OPERATION-LEVEL SEQUENCE-DEPENDENT SETUP TIME REDUCTION IN DYNAMIC CELLULAR MANUFACTURING SYSTEMS

Shahram Sharifi

A Thesis
In the Department
of
Mechanical and Industrial Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of Doctor of Philosophy
Concordia University
Montreal, Quebec, Canada

May 2012

CONCORDIA UNIVERSITY
SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: Shahram Sharifi

Entitled: Operation-Level Sequence- Dependent Setup Time Reduction In Dynamic Cellular Manufacturing Systems

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

Catherine Mulligan — Chair

Mustapha Nour El Fath — External Examiner

_____ — External to Program

Andrea Schiffauerova — Examiner

Onur Kuzgungkaya — Examiner

Nadia Bhuiyan - Satyaveer S. Chauhan — Thesis Supervisor

Approved by

_____
Chair of Department or Graduate Program Director

_____ _____
Dean of Faculty

ABSTRACT

Operation-Level Sequence- Dependent Setup Time Reduction in Dynamic Cellular Manufacturing Systems

Shahram Sharifi

In closed job shop, in which a fixed number of products are produced on a repetitive basis, when there are significant sequence dependent setup times and costs involved, cell formation (CF) problem should consider minimizing the sequence-dependent setup times in order to minimize the production cost. Setup time reduction in CMS has gained little to modest attention in the literature. This could be attributed to the fact that the fundamental problem in cell formation in CMS has been mainly about material handling and machine utilization while setup time was presumed to normally decrease as a result of grouping similar parts in a manufacturing cell. Despite more than three decades of history of CMS's it has been relatively recent that setup time has been included in cell formation problems and found a place in the existing models. Sequence-dependent setup time in the literature has been dealt with mostly for scheduling part-families in a single manufacturing cell or in allocation of parts to cells in a pure flow shop. In this thesis, the issue of setup time has been extended to the members of a part family and to its lowest level which is operation-level and incorporated in general cell formation problem in a dynamic CMS.

In this thesis we have developed a multi-period integer programming CF model to address the reduction of the sequence-dependent setup time as well as considering the dynamic nature of today's manufacturing environment in CMS, where the product mix demanded would change in different time periods. Due to time complexity of the problem, a two stage solution approach has been adopted. First a GA-based heuristic was developed that provides near optimal solutions for single-period problems of the global model. The performance of the GA-based heuristic was successfully evaluated versus optimization software. Second, a dynamic programming (DP)-based heuristic was developed that reintegrates the single-period solutions into a multi-period solution. The performance of the DP-based heuristic was also evaluated against optimization software.

Acknowledgments

I would like to express my utmost gratitude and appreciation to my co-supervisors Dr. Nadia Bhuiyan and Dr. Satyaveer S. Chauhan for their invaluable supervision, technical advices and strong support that helped me throughout my research. I am greatly indebted to their kindness and support, without which I could not complete my dissertation. Their generosity, patience and determination to make me learn, progress and succeed may not be put in words.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF TABLES (APPENDIX)

CHAPTER 1

INTRODUCTION

Today's firms are constantly trying to figure out better ways of exploiting economies of scale, while also satisfying the increasing demand for highly customized products (Fernandez et al., 2012).This issue gains higher momentum, when manufacturers strive to retain their market demand and regain competitive advantage by satisfying their customers through increasing product variety (Berry et al, 1999). While higher variety of products may appeal to a larger and more diverse range of customers, it also introduces complexity in manufacturing (Wan et al., 2012; Wang et al., 2011). Higher variety in general slows down the production rate and increases production cost per unit. In fact, with increase in product variety, a firm is likely to experience lower performance of its internal operations (Anderson, 1993; Child et al, 1993).

There is a trade off between economies of scale and economies of scope in the market and in the industry. In a micro scale this trade-off and counterbalance maps into a business entity, e.g. a manufacturing company, between production volume and product variety (Vilas and Vandael, 2002).This trade-off is presented through classical volume-variety curve in the literature. There are several ways to utilize this trade-off, in a way that maximizes the gain and minimizes the loss. One way, suggested commonly by research and practice, states that "firms may mitigate this trade-off by deliberately pursuing modularity in final product design, obtaining final product configuration by mixing and matching sets of standard components" (Starr, 1965). Modularity, also

referred to as component sharing, is increasingly viewed as a way to offer high product variety while retaining low variability in the production system. (Salvador et al, 2002). The general purpose of modularity is to decompose the complex system into constituent parts (Pandremenos et al. 2009). The basic idea behind modularity is that it increases the combination opportunities between two or more component parts (Starr, 1965). Modularity has been treated as a means of increasing commonality across different product variants within a product family (Evans, 1963). Considering commonality as the extreme end of similarity, modularity therefore reduces variation between product variants by eliminating variability between some of their component parts. In the following section group technology (GT), will be introduced as another way of handling the drawbacks of increasing variety.

1.1 Raison D'etre of CMS: A Macro Analysis through Volume-Variety curve

In this section, CMS and its raison d'etre have been looked at, from a volume-variety standpoint in the hopes of having a better understanding of any fundamental relationships that may be involved. Mass and batch production are two major front ends of the volume-variety curve, each fulfilling the requirements of the market condition depending on whether the production volume or the product variety is the major issue.

However, many times the midpoint of this spectrum is the issue, where the market demands a moderate amount of production level for a moderate number of product variants. Therefore a compromise is needed for this purpose to enjoy the efficiency and speed of mass production as well as the flexibility of the batch production system (Al-

Mubarak et al, 2002; Chen, 2004; Irizarry et al, 2001). This sets cellular manufacturing in between the two major production systems.

In order to make this compromise happen, it has been necessary to find a way to provide an environment similar to that of mass production for various products. In other words, to reduce the dissimilarity of the production system rather than the dissimilarity of the products themselves, by grouping their similar component parts in a temporary family, where they undergo more or less the same production procedure (Chang and Lee, 2000; Snead, 1989). To be able to elaborate on this concept and draw some conclusions, we try to put this into mathematical form. Let's assume that the volume-variety relationship can be graphically represented by a well-behaved mathematical curve. If this assumption holds then the simplest mathematical form to represent it would perhaps be a homographic curve, expressed by Eq. (1.1).

$$K=V.\nu \qquad (1.1)$$

where :

V: production volume

ν : product variety

K: constant

A physical interpretation of K could be the constant potential of a system which can trade-off between volume and variety factors. We may refer to this constant potential as the capacity of the system, in its broadest sense, which would be a resultant of all resources such as equipment,

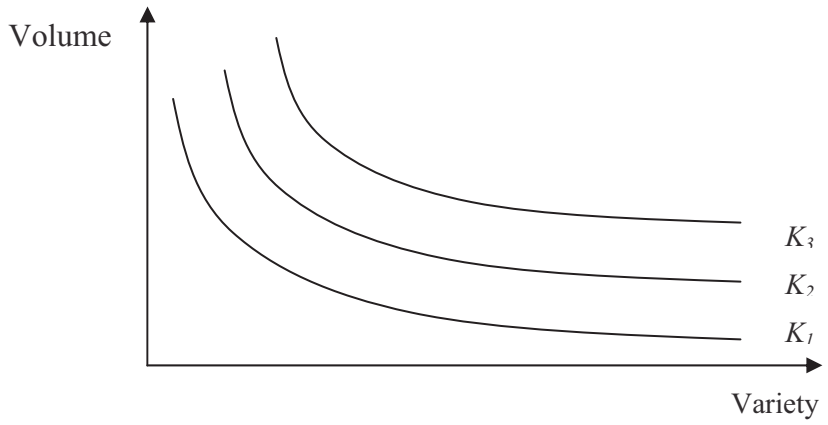

Figure 1.1-Different trade-off levels in volume-variety relationship: $K_3 > K_2 > K_1$

manpower, financial resources, etc. In Figure 1.1, different curves offer different trade-offs between production volume and product variety given different K's, e.g. K_1, K_2, K_3, etc.

Although Eq. (1.1) may be deemed too simplistic to include the complexities of the so-called volume-variety relationship, at least it reflects the fact that larger capacity is required, in order to maintain higher production volume for the same amount of variety and vice versa. On the other hand, a closer consideration of this issue would suggest that, given the capacity, the extent to which the product variety might adversely impact the production volume, depends significantly on the dissimilarity between the product variants too. In other words, the concept of variety per se, may not sufficiently reflect the dissimilarity between different product variants, since for example two sets of n products have the same index of variety but do not necessarily have the same index of dissimilarity. In order to include this consideration, we consider a modified mathematical

form that would convey a more detailed picture of the underlying relationships. One such mathematical form could look like equation (1.2) which is a modification of Eq. (1.1) based on the above discussion.

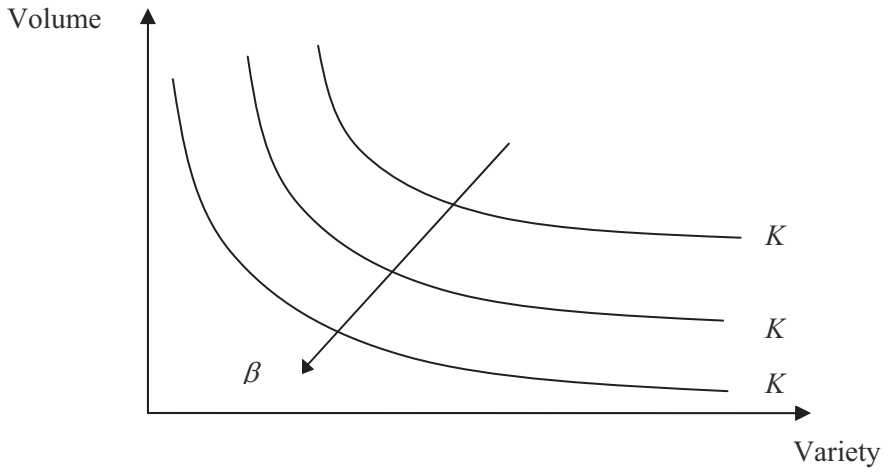

Figure 1.2- β : dissimilarity in volume-variety trade-off

$$K=V.\nu^{\beta} \qquad 0 \leq \beta \leq 1 \qquad (1.2)$$

where β, is representative of dissimilarity. Taking into consideration the above more informative relationship, there is a window of opportunity for engineering the trade-off.

Figure 1.2

represents this modification where, different volume- variety curves may now represent the same capacity *K*, given different β*'s*.

Therefore equation (1.2) communicates the fact that the extent to which two or more products are dissimilar, can affect the volume-variety relationship, thence the role of dissimilarity. The effect of dissimilarity in the production system as indicated in Figure

1.2 implies that there is a chance to engineer the trade-off by reducing the dissimilarity in the system.

Group Technology (GT) has been able to successfully engineer the aforementioned trade-off between volume and variety in a manufacturing system by reducing β, the dissimilarity index or equivalently increasing the similarity. Cellular manufacturing, an application of GT, has achieved this goal by grouping similar components of different product variants in families together with their corresponding machines in relatively independent cells.

1.2 Cellular Manufacturing as a Hybrid Configuration

In order to be successful in today's competitive manufacturing environment, managers have had to look for innovative approaches to facilities planning. Estimates imply that over $250 billion is spent annually in the United States alone on layout and machine reconfiguration. Further, between 20% and 50% of the total costs within manufacturing are related to material handling whereas effective facilities planning can reduce these costs by 10–30% (Tompkins et al., 2003; Balakrishnan and Cheng, 2007).

In a flow line, machines are laid out in line with the processing requirements of each component part in the sequence of the corresponding operations as shown in Figure 1.3. Since each component part has its own dedicated machine line and the machines are not shared among different parts, machine investment is high and the flexibility in terms of part variety is minimal.

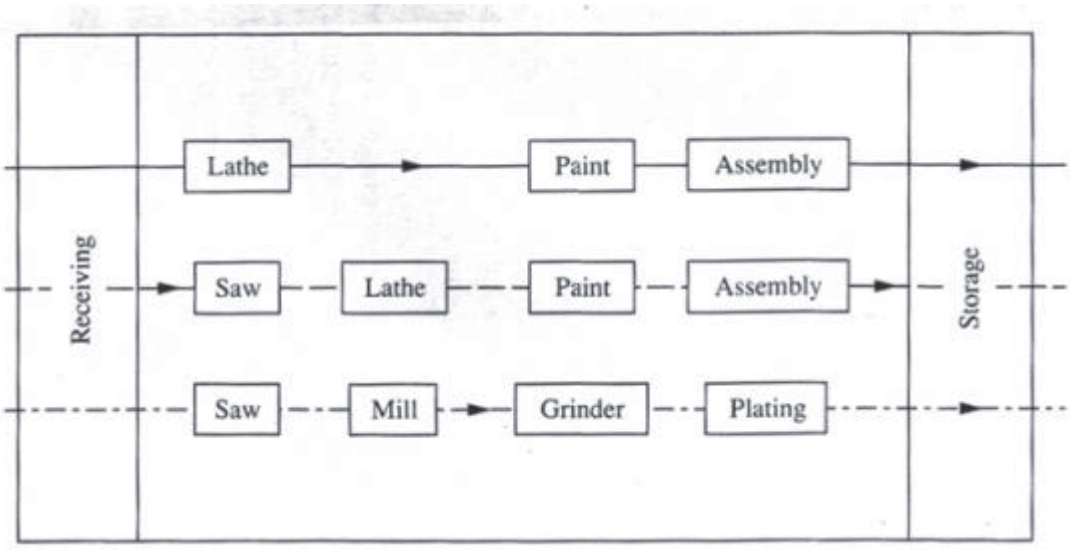

Figure 1.3-Flow line configuration
(alkolas3d.tripod.com)

On the other hand, this configuration is highly efficient resulting in faster flow of material, high production rate and high throughput as well as low transportation costs. In a job shop system, machines are grouped according to their processing characteristics in different departments: for example presses group together to form the press shop, where every part that has a press operation will have to go to that shop, and the same for lathe machines, drills, milling machines and so on. When a part is processed in a department it has to travel a rather long distance to enter another department for its next processing requirements. Since moving parts in single units is not economical, they are being transported in batches. Therefore each part has to wait until other units of that part are completed before moving to the next department which in turn leads to longer production times, larger work-in–progress, slower flow of material and even congestion and higher production costs. In order to complete all processing requirements, a part might need to

travel around the entire facility as shown in Figure 1.4. In such a configuration, high flexibility is attained since a variety of parts in relatively small batches can be processed and the general

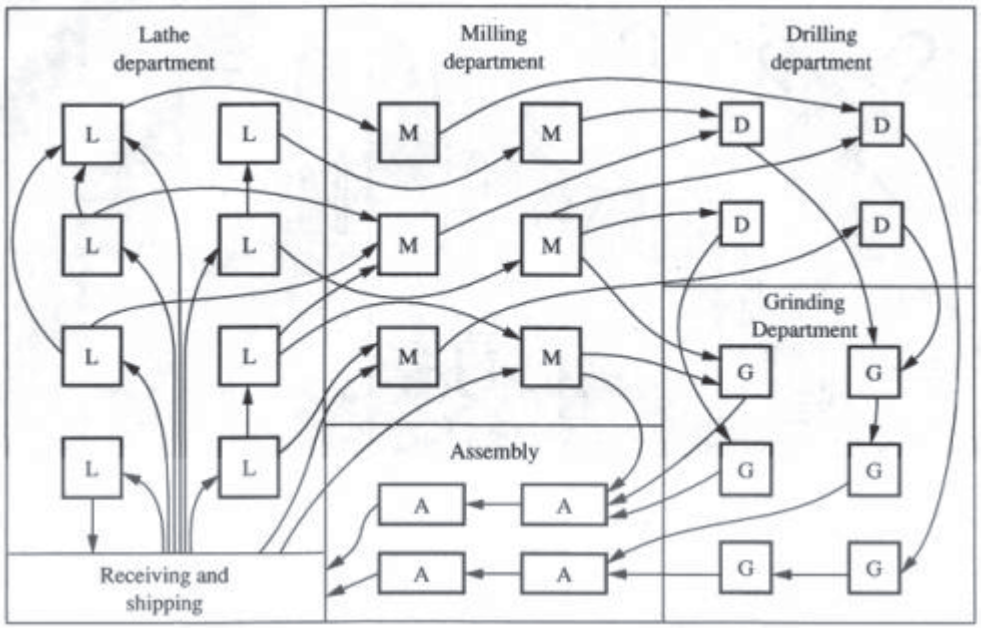

Figure 1.4- Job shop configuration
(alkolas3d.tripod.com)

purpose equipment are used to minimize machine investment. However the drawback is that the efficiency is low. One innovative approach to the above shortcomings is called Group Technology (GT). GT is based on the principle of grouping parts into families based on similarities in design or manufacturing processes. Manufacturing cells are created by grouping the parts that are produced into families. This is based on the operations required by the parts. These cells, consisting of machines or workstations, are then physically configured and dedicated to producing part families (Balakrishnan and Cheng, 2007).

