* [8] who also have developed a branch-and-cut procedure to cast the job scheduling and tool switching problem as a nonlinear Hamiltonian cycle problem. Although efficient in terms of solving more instances (up to 45 jobs and 30 tools as compared to 25 instances by Laporte *et al.*), like Tang and Denardo, the authors have not considered multiple process plans or tool life into their model. Hertz *et al.* [9] have formulated three constructive methods, i.e., FI, GENI and GENIUS; and additionally, the authors have considered nearest neighbour (NN) and 2-opt search too. Amaya *et al.* [10] have proposed a memetic algorithm while considering genetic algorithm with hill climbing procedure. However, neither of the researchers has tried Dynamic Programming to solve the SSP nor have they considered tool life or multiple process plans in their models.

The literature on tool life is rich. It considers various tool wear analysis and tool life models. Taylor [11] is the first to introduce a deterministic model to predict tool life in the metal cutting industry. New tool materials and coating technologies have emerged as a solution to machining process, which significantly reduces the machining time [12]. According to Kalpakjian and

Schmid [12], the cost of tools is often assumed to be only 2-4% of the manufacturing cost. However, some research also suggests that tool related activities in flexible manufacturing systems account for 25% of the operating cost [13]. Gray *et al.* [1] have also supported the findings which indicate that tooling can account for 25-30% of both fixed and variable production costs in automated machining environments. Despite the impact of tool life on the bottom line of the manufacturing cost, tools are often replaced before the end of their useful life [14]. Besides numerous deterministic models to predict tool life, Vagnorius *et al.* [14] have introduced a stochastic model to predict the tool life to determine the optimal replacement time. Although tool life literature is very comprehensive in terms of determining its effect on the manufacturing process as well as predicting the optimal replacement time, there is any research that integrates tool life with the job scheduling and the tool loading problem.

Manufacturing process planning is another area where extensive research has been done on process modeling, selection, planning, optimization and control [15]. ‘Autonomous distributed manufacturing system’ research has been focused on integrating the manufacturing process planning with job scheduling. For instance, Shrestha *et al.* [16] have introduced an integrated approach for generating process plans for a single product scheduling case. This model has later been extended for the case of multiple products [17]. However, this research has not been sufficiently investigated in the literature. The process planning solutions commercially available, such as MetCAPP, KAPES and UGS, has not considered tooling management as a constraint to manufacturing process planning [15]. An interesting piece of research has been carried out by Chang and Chen [18], introducing a dynamic programming based process planning while integrating different parameters of shop-floor activities. However, within the scope of shop-floor activities, the authors do not consider tool management as an input to the process planning problem.

Blazewicz *et al.* [19] have considered machine scheduling problem under resource constraints; however, they have neither considered tool loading nor a tool life constraint at all. Mass customization and shorter product life cycles forced the manufacturers to implement high part mixes and to reduce tool setup [20]. Zhiyang *et al.* [20] have introduced multiple process plans to solve a cutter selection problem on a multi-part milling machine. However, they made an assumption that the magazine is able to hold all the tools needed to process all the jobs and

therefore, tool switching is not required. The model developed in this thesis considers tool switching.

Job scheduling is another highly researched area where the tool loading problems have received due attention. Bard *et al.* [21] have articulated a non-linear integer programming model to solve the job scheduling problem to minimize the number of tool switch. The ultimate result of this problem is to minimize the makespan. The study has assumed that the tool setup is uniform and has developed a dual-based Lagrangian relaxation heuristic to solve the problem [22]. Akturk *et al.* [23] have introduced a scheduling problem considering tool change requirements due to tool wear in a flexible manufacturing system, focusing on the minimization of the total job completion time. The model has been developed for a single tool with assumptions that the tool life and the job processing time are constant and given. However, considering a single tool may not be appropriate for a flexible machine. The model developed in this thesis considers tool life for multiple tools.

The literature on job scheduling under tooling constraints is also of relevance to the thesis. Widmer [24] may be the first to consider minimizing the makespan where the capacity of the tool magazine is a reason for tool switching. However, the author has considered uniform tool setup time and has used the KTNS policy, discussed in Tang *et al.* [4], to determine the optimal tool loading and has adopted a TABU search technique to solve the job scheduling problem. Ecker *et al.* [25] have emphasised on the precedence constraints of each job and have considered non-uniform tool setup times. They have merged two ideas, a dynamic programming approach to solve a job scheduling problem being introduced by Moehring [25], and a concept of the SIT-graph being introduced by Ecker [25]. However, they have not considered multiple process plans or tool life as an input to the model.

### **1.3 Summary**

Based on the above literature survey, the research gaps that have been identified are:

- The job scheduling and the tool loading literature are mostly trying to formulate the tool loading problem in the absence of multiple process plans and therefore, the objective is to minimize the number of tool switching rather than to minimize the makespan. Moreover,

in most of the cases, the tool setup is considered as uniform and the tool life is not incorporated into the tool loading problem.

- Tool life literatures are highly focused on the geometric and mechanical properties of the tool. On the other hand, both the job scheduling and the process planning literature do not consider tool life in their problem. However, the research on tool life has appeared to be very helpful in adding this constraint into the tool loading problem.

This thesis extends the work of Zhiyang *et al.* [20] of multiple process plans to the tool loading problem with a tool magazine's capacity constraint given a fixed job sequence. The inclusion of multiple process plans requires a shift from focusing on minimizing number of tool switch to minimizing the makespan. A novel approach using Dynamic Programming is developed to solve the problem optimally. This thesis also combines the idea of tool life considerations from the tool life literature into the tool loading problem. Finally, for the case where the sequence of jobs is not fixed, several heuristics are developed to first sequence the number of jobs and then the dynamic programming method is applied to determine the optimal makespan. In doing so, this thesis will fill out the gap in the literature and will initiate some new ideas in this area of interest.

The rest of the thesis is organized as follows. In chapter 2, the problem is defined. In Chapter 3, a dynamic programming approach is presented that can be used to solve the tool loading problem for a fixed job sequence to optimally minimize the makespan. In chapters 4 and 5, the sequence of jobs to be processed is not assumed to be fixed and several heuristics are presented which can be used to first sequence the number of jobs. Once the sequence is determined, the dynamic programming approach as has been developed in Chapter 3 is then applied to minimize the makespan. In chapter 6, the conclusion and a recommendation for future research are proposed.

## **Chapter 2. Problem Definition and Assumptions**

### **2.1 Introduction & Problem Definition**

This chapter defines the job scheduling and the tool loading problem of a flexible machine such as a CNC mill or lathe machine. The machine has a tool magazine that holds tools for processing a number of jobs. Each job may be a single part or a batch of the same parts. Without loss of generality, each job is assumed to be one part in this study and so part or job can be used interchangeably. Due to the wide variety of parts made, the tool magazine often cannot hold all the tools required to process all the jobs. In addition, different tooling combinations are available to help build several process plans which make the flexible machine a better choice. On the other hand, each tool is restricted to process a certain number of jobs due to tool wear out (tool life).

If all of the required tools to process all the jobs are present in the tool magazine and the tool life is long enough to process all the jobs, then sequencing the jobs to minimize the makespan is a Travelling Salesman Problem (TSP) [8]. However, in practice, the magazine may have insufficient capacity to hold all the tools needed and that a tool may wear out before all the jobs are processed. In this case, certain tool(s) must be taken out from the magazine to accommodate new tool(s). There may be a scenario within the tool switching problem where alternative sets of tools (multiple process plans) are there to process a particular job. In this case, a set of tools may be faster in terms of processing the job but their use may be limited, causing frequent tool switching which affects the makespan.

The objective of this research is to solve a job scheduling and tool switching problem where the scheduling is to sequence the jobs while loading the right tools into the tool magazine, such that the makespan is minimized subject to multiple process plans, limited magazine capacity and tool life.

### **2.2 Model Development and Assumptions**

In order to solve the problem as described in section 2.1, a deterministic mathematical model has been developed to minimize the makespan of a flexible machine considering the following assumptions which have been discussed in Sections 2.3.1-2.3.7:

- i. All jobs are ready to be processed at time zero. Incoming or outgoing inventory does not have any effect on the jobs being processed and the machine does not break down during processing all the jobs.
- ii. The capacity of the tool magazine is insufficient to hold all the tools to process all the jobs. However, the magazine has sufficient capacity to hold all the tools to process a single job and no tool switch is required during processing a single job.
- iii. The magazine may be fully loaded or empty. However, in the example, the initial setup time is considered as zero.
- iv. The setup time for each tool switching is non-uniform, i.e., switching from tool  $i$  to tool  $k$  is different than switching from  $k$  to  $i$ . If switching from tool  $i$  to tool  $k$  is  $t_{ik}$  and switching from tool  $k$  to tool  $i$  is  $t_{ki}$ , then non-uniform setup time implies  $t_{ik} \neq t_{ki}$ . This thesis considers both manual and automatic tool setup. However, the example in this thesis considers a manual setup with larger variation.
- v. A job may be processed by multiple process plans, i.e., there may be more than one sets of tool to process some or all of the jobs. Among the sets of tool capable of processing a particular job, one set of tools may be faster in processing the job than using other sets. However, a faster tool set may process a smaller variety of jobs.
- vi. The thesis considers a single CNC machine and the process plan for any job refers to a set of tools that processes the jobs. The order of tool sequence is ignored for simplicity.
- vii. Tool will not break during processing a job.
- viii. Each tool has a life shorter than the time to process the all the jobs in the makespan. If tool  $i$  can process  $X_i$  job(s) before it wears out and if there are  $N$  jobs to be processed, then  $X_i < N$ .
- ix. The tool will continue to stay in the magazine unless the next job in the sequence requires that tool to be removed even if the tool life comes to an end during processing the current job.

### **2.2.1 Tool Requirement by Each Job**

Each job requires certain tool(s) to be processed. It may be helpful to define this tool requirement by a matrix. Suppose  $N$  jobs to be processed, one at a time, by a subset of tools. Also suppose there are  $M$  tools to process all the jobs.

The tool requirement data matrix, as has been originally introduced by Tang and Denardo [4], is an  $M \times N$  tool-job matrix  $P$ , with:

$$P_{ij} = \begin{cases} 1 & \text{if job } j \text{ requires tool } i, \\ 0 & \text{otherwise,} \end{cases}$$

For  $i = 1, 2, \dots, M$  and  $j = 1, 2, \dots, N$

### **2.2.2 Tool Life**

The thesis assumes that each tool is capable of processing a certain number of jobs before it wears out and the number of jobs a tool can process is always less than the total number of jobs in the makespan. This parameter may be defined as ‘Tool Life’ and, according to the assumption of this thesis, the tool life is a function of the number of jobs rather than a function of the job processing time by each tool. However, if the time consumes by a tool to process different jobs varies significantly then considering a time-based tool life would be more appropriate. This thesis does not consider a time based tool life analysis.

### **2.2.3 Tool Setup**

Tool switching consists of removing a tool from the tool magazine and inserting a new one into the magazine and the whole activity is considered as ‘Tool Setup’. Due to the assumptions listed in section 2.2, it may be necessary to replace tool(s) from the tool magazine to insert new ones due to any or all of the following causes:

- a) Capacity of the tool magazine – a tool switch is required because the magazine cannot hold all the tools to process all the jobs in the makespan.
- b) Tool life – a tool switch is required to process the next job since its life has come to an end.
- c) Multiple process plans – a tool switch is required because the new tool may process the next job much quicker than using a tool that is currently in the magazine. However, this will only happen when summation of tool setup time and the job processing time by the

new tool set is less than the job processing time by an existing tool set remained in the magazine.

The tool switching time is represented by an  $M \times M$  tool-tool matrix with  $t_{ik}$  equal to the setup time for switching from tool  $i$  to tool  $k$ , where  $i, k=1, 2, \dots, M$ . The tool switching time is shown in the Table 2.2 below.