Cellular manufacturing, a major application of GT, emerges as a hybrid of flow line and job shop, and attempts to benefit from advantages of both aforementioned manufacturing systems. Therefore it proves less disadvantageous as compared with the disadvantages of either system.

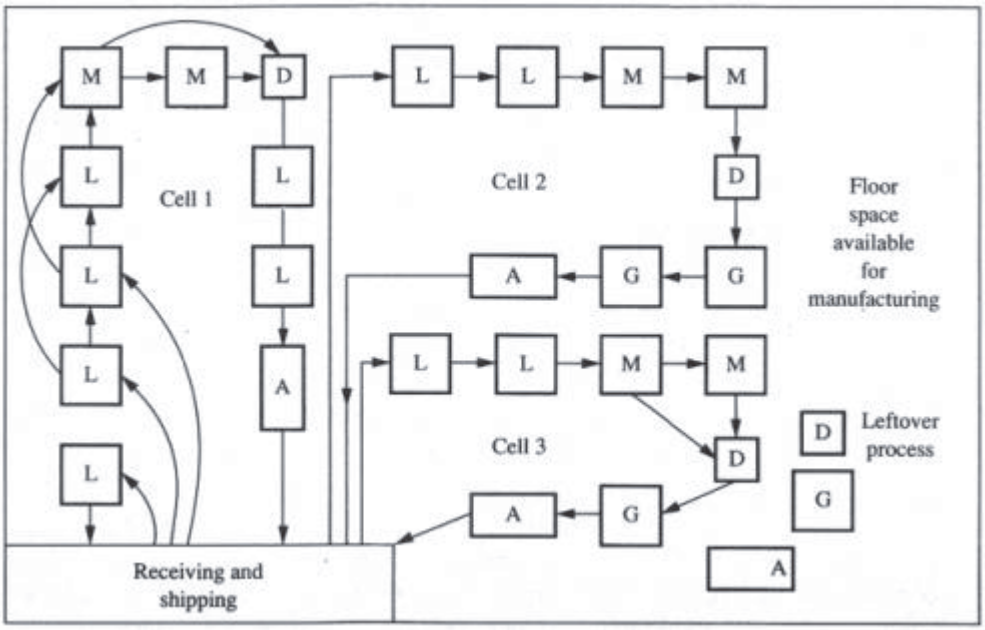

Figure 1.5- Cellular manufacturing configuration
(alkolas3d.tripod.com)

This implies that cellular manufacturing would provide a flexibility close to that of a job shop and simultaneously the efficiency close to that of a flow line. Each manufacturing cell provides an environment where similar parts represent a family that would more or less represent a single part while the cell would represent a dedicated flow line (Figure 1.5). On the other hand, the collection of cells process a large variety of parts in relatively small batches thus providing an environment which is sufficiently efficient and flexible at the same time. The result of this conversion would be indicated in less machines used

than in flow lines, less distances travelled and less time wasted in non-productive activities than in a job shop. Also since similar pars are grouped in families, it is expected that the setup time would tend to decrease more or less similarly as is the case in flow line configuration.

1.3 The Advantages of CMS

Cellular manufacturing has been credited for quite a few benefits. Several researchers have conducted surveys and or explored literature in order to outline these benefits. Burbidge (1996), Wemmerlov and Hyer (1997), Greene & Sadowski (1984), Askin & Standridge (1993), Suresh & Meredith (1994), Singh & Rajamani (1996), Mungawattana (2000) and Tariq (2010) are among sources where these benefits have been addressed. Some of these advantages noted by Burbidge (1996) are indicated in Figure 1.6.

No	ADVANTAGES	Features of GT which Cause Advantages				
		Groups Complete Parts	Machines Close Together	Total Div. into Groups	One Foreman	Team Work
1	SHORT THROUGHPUT TIMES	○	○	○	○	○
	a) Low stocks	◎	◎	◎	◎	◎
	b) Low stockholding costs	◎	◎	◎	◎	◎
2	BETTER QUALITY	○	○			○
3	REDUCED MATERIALS HANDLING COST	○	○		○	
4	BETTER ACCOUNTABILITY	○	○	○	○	
5	REDUCED INDIRECT LABOUR	○		○		○
6	REDUCED SET-UP TIME	○			○	○
7	CAPACITY INCREASED	○		○	○	
8	AUTOMATION SIMPLIFIED	○	○			
9	EASIER IN-COMPANY PROMOTION	○		○		○
10	REDUCED MATERIAL OBSOLESCENCE	○				○
11	JOB SATISFACTION	○				○

Figure 1.6 –Advantages of GT (Burbidge, 1996)

A more recent outline of benefits can be found in Tariq (2010) which has been summarized as follows:

- **Reduction of mater ial handling time and cost** : CM design is expected to minimize material handling cost during the cell formation process which means the cells are more or less independent. In other words the intercellular movements are minimized, thence savings on material handling time and cost.

- **Reduction in the number and types of tools and setup times** : Part families in each cell are expected to have similarities either in geometry, shape and size or in processing requirement. One way or another each cell would be more or less similar to a dedicated line in flow shop, which in turn would reduce the number and variety of tooling required in the cell. Also the changeover times between parts in the same family are expected to be less than that in a job shop for the above-mentioned reason.

- **Smaller inventory levels:** Due to efficient and smooth flow of material, smaller lot sizes and setup times, the level of inventory both in terms of in-process and finished items is reduced. Another aspect of this efficient material flow is the possibility of producing parts either Just-In-Time (JIT) or in small lot sizes

- **Flow time and WIP reduction**: In CMS, unlike in the job shop, parts do not have to wait in long lines since the machines are more or less dedicated to the part processing requirements. Subsequently the flow of the material would be

smoother and faster. This translates into smaller WIP and shorter flow time in the system.

- **Less space required**: Due to reduction of WIP, as a direct result of reduction of setup time and lot sizes, this would lead to less space required for carrying out the production plan.

- **Decrease in throughput times**: In a job shop environment, parts are processed in almost the entire facility and thus have to travel through long distances. In contrast, in CM each part family is processed more or less inside a particular cell, travelling shorter distances thus spending less time in transportation. This would result in shorter throughput time.

- **Quality Im provement**: Shorter travelling distances are less likely to cause damage to the parts during material handling and focused control and supervision in the cells would result in improvement in the quality of the products.

1.4 Design of Cellular Manufacturing Systems

The design methodology of CMS can be divided into two main stages. The first stage is a soft design which is the process of forming manufacturing cells. The second stage is hard design which is about actual design of the manufacturing cells including the layout of the machine and the hardware. The second stage of the design is not the subject of this thesis and only the first stage of CMS design is studied in this research. The first stage of the design, namely cell formation, consists of the following fundamental steps:

1-Forming part families based on the similarity of their processing requirements

2-Machines are grouped into manufacturing cells

3-Part families are assigned to the manufacturing cells.

Based on the cell formation approach adopted, the above three key steps may or may not be sequential. In fact many times part families and grouping of the machines together with the assignment of the part families to the cells may all be simultaneous (Mungawattana, 2000).

1.4.1 Shortcoming of the Contemporary CMS Design

The contemporary CMS design methodology has addressed quite a few real manufacturing attributes. For example, see Defersha and Chen (2006) and (2008) where the authors have considered a complete set of real life manufacturing attributes in their CMS design while only subsets had been previously considered in other publications. However, consideration of sequence-dependent setup time, as the most general form of setup time, is missing in the contemporary system design. This consideration would be essential especially in closed job shop situations where the repetitive manufacturing of a group of parts is needed and in each production cycle, the changeover time from one part type to another is relatively significant.

1.4.2 Objectives of the Thesis

The objective of this thesis is to:

-Design a CMS that accounts for the sequence-dependent setup time as the general form of changeover time between two parts on a common machine in the in order to enable the cost reduction through the order by which they are processed.

-Enable multi-period planning of the parts and machine-cell re-configurations in advance in a dynamic environment where the part mix may change from period to period during the planning horizon, in order to fulfill the contractual obligations of the manufacturing company or the market seasonal demand.

1.4.3 CMS Design Methodology

In this thesis, it was determined that the design methodology should consist of developing a mathematical model that properly represents the first stage of the CMS design of the problem under research. Mathematical modeling has the advantage of providing mathematical precision in the design stage as compared with other design methodologies. The drawback however, is that the solution would be a challenge due to computational burden and the complexity of the problem. In the case of our specific problem, the issue is significantly intensified since the presence of the sequence-dependent setup time generates massive amounts of integer variables, making the problem one of the most computationally challenging among current CMS design problems. Therefore approximate methods and heuristics seem inevitable in the solution approach. In this thesis mathematical modeling will be considered as the first step in tackling the research problem. Despite the inability of the optimization software in solving highly complex models as such, limited use of the software will help both in gaining insight about the computational intensity of the problem and evaluation of the heuristic performance. Due to the novelty of the complex model, a step by step solution approach would be helpful.

1.5 Outline of the Thesis

The rest of this thesis is organized as follows. Chapter 2 presents a brief review of the literature on CMS in general and the application of setup time in CMS related research. Chapter 3 introduces the research problem and develops a mathematical model of the CMS design to address the research problem. Chapter 4 discusses the solution methodology based on GA-based heuristic first for a static and dynamic version of the model. The performance of the GA-based heuristic is evaluated against commercial optimization software package available in the market. Chapter 5 develops a dynamic programming based heuristic to address the shortcomings of the GA-based heuristic for a multi-period solution. Chapter 6 concludes this thesis and outlines the path for future research.

CHAPTER 2

LITERATURE REVIEW

Cellular manufacturing, which is an application of group technology (GT) in manufacturing, has been recognized as one of the most recent technological innovations in job shop or batch-type production to gain economic advantages similar to those of mass production (Chang and Lee, 2000), (Singh and Rajamani, 1996), (Chu and Tsai, 1990). Group technology is a manufacturing philosophy that identifies similarities of parts and partitions a manufacturing system to its subsystem based on these similarities (Yasuda and Yin, 2002; Caux et al, 2000; Chang and Lee, 2000). For decades, researchers have been involved in different aspects of cellular manufacturing, among which, cell formation, during the past decade, has been a source of a great attention for practitioners as well as academics.

2.1. Cellular Manufacturing and Cell Formation

Cell formation (CF), which involves identification of machine groups and part families, has been known to be the first and most fundamental step in designing a cellular manufacturing system (Chen, 2004;Yasuda and Yin, 2002; Goncalves and Resende, 2002). The fundamental step in GT is to identify part families and plan a total division into groups and families, in which each group completes all the parts it makes (Burbidge, 1996; Jeon et al., 1998).

Cell formation is considered as a reorganization of an existing job shop into GT shops using information given about the processing requirements of parts. This information is commonly represented in a matrix called the machine-part incidence or machine-component incidence matrix (MCIM) with 0 or 1 entries. A 1 indicates that part *j* requires machine *i* for an operation, and 0 indicates otherwise (Caux et al, 2000; Singh and Rajamani, 1996). There are two major categories in a CF problem. A first category considers one possible way to produce each component part type i.e. fixed routing. A second category assumes several possible ways to manufacture the same component part, i.e. multiple routing (Caux et al, 2000).

Different cell formation approaches can be classified as follows: array based approaches, cluster analysis techniques, graph theoretic and mathematical programming methods which are among the major categories that have been developed to group parts and machines into cellular configurations (LaScola Needy et al, 1998). The common goal of all these methods is to maximize the number of operations performed on the part families within their corresponding machine cells. A machine cell is a manufacturing unit capable of processing a part family, a family of parts with similar manufacturing requirements, for its entire set of operations (Seifoddini and Djasemmi, 1995).

2.2. Taxonomy of Cell Formation Techniques

Versions of the taxonomy of cell formation methods can be found in the following: King (1980), King and Nakornchai (1982), Chandrasekharan and Rajagopalan (1987), Khator and Irani (1987), and Kusiak and Chow (1987), Offodile et al. (1991), Mungwattana

(2000), Murugan and Selladurai (2005), Tariq (2010) and Arora et al.(2011). A hierarchical graph of a version of the taxonomy of cell formation techniques is depicted in Figure 2.1

Visual Inspection Based Method

The visual inspection based method of GT arranges parts by studying part geometries and grouping similar parts into families. This approach is subject to error and heavily relies on the expert opinion and experience of the user, and is rarely used in practice except with a fairly small number of parts (Offodile et al., 1991) .

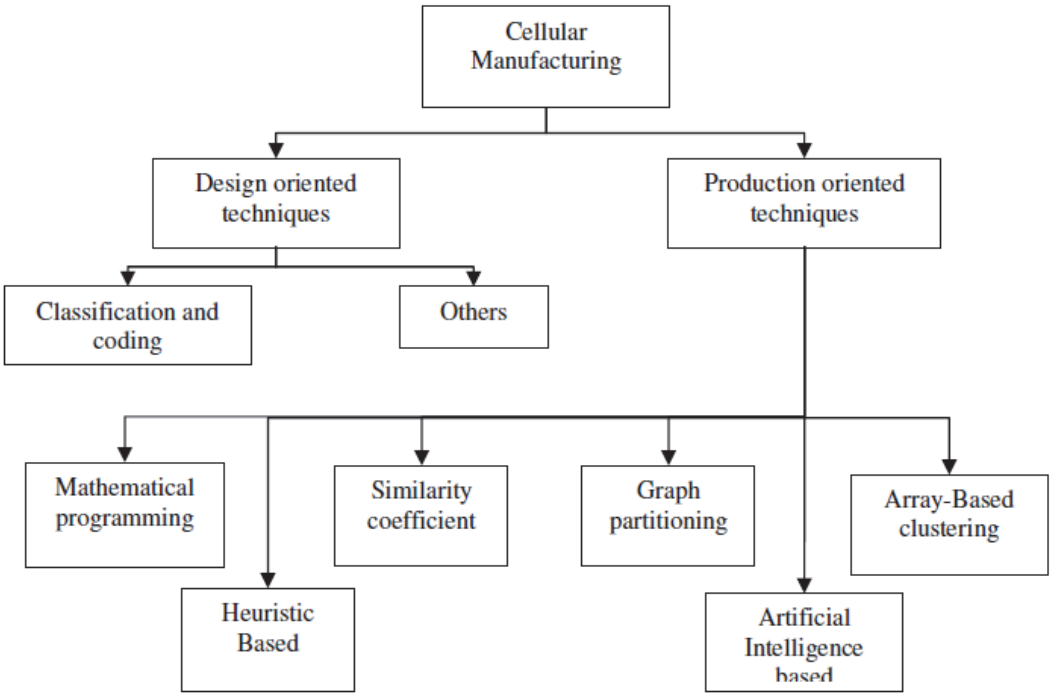

Figure 2.1-Classification of various cell formation techniques
(Tariq, 2010)

Part Characteristic Based Systems (Design Oriented Techniques)

Known as part coding and classification analysis (PCA), this approach uses a coding system to assign numerical weights to parts characteristics and identifies part families using some classification (Offodile et al., 1991).

Array-Based Clustering

Array-based clustering methods perform a series of column and row permutations to form product and machine cells simultaneously. In array-based clustering, the processing requirements of components on machines can be represented by an incidence matrix. This is referred to as the machine-component incidence matrix (MCIM) or simply machine-part matrix. The machine-part matrix has zero and one entries. A "1" entry in row *i* and column *j* of the matrix indicates that component *j* has an operation on machine *i*, whereas a "0" entry indicates otherwise. The array-based techniques allocate machines and parts to cells by manipulating the order of rows and columns to find a form as close as possible to a block diagonal form in the matrix. The following array-based clustering algorithms can be found in the literature: Bond Energy Analysis by McCormick et al. (1972), Rank Order Clustering by King (1980) and King and Nakoranchi (1982), Cluster Identification method by Kusiak and Chow (1987), Modified Rank Order Clustering by Chandrasekharan and Rajagopalan (1989), Direct Clustering Analysis by Chan and Milner (1989), and the Hamiltonian Path Heuristic by Askin et al. (1990) (Arora et al. 2011).

Hierarchical Clustering

Hierarchical classifications may be represented by inverted tree structures or dendograms, which are two-dimensional diagrams illustrating the bursts or divisions which have been made at each successive stage of the analysis. In hierarchical clustering, the data in the machine-part matrix are partitioned into groups or cells in several steps rather than in one step. They are first separated into a few large cells, each of which is further divided into smaller ones, and each of these further partitioned, and so on until terminal groups are generated which cannot be subdivided (Arora et al. 2011).

Non-Hierarchical Clustering Techniques.

The non-hierarchical procedure was developed by Chandrasekharan and Rajagopalan (1989). Non-hierarchical clustering methods are iterative methods and begin with either an initial partition of the data set or the choice of a few seed points. In either case, one has to decide the number of clusters in advance. The fact that the choice of seed points (or initial partition of data) are arbitrary, could lead to unsatisfactory results (Arora et al. 2011).

Graph Theoretic Method (Graph Partitioning)

Vannelli and Kumar (1986) developed graph theoretic models that would lead to a perfect block diagonal structure by determining the machines which need to be duplicated in a block. Each block would represent a machine group and part family namely a cell.

Production Flow Analysis

Burbidge (1971) developed production process based analysis. This method has gained overwhelming attention by researchers since it was first developed and popularized by its

developer. The goal of the production- based approach is to identify and group parts based on the similarity of their processes. Production-oriented systems or production flow analysis (PFA) systems use route sheets to record the relationship between parts and the machines that process them (Offodile et al., 1991).

Mathematical programming

Mathematical programming is a powerful tool for solving complicated production planning problems, when product structures with multi-item and multi-level (Defersha and Chen, 2008). Mathematical Programming (MP) approaches to formulate the Machine-Part grouping problem as an optimization problem. Due to their ability to consider and incorporate a number of critical system design information, MP based approaches have been extensively used to solve the CF problems. The mathematical optimization approaches applied to the cell formation are either linear or nonlinear integer programming problems. Kusiak (1987 & 1988) and Boctor (1991) have shown that these approaches have the ability to incorporate a lot of production related data, for example; processing sequence, routing flexibility, setup and processing time etc. While clustering parts and machines, as an optimization technique the objective could be to maximize the total sum of similarities between each pair of machine-part (Tariq, 2010).

The main advantage of MP approaches is that they formulate the problem with mathematical precision at the design stage, thus can guarantee an optimal solution if the problem would be tractable enough to be optimally solved in a reasonably short amount of time. Whereas other approaches introduce the approximation early in the design stage by adopting some sort of heuristic approach. The drawback of MP, however, is that CF problems in general are NP-hard and most definitely the application of approximate

methods of some kind would be inevitable at the solution stage, in order to obtain a "good enough" solution for real life problem instances.

Artificial Intelligence (AI) Approaches

Karapathi and Suresh (1992) were among the first that proposed an application of artificial neural networks to CF problems. Elmaghraby and Gu (1998) presented an approach for using domain specific knowledge rules and a prototype feature based modeling system that would automate the process of identifying parts attributes and assigning the parts to the most appropriate manufacturing cells. The expert assignment system is based on the geometric features of the parts, characteristics of formed manufacturing cells, parts functional characteristics and attributes, as well as domain specific manufacturing knowledge. Kusiak (1987) developed a pattern recognition based parts grouping which is basically similar to the grouping procedure in GT. The basic difference between these two, however, is in the degree of automation (Arora et al., 2011).

2.3 Setup Time in Manufacturing and CMS

Quite a considerable amount of work has been done in the field of setup reduction and its effects in the general scheme of production systems. Setup time reduction in manufacturing operations is widely recognized to provide significant benefits in areas such as cost, agility and quality (Nye et al, 2001). Cellular manufacturing is no exception to this rule. Introduction of cellular manufacturing systems is expected to result in reduced setup time, production lead time, work-in-process, labor, tooling, rework and

scrap materials, delivery time, and paper works (Kusiak, 1992; Seifoddini and Djasemmi, 1995).

The main improvements that can be expected from cellular manufacturing are reductions in throughput time, material handling and setup time as well as part quality (Burbridge, 1992; Wemmerlov and Hyer, 1989; Hyer 1984). A number of researchers have compared job shop layouts to those of CMS and have generally concluded that the latter provides shorter setup times, lower machine utilization and shorter travel distances (Agarwal and Sarkis, 1998; Al-Mubarak et al., 2002). Some surveys show that setup reduction from GT is a key determinant of the relative advantage of CMS over job shop and functional layout (FL). In fact there exists a breakeven level of setup reduction which must be achieved to make partitioning to cell worthwhile (Shambu and Suresh, 2002).

On the other hand, a simulation study by Garza et al (1991) shows that the performance of cellular manufacturing as measured by mean flow time or work-in-process inventory is better than that of a job shop when conversion into CMS results in low intercellular flow, even when other operating factors do not improve after the conversion. Morris and Tersine (1990) examined the effect of cellular performance of ratio of setup to processing time, the time to travel between work centers and some other factors (Goncalves and Resende, 2002). Suresh (1992) found in his study that flow time and WIP were sensitive to the magnitude of the setup time. The simulation study by Flynn and Jacobs (1986) showed that rules designed to minimize setup may have a significant impact on the performance of CMS. They stated that CMS would be most beneficial when setup time is very small relative to the average processing time. Yang and Deane (1993) studied the relationship between setup time reduction and performance improvement in the stochastic

closed manufacturing cell and showed that marginal reductions in optimal product batch sizes and flow time variance would alleviate as job setup time decreases.

2.4 Setup Time in Manufacturing Cell Scheduling

A series of studies that could be considered pertinent to the field of this research proposal are discussed in this section. Surveys by Allahverdi et al. (1999) and Cheng et al. (2000), involving separable setup times, report that most prior research on manufacturing cell scheduling has assumed sequence independent setup times (Schaller et.al., 2000). However, Sammadar et al. (1999) developed the mathematical model for resource sharing and scheduling for a class of computer-integrated manufacturing cells with sequence-dependent setup times. A similar work has been done by Schaller et al. (2000), who have addressed the problem of scheduling part families and jobs within each part family in a flow line manufacturing cell, where the setup times for each family are sequence-dependent and it is desired to minimize the makespan while processing parts together in each family. Franca et al. (2005) proposed six evolutionary heuristic algorithms to minimize makespan for permutation schedules in a pure flow shop manufacturing cell with sequence-dependent setup times between families of jobs. Lin et al. (2009) developed a meta-heuristic for a non-permutation flow line manufacturing cell with sequence-dependent family setup time. Hendizadeh et al. (2008) applied various Tabu Search based meta-heuristics to scheduling of the part families and jobs within each family in a flow line manufacturing cell with sequence dependent family setup times.

2.5 A Critical Review of Setup Time and Cost in CF Problems

LaScola Needy et al. (1998) developed a practical yet effective procedure that is basically a numeric calculation of material handling cost, setup cost and investment cost for each cellular configuration of a series of cellular configurations in order to identify the minimum cost alternative. In fact the cell formation procedure, in their work, takes place by generating alternative cell configuration through genetic algorithm approach and then the setup time is calculated for each alternative.

Lashkari et al. (2004) developed an operation allocation (OA) model which assigns a set of part types to a group of machines considering setup cost and provides information as an input to the material handling sub-system (MHSS) model. Moon and Gen (1999) presented a 0-1 integer programming model to design independent manufacturing cells with alternative process plans and machine duplication consideration. Atmani (1995) developed a mathematical programming model for production planning problem of determining optimal machine selection and operation allocation in flexible manufacturing systems to minimize transportation and setup costs.

Despite the fact that setup time has always been quite a major issue in the context of CMS, it has rarely been actually incorporated in the cell formation procedure as a criteria. There are, however some few exceptions. Damodaran et al. (1992) proposed a mixed-integer programming model for the OA problem in the context of multi-machine and multiple cell operations considering re-fixturing costs. Atmani et al. (1995) proposed a 0-1 integer programming model that jointly considers OA and cell formation in cellular manufacturing with re-fixturing cost. The paper by Ohta et al, (2002) who developed a

heuristic algorithm based on a new similarity measure, includes the setup time as one of the two criteria in the similarity measure for cell formation.

Allahverdi and Sorush (2006) emphasized the importance and benefits of reducing setup time. The fact that treating setup times separate from processing times improves resource utilization. This is particularly important in modern manufacturing management systems such as JIT, GT and CMS.

Setup time is a critical issue in every production system. In fact setup time is literally a key factor that can adjust the parameters of a production system such as lead time, WIP, batch size, machine utilization, production cost as well as smooth flow, agility, responsiveness, etc. Reduction of setup time has always been a major issue as well as motive in CMS due to its major contribution to the cost effectiveness and other crucial impacts on the production system. The results of a simulation study by Suresh and Meredith (1994) indicated that reduction in setup time and processing times have the greatest impact on CMS performance (Agarwal and Sarkis, 1998).

The simulation study by Flynn and Jacobs (1986) showed that rules designed to minimize setup may have a significant impact on the performance of CMS. They stated that CMS would be most beneficial when setup time is very small relative to the average processing time. The study by Suresh (1991) concluded that a certain level of setup reduction has to be achieved before a CMS will outperform a functional layout (FL) system. Another case study of a small manufacturing company concluded that reduction of setup times and trained workers helped the CM system achieve goals like reduction in process time and less space due to less WIP (Agarwal and Sarkis, 1998).

In the entire context of CMS, similarity has been assumed as the source of reduction in material handling costs, reduction of setup times and increase in machine utilization. It has been assumed that since parts grouped in a cell need less set up time due to similarity among them, the overall setup time is expected to reduce in the entire system. In other words the conventional CF approaches attempt to increase similarity or equivalently reduce dissimilarity to achieve reduction in setup time. It has been mentioned that since parts are assigned to families on tooling and setup requirements, usually a negligible or minor setup is required to change from one part to another within a family and hence can be included in the processing times of each job (Schaller, 2000). Although the above assumption sounds reasonable, the impact of cellular manufacturing on setup reduction should not be exaggerated. In fact, some mixed results from some surveys can exemplify the definition of setup reduction in conventional CF approaches based on MCIM. For example, Wemmerlov and Hyer (1989) stated that a well-organized job shop results in better flow time and less work-in-process inventories than cellular manufacturing. They claim that surveys of firms adopting cellular manufacturing report better queue related results mainly because they are comparing their new layout to their previous poorly designed and operated job shop. Some literature raises doubts on whether CMS produces better performance for queue related criteria (Al-Mubarak et al, 2004). Roughly speaking, the so-called similarity is assumed to be attained by grouping parts with highest number of machines in common in each cell. To do so, almost all CF approaches in CMS deal with some manipulation of machine-part or machine-component incidence matrix, MCIM (Moleman et al, 2002; Ohta and Nakamura, 2002). In essence, the basic information required to solve a CM problem is the machine-part incidence matrix (Ohta

and Nakamura, 2002). It is a matrix $A_{ij,}$ with machines in rows and component parts in columns consisting of elements $a_{ij=1}$ where part j visits machine i and $a_{ij=0}$ otherwise. Once we know what machines are required by what parts, we would be able to minimize intercellular movements as well as maximize machine utilization through the MCIM with the maximum 1's inside the cells and minimum 1's outside the cells. Thus we can minimize the overall material handling costs and machine cost by proper cell formation.

Reduction of setup time in CMS is justified by referring to the fact that cells are formed on the basis of similarity of processes for the parts in the cell. In the MCIM, parts are matched with their pertinent machines in a matrix. This way one can see the commonality of machines between different parts and group them accordingly. However, what is attained would be an increased commonality of machines between parts in a cell.

Actually the MCIM in its utmost can only guarantee the commonality of machines among parts in a cell which is not necessarily the same as the similarity of operations. In other words, the MCIM does not tell us much about the quantity of the setup times, thus a setup reduction is not controlled.

In this research, it is presumed that part of the ambiguity about setup time reduction in the CF procedure can be attributed to the definition of setup time in the literature. Some works that deal with introducing parameters or coefficients concerning setup cost or setup time are implicitly based on consecutive operations of parts on various machines (as an example, see Damodaran et al. 1992 and Ohta and Nakamura 2002). In some others, e.g. Atmani et al. (1992) and Lashkari et al. (2004), setup cost has been associated solely with one part and the machine. However Cox et al. (1995*),* defines setup as the work required to change a specific machine, work centre, or line from making the last good piece of a

unit A to the first good piece of unit B (LaScola Needy et al., 1998). The above definition is the closest to real life manufacturing. In general, setup time can be defined as the transition time from state A to state B while state A refers to part A and state B refers to part B. The significance of this definition is that it implies the important concept of *changeover* between two parts. Besides, common sense and real practice both imply that changeover must take place on a *common* machine. The term 'common' should be stressed since it is a key factor in the definition that might lead to confusion if neglected or understated. In fact, setup time depends on three elements without which part of the information for obtaining the changeover time would be missing. These elements are: incoming component part *i*, outgoing component part *i'* and the common machine *k,* as shown in Figure 2.2.

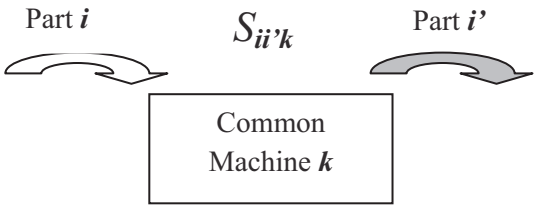

Figure 2.2- Setup as joint changeover

Allahverdi and Sorush (2006) note two types of setup times: sequence-dependent and sequence-independent. In a sequence-independent setup time, the setup time of the incoming part is independent and has nothing to do with what it is going to replace on the common machine. In sequence-dependent setup time, the setup time of the incoming part may vary based on the part it is going to replace, implying a joint changeover time

between the two interchanging parts. This situation could occur in many situations where the size, shape, or complexity of the parts would force more or less time for the adjustment of the machine and setup of the part than it would if a different part was to be replaced. For example, removing a previously installed die on a press processing part A could take more or less time than removing that of part A'. Therefore the overall time for part B to remove the previous part and get installed would depend on the previous part.