**Table 2.1: Tool Switching Time**

	1	2	.	.	.	$M$
1	$t_{11}$	$t_{12}$	.	.	.	$t_{1M}$
2	$t_{21}$	$t_{22}$	.	.	.	$t_{2M}$
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
$M$	$t_{M1}$	$t_{M2}$	.	.	.	$t_{MM}$

#### 2.2.4 Uniform vs. Non-uniform Tool Setup Time

Tool switching may occur between two similar or dissimilar tools. The setup time to perform a switch can be assumed to be uniform or non-uniform. For instance, for three tools  $A$ ,  $B$ , and  $C$ , if  $t_{ab}$  is the setup time for switching tools from tool  $A$  to tool  $B$ ,  $t_{ac}$  is the setup time for switching tools from tool  $A$  to tool  $C$  and  $t_{bc}$  is the setup time for switching tools from tool  $B$  to tool  $C$ , then for the uniform setup time,  $t_{ab} = t_{ac} = t_{bc}$  and for the non-uniform setup time,  $t_{ab} \neq t_{ac} \neq t_{bc}$ . Similarly,  $t_{aa}$  represents the setup time for switching tools from tool  $A$  to tool  $A$  due to tool life. The model in this thesis will assume non-uniform setup time since uniform setup time is simply a special case of non-uniform setup time.

#### 2.2.5 Tool Combinations

For a magazine with capacity  $C$  and for  $M$  tools, the maximum possible combinations of tool set to process any job is  $\binom{M}{C}$ . For jobs that require less number of tools than the capacity of the tool magazine, certain tool set is always present within the available tooling combinations with tool

magazine fully loaded. For instance, if there are 5 tools:  $A B C D E$  and the capacity of the tool magazine is 3, then there are  $\binom{5}{3}$  or 10 possible tool sets to process any job, they are:

$$C^T = \{ABC, ABD, ABE, ACD, ACE, ADE, BCD, BCE, BDE, CDE\}$$

The set  $AB$  is always available in  $ABC$ ,  $ABD$ , and  $ABE$  tool sets if any of the jobs requires such tool set.

### 2.2.6 Job Processing Time and Process Plan

Job processing time is the time to process a job by a particular tool-set. If there are  $C_j$  sets of tool to process a particular job  $j$  and if  $t_k^j$  is one of the sets for processing  $j^{th}$  job where  $t_k^j \in C_j$  then there is a specific job processing time for each tool loading and each tool set  $t_k^j$  represents a process plan for a particular job. Due to the flexible nature of the machine, there may be more than one process plan to process a particular job.

### 2.2.7 Commonality Factor

Commonality factor ( $F_c$ ) is the number of tool set(s) common amongst the jobs to be processed. If there are more jobs in the makespan than the number of available tool sets, then there are some jobs that must be processed by tool set(s) common to more than one job. For instance, if there are  $N$  jobs to be processed, one at a time, by any of the tool set from  $C^T$  possible tool sets and if  $N > C^T$ , then some tool set(s) must be common in  $(N - C^T + 1)$  number of jobs. Below are all possible commonality factors:

- a) All tool sets are common to all of the jobs. If  $C_1, C_2, \dots, C_N$  be the tool sets to process corresponding jobs  $1, 2, \dots, N$ , then this commonality factor holds the relationship of  $C_1 = C_2 = \dots = C_N$
- b) Some or at least one tool set is common to all jobs. If  $C_1, C_2, \dots, C_N$  be the tool sets to process corresponding jobs  $1, 2, \dots, N$ , then this commonality factor holds the relationship of  $C_N \subset C_{N-1} \subset \dots \subset C_2 \subset C_1$

- c) Some or at least one tool set is common between any pair of jobs. If  $C_1, C_2, \dots, C_N$  be the tool sets to process corresponding jobs  $1, 2, \dots, N$ , then this commonality factor holds the relationship of  $C_N \subset C_{N-1}$  and  $C_N \subset C_{N-2}$ , however,  $C_{N-1} \not\subset C_{N-2}$
- d) No tool set is common to any of the jobs. If  $C_1, C_2, \dots, C_N$  be the tool sets to process corresponding jobs  $1, 2, \dots, N$ , then this commonality factor holds the relationship of  $C_N \not\subset C_{N-1} \not\subset \dots \not\subset C_2 \not\subset C_1$ . In this case, the number of jobs must be equal to or less than the available tool set.

## Chapter 3. Dynamic Programming Approach

### 3.1 Introduction

Dynamic Programming (DP) is a technique that can be used to solve many types of optimization problems. The technique requires the optimization problem to be cast as a network where the objective is to find the shortest path from the start node to the end node. DP determines the optimal solution by working backward or forward from one end of the network towards the other end through recursion, breaking up a large problem into a series of smaller, more tractable problems. The main contribution of dynamic programming is the *principle of optimality*, as mentioned in Taha [26]. In general, there are five distinctive characteristics of the dynamic programming technique, as presented by Winston [27], they are:

- 1) The problem may be divided into sub-problems of several stages and a decision is required at each stage,
- 2) Each stage consists of number of states,
- 3) The decision obtained at any stage shows how the state at the present stage is transformed into the state at the next stage,
- 4) Principle of optimality - this means the optimal decision for each of the remaining sub-problem must not depend on previously obtained decisions, and
- 5) If the states for the problem is classified into one of the  $T$  sub-problems, there must be a recursion that relates the result obtained during stages  $t, t+1, \dots, T$  to the result obtained from the stages  $t+1, t+2, \dots, T$ .

### 3.2 Formulation of DP for Optimal Tool Loading

In this section, a Dynamic Programming model is formulated to find the optimal tool loading to minimize the makespan while processing a number of jobs for a fixed job sequence.

- The sequence of jobs to be processed represents the *stages* in the network. Since there are  $N$  jobs to be processed, the network will have  $N$  stages.

- The tool sets available for processing any job is denoted as  $C_j$  where  $C_j = \{t_l^j \mid l=1,2,\dots,n_j\}$
- A node (*state*) in the network represents a particular tool loading  $t_k^j$  to process a given job  $j$ .
- The path between two nodes in the network represents the job processing time for job  $j$  by tool loading  $t_k^j$  plus the tool setup time due to tool switching.
- The time to switch from tool  $i$  to tool  $k$  is  $t_{ik}$  and the setup time is non-uniform. This switching time includes switching between two similar tools (due to tool wear out) or switching between two different tools due to magazine constraints or quicker job processing time.

While moving from job  $j-1$  to job  $j$  (stage  $j-1$  to stage  $j$ ) the *decision* is to determine which tool set to load to process the  $j^{\text{th}}$  job from among the available tool sets being used for processing job  $j-1$ .

For a given stage, the following recursive equation gives the optimal makespan of a particular tool loading up to that stage.

$$M [t_k^j] = \text{Min}_{t_l^{j-1}} \left\{ p[t_k^j] + S[t_l^{j-1}, t_k^j] + M [t_k^{j-1}] \right\} \quad (1)$$

Here,

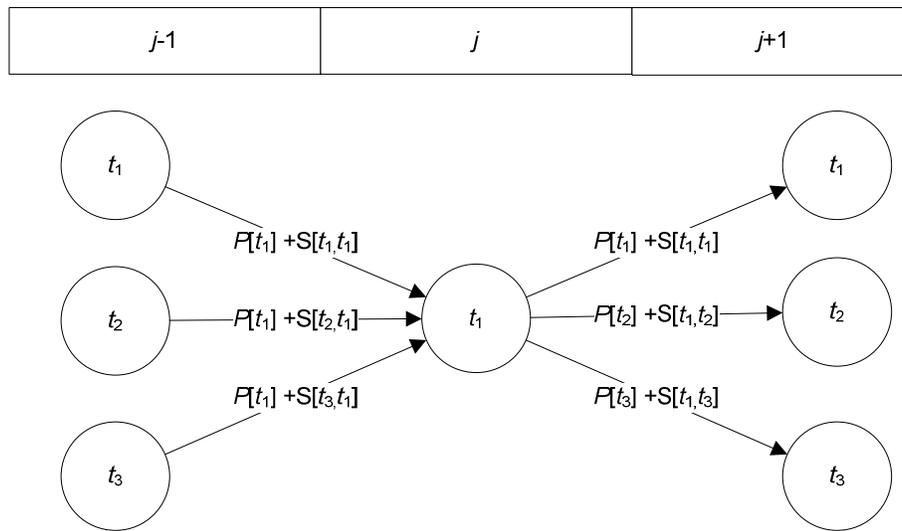
$M[t_k^j]$ , the optimal makespan up to and including job  $j$  for a specific tool loading  $t_k$

$p[t_k^j]$ , the job processing time of job  $j$  by a specific tool loading  $t_k$

$S[t_l^{j-1}, t_k^j]$ , total tool switching time from all tool loadings corresponding to job  $j-1$  to the specific tool loading corresponding to job  $j$

### 3.2.1 Construction of the Network Diagram

Construction of a network diagram is always helpful in visualizing the stages, states and arcs connecting the states. In the network, each job is considered as a stage and the tool loading for processing of that particular job are states. In Figure 3.1 below, a general network diagram is constructed with three jobs for a particular fixed sequence, their corresponding tool loading from three possible tool sets and the connecting paths from one tool loading to another when moving from one job to another.



**Figure 3.1 Basic Network Diagram**

As seen from Figure 3.1, job  $j-1$  can be processed by any of the three tool loadings (process plans),  $t_1$ ,  $t_2$ , and  $t_3$ . Job  $j$  can be processed by one tool loading  $t_1$ . Job  $j+1$  can be processed by three tool loadings,  $t_1$ ,  $t_2$ , and  $t_3$ . When moving from job  $j-1$  to job  $j$ , three tool loadings are available to be transformed into one tool loading to process job  $j$ . However, there is one tool loading available to transform to three tool loadings to process job  $j+1$  when moving from job  $j$  to job  $j+1$ .

Each path connecting two nodes represents processing time plus the tool setup time associated with using that path. For instance, the path connecting nodes  $t_1$  of job  $j-1$  and  $t_1$  of job  $j$  represents a time of which the first part is the processing time of job  $j$  by the tool loading  $t_1$  and the second part is summation of all tool switching times for tool loading transforming from  $t_1^{j-1}$

to  $t_1^j$  and is denoted as  $S[t_1, t_1]$  where the first element within the square bracket is the tool loading for the previous job and the second element is the tool loading for the current job. This tool switching time includes the switching time between two similar tools due to tool life or between two dissimilar tools due to multiple process plans and due to the capacity of the tool magazine.

### 3.2.2 Example of Dynamic Programming

Two numerical examples will be used to illustrate the use of Dynamic Programming to solve the tool loading problem for a fixed job sequence for multiple process plans where tool switch is required. In the first example tool life will not be considered and in the second example, tool life will also be considered as a contributing factor to tool switch.

#### 3.2.2(a) Example of Dynamic Programming without Tool Life Consideration

In this example, the life of each tool is sufficient to process the required number of jobs so that no tool switch is required due to wear out. The objective is to determine the optimal tool loading for a fixed job sequence of 5 jobs to be processed by 3 tools when the magazine can only hold a maximum of 2 tools at a time. The job processing and tool requirements for each job and the tool setup time are shown in the respective tables below:

**Table 3.1: Job Processing & Tool Requirement Data**

Job	Process Plan	Tool Required	Processing Time (min)
1	1	1&2	7
	2	2&3	4
	3	1&3	6
2	1	1&2	9
	2	1&3	8
3	1	2	8
	2	3	6
4	1	3	7
5	1	2&3	6

**Table 3.2: Switching Time between the Tools**

<i>Tool</i>	1	2	3	
$T =$	1	10	12	8
	2	7	11	5
	3	6	3	15

**Solution**

Table 3.1 shows that Job 1 can be processed by any of the three available process plans, they are:

*Plan 1* = tool 1&2 with processing time 7 minutes

*Plan 2* = tool 2&3 with processing time 4 minutes

*Plan 3* = tool 1&3 with processing time 6 minutes

Similar process plan is available for other jobs as well. Table 3.2 shows that the switching from tool 1 to tool 1 requires 10 minutes and the switching from tool 1 to tool 2 requires 12 minutes, etc.