Sequence-dependent setup times are usually found in the situation where the facility has multipurpose machines. Some examples of sequence-dependent setups include chemical compounds manufacturing, where the extent of the cleansing depends on both the preceding chemical processed and the chemical about to be processed, the printing industry, where the cleaning and setting of the press for processing the next job depend on its derogation from the color of ink, size of paper and types used in the previous job. Sequence-dependent setups can also be found in many other industrial systems, some of which include the stamping operation in plastic manufacturing, die changing in metal processing shops, and roll slitting in the paper industry, etc. (Yang and Liao, 1999; Eren, 2007).

In order to err on the side of caution and maximize the inclusiveness and generality, a sequence-dependency condition has to be considered, so that:

$$S_{ii'k} = 0 \qquad \text{for} \quad i = i' \tag{2.1}$$

$$S_{ii'k} \neq S_{i'ik} \qquad \text{for} \quad i \neq i' \tag{2.2}$$

where: $S_{ii'k}$ is the sequence-dependent setup time between parts *i* and *i'* on machine *k*.

The definition of setup as sequence-dependent changeover is not only clear and straightforward, but also is the most inclusive one in the real world which includes sequence-independent ones, when the input data for $S_{ii'k}$ and $S_{i'ik}$ are equal; or additive type where the sum of the times for removal of part *i'* and installing of part *i*, would represent the single joint changeover time $S_{i'ik}$. In effect it covers all different cases of changeover e.g. from uninstalling a previous die, say on a press, and installing a new die, to switching from a previous set of adjustments, say on a lathe machine, to a new one, etc.

Consideration of all pairs of parts on different machines leads to a joint changeover matrix, $\mathbf{S}_{ii'k}$, including the sequence-dependent setup times between pairs of parts having common machines. This information in companies may be obtained from route sheets where the MCIM data is also obtained. It is obvious for a sequence-independent setup time the corresponding input data for $S_{ii'k}$ and $S_{i'ik}$ will simply be equal. Whenever a pair of component parts have a common machine but will definitely not change over due to routing or technological reasons, the joint setup time in the matrix would be infinity.

Regarding the above discussion and conclusions, in Damodaran et al. (1992) and Ohta and Nakumara (2004), setup has been considered between consecutive operations *k* and *k+1* of part *j* notwithstanding the common machine. In other words in their consideration of setup, the common machine has been tacitly ignored, i.e. the part is common and the machines could be different. In Atmani et al. (1995) and Lashkari et al. (2004), however, the outgoing part has been ignored.

Rajamani et al. (1992) considered a sequence-dependent setup time between pairs of parts notwithstanding the machine, in flow shop cells "where various parts tend to use identical production processes" (Rajamani et al., 1992). This also implies that MCIM is not required, as all elements of such incidence matrix would consist of 1's only. Consequently intercellular movement does not play a role.

2.6 Application of Meta-Heuristics to Cellular Manufacturing

Murugan and Selladurai (2005) applied genetic algorithm to a cell formation problem that would reduce the setup time, however they used the group efficacy criteria approach rather than mathematical programming and the setup time, which, while not being sequence-dependent, were measured on two different machines as opposed to a common machine.

Gosh et al. (2011) conducted a state-of-the- art generic review of application of various meta-heuristics in cellular manufacturing. Defersha and Chen (2008) embedded a linear programming sub-model in their genetic algorithm based heuristic to solve a multi-period mixed integer programming comprehensive cell formation problem. Ahmed et al (2004) conveyed a comparison among heuristic methods used for solving cellular manufacturing models in a dynamic environment. They considered a generic nonlinear mixed integer programming model for designing CMS in a dynamic environment and they solved the problem by the three well known meta-heuristics, namely Genetic Algorithm (GA), Simulated Annealing (SA) and Tabu search. Ohta and Nakumara (2002) developed a genetic algorithm based heuristic to solve a cell formation problem including setup time as one of the factors in their similarity coefficient based model.

2.7 Dynamic Cellular Manufacturing

The concept of the dynamic cellular manufacturing system (DCMS) was first introduced by Rheault et al. (1995). In the conventional CMS any changes in the product demand over time is ignored from product redesign and other factors. It assumes that the product mix and part demand is constant for the entire planning horizon. The product mix refers to a set of part types to be produced at each period. However, in the dynamic environment, a planning horizon can be divided into smaller periods where each period has different product mix and demand requirements. Kannan and Gosh (1995) noted that cells can evolve and dissolve on a dynamic, real time basis, through applying scheduling mechanisms enabling them to respond more effectively to changes in workload and to shifts in the locations of bottlenecks. Balakrishnan and Cheng (2007) conducted a comprehensive review of the research that had been done to address cellular manufacturing under conditions of multi-period planning horizons, with demand and resource uncertainties. They noted that in a study of 32 manufacturing cell life cycles at 15 plants by Marsh et al. (1997), it was found that layout changes could take place as soon as within six months of the start of the cell life cycle. Thus when manufacturing cells are created, expected changes in products and product mix have to be taken into consideration. Harhalakis et al. (1990) introduced a procedure to design robust CMS over a range of product demand variation. Chen (1998) was among the first who emphasized the importance of cell reconfiguration in a dynamic environment. He used a decomposition technique to decompose the original multi-period integer programming model into several single-period models which he would solve optimally through a commercial optimization software package. He then used dynamic programming (DP) to

re-integrate the smaller single-period problem solutions to obtain the best feasible solution for the original multi-period problem. Wilhelm et al. (1998) introduced a multi-period formation of grouping machine cell and part family. Tavakkoli-Moghaddam et al. (2005) and Safaei and Tavakkoli-Moghaddam (2009) developed a multi-period cellular manufacturing system for dynamic environments. Koren and Shpitalni (2011) studied the rationale of developing reconfigurable manufacturing systems. They discussed the core characteristics and design principles of reconfigurable manufacturing systems (RMS).

Mungwattana (2000) considered routing flexibility in dynamic environment with stochastic demand. Defersha and Chen (2008) suggest that cellular manufacturing system would be interrelated with production planning through dynamic system reconfiguration. They note that "in most research articles, CF has been considered under static conditions in which cells are formed for a single time period with known and constant product mix and demand" while, in a more realistic dynamic situation, a multi-period planning horizon shall be considered where the product mix and demand in each period may be different. Consequently, the cell configuration in one period may not be optimal in another period. Defersha and Chen (2006) mention that "As stated in a "US National Research Council document (National Research Council, 1998), reconfigurable manufacturing is considered by many manufacturing experts as one of the most important technologies in advanced manufacturing systems".

Jayakumar and Raju (2010) developed a multi-period, multi-objective non-linear mathematical CF model together and solved it with LINGO 11 commercial software for small and medium sized problems. For small size problems the problem could be solved optimally while for medium size problems solving the problem in a reasonably short

period of time was not possible. For example for a numerical example including 12 part types, 8 machines and three planning types the number of integer variables has been 504 and the total number of variables amounts up to 6125. The model incorporates real-life parameters like alternate routing, operation sequence, duplicate machines, product mix, product demand, varying batch size, processing time, machine capacity, and various cost factors.

Balakrishnan and Cheng (2005) suggested a two-stage procedure based on the generalized machine assignment problem and dynamic programming for cell formation problem under conditions of varying product demand. The objectives were to minimize the material handling and machine reconfiguration costs. They suggested a periodical reconfiguration when the cost-benefit analysis favors such a move. This way, the cellular layout will better fit the demand in each period and thus be more effective and agile during the planning horizon. In addition, they proposed examining multiple layouts when considering cell redesign in order to incorporate different qualitative and quantitative considerations. Such an analysis can also highlight the need for easily movable machines to ensure that cellular manufacturing is effective throughout the planning horizon.

2.8 Summary

In this literature review, different aspects related to the current research were reviewed in publications. First, a taxonomy of cell formation techniques was presented that provided a general review of CMS design and the corresponding cell formation approaches. Then the concept of setup time was traced in CMS related literature. More specifically,

sequence-dependent setup time was briefly reviewed in scheduling related publication, where it has gained significant attention from researchers.

A critical review of setup time in CMS related literature followed which debated the use of setup time in cell formation related works as well as examining the impact of CMS on setup reduction. It was argued that in order to achieve a controlled setup time reduction in CMS, more information than MCIM is required. The critical review of setup time related cell formation publications, revealed that different researchers have had different perceptions of setup time when using it in their cell formation process. Therefore a standard definition based on Cox et.al (1995) that was found to be conforming to the real manufacturing and common sense both, was established. In the continuation, other aspects of the subject of this thesis, i.e. application of heuristic methods especially genetic algorithm in cellular manufacturing and dynamic CMS were briefly reviewed. Finally Figure 2.3 provides a comparative view of the subject of this thesis and the closely related works in the literature and concludes the literature review.

Author (s)	Year	Manufacturing cell		Type of Cell		Type of Set up		Set up level			Type of Model		Solution Approach
		Single- Cell (Scheduling)	Multi- Cell	General	Pure Flow line	Setup cost	Sequence-dependent	Part Family	Component Part		Static	Dynamic	
									Part	Operation			
Damodaran et al.	1992		X	X		X				X	X		B&B
Rajamani et.al	1992		X		X		X		X		X		B&B
Atmani	1995		X	X		X				X	X		B&B
LaScola Needy et al.	1998		X	X		X			X		X		GA
Samaddar et al.	1999	X		X			X	X					CustomisedB &B
Schaller et al.	2000	X			X		X	X					Heuristic
Ohta &Nakamura	2002		X	X		X				X	X		GA
Lashkari et al.	2004		X	X		X				X	X		B&B
Franca et al.	2005	X			X		X	X					GA+MA
Murugan&Selladurai	2005		X	X		X				X	X		GA
Defersha&Chen	2006		X	X		X				X		X	B&B
Defersha&Chen	2008		X	X		X				X		X	GA
Lin et al.	2008	X			X		X	X					GA+SA+ TABU
Current thesis	**2012**		**X**	**X**			**X**			**X**		**X**	GA+DP

Figure 2.3- Comparing the thesis attributes with relevant the literature

CHAPTER 3

RESEARCH PROBLEM AND MATHEMATICAL MODELING

Repetitive manufacturing is one solution for situations where market demand warrants a large-scale production but market constraints prohibit continuous production. In a closed job shop, in which a fixed number of products are produced on a repetitive basis, when there are significant sequence-dependent setup times and costs involved, the cell formation problem should consider minimizing the sequence-dependent setup times in order to minimize the production cost (Rajamani et. al., 1992). Due to frequent changeovers of the machines in repetitive cycles within the planning horizon, the overall setup cost incurred could be quite considerable if it is not incorporated in the cost minimization process. To the best of our knowledge, the consideration of sequence-dependent setup time at the operation-level in the process of cost minimization of cellular manufacturing systems has not been addressed by other researchers. Furthermore Defersha and Chen (2008) suggest that cellular manufacturing systems may be interrelated with production planning through dynamic system reconfiguration. They note that in most research articles, CF has been considered under static conditions in which cells are formed for a single time period with known and constant product mix and demand while in a more realistic dynamic situation, a multi-period planning horizon shall be considered where the product mix and demand in each period may be different. Consequently, the cell configuration in one period may not be optimal in another period.

3.1 Description of the Research Problem

The research problem description can be best described through a real manufacturing example. Consider a part supplier to the automobile industry producing various component parts. The company has a policy for the first half of the year to provide a large number of a few products for the next six months while each of those products shall be delivered on a bi-weekly basis. This can be a result of a contractual obligation towards a specific car manufacturer or the auto industry as its whole market. This implies that the whole demand for the products shall be broken down to baskets of small batches of products which would be produced within repetitive cycles. The combination of all the repetitive cycles would meet the aggregate demand of the 6-month period. In the next 6-month period, the demand and the product mix would change, in a deterministic way, because of which we may need to reconfigure the manufacturing cells for the next planning horizon. Parts may have various operations on different machines and the operations of each part shall be processed in a pre-determined sequence, identified by their index numbers; thus the sequence of operations matters and shall be observed.

The processing order of the different parts in each cell and on each machine recommended by the solution of the model would minimize the amount of setup time and cost in each cycle together with other production related costs. The setup time is measured at the operation-level, that is, between every pair of different component parts in different operations. Each changeover from one part to another requires changing a set of adjustments which depends on not only on the next part but also on the set of adjustments done for the previous part as well. The setup time between identical

operations of a pair of identical parts is not considered as it is zero. The setup time between different operations of parts is obtained from the route sheets where other critical manufacturing information, such as sequence of operations, corresponding processing times, machine types, etc., is maintained.

Each machine has a limited capacity expressed in hours during each period. This implies that each machine type may have identical duplicates should it surpass its capacity limit in order to meet the corresponding demand. The objective is to design a cellular manufacturing system that simultaneously groups the machines and the component parts into cells so as to minimize the overall production cost including operation-level sequence-dependent setup time and cost, machine utilization cost, material handling (intercellular movement) cost and cell reconfiguration cost over the span of several time periods.

In achieving the aforementioned objective, there are several constraints and limitations that must be noted. Machine capacities cannot be exceeded and physical limitations shall be observed: cells can only include a certain range of machine types below or beyond which the cell is not viable. Furthermore, the model requires that operation j of any part shall precede operation $j+1$ of the same part on the same machine in the same cell.

3.1.1 Mathematical Model

Nomenclature

Indices:

i: index of the component part

j: index of the operation

p: index of the sequential position of processing the operations of component parts on a machine type

k:index of machine type

l: index of the cell

t: index of the time period

Parameters:

$Siji'j'k$ - *joint sequence - dependent setup time to replace operation j of part i with operation j' of part i' on machine k*

$B(t)$ - *number of part baskets produced in period t*

Ck - *unit cost of setup on machine type k*

Ek: - *cost of acquiring one unit of machine type k*

Ωk^{+} - *unit cost of installing machine type k*

Ωk^{-} - *unit cost of removing machine type k*

Ψ_i - *unit cost to move part i from one cell to another*

Φ_k - *unit operation cost on machine type k*

Γ_k - *capacity of one unit of machine type k*

Hi,j - *set of machine types on which operation j of part i can be processed*

θijk - *processing time of operation j of part i on machine type k*

$\eta_i(t)$ - *demand for part i in period t*

$MAXl$ - *maximum number of machines allowed in cell l*

$MINl$ - *minimum number of machines allowed in cell l*

Decision Variables :

$$x_{ijpkl}(t) = \begin{cases} 1 & \text{if operation } j \text{ of part } i \text{ visits machine type } k \text{ in position } p \text{ in cell } l \text{ in period } t \\ 0 & \text{otherwise} \end{cases}$$

$$\mu_{ijl}(t) = \begin{cases} 1 & \text{if operation subsequent to } j \text{ of part } i \text{ is done in another cell } l \text{ in period } t \\ 0 & \text{otherwise} \end{cases}$$

$y_{kl}(t)$ No. of machine type k in cell l during period t

$y^{+}_{kl}(t)$ No. of machine type k added to cell l in the beginnning of period t

$y^{-}_{kl}(t)$ No. of machine type k removed from cell l in the beginnning of period t

$$z_{ijl}(t) = \begin{cases} 1 & \text{if operation } j \text{ of part } i \text{ visits cell } l \text{ in period } t \\ 0 & \text{otherwise} \end{cases}$$

$$\beta_{ijkl}(t) = \begin{cases} 1 & \text{if operation } j \text{ of part } i \text{ visits machine type } k \text{ in cell } l \text{ in period } t \\ 0 & \text{otherwise} \end{cases}$$

Auxilliary Binary Variables :

$$\lambda_{ijpi'j'(p+1)kl}(t) = \begin{cases} 1 & \text{if operation } j \text{ of part } i \text{ immediately precedes operation } j' \text{ of part } i' \\ & \text{on machine type } k \text{ in cell } l \text{ in period } t \\ 0 & \text{otherwise} \end{cases}$$

$$\zeta_{ijp(j+1)p'kl}(t) = \begin{cases} 1 & \text{if position } p \text{ of operation } j \text{ of part } i \text{ precedes position } p' \text{ of operation } (j+1) \\ & \text{on machine type } k \text{ in cell } l \text{ in period } t \\ 0 & \text{otherwise} \end{cases}$$

$$\textbf{Minimize Z} = \sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{j=1}^{J_i}\sum_{p=1}^{P}\sum_{k=1}^{K}\sum_{l=1}^{L} B(t).\ \lambda_{ijpi'j'(p+1)kl}(t)\cdot S_{iji'j'k}\cdot C_k +$$

$$\sum_{t=1}^{T}\sum_{k=1}^{K} E_k \cdot \left(\sum_{l=1}^{L} y_{kl}(t) - \sum_{l=1}^{L} y_{kl}(t-1) \right) +$$

$$\sum_{t=1}^{T}\sum_{l=1}^{L}\sum_{i=1}^{I}\sum_{j=1}^{J_i} \Psi_i \cdot \eta_i(t) \cdot \mu_{ijl}(t) + \sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{l=1}^{L} (\Omega_k^{+} \cdot y_{kl}^{+}(t) + \Omega_{\bar{k}} \cdot y_{kl}^{-}(t)) \quad (3.1)$$

Subject to:

$$\lambda_{ijp\,i'j'(p+1)kl}(t) \geq x_{ijpkl}(t) + x_{i'j'(p+1)kl}(t) - 1 \quad (3.2)$$

$$\mu_{ijl}(t) \geq z_{ijl}(t) - z_{i(j+1)l}(t) \quad (3.3)$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J_i} x_{ijpkl}(t) \leq 1 \quad (3.4)$$

$$\sum_{l=1}^{L}\sum_{p=1}^{P}\sum_{k=1}^{K} x_{ijpkl}(t) = 1 \quad (3.5)$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J_i} x_{ijpkl}(t) \geq \sum_{i=1}^{I}\sum_{j=1}^{J_i} x_{ij(p+1)kl}(t) \quad (3.6)$$

$$\zeta_{ijp(j+1)p'kl}(t) \leq \frac{1}{2}(x_{ijpkl}(t) + x_{i(j+1)p'kl}(t)) \quad (3.7)$$

$$\zeta_{ijp(j+1)p'kl}(t) \geq x_{ijpkl}(t) + x_{i(j+1)p'kl}(t) - 1 \quad (3.8)$$

$$\zeta_{ijp(j+1)p'kl}(t).\ p' + (1 - \zeta_{ijp(j+1)p'kl}(t)).p \geq p \quad (3.9)$$

$$\sum_{k=1}^{K}\sum_{p=1}^{P} x_{ijpkl}(t) = z_{ijl}(t) \quad (3.10)$$

$$\sum_{k=1}^{K} \beta_{ijkl}(t) = z_{ijl}(t) \tag{3.11}$$

$$x_{ijpkl}(t) \leq y_{kl}(t) \tag{3.12}$$

$$y_{kl}(t) = y_{kl}(t-1) + y_{kl}^{+}(t) - y_{kl}^{-}(t) \tag{3.13}$$

$$MIN_{\,l} \leq \sum_{k=1}^{K} y_{kl}(t) \leq MAX_{\,l} \tag{3.14}$$

$$\Gamma_k \cdot y_{kl}(t) \geq \sum_{i=1}^{I} \sum_{j=1}^{J_i} \theta_{ijk} \cdot \eta_i(t) \cdot \beta_{ijkl}(t) \tag{3.15}$$

$$x_{ijpkl}(t),\ z_{ijl}(t),\ \lambda_{ijpi'j'(p+1)kl}(t),\ \mu_{ijl}(t),\ \zeta_{ijp(j+1)p'kl}(t) \in \{0,1\} \tag{3.16}$$

$$y_{kl}(t),\ y_{kl}^{+}(t),\ y_{kl}^{-}(t) \in \{0,1,2,..\} \tag{3.17}$$

Model Objective function

The objective function, Eq. (3.1), consists of four cost terms. The first term is operation-level sequence-dependent setup cost. Individual setup costs have been assumed to be a linear function of the corresponding setup times as in real manufacturing the cost is mainly associated with the time spent by the skilled worker to changeover. The second term is machine utilization cost. The third term represents the intercellular/material handing cost. The fourth term addresses the relocation (reconfiguration) cost.

Model Constraints

Eq. (3.2) forces the auxiliary variable $\lambda_{ijpi'j'(p+1)kl}$ to be positive, if and only if the corresponding pair of different operations appear in two consecutive positions *p* and *p+1* on the same machine. Eq. (3.3) enforces that the model counts a move only when part *i* exits the cell *l* in order to perform subsequent operation *j+1*. Eq. (3.4) ensures that two or more operations may not take place in the same position in the processing sequence of a given machine type in all cells. Eq. (3.5) enforces that each operation *j* of each part *i* shall choose only one machine type, out of the set of eligible machine types, to be processed on, hence no lot splitting. Eq. (3.6) prevents void positions in between the natural sequence of positions on each machine. Eqs. (3.7) - (3.9) observe that operation *j* of component part *i* shall precede operation *j+1* of the same component part on the same machine in sequence.

Eq. (3.10) and (3.11) relate the state of part *i* with respect to visiting cell *l* whether or not any operations of part *i* takes place on any machine in cell *l.* Eq. (3.12) relates the number of machine types in a cell to its corresponding visiting parts. Eq. (3.13) observes the machine type inventory balance in each period. Eq. (3.14) enforces the cell size limitations Eq. (3.15) respects the machine type capacity limitations. Integrality constraints (3.16) and (3.17) introduce the non-negative binary and general integer variables in the model.

3.1.2. Time complexity of the global model

Cell formation problems in general are known to be NP-complete (see for example Shtub, 1989), therefore exact approaches such as branch and bound algorithm would not be

computationally efficient. In LIP models, the number of the integer variables is the decisive factor in time complexity and computational burden of the model. If all n integer variables in our LIP model were binary, the maximum number of solutions representing the worst case complexity of the model, would be 2^n, an exponential function of the number of variables. To the best of our knowledge one of the most comprehensive mathematical programming CF model in CMS appears to be that in Defersha and Chen (2006) where the main theme of the paper is to introduce one comprehensive model that combines all the real manufacturing features previously introduced in separate models . Since the number of integer variables is the decisive factor in computation complexity of MIP models, we have counted the maximum number of integer variables in our model in Table 3.1 and those in Defersha and Chen (2006) in Table 3.2. As seen, the number of the integer variables in our model is significantly larger than those in Defersha and Chen (2006). This emphasizes the intense computational burden and time complexity of our model with respect to a comprehensive MIP model in the literature. In Tables 3.1 and 3.2, OP represents the sum of the number of operations of all parts.

Table 3.1- The number of integer variables in the global model

Variable	**VariableCount**
$x_{ijpkl}(t)$	$K \times L \times T \times P \times OP$
$\mu_{ijl}(t)$	$L \times T \times OP$
$y_{kl}(t)$	$K \times L \times T$
$y_{kl}^{+}(t)$	$K \times L \times T$
$y_{kl}^{-}(t)$	$K \times L \times T$
$z_{ijl}(t)$	$L \times T \times OP$
$\beta_{ijkl}(t)$	$K \times L \times T \times OP$
$\lambda_{ijpi'j'(p+1)kl}(t)$	$K \times L \times T \times (P-1) \times OP^2$
$\zeta_{ijp(j+1)p'kl}(t)$	$K \times L \times T \times P(P-1) \times (OP-I)$

Table 3.2- The number of integer variables in Defersha and Chen (2006)

$N_{kl}(t)$	Gen. integer	$K \times L \times T$
$y_{kl}^{+}(t)$	Gen. integer	$K \times L \times T$
$y_{kl}^{-}(t)$	Gen. integer	$K \times L \times T$
$r_{kl}(t)$	Binary	$K \times L \times T$
$p_{jkl}(t)$	Binary	$L \times T \times$ OP
$\xi_{k}(t)$	Binary	$K \times T$
$\beta_{ijl}(t)$	Binary	$L \times T \times$ OP

Our experimenting with LINGO, one of the available commercial optimization software packages which utilizes branch-and-bound and branch-and-cut algorithms, was not

encouraging as it failed to provide feasible solutions in a reasonable amount of time for most medium scale problem instances of the global model.

3.2 Special Case

In order to be able to experiment with LINGO, we first consider a special case of the global model which would simplify the model and reduce its complexity. Experimentation with LINGO proved to be able to provide feasible solutions in relatively short amount of time. The characteristics of the simplified model are as follows:

-The number of planning periods $t = 1$. This reduces the model from multi-period to single-period making the dynamic environment to turn static.

-Each part on any machine has at most one operation. This makes the model a BIP one.

The model has been further simplified by the following assumptions:

-Infinite capacity has been assumed for the machine types: Each machine type can process as many parts as is allocated to. This assumption will reduce the maximum number of the machines of each type in each cell to unity and thus will simplify studying the trade-off between number of machine types and setup time and drawing conclusions.

-The sequence of operation does not matter. This assumption simplifies the counting of the intercellular moves and subsequently the corresponding material handling cost.

3.2.1 Formulation of the Simplified Model

The formulation of the special case and simplified model is as follows:

Variables :

$$x_{ipkl} = \begin{cases} 1 & \text{if part } i \text{ visits machine } k \text{ in position } p \text{ in cell } l \\ 0 & \text{otherwise} \end{cases}$$

$$z_{il} = \begin{cases} 1 & \text{if part } i \text{ visits cell } l \\ 0 & \text{otherwise} \end{cases}$$

y_{kl} No. of machine k in cell l

$\lambda_{ipi'(p+1)kl}$ A binary auxilliary variable assuming the value of 1, IFF both x_{ipkl} and $x_{i'(p+1)kl}$ equal 1

Parameters :

$S_{ii'k}$ joint sequence-dependent set up time for replacing part i with part i' on machine k

C_k unit cost of setup on machine k

E_k cost of having one unit of machine k in the system

D_i cost of moving one unit of part i between cells

MAX_l maximum number of machines allowed in a cell

MIN_l minimum number of machines allowed in a cell

NOM_k available machine type k in the system

Given the above variables and parameters, since the objective is to minimize the overall setup cost, machine depreciation cost and material handling (intercellular movement) cost, then it would be expressed by the following function:

$$\sum_i \sum_p \sum_k \sum_l x_{ijkl} \cdot x_{i'(j+1)kl} \cdot S_{ii'k} \cdot C_k + \sum_k E_k \sum_l y_{kl} + \sum_i D_i \cdot (\sum_l z_{il} - 1) \qquad i \neq i' \qquad (3.18)$$

Since the quadratic function in (3.18) includes binary integers, it can be transformed to the following linear function by adding binary variable $\lambda_{ipi'(p+1)kl}$ and adding constraint (5) as follows:

$$\sum_i \sum_p \sum_k \sum_l \lambda_{ipi'(p+1)kl} S_{ii'k} \cdot C_k + \sum_k E_k \sum_l y_{kl} + \sum_i D_i \cdot \left(\sum_l z_{il} - 1 \right) \qquad i \neq i \tag{3.19}$$

$$\lambda_{ipi'(p+1)kl} \geq x_{ipkl} + x_{i'(p+1)kl} - 1 \tag{3.20}$$

Since we are dealing with sequence-dependent setup time, the sequence of changing over batches of parts on a machine affects the total setup time of the cellular configuration. Therefore index *p* specifies the position of parts in a sequence on each machine. Constraint (3.23) ensures that in each cell, on each machine, two or more parts cannot take over the same position in the processing sequence of their corresponding machine.

On the other hand, it is assumed that each part occupies one and only one position in the processing sequence of its corresponding machine in only one cell. This is restricted by constraint (3.24). Since the model seeks a minimum sum of setup times, it tends to break the link between pairs of interchanging parts and set them position-wise apart so as to further minimize the joint setup times. Constraint (3.25) is in place so as to ensure that we will not have such position voids. We should also relate machines and their corresponding visiting parts. Constraint (3.26) binds these two. Constraints (3.27) and (3.28) ensure that when a part does not visit a cell it will not be assigned to any position on any machine in that cell and vice versa. Finally the constraints (3.29) and (3.30) reflect limitations on minimum and maximum number of machine types in cells and the maximum number of each machine type available in the system respectively.

The linear integer programming model is as follows:

$$\sum_i \sum_p \sum_k \sum_l \lambda_{ipi'(p+1)kl} \cdot S_{i\,i'\,k} \cdot C_k + \sum_k E_k \sum_l y_{kl} + \sum_i D_i \cdot (\sum_l z_{il} - 1) \tag{3.21}$$

st.

$$\lambda_{ipi'(p+1)kl} \geq x_{ipkl} + x_{i'(p+1)kl} - 1 \tag{3.22}$$

$$\sum_i x_{ipkl} \leq 1 \tag{3.23}$$

$$\sum_p \sum_l x_{ipkl} = 1 \tag{3.24}$$

$$\sum_i x_{ipkl} \geq \sum_i x_{i(p+1)kl} \tag{3.25}$$

$$x_{ipkl} \leq y_{kl} \tag{3.26}$$

$$\sum_k \sum_p x_{ipkl} \leq M^{\infty} \cdot z_{il} \tag{3.27}$$

$$\sum_k \sum_p x_{ipkl} \geq z_{il} \tag{3.28}$$

$$MIN_l \leq \sum_k y_{kl} \leq MAX_l \tag{3.29}$$

$$\sum_l y_{kl} \leq NOM_k \tag{3.30}$$

$$x_{ipkl}, y_{kl}, z_{il}, \lambda_{ip\,i'(p+1)kl} \quad 0,1 \ \text{variable} \tag{3.31}$$

3.2.2 Numerical Example

Consider the following problem: Seven parts are selected to be grouped with the four different machine types in three different cells. Table 3.3 shows the corresponding machine-component matrix(MCIM). The joint changeover matrix of the above mentioned case is shown in Table 3.4 Table 3.5 indicates the number of units of machine types available in the system. It should be mentioned that for simplicity and without lack of generality the unit cost per setup time, C_k and the cost of machine, E_k have both assumed to be unity. This is just to minimize the effect of external factors such as $ values in this example, since our interest in this basic study is to behold the pure trade-off results. Furthermore, the part flow feature of the model has been neutralized in the first

stage of the experimentation, but included later. This implies that the last cost term in Eq. (3.21) has been initially excluded while a new constraint

$$\sum_{p} x_{ipkl} = \sum_{p} x_{ipk'l} \tag{3.32}$$

has been included in the formulation in order to prevent the part flow. Therefore the results show setup time of the system vs. numbers of machines required. This case is solved by LINGO 8 programming (LINDO Systems Incorporation). The experimentation by LINGO reveals that branch and bound and or branch and cut algorithms may not optimally solve even small problem instances of the simplified model in a reasonably short period of time. However, feasible solutions with upper and lower bound would be provided after a reasonably short period of time. On the other hand, when dealing with problem instances of the same size for the global model, commercial optimization software packages such as LINGO may not sometimes provide feasible solutions in a reasonably short amount of time. Naturally the primary step towards solving the global model was to examine the model in order to learn its characteristics through experimenting with commercial software. The result for this case is shown in Table 3.6. In Table 3.6 parts and machines are grouped in cells and the position (processing order) of parts on machines, from left to right, in each cell is such that the overall setup time of the entire cellular configuration/production subsystem is minimized.