In order to solve the problem, the job processing and tool requirement table needs to be modified to accommodate the tool loading into it so the spare capacity of the tool magazine can be fully utilized. Since the job 3 can be processed either by tool 2 or by tool 3, the requirement of tool 2 can be obtained from either tool loadings (1,2) or (2,3) [ $2 \in \{(1,2), (2,3)\}$ ] and the requirement of tool 3 can be obtained from either tool loadings (2,3) or (1,3) [ $3 \in \{(1,3), (2,3)\}$ ]. Similarly, the tooling requirement of tool 3 for job 4 can be obtained from either tool loadings (1,3) or (2,3) [ $3 \in \{(1,3), (2,3)\}$ ]. The modified table is given below:

**Table 3.3: Tool Requirement Table Modified**

Job	Process Plan	Tool Required	Tool Loading	Processing Time (min)
1	1	1&2	1,2	7
	2	2&3	2,3	4
	3	1&3	1,3	6
2	1	1&2	1,2	9
	2	1&3	1,3	8

Job	Process Plan	Tool Required	Tool Loading	Processing Time (min)
3	1	2	1,2	8
	1	2	2,3	8
	2	3	1,3	6
	2	3	2,3	6
4	1	3	1,3	7
	1	3	2,3	7
5	1	2&3	2,3	6

This example considers the tool loading problem is for the following fixed job sequence:

(1,2,3,4,5)

Based on the job sequence, the following network diagram can be constructed:

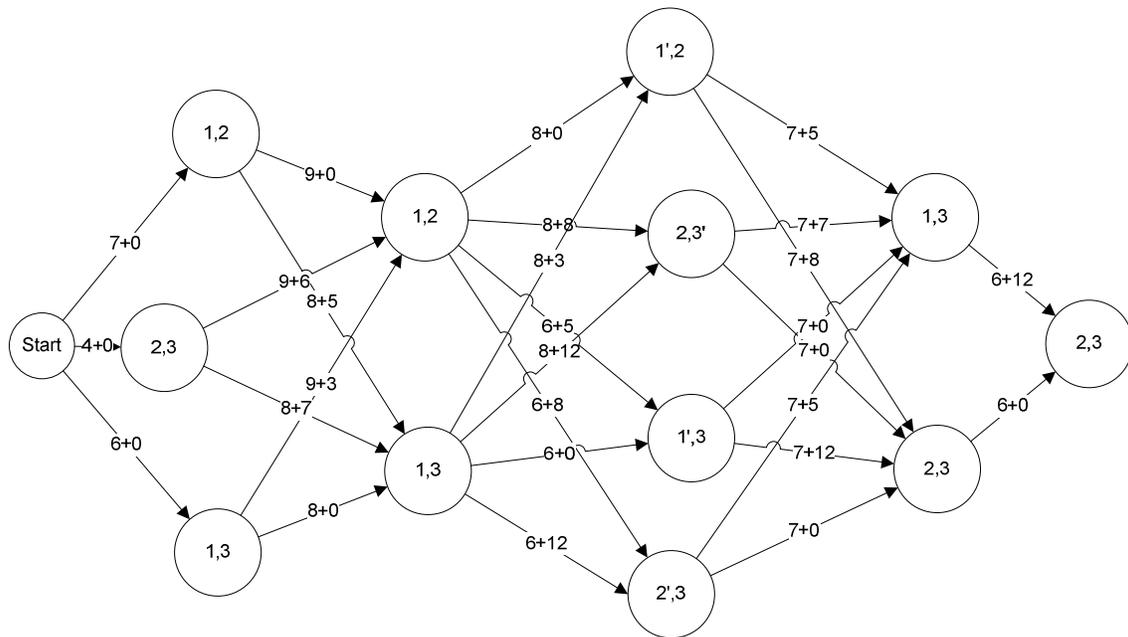
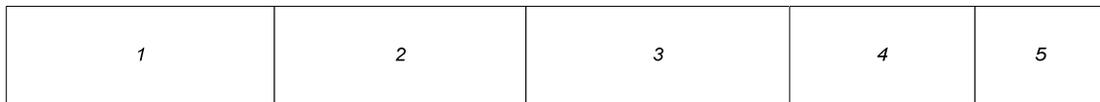


Figure 3.2 Network Diagram for Job Sequence 1-2-3-4-5

As can be seen from the Figure 3.2, for processing job 1, there are three different process plans and for each process plan there are three specific tool loading (states); they are: (1,2), (2,3) and (1,3). While moving from job 1 to job 2, the network looks for any tool loading (states) for job 2 that can be obtained from the previous tool loading for job 1. For instance, tool loading (1,2) for job 2 can be obtained from previous tool loading (1,2) for job 1 without any tool switch, and from (1,3) and (2,3) by switching from tool 3 to 2 or switching from tool 3 to 1. Similarly, tool set (1,3) can be obtained from the previous tool sets either by keeping the same tool sets or by switching the tools. In the network, the idle tools are identified too. Since job 3 can be processed by any of the three tool sets – (1,2), (1,3), (2,3) and since tool set (2,3) has a usage of two; there are four States for Stage 3 - (1',2) (2,3') (2',3) (1',3). Here, the tool with (') represents an idle tool that does not take part in the processing of the job but are used to fill out the spare capacity of the tool magazine. The calculation for the tool loading for each stage using forward recursion is shown below:

**Table 3.4: Calculation of Tool Loading for Job Sequence 1-2-3-4-5**

Job $j$	Tool Loading $t_k^j$	Process Time (1) $p[t_k^j]$	Previous Tool Loading $t_i^{j-1}$	(2) $M[t_i^{j-1}]$	Tool Switch $i > k$	Switch Time (3) $S[t_i^{j-1}, t_k^j]$	Total Time (1)+(2)+(3)	Makespan for corresponding tool loading $M[t_k^j]$
1	<b>1,2</b>	7	-	0	-	0	<b>7</b>	<b>7</b>
	2,3	4	-	0	-	0	4	4
	1,3	6	-	0	-	0	6	6
2	<b>1,2</b>	9	<b>1,2</b>	<b>7</b>	-	0	<b>16</b>	<b>16</b>
	1,2	9	2,3	4	3>1	6	19	
	1,2	9	1,3	6	3>2	3	18	
	1,3	8	1,2	7	2>3	5	20	
	1,3	8	2,3	4	2>1	7	19	
	1,3	8	1,3	6	-	0	14	
3	1,2	8	1,2	16	-	0	24	<b>30</b>
	1,2	8	1,3	14	3>2	3	25	
	2,3	8	1,2	16	1>3	8	32	
	<b>2,3</b>	6	<b>1,2</b>	<b>16</b>	1>3	8	<b>30</b>	
	2,3	6	1,3	14	1>2	12	32	
	1,3	6	1,2	14	2>3	5	25	
	1,3	6	1,3	14	-	0	20	
4	<b>2,3</b>	7	<b>2,3</b>	<b>30</b>	-	0	<b>37</b>	<b>37</b>
	2,3	7	1,3	20	1>2	12	39	
	2,3	7	1,2	24	1>3	8	39	

Job $j$	Tool Loading $t_k^j$	Process Time (1) $p[t_k^j]$	Previous Tool Loading $t_i^{j-1}$	(2) $M[t_i^{j-1}]$	Tool Switch $i > k$	Switch Time (3) $S[t_i^{j-1}, t_k^j]$	Total Time $(1)+(2)+(3)$	Makespan for corresponding tool loading $M[t_k^j]$
	1,3	7	1,3	20	-	0	27	27
5	<b>2,3</b> 2,3	6 6	<b>2,3</b> 1,3	<b>37</b> 27	- 1>2	0 12	<b>43</b> 45	<b>43</b>

Table 3.4 shows that the optimal makespan is 43 minutes for the job sequence {1, 2, 3, 4, 5}. In order to find the corresponding tool loading for each stage, moving backward from stage 5 to up to stage 1 is necessary. For instance, the makespan 43 corresponds to the tool loading of (2,3) in *stage 5*, which has been transformed from a previous tool loading of (2,3) of *stage 4*. The corresponding tool loadings in each stage are shown in bold faces to make it easier to locate.

The optimal tool loading for this particular job sequence is shown in the tool requirements matrix below:

**Table 3.5: Tools Loading for Job Sequence 1-2-3-4-5**

$$P_{ij} = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 1 & 0 & 0 & 0 \\ 2 & 1 & 1 & 1 & 1 & 1 \\ 3 & 0 & 0 & 1 & 1 & 1 \end{array}$$

Table 3.5 shows the optimal tool loading for the fixed job sequence (1,2,3,4,5). However, at this point, it is not known, if the sequence of job itself is optimal or not. This has been addressed in Chapter 4 where several heuristics are proposed to solve the job scheduling problem. Once the job sequence is determined, the same dynamic programming technique as has been used in this example will be applied to determine the optimal makespan.

### 3.2.2(b) Example of Dynamic Programming with Tool Life Consideration

The same example given in 3.3.2(a) is solved here except a deterministic tool life will now be considered where each tool is capable of processing a certain number of jobs before it wears out so that a tool change is required due to the tool life. The job processing & tool requirements for

each job; the tool switching time; and the maximum number of jobs each tool can process before it expires (tool life) are shown in the respective tables below:

**Table 3.6: Job Processing & Tool Requirement Data**

Job	Process Plan	Tool Required	Processing Time (min)
1	1	1&2	7
	2	2&3	4
	3	1&3	6
2	1	1&2	9
	2	1&3	8
3	1	2	8
	2	3	6
4	1	3	7
5	1	2&3	6

**Table 3.7: Switching Time between the Tools**

$$T = \begin{array}{c|ccc} \text{Tool} & 1 & 2 & 3 \\ \hline 1 & 10 & 12 & 8 \\ 2 & 7 & 11 & 5 \\ 3 & 6 & 3 & 15 \end{array}$$

**Table 3.8: Tool Life for Each Tool**

Tool	Maximum no of jobs the tool can process
1	3
2	2
3	1

Table 3.8 shows that tool 1, 2 and 3 can process 3 jobs, 2 jobs and 1 job respectively before wears out. Once any tool reaches that number and if the next job in the sequence requires that tool for processing, then this tool must be replaced by a new tool. Table 3.7 also shows the



tool life (switching from tool 2 to tool 2 and switching from tool 3 to tool 3) because both tools have reached their tool life while processing job 4. However, the first value of each path always represents the processing time. The calculation for each stage using forward recursion method is shown in the table below:

**Table 3.9: Calculation for Tool Loading with Tool Life**

Job $j$	Tool Load $t_k^j$	Process Time (1) $p[t_k^j]$	Previous Tool Loading $t_i^{j-1}$	Tool Use					(2) $M[t_i^{j-1}]$	Tool Switch $i > k$	Switch Time (3) $S[t_i^{j-1}, t_k^j]$	Total Time (1)+(2)+(3)	Makespan for corresponding tool loading $M[t_k^j]$	
				Tool	Current	Up to Previous	Total	Reset on Tool Switch						
1	1,2	7	-	1	1	0	1	1	0	-	0	7	7	
				2	1	0	1	1						
				3	0	0	0	0						
	2,3	4	-	-	1	0	0	0	0	0	-	0	4	4
					2	1	0	1	1					
					3	1	0	1	1					
	1,3	6	-	-	1	1	0	1	1	0	-	0	6	6
					2	0	0	0	0					
					3	1	0	1	1					
2	1,2	9	1,2	1	1	1	2	2	7	-	0	16	16	
				2	1	1	2	2						
				3	0	0	0	0						
	1,2	9	2,3	-	1	1	0	1	1	4	3>1	6	19	
					2	1	1	2	2					
					3	0	1	1	1					
	1,2	8	1,3	-	1	1	1	2	2	6	3>2	3	17	
					2	1	0	1	1					
					3	0	1	1	1					
	1,3	8	1,2	-	1	1	1	2	2	7	2>3	5	20	20
					2	0	1	1	1					
					3	1	0	1	1					
	1,3	8	2,3	-	1	1	0	1	1	4	2>1	7	34	
					2	0	1	1	1					
					3	1	1	2	1					
	1,3	8	1,3	-	1	1	1	2	2	6	-	0	29	
					2	0	0	0	0					
					3	1	1	2	1					
3	1,2	8	1,2	1	1	2	3	3	16	-	0	35	31	
				2	1	2	3	1						
				3	0	0	0	0						
	1,2	8	1,3	-	1	1	2	3	3	20	3>2	3	31	
					2	1	1	2	2					
					3	0	1	1	1					
	2,3	8	1,2	-	1	0	2	2	2	16	1>3	8	43	41
					2	1	2	3	1					
					3	1	0	1	1					

Job	Tool Load	Process Time (1)	Previous Tool Loading	Tool Use					(2)	Tool Switch	Switch Time (3)	Total Time	Makespan for corresponding tool loading	
				Tool	Current	Up to Previous	Total	Reset on Tool Switch						$M[t_i^{j-1}]$
<i>j</i>	2,3	8	1,3	1	0	2	2	2	20	1>2	12	55		
				2	1	1	2	1						
				3	1	1	2	1						
	2,3	6	1,2	1	0	2	2	2	16	1>3	8	41		
				2	1	2	3	1						
				3	1	0	1	1						
	2,3	6	1,3	1	0	2	2	2	14	1>2	12	53		
				2	1	1	2	1						
				3	1	1	2	1						
	1,3	6	1,2	1	1	2	3	3	16	2>3	5	27		27
				2	0	2	2	2						
				3	1	0	1	1						
	1,3	6	1,3	1	1	2	3	3	14	-	0	41		
				2	0	1	1	0						
				3	1	1	2	1						
4	2,3	7	1,2	1	0	3	3	3	31	1>3	8	72	63	
				2	1	2	3	1						
				3	1	1	2	1						
	2,3	7	2,3	1	0	2	2	2	41	-	0	63		
				2	1	1	2	2						
				3	1	1	2	1						
	2,3	7	1,3	1	0	3	3	3	27	1>2	12	72		
				2	1	2	3	1						
				3	1	1	2	1						
	1,3	7	1,2	1	1	3	4	1	31	2>3	5	53		53
				2	0	2	2	2						
				3	1	1	2	1						
	1,3	7	2,3	1	1	2	3	3	41	2>1	7	70		
				2	0	1	1	1						
				3	1	1	2	2						
	1,3	7	1,3	1	1	3	4	1	27	-	0	59		
				2	0	2	2	2						
				3	1	1	2	1						
5	2,3	6	2,3	1	0	2	2	2	63	-	0	95	95	
				2	1	2	3	1						
				3	1	1	2	1						
	2,3	6	1,3	1	0	1	1	1	53	1>2	12	97		
				2	1	2	3	1						
				3	1	1	2	1						

The makespan is 95 minutes and the optimal tool loading for the job sequence is shown in the tool requirement matrix table below:

**Table 3.10: Tool Loading for Job Sequence 1-2-3-4-5 with Tool Life**

$$P_{ij} = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 1 & 0 & 0 & 0 \\ 2 & 1 & 1 & 1 & 1 & 1 \\ 3 & 0 & 0 & 1 & 1 & 1 \end{array}$$

### 3.3 Summary

In this chapter, a dynamic programming approach has been developed to solve two numerical tool loading problems, one without considering the tool life and another with due consideration of the tool life as a contributing factors for tool switch. The dynamic programming approach has successfully accommodated all the assumptions of the problem and has provided an optimal tool loading for a fixed job sequence. However, an optimal tool loading does not necessarily guarantee an optimal makespan, particularly when the job sequence is flexible and a re-sequencing of the jobs may provide a better makespan. In the next two chapters, a heuristic based approach will be used to sequence the jobs first and then the same dynamic programming approach as has been developed here can be applied to obtain a better makespan with the optimal tool loading.

## Chapter 4. Heuristics for Job Sequencing without Tool Life

### 4.1 Introduction

Tang and Denardo [4] have proved that the scheduling of a number of jobs to be processed in conjunction with deciding how tools should be loaded into the magazine is NP-hard [4] for a magazine capacity greater than or equal to 2. In the literature, there are several heuristic based procedures for finding a sub-optimal solution to this problem. However, none of the heuristics have considered multiple process plans or tool life as contributing factors for tool switch. Rather, most have been developed to solely address tool switch due to the capacity of the tool magazine. In this thesis, multiple process plans play a key role in selecting a particular tool set to process a particular job. Keeping this in mind, two sets of heuristics have been formulated for sequencing the jobs. In the first set, three heuristics have been modelled considering multiple process plans and capacity of the tool magazine as factors affecting the makespan. In the second set, one heuristic has been modelled considering multiple process plans, capacity of the tool magazine and tool life as factors affecting the makespan.

### 4.2 Sequencing the Jobs without Tool Life Consideration

As explained in Section 2.2.7, there may be different commonality factors ( $F_c$ ) among different tool sets for processing different jobs. The following three heuristics are developed based on this concept of commonality factors. The heuristics assume tool life will not play a role in tool switch and tool setup time can either be uniform or non-uniform.

#### 4.2.1 Heuristic 1: Tool Set Common among All Jobs

Jobs can be sequenced in any order if there is at least one tool set common among all the tool sets to process all the jobs. The idea behind this heuristic is eliminate tool switches due to the capacity of the tool magazine.

Let  $C_1, C_2, \dots, C_N$  be the tool sets to process corresponding jobs  $1, 2, \dots, N$ , such that  $C_N \subseteq C_{N-1} \subseteq \dots \subseteq C_1$ , then the tool set to process  $N^{\text{th}}$  job is common for all of the jobs. Therefore tool set for  $N^{\text{th}}$  job can be used to process all of the jobs including the  $N^{\text{th}}$  job, and no

tool switch is required due to the capacity constraint of the tool magazine. In such case, jobs  $(1,2,\dots,N)$  can be sequenced in any order.

**Illustration:**

For 5 tools,  $A B C D E$ , and with magazine capacity 3, there are  $\binom{5}{3}$  or 10 possible tool sets, they are:

$$C^T = \{ABC, ABD, ABE, ACD, ACE, ADE, BCD, BCE, BDE, CED\}$$

Suppose there are  $N$  jobs to be processed by the available tool sets. Also suppose,  $C_1, C_2, \dots, C_N$  be the number of tool sets to process corresponding jobs  $1, 2, \dots, N$ , where:

$$C_1 = \{ABC, ABD, ABE, ACD, ACE, ADE, BCD, BCE, BDE, CED\}$$

$$C_2 = \{ABC, ABD, ABE, ACD, ACE, ADE, BCD, BCE, BDE\}$$

.

.

$$C_{N-1} = \{ABC, ABD\}$$

$$C_N = \{ABC\}$$

This shows that  $C_N \subset C_{N-1} \subset \dots \subset C_1$ . Thus, tool set  $ABC$  is common to all of the tool sets to process corresponding jobs. Therefore, loading tool set  $ABC$  to process all the jobs will not require a tool switch due to the capacity of the tool magazine and the jobs can be sequenced in any order.

**4.2.2 Heuristic 2: Tool Set Common in Two or More Jobs but not in All Jobs**

Starting from the job being processed by maximum tool sets, the rest of the jobs can be sequenced according to the decreasing order of the number of commonality factors of the tool sets among successive jobs. The idea behind this heuristic is to keep two jobs together that most likely will be processed by the same set of tools.

Let  $C_{j-1}, C_j, C_{j+1}, C_{j+2}$  be the tool sets to process corresponding jobs  $J-1, J, J+1, J+2$ , such that number of items contained in  $C_{j-1}$  is greater than the number of items contained in any other tool sets to process other jobs. In that case, job  $J-1$  will be the first job in the sequence. Suppose,  $x_{j-1,j} = C_{j-1} \cap C_j, x_{j-1,j+1} = C_{j-1} \cap C_{j+1}$  and  $x_{j-1,j+2} = C_{j-1} \cap C_{j+2}$ . Also  $N[x_{j-1,j}]$  is the number of items common between two sets  $C_{j-1}$  and  $C_j$ ,  $N[x_{j-1,j+1}]$  is the number items common between  $C_{j-1}$  and  $C_{j+1}$ , and  $N[x_{j-1,j+2}]$  is the number of items common between  $C_{j-1}$  and  $C_{j+2}$ . If  $N[x_{j-1,j}] > N[x_{j-1,j+1}] > N[x_{j-1,j+2}]$ , then the next job in the sequence is  $J$  and no tool switch is required between these two jobs. The rest of the jobs will follow the same steps to be sequenced.

### Tie Braking

If there is a tie such that  $N[x_{j-1,j}] = N[x_{j-1,j+1}]$ , then the two jobs, job  $J$  and job  $J+2$ , can be sequenced in either order depending on their relative relationship with previous jobs and that the job with higher tool sets being used more in previous job(s) will get the preference over the tool sets being used less. There are two steps to break the tie, they are:

Step 1- To find the tool set that is used most in previous jobs

Step 2- To check if the most used tool set is used in tied jobs. The job with most used tool set will get the preference over the other job

### Illustration:

For 5 tools,  $A B C D E$ , and with magazine capacity 3, there are  $\binom{5}{3}$  or 10 possible tool sets; they are:

$$C^T = \{ABC, ABD, ABE, ACD, ACE, ADE, BCD, BCE, BDE, CED\}$$

Suppose there are 4 jobs to be processed, one at a time, by the available tool sets. Also suppose,  $C_1, C_2, \dots, C_4$  be the tool sets to process corresponding jobs (1, 2, 3, 4), where:

$$C_1 = \{ABC, ABD, ABE, ACD, BDE, CED\}$$

$$C_2 = \{ABC, ABD, ABE, ADE, CED\}$$

$$C_3 = \{ABC, ABD, ABE, ADE, BCD\}$$

$$C_4 = \{ACD, ADE, BCD, BCE\}$$

Since  $C_1$  contains more items than any other tool sets corresponding to other jobs; the starting job will be 1. The next job in the sequence is determined based on the commonality factors between job 1 and other jobs. From the tool sets, the set of common tool sets are:

$$x_{12} = \{C_1 \cap C_2\} = \{ABC, ABD, ABE, CED\}$$

$$x_{13} = \{C_1 \cap C_3\} = \{ABC, ABD, ABE\}$$

$$x_{14} = \{C_1 \cap C_4\} = \{ACD\}$$

Since  $N[x_{12}] = 4$ ,  $N[x_{13}] = 3$  and  $N[x_{14}] = 1$ , the next job in the sequence is job 2 and the jobs can be sequenced as (1,2) Again, the next job in the sequence is to be determined based on the commonality factors between job 2 and other jobs except job 1 since the job 1 already has secured the first position. From the tool sets, the set of common tool sets are:

$$x_{23} = \{C_2 \cap C_3\} = \{ABC, ABD, ABE, ADE\}$$

$$x_{24} = \{C_2 \cap C_4\} = \{none\}$$

Since  $N[x_{23}] = 4$  and  $N[x_{24}] = 0$ , the next job in the sequence is 3 and since job 4 the only job that is left, the jobs can be sequenced as (1,2,3,4)

### 4.2.3 Heuristic 3: No Tool Set Common among the Jobs

Sequence the jobs in increasing order of tool switching time for corresponding jobs if no tool set is common among all the jobs. There are  $N$  jobs to be processed, one at a time. Suppose  $C_1, C_2, \dots, C_N$  are the tool sets to process corresponding jobs 1,2,...,N and  $C_N \not\subset C_{N-1} \not\subset \dots \not\subset C_1$ . If  $t_1, t_2, \dots, t_n$  are the minimum tool switching times for switching from other jobs to the corresponding jobs 1,2,...,N, such that  $t_1 < t_2 < \dots < t_n$ ; then sequence jobs as (1,2,...,N).

**Illustration:**

Suppose there are 4 tools  $A B C D$  and capacity of the magazine is 2, then there will be 6 possible tool sets to process a maximum of 6 jobs since no tool set is common to any of the jobs. The possible sets are:  $\{AB, AC, AD, BC, BD, \text{ and } CD\}$ . The tool-job matrix below shows the tool requirement for each of the jobs:

**Table 4.1: Tool Requirements for Each Job**

	1	2	3	4	5	6
$A$	1	1	1	0	0	0
$B$	1	0	0	1	1	0
$C$	0	1	0	1	0	1
$D$	0	0	1	0	1	1

The Table 4.1 shows that the job 1 requires tools  $A$  &  $B$ , job 2 requires tools  $A$  &  $C$  and so on. The Table 4.2 shows the tool switching time between two tools. Since the tool life does not affect the tool switch, the table does not hold any value for switching between two similar tools.

**Table 4.2: Tool Switching Time between Two Tools**

	$A$	$B$	$C$	$D$
$A$	–	$t_{ab}$	$t_{ac}$	$t_{ad}$
$B$	$t_{ba}$	–	$t_{bc}$	$t_{bd}$
$C$	$t_{ca}$	$t_{cb}$	–	$t_{cd}$
$D$	$t_{da}$	$t_{db}$	$t_{dc}$	–

Assuming that the switching time between two tools always maintains the following relationships:  $t_{ab} < t_{ac} < t_{ad} < t_{ba} < t_{bc} < t_{bd} < t_{ca} < t_{cb} < t_{cd} < t_{da} < t_{db} < t_{dc}$ . For 6 jobs and 4 tools, Table 4.3 shows the tool switching requirement from one job to another:

**Table 4.3: Requirement of Tool Switch for Each Job**

<i>Job</i>		1	2	3	4	5	6	
	<i>Combo</i>	<i>AB</i>	<i>AC</i>	<i>AD</i>	<i>BC</i>	<i>BD</i>	<i>CD</i>	<i>Min</i>
1	<i>AB</i>	–	$t_{bc}$	$t_{bd}$	$t_{ac}$	$t_{ad}$	$t_{ac} + t_{bd}$	$t_{ac}$
2	<i>AC</i>	$t_{cb}$	–	$t_{cd}$	$t_{ab}$	$t_{ab} + t_{cd}$	$t_{ad}$	$t_{ad}$
3	<i>AD</i>	$t_{db}$	$t_{dc}$	–	$t_{ab} + t_{dc}$	$t_{ab}$	$t_{ac}$	$t_{ab}$
4	<i>BC</i>	$t_{ca}$	$t_{ba}$	$t_{ba} + t_{cd}$	–	$t_{cd}$	$t_{bd}$	$t_{ba}$
5	<i>BD</i>	$t_{da}$	$t_{ba} + t_{dc}$	$t_{ba}$	$t_{dc}$	–	$t_{bc}$	$t_{ba}$
6	<i>CD</i>	$t_{ca} + t_{db}$	$t_{da}$	$t_{ca}$	$t_{db}$	$t_{cb}$	–	$t_{ca}$

If the job starts from 1, then the next minimum tool switch time from job 1 is  $t_{ac}$  which corresponds to job 4, then from job 4, the minimum tool switch time is  $t_{ba}$  which corresponds to job 2. Based on this, the job sequence (1,4,2,6,3,5) is obtained.

### 4.3 Example of Job Sequencing without Tool Life Consideration

The problem described in 3.3.2(a) will be solved here. However, rather than assuming the job sequence is pre-specified and fixed as (1,2,3,4,5), the set of heuristics described in Sections 4.2 will be applied to sequence the jobs. Once a sequence is determined, the Dynamic Programming technique as has been developed in Chapter 3 will be applied to this sequence to find the optimal tool loading.

The problem assumes each tool is capable of processing a sufficient number of jobs so that no tool switching is required due to tool life. The objective is to minimize the makespan while determining the sequence a set of 5 jobs and loading 3 tools into a magazine that can hold a maximum of 2 tools. The job processing and tool requirements for each job and the tool switching time are shown in the respective tables below:

**Table 4.4: Job Processing & Tool Requirement Data**

Job	Process Plan	Tool Required	Processing Time (min)
1	1	1&2	7
	2	2&3	4
	3	1&3	6

Job	Process Plan	Tool Required	Processing Time (min)
2	1	1&2	9
	2	1&3	8
3	1	2	8
	2	3	6
4	1	3	7
5	1	2&3	6

**Table 4.5: Switching Time between the Tools**

<i>Tool</i>	1	2	3
1	10	12	8
2	7	11	5
3	6	3	15

## Solutions

Similar approach has been followed as has been used to solve 3.3.2(a) except that the sequencing the jobs is done first based on the heuristic presented in Section 4.2 and then the dynamic programming technique is applied to solve the tool loading problem once a sequence is found by the set of heuristics. Since details regarding the modification of the tool requirement table have already been explained in Section 3.3.2(a), it is not repeated here. The modified table is:

**Table 4.6: Tool Requirement Table Modified**

Job	Process Plan	Tool Required	Tool Loading	Processing Time (min)
1	1	1&2	1,2	7
	2	2&3	2,3	4
	3	1&3	1,3	6
2	1	1&2	1,2	9
	2	1&3	1,3	8
3	1	2	1,2	8
	1	2	2,3	8
	2	3	1,3	6

Job	Process Plan	Tool Required	Tool Loading	Processing Time (min)
	2	3	2,3	6
4	1	3	1,3	7
	1	3	2,3	7
5	1	2&3	2,3	6

### 4.3.1 Sequencing the Jobs

Since there are more jobs than the available tool sets (5 jobs and 3 tool sets), therefore, heuristic 4.2.3 does not apply. On the other hand, tool set (1,2) is common to jobs 1, 2 & 3. Tool set (1,3) is common to jobs 1, 2, 3 & 4. Tool set (2,3) is common to jobs 1, 3, 4 & 5. Since there is not a single tool set that is common to all of the jobs; heuristic 4.2.1 cannot be applied. Therefore, 4.2.2 can be applied to sequence the jobs. The table below shows the tool sets for processing all the jobs:

**Table 4.7: Tool Sets to Process All Jobs**

$t_i$	1	2	3	4	5
(1,2)	1	1	1	0	0
(1,3)	1	1	1	1	0
(2,3)	1	0	2	1	1
Total	3	2	4	2	1

Table 4.7 shows that job 3 can be processed by a maximum of 4 tool sets followed by job 1. Therefore, the starting job should be job 3. The next job in the sequence can be determined based on the commonality factors between the job 3 and other jobs. The commonality factors as obtained from the tool sets are:

$$\begin{aligned}
 x_{31} &= \{(1,2), (1,3), (2,3)\} & \text{and } N[x_{31}] &= 3 \\
 x_{32} &= \{(1,2), (1,3)\} & \text{and } N[x_{32}] &= 2 \\
 x_{34} &= \{(1,3), (2,3)\} & \text{and } N[x_{34}] &= 2 \\
 x_{35} &= (2,3) & \text{and } N[x_{35}] &= 1
 \end{aligned}$$

Since,  $N[x_{31}] > N[x_{32}] = N[x_{34}] > N[x_{35}]$ , the next job in the sequence after job 3 is job 1. The next job in the sequence can be determined based on the commonality factors between job 1 and rest of the jobs:

$$\begin{aligned} x_{12} &= \{(1,2) (1,3)\} && \text{and } N[x_{12}] = 2 \\ x_{14} &= \{(1,3), (2,3)\} && \text{and } N[x_{14}] = 2 \\ x_{15} &= (2,3) && \text{and } N[x_{15}] = 1 \end{aligned}$$

Since,  $N[x_{12}] = N[x_{14}]$ ; applying the tie breaking rule is necessary to break the tie.

### **Tie Breaking**

#### **Step 1- To find the tool set that is used most in previous jobs**

Following table helps determine the usage of tool sets in previous jobs:

**Table 4.8: Usage of Tool Sets in Previous Jobs**

<i>Combo</i>	<i>job3</i>	<i>job1</i>	<i>Total</i>
(1,2)	1	1	2
(1,3)	1	1	2
(2,3)	2	1	3

Thus, tool set (2,3) is the most used tool set.

#### **Step 2- To check if the most used tool set is used in tied jobs**

Since (2,3) is the most used tool set, the solution will now check if this tool set has been used in those tied jobs. As seen, the tool set (2,3) will be used by job 4 and is a common item for  $x_{14}$ . Therefore, the next job in the sequence will be 4. The next job in the sequence can be determined based on the commonality factors between job 4 and rest of the jobs, as:

$$x_{42} = (1,3) \quad \text{and } N[x_{42}] = 1$$

$$x_{45} = (2,3) \quad \text{and } N[x_{45}] = 1$$

Since,  $N[x_{42}] = N[x_{45}]$ ; applying the tie break rule is necessary to break the tie.

## Tie Breaking

### Step 1- To find the tool set that is used most in previous jobs

The following table helps determine the usage of tool sets in previous jobs:

**Table 4.9: Usage of Tool Sets in Previous Jobs**

<i>Combo</i>	<i>job3</i>	<i>job1</i>	<i>job4</i>	<i>Total</i>
(1,2)	1	1	0	2
(1,3)	1	1	1	3
(2,3)	2	1	1	4

Thus, tool set (2,3) is the most used tool set.