Table 3.3-Machine-part Incidence matrix

	Part						
Machine	*1*	*2*	*3*	*4*	*5*	*6*	*7*
M1	1	1	1	1	1	1	0
M2	1	1	1	0	0	1	1
M3	0	1	1	0	1	0	0
M4	0	0	1	1	1	1	1

Table 3.4- Joint changeover matrix of the numerical example

Machine 1

Parts	1	2	3	4	5	6	7
1		11	23	18	15	19	N/A
2	13		25	24	19	26	N/A
3	20	28		34	25	13	N/A
4	14	21	15		16	18	N/A
5	11	22	10	15		17	N/A
6	16	12	14	21	18		N/A
7	N/A	N/A	N/A	N/A	N/A	N/A	

Machine 2

Parts	1	2	3	4	5	6	7
1		22	35	N/A	N/A	31	41
2	37		40	N/A	N/A	39	29
3	45	36		N/A	N/A	40	39
4	N/A	N/A	N/A		N/A	N/A	N/A
5	N/A	N/A	N/A	N/A		N/A	N/A
6	20	35	29	N/A	N/A		28
7	41	28	33	N/A	N/A	34	

Machine 3

Parts	1	2	3	4	5	6	7
1		N/A	N/A	N/A	N/A	N/A	N/A
2	N/A		23	N/A	25	N/A	N/A
3	N/A	23		N/A	28	N/A	N/A
4	N/A	N/A	N/A		N/A	N/A	N/A
5	N/A	20	26	N/A		N/A	N/A
6	N/A	N/A	N/A	N/A	N/A		N/A
7	N/A	N/A	N/A	N/A	N/A	N/A	

Machine 4

Parts	1	2	3	4	5	6	7
1		N/A	N/A	N/A	N/A	N/A	N/A
2	N/A		N/A	N/A	N/A	N/A	N/A
3	N/A	N/A		7	11	9	15
4	N/A	N/A	7		16	13	8
5	N/A	N/A	16	12		14	16
6	N/A	N/A	13	17	11		8
7	N/A	N/A	18	14	12	16	

Table 3.5- No. of machines available

M1	M2	M3	M4
3	3	2	2

Table 3.6- Three-cell configuration with minimum overall sequence-dependent setup time under machine restriction

	C1	C2	C3
M1	P5-P3-P6	P1-P2	P4
M2	P3-P6	P1-P2	P7
M3	P5-P3	P2	
M4	P3-P6-P5		P4-P7

To see the effect of machine restriction on the setup time for a 3-cell configuration, different scenarios have been considered as shown in Table 3.7. Figure 3.1 shows the impact of machine restriction on increasing the system setup time. In Table 3.7, overall setup time in the system decreases as the number of machines available in the 3-cell configuration increases. This is due to the fact that adding more machines to the pool of available machines increases the degree of freedom of the system in the search of lower setup time solutions. Thus, given a certain number of cells, the objective value for machine restriction-free scenario (in this case 12 machines) shall provide us with the lower bound for that cell configuration.

Table 3.7- Setup time under machine restriction for a 3-cellconfiguration

Obj.value	229	173	149	113
Avail. machines	8	9	10	12
Setup time	221	164	139	101

A series of situations ranging from 1 to 5 cell configurations have been solved for the model (Table 3.8). For this series of cell configurations, no preliminary machine

restriction has been imposed on the system, i.e. constraint (3.30) has been relaxed. This allows the model to obtain machine restriction-free results for a free trade-off between the number of machines and setup time in different scenarios or cell configurations. As is shown in Figure 3.2, as the number of cells increase, the setup time decreases while the number of machines required increases.

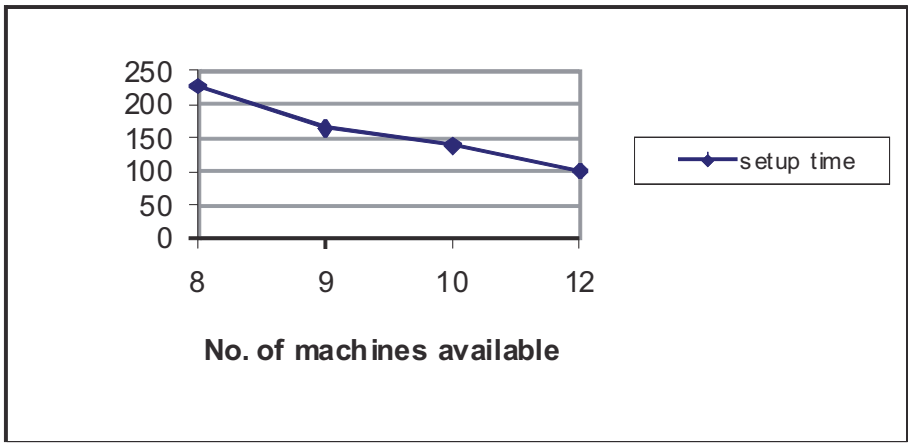

Figure 3.1-The impact of machine restriction on setup time in 3-cell

Table 3.8 - Trade-off between No. of machines, setup time in different scenarios

No. of cells	5	4	3	2	1
Obj.value	36	74	113	181	261
Req. machines	17	14	12	8	4
Setup time	19	60	101	173	257

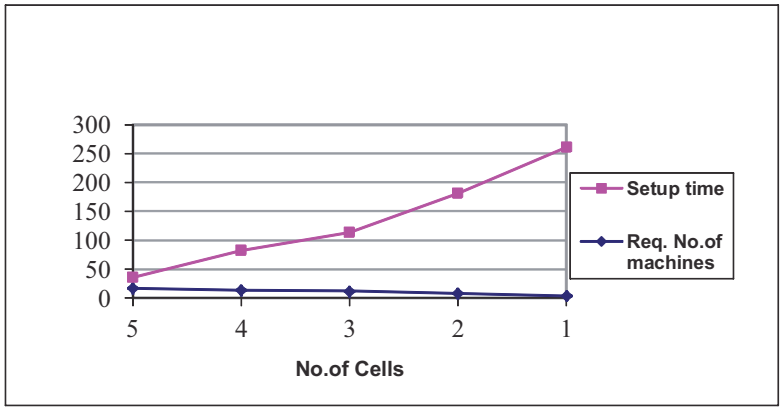

Figure 3.2- Setup time vs. machine trade-off for different cell configurations

To show the behavior of the three factors together, we have considered the complete model which includes the intercellular transfers. Table 3.9 represents the above example when solved for the case with intercellular movement. The result of which is shown in Figure 3.3. As expected the objective values in case with part flow are lower than that of the case with no part flow, since the model has the option to take advantage of the intercellular movement, should it help minimize the overall cost.

Table 3.9 -Trade-off between setup time, No .of machines and No. of movements

Number of cells	5	4	3	2	1
Obj.Value	32	68	110	178	261
Number of moves	3	5	6	4	0
Req.machines	18	15	12	8	4
Setup time	11	48	92	166	257

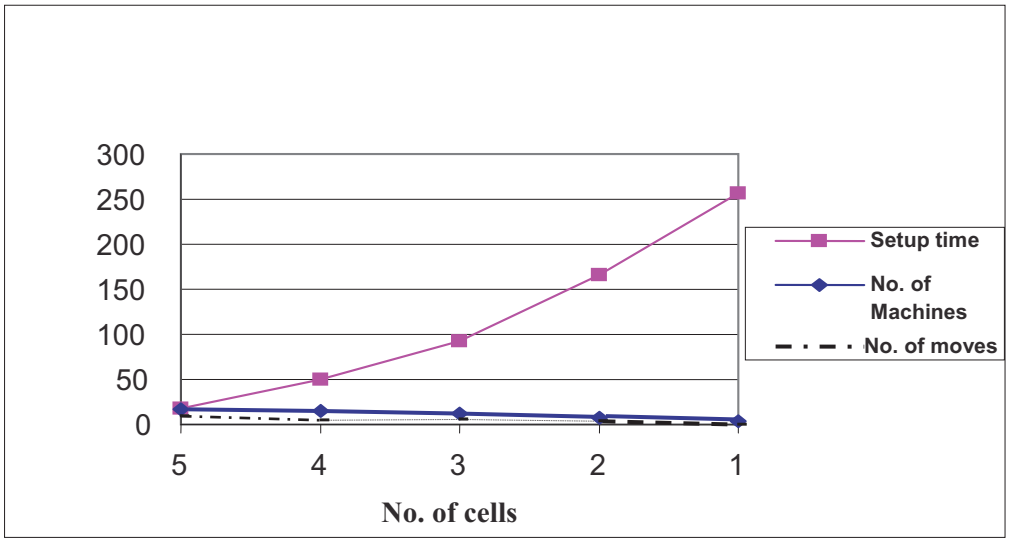

Figure 3.3- Setup time vs. No. of machines and No. of moves for different cells

3.3 Discussion and Summary

In this chapter, an integer programming model was presented that includes sequence-dependent setup time and cost as part of the overall production cost in manufacturing cells over multiple planning horizons in a dynamic environment. The developed model is flexible in the following senses: considering a sequence-dependent setup time does not restrict the application of the model to a limited number of situations where the setup times are sequence-dependent. On the contrary it just ensures that the model can handle situations where some or all setup times are sequence-dependent. In situations where sequence-independent times are available, $Sii'k = Si'ik$ and simply the same time would be recorded in the joint changeover input matrix when the incoming and outgoing operations switch places. Additive setup times would also easily fit in the input matrix of

the model since they are another version of sequence independent setup time. This makes the presented model a general one and the most inclusive when setup times are deemed significant. Another feature of the model is the ability of optimal multi-period production planning in a dynamic environment when the product mix would change from one planning horizon to another.

In order to be able to freely study the basic characteristics of the model we further considered a special case assumptions of which would simplify the global model. The simplified version was experimented with LINGO optimization software and preliminary conclusions were drawn. These conclusions provides insight on the characteristics of the simplified model, which in turn helps with developing a pilot GA-based heuristic to solve the simplified version before the global model is tackled. The above conclusions will be helpful especially in generating initial feasible solutions in the GA-based heuristic.

CHAPTER 4

SINGLE-PERIOD SOLUTION APPROACH: GA-BASED HEURISTIC

The cell formation problem has been shown to be NP-complete by several researchers. One such example could be found in Shtub (1989) where the CF problem was modeled as a generalized assignment problem. This suggests that enumerative algorithms such as branch and bound (B&B) would not be computationally efficient in solving these problems. In a BIP model as in the special case introduced in Chapter 3, exponential number of iterations, 2^n , would be required in the worst case. As was discussed in Chapter 3, the intensity of the computational burden of the global model resulted in the inability of the optimization software to provide optimal or in some cases even a feasible solution for medium sized problem instances of the global model. Therefore it was decided that primarily a special case of the model that would lead to a dimensionality collapse in the global model and would alleviate the computational burden be considered for the purpose of ease of the driving on the solution approach. The simplified version of the model represented by the special case would still be NP-hard, however, it would bring about a dimensionality collapse in the global model which makes it less complex in comparison to the global model.

Due to the NP-hard nature of the problem, approximate methods shall be considered if a reasonably short amount of computation time is in quest. A variety of researchers have applied GA to CF in CMS (see for example Wu[a] et al., 2007; De Lit et al. 2000, Mahdavi

and Mahadevan, 2008, Wu[b] et al. 2007, Defersha and Chen, 2008, Goncalves and Resende, 2002, Safaei and Tavakkoli-Moghaddam, 2009, Tavakkoli-Moghaddam et al., 2005). Murugan, and Selladurai (2005) applied genetic algorithm to a cell formation problem that would reduce the setup time, however they used group efficacy criteria approach rather than mathematical programming and the setup time, while not being sequence-dependent, were measured on two different machines as opposed to a common machine. Gosh et al. (2011) conducted a state-of-the- art generic review of application of various meta-heuristics in cellular manufacturing. Ahmed and Tavakkoli-Moghaddam (2004) compared various heuristic methods including GA in solving cellular manufacturing problems in dynamic environment. In this thesis, based on the fact that GA has a good record of successful application in the literature in application to CF problems, genetic algorithm has been chosen as the platform for developing a tailor-made heuristic as part of the solution approach. Due to the novelty of the model itself to the best of our knowledge, the GA application to the model would also be novel.

4.1 Genetic Algorithm

Genetic algorithm (GA) was first developed in 1970s by Holland (1975) and has gained ever increasing attention and application in solving many combinatorial optimization problems. A number of works are also published indicating the development and use of genetic algorithm for various problems in different disciplines. Goldberg (1989) provided a good introduction on the fundamentals of genetic algorithms and Gen and Cheng (1996) elaborate on the use of genetic algorithm in engineering design problems (Defersha and Chen, 2008).

Genetic algorithms are intelligent search algorithms inspired by the mechanics of natural selection and natural genetics. They combine survival of the fittest among string structures, namely chromosomes, with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search. In every generation, a new set of artificial creatures (strings) is created using bits and pieces of the fittest of the old, namely genes. While randomized, genetic algorithms are no simple random walk. They efficiently use historical information to speculate on new search points with expected improved performance (Goldberg, 1989). GA may be considered as a controlled evolutionary stochastic search which follows the principles of natural selection and genetics. Genetic algorithms start from an initial solution which is either available or may be made up, presumably randomly, then start evolving the initial solution to a series of solutions which are consistently fitter with respect to a certain measure, thence the term evolutionary algorithms (EA). The general evolutionary procedure is shown in Figure 4.1.

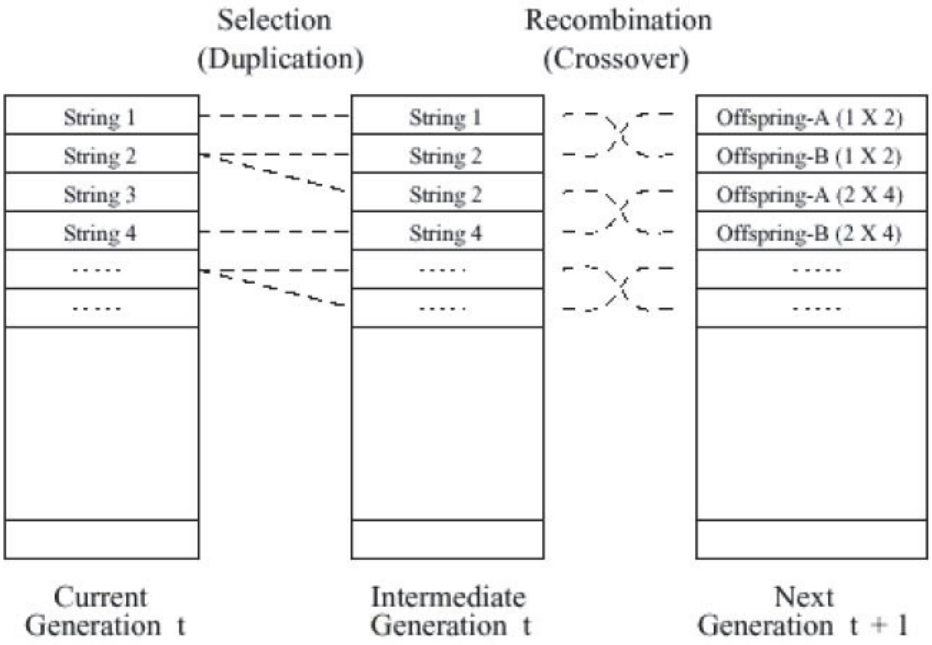

Fig. 4.1-Graphical representation of evolutionary process in GA
(www.scribd.com,)

Each string, namely chromosome, contains various information, namely genes. As an example in a linear binary integer programming optimization problem, the genes which carry the information consists of 0-1 integers and each chromosome (string) represents a solution vector in the *n* dimensional solution space. During the selection phase, the fitter strings - those solutions which better serve the objective function- shall have more chance of being selected for recombination. As shown in figure 4.1, string 2 has been selected twice most likely due to its being a better fit than strings 2 or 4.

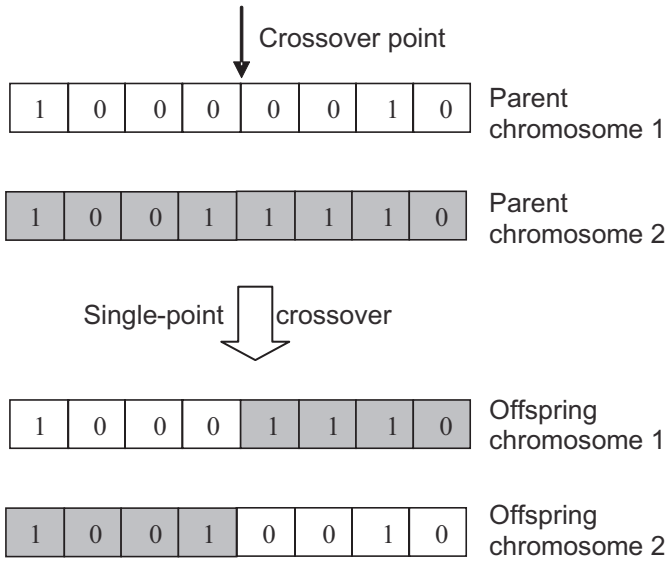

Fig. 4.2-Mechanics of a single point crossover
(www.cs.kent.ac.uk/people/staff/aaf/)

The selected strings form the mating pool, where they recombine to generate a population of offsprings representing the next generation. Usually this recombination occurs in the form of crossover between a pair of chromosomes. Various forms of crossover may be considered the simplest of which would be the single point crossover. As depicted in Figure 4.2, in a single point crossover, once the crossover point is determined randomly for each pair, the genes to the right of each chromosome are swapped with that of the mate.

4.2 Simplified Model: Characteristics of the GA-based Heuristic

4.2.1 Chromosomal Encoding

As was discussed in the introductory of stage of this Chapter, using a GA platform, first a heuristic for the simplified version of the global model will be developed in order to

assess the complexity of the problem and gain insight in tackling the solution of the global model thereafter. Chromosomal encoding is the first step in developing any heuristic in a GA platform. The chromosomal structure of the improvised GA-based heuristic for the simplified version of the global model is depicted in Figure 4.3. As shown, in this specific encoding, machine, encompasses the cell as opposed to the physical reality. This specific structure has been tailored in order to facilitate the handling of the constraints during the recombination process. The interpretation of the Figure 4.3 would be as follows:

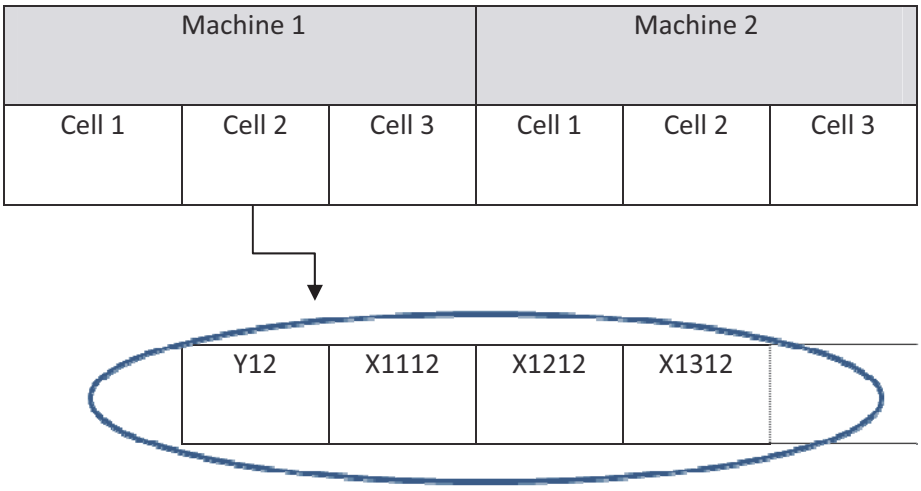

Fig 4.3-Chromosomal encoding for the simplified GA-based heuristic

The 0-1 decision variable, y_{kl} , indicates whether cell *l* is contained by machine *k* or as in terms of physical reality whether machine type *k* is in cell *l* . When cell *l* is contained within machine *k*, then the binary decision variable, x_{ipkl} , indicates whether part *i* would

lie in that contained cell in position p. The aforementioned decision variables are read directly from the chromosomes. Other variables such as z_{il} are indirectly decoded.

4.2.2 Initia l Solution

The initial population is being generated as follows briefly: Starting from machine type 1, part number 1 is tried for the chance to appear at position 1 in cell 1 on machine 1; then part number 2 would be tried for that position and so on. Then for the second position, the unallocated parts will be tried again. The assignment of the parts is a process without replacement in order to observe the corresponding constraints in the model. This process continues until all eligible parts are assigned to the cells on their corresponding machines. MCIM is the main tool for constraint handling at this phase.

4.2.3 Constraint Handling

Genetic algorithm is a bi-faceted meta-heuristic which relies on a combination of stochastic search and evolutionary process. Neither of the above two properties can be compromised without adversely affecting the quality of the search. This means that forcing deterministic decision variable values could drastically downgrade the GA-based heuristic. However maintaining the feasibility of the evolved solution population in each generation requires that the constraints of the problem are fully observed. This in turn necessitates the use of some repair procedures or repair heuristics that would retain the feasibility of the solution when the random numbers cause some constraints to be violated. However the extent and the nature of the repair heuristics can adversely affect the quality of the intelligent search process in the GA–based heuristic. Due to this, in the current GA-based heuristic it has been tried to minimize the extent of the repair

procedures. One such example would be consideration of a suitable proper chromosomal encoding, that is machine encompassing the cell as opposed to the physical reality, which drastically reduces the extent of the repair needed after recombination. Also proper genetic operators have been developed to cut the chromosomes at points where minimal distortion is caused to the feasibility of the chromosome.

4.2.4 Genetic Operators

Genetic operators play the main role in evolutionary process of the chromosomal population by generating promising solutions. The genetic operators designed for the GA- based heuristic are explained as follows.

Selection operator

The selection process is done through a *biased roulette wheel*. For the current research problem, in order to accommodate a fair selection mechanism that properly would serve the purpose, a frequency is defined for each chromosome. This frequency would be the percentage of the fitness of the chromosome with respect to the total fitness of all chromosomes using Eq.(4.1) where f_i is the reciprocal frequency, and F_i is the Fitness of chromosome i, here the objective function value, respectively.

$$f_i = \left\lceil \frac{\sum_i F_i}{F_i} \right\rceil \tag{4.1}$$

The reciprocal frequencies are then normalized to integers Nf_i and the pseudo random numbers generated by the C++ random function within the range of minimum and

maximum of the cumulative frequencies, $\left[0, \sum_i f_i\right]$, would select the parent in each generation, with replacement, based in which cumulative frequency bandwidth it would fit. When the frequency values of the chromosomes are added up in the order of the chromosome numbers, the frequency bandwidth represents the position of the frequency value of each chromosome in the cumulative frequencies as shown in Figure 4.4. Since the frequency bandwidths are proportional to the fitness of different chromosomes, those with higher frequency will have a larger bandwidth, thence a higher chance to be hit by the pseudo randomly generated numbers providing the necessary discriminatory selection of the chromosomes inspired by the principle of natural selection. In Figure 4.4, the frequency of chromosome 5 has been 3, starting from point 18, which is the cumulative frequency of all chromosome before chromosome 5. if the randomly generated number is any integer $\rho \in [18, 21)$, then chromosome number 5 will be selected.

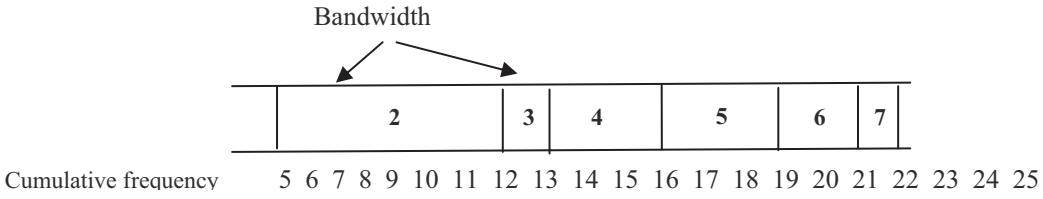

Fig. 4.4 – Selecting process discriminated by the frequency bandwidths

Crossover operators

Crossover operators generate offspring of the next generation by recombining the selected parents through swapping genes between pairs in the mating pool. Two types of crossover operators have been considered for the simplified model: machine-level and part-level.

Machine –level crossover: For this operator a single-point crossover has been considered. First a machine type number is randomly selected as the single crossover point. Then between a pair of selected parental chromosomes in the mating pool, all the content of the two chromosomes at the machine level of chromosomal structure will be swapped from the selected machine to the right.

The advantage of the above machine–level crossover is that the feasibility of the chromosomal solution will not be compromised or disrupted therefore constraint handling controls will not be required. The drawback however is that machine-level crossover is a high-level crossover which does not interchange the information of the chromosomes at low levels.

Part-level operator: The part-level operator randomly selects a part in the pair of selected parental chromosomes and interchanges all the information relating to this part, including the cell and the position on the machine between the two selected parental chromosomes. This operator requires constraint handling controls to maintain and or regain the feasibility of the chromosomal solution during or after the operator functions.

Mutation

In order to introduce a new gene to the chromosomal population, mutation operator need to be developed since it is the only source that can diversify the exploration thus preventing the solution procedure from premature convergence to a local optima. If we consider the search process as a golf ball, mutation acts as hitting the running ball to move the ball past the local optima toward the global optimum. However if the impulse is too forceful, it may overshoot the search ball past the global optimum too. For this reason, mutation operators shall be used with caution which implies a low probability rate

of application in the search process shall be used. For this purpose, the developed mutation operator is applied only to a small percentage of the offspring so as to avoid overshooting of the search area while preventing premature local optimization either.

The mutation operator in the GA-based heuristic for the simplified version of the global model randomly picks a machine type and randomly alters the information regarding the parts in the cells and their positions on the selected machines by randomly regenerating the information. This mutation takes place with very small likelihood and affects only one of the two mating parental chromosomes that have been drawn for mutation.

Rejuvenation

After a certain number of consequent generations, where no improvement is made in the best objective value (fitness), we may assume that the population is in the neighbourhood of a promising area. In order to focus and reduce the diversity, certain number of chromosomes of the current generation will be replaced by some of the best individuals found so far.

The simplified model GA-based heuristic is featuring a full rejuvenation procedure where after every g_{max} generations, the population of the last generation will be replaced with a newly generated population and resumes the algorithm R rounds, parameter R being the number of the full rejuvenations considered before termination of the search.

4.2.5 Computational Performance

In order to test the functionality and evaluate the computational performance of the GA based heuristic, 6 problem sizes each with 6 data sets were considered which results in 36 test problem instances. The 6 problem sizes were chosen to include 6,7,8,9,10,12 and 15

parts. The sizes were adopted based on experience, such that they could be handled by LINGO in order to make the comparison possible. For each problem size, six different types of input data were used. Input data consist mainly of MCIM and corresponding setup times which were generated by pseudo random function in C++ programming language to maintain a random and unbiased distribution of the input data for the problem instances.

Other parameters are common among all problem sizes: parts are selected to be grouped with four different machine types in three different cells so as to minimize the overall cost of setup, machine and intercellular movement. Besides, while the cell upper bound and lower bound on the number of machines are 4 and 1 respectively, constraint 12 has been relaxed. These problem instances were run on LINGO 9 (LINDO Systems Incorporation) on a 2 MHZ PC. The setup times were generated pseudo randomly within the range of 10 and 40 minutes while the MCIM is formed by randomly picked binary variables. As for the GA based heuristic for simplified model, the termination criteria were set at 120 generations after which the program would terminate the computation. The best objective values from the GA were picked after 10 rounds of full rejuvenation of the program and the computation time was measured against the generation in which the best objective was achieved for the first time during the run. Table 4.1 shows the results from GA based heuristic in comparison with LINGO.

Table 4.1-Results for GA-based heuristic and LINGO for 36 problem instances in 6 sizes

	GA		LINGO			GA		LINGO	
	Best obj	Time	Best obj	Time		Best obj	Time	Best obj	Time
7 parts		sec		hrs	**10 parts**		sec		hrs
	50	*0.250*	50	1		124	*0.219*	140	*1*
	22	*0.250*	22	*		83	*0.218*	92	*1*
	11	*0.234*	10	*		122	*0.203*	120	*1*
	22	*0.250*	21	*		166	*0.219*	189	*1*
	60	*0.156*	63	1		96	*0.219*	111	*1*
	32	*0.250*	32	1		117	*0.218*	129	*1*
8 parts				1	**12 parts**				*1*
	63	*0.782*	63	1		214	*0.203*	236	*1*
	118	*0.500*	175	1		112	*0.297*	123	*1*
	76	*0.469*	76	1		122	*0.297*	133	*1*
	46	*0.500*	44	1		156	*0.281*	152	*1*
	55	*0.515*	55	1		256	*0.281*	256	*1*
	75	*0.781*	74	1		142	*0.328*	159	*1*
9 parts				1	**15 parts**				*1*
	95	*0.127*	107	1		347	*0.764*	**	*1*
	84	*0.172*	105	1		237	*0.503*	216	*1*
	97	*0.172*	116	1		366	*0.331*	**	*1*
	108	*0.187*	114	1		344	*0.489*	379	*1*
	80	*0.172*	86	1		298	*0.564*	336	*1*
	125	*0.141*	150	1		307	*0.437*	393	*1*

*Optimal solution was found by LINGO
**LINGO did not provide a feasible solution within 1 hour of computation time

The results from the GA based heuristic tailored for the current model deem reasonably good delivered in a short period of time as compared with those from LINGO within 1 hour of computation. On average, over 80% of the time, the GA-base heuristic has either outperformed or equalled the best objective values provided by LINGO within 1 hour of computation time. The fact that the heuristic has delivered results in a very short amount of computation time provides a practical tool to solve different problem instances of the presented model. However improvement is needed when developing the GA-based heuristic for the global model.

4.3 Global model: Characteristics of the GA-based Algorithm

4.3.1 Chromosomal encoding

The chromosomal structure of the improvised GA-based algorithm is depicted in Figure 4.5. Again in this encoding, machine, encompasses the cell as opposed to the physical reality. This specific structure has been tailored in order to facilitate the handling of the constraints during the recombination process as was explained in 4.1.2. The interpretation of Figure 4.5 in physical reality would be as follows: when Machine 1 lies in cell 1 in period 1, the binary decision variable, x_{31}, indicates that operation *1* of part *3* lies in that contained cell in position p_1 Also decision variable $y_{11,}$ indicates the number of machine type 1 in cell 1.

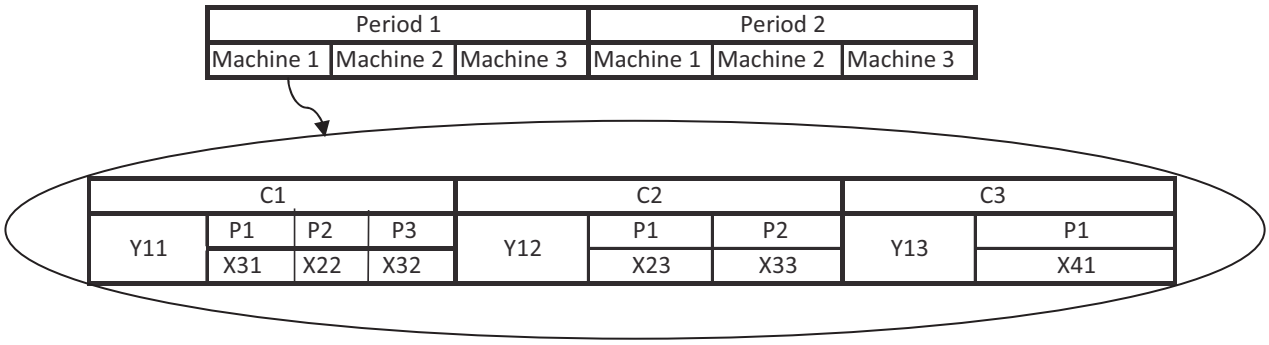

Fig. 4.5- Chromosomal encoding of the GA-based

4.3.2 Initia l Solution

The initial solution generator in the global GA-based heuristic is fundamentally different from that of the simplified model GA-based heuristic. The shortcoming of the simplified model GA-based heuristic was found to be the following: While once a part was not nominated for a position on a machine in the first cell, the second part was tried and so on. When the first position was eventually occupied by a part, the second position was tried for all the other parts regardless of whether they had failed their chance for the first position. This procedure found to give more chance to the initial cell to contain more parts than other cells, consistently leaving other cells with fewer parts or even empty. This would lead to a biased allocation of parts to the cells which in turn would deny the homogenous distribution of the parts in the cells.