### Step 2- To check if the most used tool set is used in tied jobs

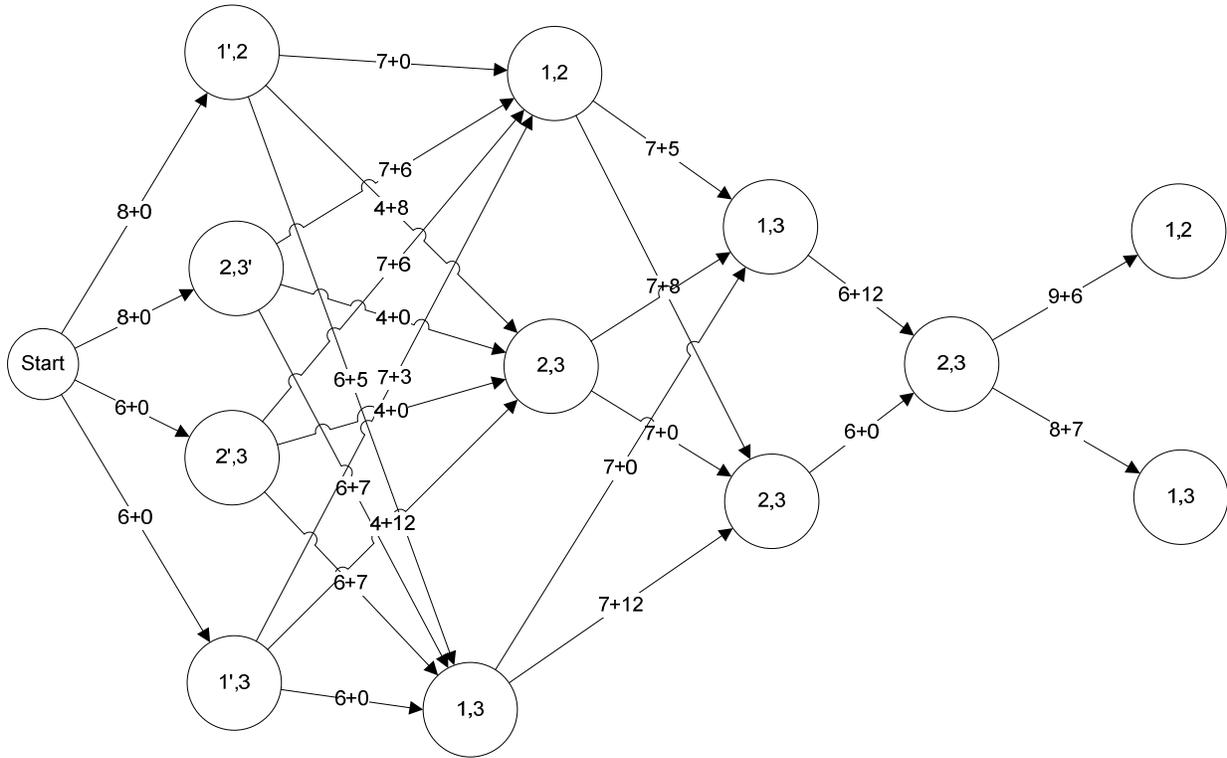
Since (2,3) is the most used tool set, the solution will now check if this tool set has been used in those tied jobs. As seen, the tool set (2,3) will be used by job 5 and is a common item for  $x_{45}$ . Therefore, the next job in the sequence will be 5. Since, there is only one job left, the final job sequence is

$$(3,1,4,5,2)$$

## 4.3.2 Tool Loading

In order to solve the tool loading problem for the job sequence obtained in section 4.3.1, the Dynamic Programming technique as has been developed in Chapter 3 can be applied. The network diagram based on the job sequence (3,1,4,5,2) is shown below:

3	1	4	5	2
---	---	---	---	---



**Figure 4.1: Network Diagram for Job Sequence 3-1-4-5-2**

In the Figure 4.1, a network diagram is drawn with a starting node followed by four nodes representing each *state* of job 3 and each line connecting the starting node to four nodes represent respective times to reach to those nodes. Inside each node, there are two numbers that corresponds to the tool loading. The tool with (') represents a dummy tool that does not take part in the job processing but occupies the magazine. Each line connecting two nodes holds two values; the first one being the job processing time by the given tool set and the second one being the tool setup time if the tool loading requires a tool switch from the previous job to the current job. For instance, there are three paths from each node at 3 to go to each node at 1 and moving from the tool loading (1',2) of job 3 to the tool loading (1,2) for job 1 does not require a tool switch, but moving from the tool loading (1',2) of job 3 to the tool loading (2,3) for job 1 requires a tool switch from tool 1 to 3. Therefore, the path connecting (1',2) to (1,2) holds the value of the job processing time of 7 minutes by the tool loading (1,2) for the job 1 plus a zero for the tool switching time. The path connecting (1',2) to (2,3) holds the value of the job processing time of

4 minutes by the tool loading (2,3) for the job 1 plus the tool switching time of 8 minutes due to tool switch from tool 1 to 3. The calculation using forward recursion for each stage is shown below:

**Table 4.10: Calculation for Tool Loading for Job Sequence 3-1-4-5-2**

Job	Tool Loading	Process Time (1)	Previous Tool Loading	(2)	Tool Switch (3)	Switch Time	Total Time	Makespan for corresponding tool loading	
$j$	$t_k^j$	$p[t_k^j]$	$t_l^{j-1}$	$M[t_l^{j-1}]$	$i>k$	$S[t_l^{j-1}, t_k^j]$	$(1)+(2)+(3)$	$M[t_k^j]$	
3	2,3	8	-	0	-	0	8	<b>6</b>	
	<b>2,3</b>	<b>6</b>	-	0	-	0	<b>6</b>		
	1,2	8	-	0	-	0	8		
	1,3	6	-	0	-	0	6		
1	<b>2,3</b>	4	<b>2,3</b>	<b>6</b>	-	0	<b>10</b>	<b>10</b>	
	2,3	4	1,2	8	1>3	8	20		
	2,3	4	1,3	6	1>2	12	22		
	1,2	7	2,3	6	3>1	6	19		15
	1,2	7	1,2	8	-	0	15		
	1,2	7	1,3	6	3>2	3	16		
	1,3	6	2,3	6	2>1	7	19		12
	1,3	6	1,2	8	2>3	5	19		
	1,3	6	1,3	6	-	0	12		
4	<b>2,3</b>	7	<b>2,3</b>	<b>10</b>	-	0	<b>17</b>	<b>17</b>	
	2,3	7	1,2	15	1>3	8	30		
	2,3	7	1,3	12	1>2	12	31		
	1,3	7	2,3	10	2>1	7	24		19
	1,3	7	1,2	15	2>3	5	27		
	1,3	7	1,3	12	-	0	19		
5	<b>2,3</b>	6	<b>2,3</b>	<b>17</b>	-	0	<b>23</b>	<b>23</b>	
	2,3	6	1,3	19	1>2	12	37		
2	<b>1,2</b>	9	<b>2,3</b>	<b>23</b>	3>1	6	38	<b>38</b>	
	1,3	8	2,3	23	2>1	7	38		

For the job sequence (3,1,4,5,2), the minimum processing time for all 5 jobs is 38 minutes with the following tool loading:

**Table 4.11: Optimal Tool Loading for Job Sequence 3-1-4-5-2**

$$P_{ij} = \begin{array}{c|ccccc} & 3 & 1 & 4 & 5 & 2 \\ \hline & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{array}$$

#### 4.4 Analysis

In example 3.3.2(a), the makespan was 43 minutes with job sequence (1-2-3-4-5) and the corresponding tool loading was also different. After applying the heuristic, a better job sequence (3-1-4-5-2) is determined and the makespan has been reduced to 38 minutes. However, it has been seen in both cases that there is only one tool switch. For instance, during the optimal tool loading for the job sequence (1-2-3-4-5), the tool switch has happened while moving from job 2 to job 3 (tool 1 to tool 3) and during the optimal tool loading for the job sequence (3-1-4-5-2), the tool switch has happened while moving from job 5 to job 2 (tool 3 to tool 1). This also proves that minimizing number of tool switch may not necessarily minimize the makespan.

Heuristics in general do not guarantee an optimal solution. In order to determine the performance of the heuristic, the dynamic programming model has been run for all 120 possible ways of scheduling the 5 jobs. The results indicate there are 3 optimal solutions shown below:

4-2-1-3-5 and the makespan is 35 minutes

4-2-1-5-3 and the makespan is 35 minutes

4-3-2-1-5 and the makespan is 35 minutes

There are 37 different sequences for the jobs where the makespan had the same value as the one obtained using the heuristic. The upper bound (worst) makespan was found to be 51 minutes. The summary of the 120-run is shown in “Appendix-A”.

## Chapter 5. Heuristic for Job Sequencing with Tool Life Consideration

### 5.1 Introduction

It is assumed the number of jobs a tool can process before it expires is less than the number of jobs in the set and so tool switch may be required due to end of tool life. Based on this assumption, one heuristic will be presented in this section. It is assumed that each tool follows a deterministic life as has been initially modeled by Taylor [11] and has been improvised by subsequent researchers. However, an average or an expected tool life data may be obtained from a probabilistic model as well and may also be used in this heuristic.

### 5.2 Heuristic 4: Sequence the Jobs Based On Total Tool Life for Each Tool Set

Sequence the jobs in decreasing order of the total tool life of corresponding tool set. The assumption is that the number of times a tool appears in different tool set for processing different jobs is more than the number of jobs the tool can process (tool life).

Suppose, there are  $N$  jobs to be processed, one at a time, by  $M$  number of tools and a magazine has a capacity to hold  $C$  tools, where  $C \leq M$ , and further suppose,  $C_1, C_2, \dots, C_N$  be the tool sets to process corresponding jobs  $(1, 2, \dots, N)$  Also suppose,

$X_i$  = number of jobs tool  $i$  ( $i = 1, 2, \dots, M$ ) can process before wears out

$t_l$  = tool loading for processing of any job

$t_l^j$  = tool loading for processing of  $j^{\text{th}}$  job, where  $j=1, 2, \dots, N$

$C_j$  = Set of tool combinations that can process job  $j$ , and  $C_j = \{t_l^j \mid l = 1, 2, \dots, n_j\}$

$y_l$  = number of jobs a tool set  $t_l$  can process before making a tool switch due to tool life. If  $t_l$  consists of  $(1, 2, \dots, M)$ , then  $y_l = \min\{X_1, X_2, \dots, X_M\}$ .

$Y_j$  = maximum among all  $y_l$  from corresponding tool loadings present in  $C_j$

If  $Y_{j-1}$  corresponds to job  $J-1$ ,  $Y_j$  corresponds to job  $J$  and  $Y_{j+1}$  corresponds to job  $J+1$  then sequence the jobs as  $(J-1, J, J+1)$  if  $Y_{j-1} \geq Y_j \geq Y_{j+1}$ .

### Tie Breaking Rule

If there is a tie such that  $Y_{j-1} = Y_j = Y_{j+1}$ , then apply heuristic 4.2.2 or 4.2.3 to sequence the jobs being tied depending on the situation.

### Illustration:

For 5 tools,  $A B C D E$ , and with magazine capacity 3, there are  $\binom{5}{3}$  or 10 possible tool sets, they are :  $\{ ABC, ABD, ABE, ACD, ACE, ADE, BCD, BCE, BDE, CED \}$ . Suppose there are 4 jobs to be processed by the available tool sets and  $C_1, C_2, \dots, C_4$  be the set of tools to process corresponding jobs (1,2,3,4), where:

$$C_1 = \{ABC, ABD, ABE, ACD\}$$

$$C_2 = \{ABE, ACD, ACE, ADE\}$$

$$C_3 = \{ACE, ADE, BCD, BCE\}$$

$$C_4 = \{BDE, CDE\}$$

Also suppose,  $X_1, X_2, X_3, X_4, X_5$  are the maximum number of jobs the tool  $A, B, C, D, E$  can process before wares out and  $X_1 > X_2 > X_3 > X_4 > X_5$ . Since, there are 4 tool sets to process job 1, therefore:

$$t_1^1 = \{A, B, C\}. \text{ Thus, } y_1 = X_3$$

$$t_2^1 = \{A, B, D\}. \text{ Thus, } y_2 = X_4$$

$$t_3^1 = \{A, B, E\}. \text{ Thus, } y_3 = X_5$$

$$t_4^1 = \{A, C, D\}. \text{ Thus, } y_4 = X_4$$

Thus,

$$\begin{aligned} Y_1 &= \max \{ \min(X_1, X_2, X_3), \min(X_1, X_2, X_4), \min(X_1, X_2, X_5), \min(X_1, X_3, X_4) \} \\ &= \max \{ X_3, X_4, X_5, X_4 \} = X_3 \end{aligned}$$

Similarly  $Y_2 = X_4$ ,  $Y_3 = X_4$  and  $Y_4 = X_5$ .

### Tie Breaking

Now that there is a tie between job 2 and job 3 since  $Y_2 = Y_3$ . Apply heuristic 4.2.2 to break the tie.

$$x_{12} = \{C_1 \cap C_2\} = (ABE, ACD)$$

$$x_{13} = \{C_1 \cap C_3\} = \text{none}$$

Since  $N[x_{12}] = 2$  and  $N[x_{13}] = 0$ , the next job in the sequence is job 2 and the jobs can be sequenced as (1,2,3).

The final sequence of the jobs is (1,2,3,4)

### 5.3 Example of Job Sequencing with Tool Life Consideration

The same problem described in 3.3.2(b) will be solved here except in this example the heuristic in Section 5.2 will be applied first to sequence the jobs and then Dynamic Programming technique will be applied next to find the optimal tool loading.

The objective is to minimize the makespan while determining the sequence of 5 jobs and loading of 3 tools into a magazine that can hold a maximum of 2 tools. The job processing and tool requirements for each job, the tool setup time and the maximum number of jobs each tool can process before expires (tool life) are shown in the respective tables below:

**Table 5.1: Job Processing & Tool Requirement Data**

Job	Process Plan	Tool Required	Processing Time (min)
1	1	1&2	7
	2	2&3	4
	3	1&3	6
2	1	1&2	9
	2	1&3	8

Job	Process Plan	Tool Required	Processing Time (min)
3	1	2	8
	2	3	6
4	1	3	7
5	1	2&3	6

**Table 5.2: Switching Time between the Tools**

<i>Tool</i>	1	2	3
1	10	12	8
2	7	11	5
3	6	3	15

**Table 5.3: Tool Life Data**

Tool	Maximum no of jobs the tool can process
1	3
2	2
3	1

## Solutions

Table 5.3 shows that the tool 1, 2 and 3 can process 3 jobs, 2 jobs and 1 job respectively before wears out. Once any tool reaches to that number and if the next job in the sequence requires that tool to process the job, then this tool must be replaced by a new tool. Table 5.2 provides the switching time between two similar tools too. Table 5.1 has to be modified to show all possible tool loadings to process the jobs. This procedure has been explained in Section 3.2.2(a).

### 5.3.1 Sequencing the Jobs

Since the tool life is a contributing factor to switching tool, the heuristic in Section 5.2 can be applied to sequence the jobs. Thus:

$$X_1 = 3, X_2 = 2 \text{ and } X_3 = 1$$

There are 3 tool sets to process job 1, therefore:

$$t_1^1 = \{1,2\}, \text{ Thus, } y_1 = 2$$

$$t_2^1 = \{2,3\}, \text{ Thus, } y_2 = 1$$

$$t_3^1 = \{1,3\}, \text{ Thus, } y_3 = 1$$

Therefore,

$$\begin{aligned} Y_1 &= \max\{\min(X_1, X_2), \min(X_2, X_3), \min(X_1, X_3)\} \\ &= \max\{2,1,1\} = 2 \end{aligned}$$

Similarly,  $Y_2 = 2, Y_3=2, Y_4=1$  and  $Y_5=1$ .

### **Tie Breaking**

Since there is a tie among the jobs 1, 2 and 3, apply the heuristic 4.2.2 to break the tie. While solving the problem 4.3, the heuristic 4.2.2 has been applied and the same is applicable here. Therefore, for sequencing the jobs 1, 2 and 3, that procedure may be used here. Based on the solution, the sequence for the three jobs is (3, 1, 2)

Since there is another tie between the jobs 4 and 5, apply the heuristic 4.2.2 to break the tie. The tool commonality factors between the jobs 2 and 4, and between the jobs 4 and 5 are:

$$x_{24} = \{C_2 \cap C_4\} = (1,3)$$

$$x_{25} = \{C_2 \cap C_5\} = \textit{none}$$

Since  $N[x_{24}] = 1$  and  $N[x_{25}] = 0$ , the next job in the sequence is job 4 and the jobs will be sequenced as (3,1,2,4).

The final sequence the jobs will be (3,1,2,4,5).

### 5.3.2 Tool Loading

Dynamic Programming technique from Chapter 3 will be applied to find the optimal tool loading. For simplicity, drawing the network diagram is not repeated here. However, the calculation for each stage to determine the optimal tool loading is done and is shown below:

**Table 5.4: Calculation for Tool Loading for Job Sequence 3-1-4-5-2 with Tool Life**

Job $j$	Tool Loading $t_k^j$	Process Time (1) $p[t_k^j]$	Previous Tool Loading $t_i^{j-1}$	Tool Use					(2) $M[t_i^{j-1}]$	Tool Switch $i > k$	Switch Time (3) $S[t_i^{j-1}, t_k^j]$	Total Time 1+2+3	Makespan for corresponding tool loading $M[t_k^j]$	
				Tool	Current $t$	Up to Previous	Total	Reset on Tool Switch						
3	1,2	8	-	1	1	0	1	1	0	-	0	8	8	
				2	1	0	1	1						
				3	0	0	0	0						
	2,3	8	-	-	1	0	0	0	0	0	-	0	8	6
					2	1	0	1	1					
					3	1	0	1	1					
	2,3	6	-	-	1	0	0	0	0	0	-	0	6	6
					2	1	0	1	1					
					3	1	0	1	1					
1,3	6	-	-	1	1	0	1	1	0	-	0	6	6	
				2	0	0	0	0						
				3	1	0	1	1						
1	1,2	7	1,2	1	1	1	2	2	8	-	0	15	15	
				2	1	1	2	2						
				3	0	0	0	0						
	1,2	7	7	2,3	1	1	0	1	1	6	3>1	6	19	19
					2	1	1	2	2					
					3	0	1	1	1					
	1,2	7	7	1,3	1	1	1	2	2	6	3>2	3	16	16
					2	1	0	1	1					
					3	0	1	1	1					
	2,3	4	4	1,2	1	1	1	2	2	8	1>3	8	20	20
					2	0	1	1	1					
					3	1	0	1	1					
	2,3	4	4	2,3	1	1	0	1	1	6	-	0	25	25
					2	0	1	1	1					
					3	1	1	2	1					
2,3	4	4	1,3	1	1	1	2	2	6	1>2	12	37	37	
				2	0	1	1	1						
				3	1	1	2	1						

Job $j$	Tool Loading $t_k^j$	Process Time (1) $p[t_k^j]$	Previous Tool Loading $t_l^{j-1}$	Tool Use					(2) $M[t_l^{j-1}]$	Tool Switch $i>k$	Switch Time (3) $S[t_l^{j-1}, t_k^j]$	Total Time 1+2+3	Makespan for corresponding tool loading $M[t_k^j]$
				Tool	Current	Up to Previous	Total	Reset on Tool Switch					
				2	0	0	0	0					
				3	1	1	2	1		3>3	15		
	1,3	6	1,2	1	1	1	2	2	8	2>3	5	19	19
				2	0	1	1	1					
				3	1	0	1	1					
	1,3	6	2,3	1	1	0	1	1	6	2>1	7	34	
				2	0	1	1	1					
				3	1	1	2	1		3>3	15		
	1,3	6	1,3	1	1	1	2	2	6	-	0	27	
				2	0	0	0	0					
				3	1	1	2	1		3>3	15		
2	1,2	9	1,2	1	1	2	3	3	15	-	0	35	31
				2	1	2	3	1		2>2	11		
				3	0	0	0	0					
	1,2	9	2,3	1	1	2	3	3	20	3>1	6	35	
				2	1	1	2	2					
				3	0	1	1	1					
	1,2	9	1,3	1	1	2	3	3	19	3>2	3	31	
				2	1	1	2	2					
				3	0	1	1	1					
	1,3	8	1,2	1	1	2	3	3	15	2>3	5	28	28
				2	0	2	2	2					
				3	1	0	1	1					
	1,3	8	2,3	1	1	2	3	3	20	2>1	7	50	
				2	0	1	1	1					
				3	1	1	2	1		3>3	15		
	1,3	8	1,3	1	1	2	3	3	19	-	0	42	
				2	0	1	1	1					
				3	1	1	2	1		3>3	15		
4	2,3	7	1,2	1	0	3	3	3	31	1>3	8	72	72
				2	1	2	3	1		2>2	11		
				3	1	1	2	1		3>3	15		
	2,3	7	1,3	1	0	3	3	3	28	1>2	12	73	
				2	1	2	3	1		2>2	11		
				3	1	1	2	1		3>3	15		
	1,3	7	1,2	1	1	3	4	1	31	2>3	5	53	53
				2	0	2	2	2		1>1	10		
				3	1	1	2	1		3>3	15		

Job $j$	Tool Loading $t_k^j$	Process Time (1) $p[t_k^j]$	Previous Tool Loading $t_l^{j-1}$	Tool Use					(2) $M[t_l^{j-1}]$	Tool Switch $i>k$	Switch Time (3) $S[t_l^{j-1}, t_k^j]$	Total Time 1+2+3	Makespan for corresponding tool loading $M[t_k^j]$
				Tool	Current	Up to Previous	Total	Reset on Tool Switch					
	1,3	7	1,3	1	1	3	4	1	28	-	0	60	
				2	0	2	2	2		1>1	10		
				3	1	1	2	1		3>3	15		
5	2,3	6	2,3	1	0	3	3	3	72	-	0	93	85
				2	1	1	2	2					
				3	1	1	2	1		3>3	15		
	2,3	6	1,3	1	0	1	1	1	53	1>2	0	85	
				2	1	2	3	1		2>2	11		
				3	1	1	2	1		3>3	15		

As seen from the Table 5.4, the makespan has now reduced to 85 minutes with job sequence and the tool loading for this job sequence is:

**Table 5.5: Tool Loading for Job Sequence 3-1-2-4-5**

$$P_{ij} = \begin{array}{c|ccccc} & 3 & 1 & 2 & 4 & 5 \\ \hline 1 & 1 & 1 & 1 & 1 & 0 \\ 2 & 1 & 0 & 1 & 0 & 1 \\ 3 & 0 & 1 & 0 & 1 & 1 \end{array}$$

### 5.3.3 Analysis

The makespan has been minimized after applying the heuristic. The makespan for the job sequence 1-2-3-4-5 with tool life was 95 minutes in Section 3.3.2(b) and has come down to 85 minutes. On the other hand, there is more tool switches in the job sequence 3-1-2-4-5 than in the job sequence 1-2-3-4-5 (3 tool switches compared to 1 in the previous instance). This also proves that minimizing number of tool switch may not minimize the makespan. The tool life and the processing plans play a significant role in determining the makespan.

## **Chapter 6. Concluding Remarks and Future Work**

### **6.1 Conclusion**

For a flexible manufacturing machine, the makespan is the summation of the job processing time and the tool setup time to process all the jobs. Tool management, in general and the tool loading into a machine in particular, have been investigated by many researchers. However, multiple process plans have not been considered. As a result, there has been a significant amount of research concentrating on the minimization of the number of tool switches as opposed to minimization of the makespan.

In this thesis, it has been shown that simply minimizing the number of tool switches may not minimize the makespan in the presence of multiple process plans or if the tool life also becomes a contributing factor for tool switch. The Dynamic Programming approach presented here can be used to solve the tool loading problem to minimize the makespan for a fixed job sequence optimally. In addition, two sets of heuristics have been proposed to solve both the job sequencing problem and the tool loading problem simultaneously. This thesis also considers non-uniform tool setup time, as well as tool life as a contributor to tool switch.

### **6.2 Contributions**

The first contribution of this research is to formulate the tool loading problem using a Dynamic Programming (DP) approach. The Dynamic Programming technique has many applications in different areas of operations research and is a very effective tool that will guarantee an optimal solution. However, within the tool management arena, particularly in solving the tool loading problem, this technique has never been tried. By successfully using DP to solve several examples, it can be seen that this method has the flexibility to accommodate various assumptions within the tool loading problem. It may be recalled that the most widely used KTNS policy is not even flexible enough to accommodate non-uniform tool setup time and the Network Policy, which has been proposed to address both the uniform and non-uniform tool setup time is not flexible to accommodate multiple process plans.

The second contribution of this thesis is to challenge a myth – a myth that has been active for decades, focusing researchers to find the optimal tool loading through minimizing the number of tool switches. In the absence of multiple process plans or tool life, this is, no doubt, the correct approach. However, since the tool loading problem has been for a flexible machine and the flexible machine is, by default, good enough to handle multiple process plans, therefore, the tool loading problem should not be focused solely on the number of tool switches; rather it should be focused on the makespan. The heuristics, as presented in this research, are just an introduction as to how multiple process plans can be tackled within the tool loading problem.

### **6.3 Direction for Future Work**

The application of the Dynamic Programming (DP) is robust and has opened the door for researchers to solve the tool loading problems with a wide array of assumptions and requirements. However, this research did not conduct performance measures on how large a problem (number of jobs, tools) can be solved using the dynamic programming approach. A collection of real life data may help future researchers to determine the limitation and the performance of this approach. Another approach to solving the job scheduling and the tool loading problem in the presence of probabilistic tool life is to use simulation and a time based tool life is desirable over number of jobs based tool life. As the tools wear out, the processing time may increase and so does increase the number of defective items. This phenomena may also be considered in future research too. Moreover, the Dynamic Programming approach has not been tried to solve both the job sequencing problem and the tool loading problem in a simultaneous fashion and future researchers may consider formulating a DP within a DP to solve this problem.

## References

- [1] E. Gray, A. Seidmann, and K.E. Stecke, 1993, *A synthesis of decision models for tool management in automated manufacturing*, Management Science, Vol. 39. 549-567.
- [2] R. Shirazi and G D M. Frizelle, 2001, *Minimizing the number of tool switches on a flexible machine: an experimental study*, Journal of Engineering Manufacture, Vol. 215 Part B. 253-261
- [3] Y. Crama, A.W.J. Kolen and A. G. Oerlemans, F. C. R. Spijksma, 1994, *Minimizing the Number of Tool Switches on a Flexible Machine*, The International Journal of Flexible Manufacturing Systems, Vol. 6. 33-54.
- [4] C. S. Tang and, E.V. Denardo 1988, *Models Arising from a Flexible Manufacturing Machine, Part I: Minimization of the Number of Tool Switches*, Operations Research, Vol. 36. 767-777.
- [5] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan and D.B. Shmoys, 1989, *Sequencing and scheduling : algorithms and complexity*. Amsterdam : Centrum voor Wiskunde en Informatica, BS-R89xx.
- [6] C. Privault and G. Finke, 1995, *Modeling a tool switching problem on a single NC-machine*. Journal of Intelligent Manufacturing, Vol. 6. 87-94.
- [7] G. Laporte, J. S. Gonzalez, F. Semet, 2004, *Exact algorithms for the job sequencing and tool switching problem*, IIE Transaction, Vol. 36. 37-45.
- [8] G. Ghiani, A. Grieco, and E. Guerriero, 2010, *Solving the Job Sequencing and Tool Switching Problem as a nonlinear least cost Hamiltonian cycle problem*, Networks, Vol. 55. 379-385.
- [9] A. Hertz, G. Laporte, M. Mittaz and K. E. Stecke, 1998, *Heuristics for minimizing tool switches when scheduling part types on a flexible machine*, IIE Transactions, Vol. 30. 689-694.

- [10] J. E. Amaya, C. Cotta, and A. J. Fernandez, 2008, *A Memetic Algorithm for the Tool Switching Problem*, LNCS - Lecture Notes in Computer Science, 2008, Vol. 5296. 190-202.
- [11] F. W. Taylor 1906, *On the art of cutting metals.*, American Society of Mechanical Engineers, 28(3).
- [12] S. Kalpakjian and S.R. Schmid, 2006, *Manufacturing Engineering and Technology*. Upper Saddle River, NJ 07458 : Printice-Hall, 2006.
- [13] J. Sharit and S. Elhence, 1989, *Computerization of tool replacement decision making in flexible manufacturing systems: a human-systems perspective*, International Journal of Production Research, Vol. 27. 2027-2039.
- [14] Z. Vagnorious, M. Rausand and K. Sorby, 2010, *Determining optimal replacement time for metal cutting tools*, European Journal of Operational Research, Vol. 206. 407-416.
- [15] X.G. Ming, J.Q. Yan, X.H. Wang and W.F. Lu, 2008, *Collaborative manufacturing process planning for distributed integration in tooling industry*, International Journal Manufacturing Technology and Management, Vol. 14. 17-34.
- [16] R. Shrestha, T. Takemoto and N. Sugimura, 2003, *A study on process planning system for holonic manufacturing – process planning considering both machining time and machining cost*, Proc. LEM21, 753–756.
- [17] R. Shrestha, T. Takemoto, K. Ichinose and N. Sugimura, 2008, *A study on integration of process planning and scheduling system for holonic manufacturing with modification of process plans*, Int. J. Manufacturing Technology and Management, Vol. 14. 359-378.
- [18] H. C. Chang and F. F. Chen, 2002, *A Dynamic Programming Based Process Planning Selection Strategy Considering Utilization of Machine*, International Journal of Advanced Manufacturing Technology, Vol. 19. 97-105.
- [19] J. Blazewicz, G. Finke, R. Haupt and G. Schmidt, 1988, *New trends in machine scheduling*, European Journal of Operational Research, Vol. 37. 303-317.

- [20] Z. Yao, S. K. Gupta, D. S. Nau, 2003, *Algorithm for selecting cutters in multi-part milling problems*, Computer-Aided Design, Vol. 35. 825-839.
- [21] J. F. Bard, 1988, *A heuristic for minimizing the number of tool switches on a flexible machine*, IIE Transaction, Vol. 20. 382-391.
- [22] K. Das, M.F. Baki, and X. Li, 2009, *Optimization of operation and changeover time for production planning and scheduling in a flexible manufacturing system*, Computers & Industrial Engineering, Vol. 56. 283-293.
- [23] M. S. Akturk, J. B. Ghosh, E. D. Gunes 2003, *Scheduling with Tool Change to Minimize Total Completion Time: A Study of Heuristics and Their Performance*, Naval Research Logistics, Vol. 50. 15-30.
- [24] M. Widmer, 1991, *Job Shop Scheduling with Tooling Constraints: A Tabu Search Approach*, The Journal of the Operational Research Society, Vol. 42. 75-82.
- [25] K. H. Ecker and J. N. D. Gupta, 2005, *Scheduling tasks on a flexible manufacturing machine to minimize tool change delays*, European Journal of Operational Research, Vol. 164. 627-638.
- [26] H. A. Taha, 1997, *Operations Research - An Introduction, 6th Edition*. New Jersey : Prentice Hall, ISBN 0-13-272915-6.
- [27] W. L. Winston, 2004, *Operations Research - Application and Algorithms, 4th edition*. Toronto : Thomson Books/Cole, ISBN 0-534-42362-0.

## Appendix-A

### Summary of Optimum Tool Loading for All Job Sequence

Sl. No	Job Sequence					Makespan	Optimal
1	1	2	3	4	5	43	35
2	1	2	3	5	4	43	
3	1	2	4	5	3	43	
4	1	2	4	3	5	43	
5	1	2	5	3	4	43	
6	1	2	5	4	3	45	
7	1	3	2	4	5	45	
8	1	3	2	5	4	45	
9	1	3	4	2	5	45	
10	1	3	4	5	2	51	
11	1	3	5	2	4	51	
12	1	3	5	4	2	51	
13	1	4	2	3	5	45	
14	1	4	2	5	3	45	
15	1	4	3	2	5	45	
16	1	4	3	5	2	38	
17	1	4	5	2	3	38	
18	1	4	5	3	2	38	
19	1	5	2	3	4	38	
20	1	5	2	4	3	38	
21	1	5	3	2	4	38	
22	1	5	3	4	2	38	
23	1	5	4	2	3	40	
24	1	5	4	3	2	38	
25	2	1	3	4	5	38	
26	2	1	3	5	4	40	
27	2	1	4	5	3	40	
28	2	1	4	3	5	40	
29	2	1	5	3	4	40	
30	2	1	5	4	3	40	
31	2	3	1	4	5	42	
32	2	3	1	5	4	42	
33	2	3	4	1	5	42	
34	2	3	4	5	1	42	
35	2	3	5	1	4	42	
36	2	3	5	4	1	42	
37	2	4	1	3	5	40	
38	2	4	1	5	3	40	
39	2	4	3	1	5	40	
40	2	4	3	5	1	40	
41	2	4	5	1	3	40	
42	2	4	5	3	1	40	
43	2	5	1	3	4	40	

### Summary of Optimum Tool Loading for All Job Sequence

Sl. No	Job Sequence					Makespan	Optimal
44	2	5	1	4	3	40	
45	2	5	3	1	4	40	
46	2	5	3	4	1	40	
47	2	5	4	1	3	40	
48	2	5	4	3	1	40	
49	3	1	2	4	5	45	
50	3	1	2	5	4	45	
51	3	1	4	2	5	45	
52	3	1	4	5	2	38	
53	3	1	5	2	4	38	
54	3	1	5	4	2	38	
55	3	2	1	4	5	42	
56	3	2	1	5	4	42	
57	3	2	4	1	5	42	
58	3	2	4	5	1	42	
59	3	2	5	1	4	42	
60	3	2	5	4	1	42	
61	3	4	1	2	5	45	
62	3	4	1	5	2	38	
63	3	4	2	1	5	43	
64	3	4	2	5	1	43	
65	3	4	5	1	2	38	
66	3	4	5	2	1	40	
67	3	5	1	2	4	38	
68	3	5	1	4	2	38	
69	3	5	2	1	4	40	
70	3	5	2	4	1	40	
71	3	5	4	1	2	38	
72	3	5	4	2	1	40	
73	4	1	2	3	5	45	
74	4	1	2	5	3	47	
75	4	1	3	2	5	45	
76	4	1	3	5	2	38	
77	4	1	5	2	3	38	
78	4	1	5	3	2	38	
79	4	2	1	3	5	35	
80	4	2	1	5	3	35	
81	4	2	3	1	5	43	
82	4	2	3	5	1	43	
83	4	2	5	1	3	43	
84	4	2	5	3	1	43	
85	4	3	1	2	5	45	
86	4	3	1	5	2	38	
87	4	3	2	1	5	35	
88	4	3	2	5	1	43	
89	4	3	5	1	2	38	
90	4	3	5	2	1	40	

### Summary of Optimum Tool Loading for All Job Sequence

Sl. No	Job Sequence					Makespan	Optimal
91	4	5	1	2	3	38	
92	4	5	1	3	2	38	
93	4	5	2	1	3	40	
94	4	5	2	3	1	40	
95	4	5	3	1	2	38	
96	4	5	3	2	1	40	
97	5	1	2	3	4	38	
98	5	1	2	4	3	38	
99	5	1	3	2	4	38	
100	5	1	3	4	2	38	
101	5	1	4	2	3	38	
102	5	1	4	3	2	38	
103	5	2	1	3	4	40	
104	5	2	1	4	3	40	
105	5	2	3	1	4	40	
106	5	2	3	4	1	40	
107	5	2	4	1	3	40	
108	5	2	4	3	1	40	
109	5	3	1	2	4	38	
110	5	3	1	4	2	38	
111	5	3	2	1	4	40	
112	5	3	2	4	1	40	
113	5	3	4	1	2	38	
114	5	3	4	2	1	40	
115	5	4	1	2	3	38	
116	5	4	1	3	2	38	
117	5	4	2	1	3	40	
118	5	4	2	3	1	40	
119	5	4	3	1	2	38	
120	5	4	3	2	1	40	