In the initial population generation function of the global model heuristic, this deficiency was remedied as follows: first, it is determined randomly whether a certain machine will go to a certain cell. Once that takes place, it will then be determined, randomly, whether a certain part and its corresponding operation will be processed in a certain position (sequence) on that machine in that cell. The process continues until all parts and their corresponding operations are assigned to their corresponding machines in the cells. This change in the initial solution generator had quite an impact in diversified solution generating that would better explore the solution space of the problem, thus providing better chance for more accurate results to generate. The process is continuously monitored by MCIM and controlled by relevant constraints feasible initial solutions but no repair is required at this phase.

4.3.3 Constraint Handling

Constraint handling in the global model GA-based heuristic is much more complex due to the drastic increase in the number of the constraints as well as the number of the indices in the global model. These controls will either maintain the feasibility or regain it, as applicable, based on the type of the genetic operator affecting the genome of the chromosomes. Whenever a genetic operator requires constraint handling mechanism, it will be mentioned therein.

4.3.4 Genetic Operators

Selection operator

The selection process is done through a biased roulette wheel and in the same way that was explained for the simplified model GA-based heuristic.

Crossover operators

In the global GA-based heuristic, four crossover operators have been designed: period-level, machine-level, part-level and operation-level crossover operators.

Period swap operator: The period-swap operator works only when a multi-period solution is sought. It randomly picks a period and then interchanges that period portion of the chromosomes of a selected pair of parents.

*Machine swap operator***:** The machine-swap operator randomly selects a machine type and interchanges all the cell genes and its corresponding sub genes between the selected pair of machines.

Part-swap operator: The part-swap operator randomly picks a part and exchanges the corresponding information of that part between the two selected parents.

Operation-swap o*perator:* Finally the operation-swap operator randomly selects an operation of a selected part and interchanges the operation information including the cell and position between the selected parental chromosome pair. This operator performs at the lowest level and thus plays a critical role in diversifying the search process.

Part-swap and operation swap are the most challenging genetic operators in terms of constraint handling. These operators completely violate quite a few constraints as the order of the information observing the constraints will be disrupted. The swap requires that parts and their corresponding operations change their cell to find a position in the swapping cell. However entering a new cell, the swapped operation may face many challenges. One such challenge would be that the sought after position does not exist in the host cell. Another one is that the swapped operation shall assume a position that does not violate Eq. (3.7)-(3.9).

Mutation operator

For the global model GA-based heuristic, the developed mutation operator is different from the one used in the simplified model heuristic. The designed operator alters the position of the randomly selected operation of selected parts by stepping up or down the position number of the operation by unity. This will lead to interchanging the position number of the selected operation with that of a neighbouring operation on the corresponding machine in the cell. It should be first checked whether at least one neighbour exists, or the operation is a solo one in that cell on the corresponding machine. If so, the mutation will not apply to that particular operation. In case the operation has both neighbours on its left and right, then one of them will be randomly picked. It is important to maintain the randomness of the heuristic as much as possible and not

determine the neighbour by a pre-planned procedure. Constraint handling controls have been considered to guarantee that the feasibility of the solution chromosome involved will be maintained after the mutation.

Rejuvenation

Unlike the simplified model heuristic, the global GA-based heuristic is featuring two rejuvenation procedures. The two types of rejuvenation consist of partial rejuvenation and full rejuvenation. As for partial rejuvenation, after any σ successive generation without incumbent fitness value being improved, m best chromosome of each of the past n most recent generations will replace $m.n$ individuals of the current generation and mix up with the rest of the current population so that: $\sigma \geq m.n$ where σ is the population size and m and n are the parameters of the devised GA-based algorithm. This procedure boosts up genome and centers the search around a promising solution by refreshing the stalled population thence rejuvenation. Also after g_{max} generations, a full rejuvenation will occur which replaces all the population of the last generation with a new population and resumes the algorithm, parameter R being the number of the full rejuvenations. Figure 4.6 depicts the flow chart of the main steps in the global model GA-based heuristic.

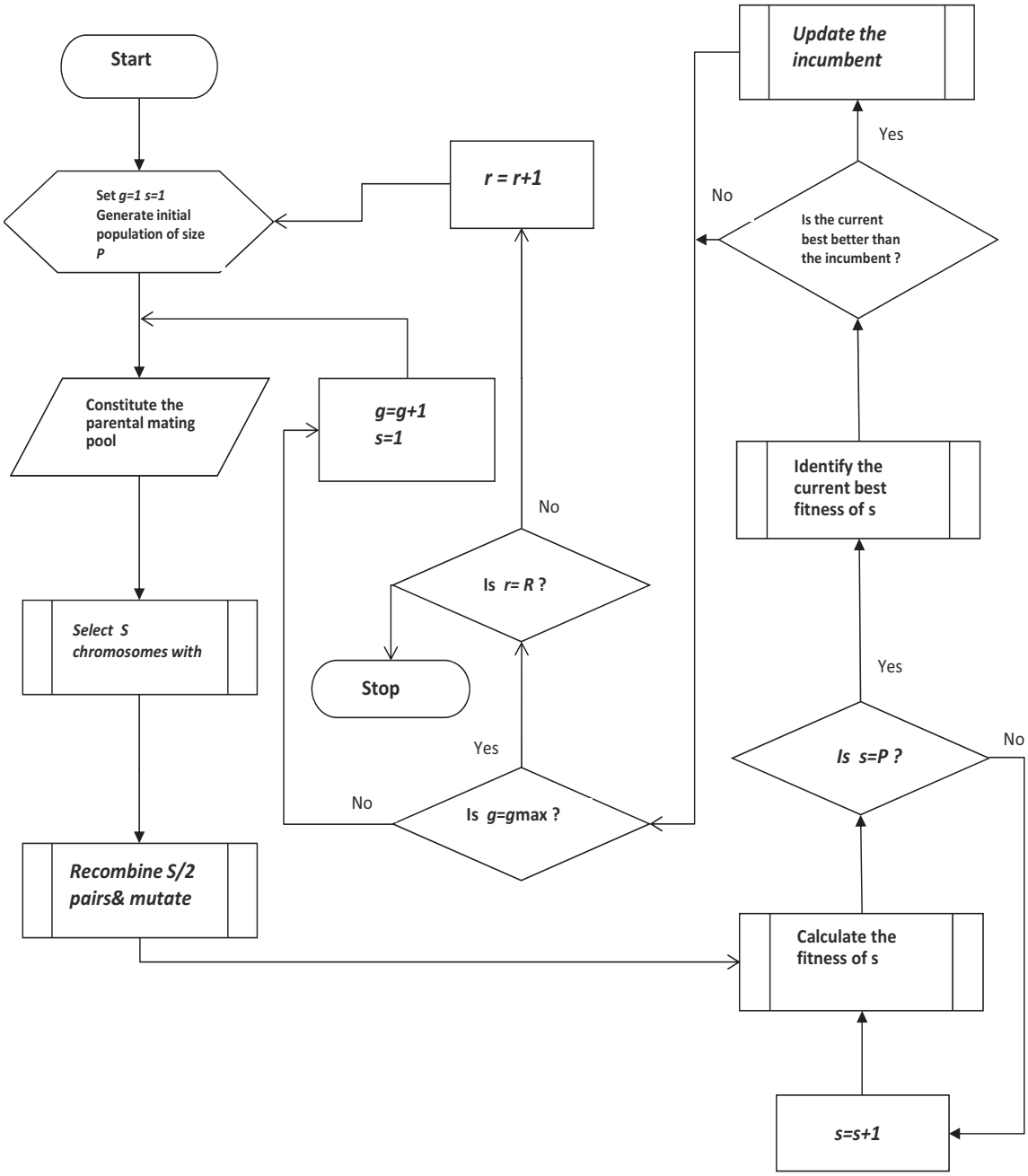

Figure 4.6 -Main steps in the current GA-based heuristic

4.3.5 Numerical Example

Consider a part supplier to the automobile industry, producing various automobile components. The company has to produce large quantities of a few selected products for the next six months where each of those products will be delivered on a bi-weekly basis. This can be the result of a contractual obligation towards a specific car manufacturer or the auto industry as its whole market. This implies that the whole demand for the products will be broken down to *baskets* of batches of products, as shown in Figure 4.7, which would be produced within repetitive cycles. The fixed number of the baskets to be produced in each planning horizon is determined by the delivery policy and market requirement. For example in a six month planning horizon a bi-weekly delivery of parts could imply twelve baskets, given the working calendar. The number of units of each part type in the basket represent the batch size of that part. The relation between the number of the product baskets produced in period *t*, *B(t),* and the batch size, *h(i), are* denoted in Eq. (4.2) and Eq. (4.3).

$$B(t)=T/w \tag{4.2}$$

$$h(i) = \left\lceil d(i,t)/B(t) \right\rceil \tag{4.3}$$

where *T* is the length of period *t*, *w* is the production cycle or equivalently the production lead time based on delivery terms and *d(i,t)* represent the demand of part *i* in period *t* respectively.

The combination of all the repetitive cycles would meet the aggregate demand of 6 month planning horizon. In the next 6 months, following the first 6-month period, the demand and the product mix may change, because of which we may need to reconfigure the

manufacturing cells for the next planning horizon. The processing order recommended by the solution of the model would minimize the planned production cost including the amount of setup time and cost in each cycle. Twenty-five parts and a maximum of 9 operations for each part are to be simultaneously grouped with six different machine types in three manufacturing cells assigned. All constraints and features of the research problem apply to this numerical example. We are seeking the solution of this problem for two 6-month periods. Table 4.2 includes the data relating to the parts. Tables A.4.1, A.4.2.1-.4 and A.4.3 (Appendix) show the MCIM, setup times and processing times for the operations of these parts. The processing times and the setup times, have been pseudo randomly generated within the range of 1 to 6 and 10 to 40 minutes respectively. Parts have different number of operations on different machines based on the MCIM. In table A.4.1, the data regarding MCIM the digits separated by comma form a triplet, each of which indicate the machine number, the part number and the operation number respectively. For example the first triplet in Table A.4.1, i.e. 1,1,1 implies that on machine 1, part 1 has its operation 1 to be processed and so on. In Table A.4.3, however, the digits separated by comma, form a quadruplet, each of which indicate the machine number, the part number, the operation number and processing time in minutes, respectively. Note that in this table, the ordinal numbers of machine types, part types and their corresponding operation exceptionally start from 0 rather than 1. Table A.4.2.1- 4, show the sequence-dependent setup times between various operations of the parts. This table has to be interpreted with respect to MCIM, Table A.4.1.The first triplet on machine 1 in MCIM (Table A.4.1) will have a changeover time with all other triplets on the same machine i.e. with the second one, third one, etc. Then the second triplet on that machine,

will have changeover times with the first one, the third one, etc. and so on. Therefore number of the possible changeover times for machine 1, would be *q(q-1), q* being the total number of the triplets (operations) on machine 1.

Since the parts are being moved around in batches rather than units, intercellular moving cost for each unit has been calculated by dividing the cost estimates of handling a batch of the part in each cycle by the number of the parts in each batch.

Table 4.3 indicates the parameters relating to the machine types. Machine capacities are in terms of hours of availability in each period. Duplicates of machine types may be needed in a cell if the capacity of one unit is not enough to fulfill the planned obligations. It has been assumed that the removal of a machine would cost virtually the same amount as re-installing it in a cell. Machine cost represents the utilization (depreciation) cost of the machine for one period.

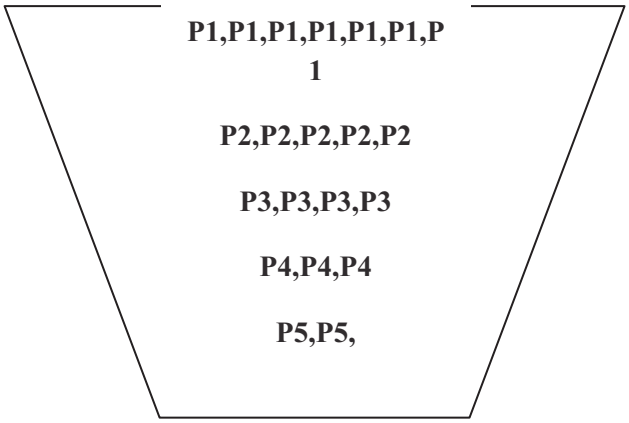

Fig. 4.7 -Product basket

Table 4.2-Part data

Part	Demand during period t		Intercell
	t=1	*t=2*	cost
1	1200	0	0.5
2	1000	0	0.5
3	0	1480	0.5
4	0	1800	1
5	3200	2800	0.5
6	1500	0	1
7	1680	0	0.5
8	2200	0	1.5
9	0	1780	1
10	980	0	0.5
11	0	900	0.5
12	740	1350	0.5
13	1800	1400	0.5
14	860	0	1
15	0	2500	1
16	0	960	0.5
17	0	1000	1.5
18	0	1700	1.5
19	1200	2000	1
20	1520	1800	0.5
21	2200	0	0.5
22	0	0	0.5
23	0	0	0.5
24	0	1890	1
25	0	1400	1.5

Table 4.3-Machine data

Machin type	Capacity (hrs/period	Utilization cost ($ per period)	Reconfiguration cost ($)	Setup cost($)
1	1200	1250	75	40
2	1200	1180	100	35
3	1200	1000	140	38
4	1350	1120	90	35
5	1250	1720	80	28
6	1200	1980	120	34

Table 4.4- The overall cost of the two period solution and its corresponding cost terms

Objective value ($)	Reconfiguration. cost($)	Machine cost($)	Setup cost($)	Intercell cost($)
65460.7	1235	32270	20010.7	11945

Table 4.4 shows the amount of overall production cost of an obtained solution by the GA-based heuristic and its components. Table 4.5 and 4.6 provide the assignments of the operations of each part to cells on their corresponding machines. These tables indicate the recommended machine-cell configuration and part-operation families in each cell in each period.

The number of the duplicates of each machine type in each cell have been indicated under "Qty" column. The triplet inside each parenthesis indicates the part number, the operation number and its processing order (position) on its corresponding machine in the assigned cell. Note that the numbering of the positions only start from 0 rather than 1. For example in Table 4.5, in cell number 3 (C3), only one unit of machine type number 3 (M3), is required, which processes operations number six of part 13, three of part 5, seven of part 13, three of part 1 and eight of part 5 in the above order while observing the constraints of the global model. In the same period in cell number 2, however, we have one unit of machine types 2 and 4 each, processing operation one of part 12 and operation two of part 2 respectively.

Table 4.5- Part-cell assignment for period 1

Cell	Machine Type	Machine Qty	(Part, Operation, Position)									
C1	M1	0										
	M2	0										
	M3	1	(12,2,0)									
	M4	1	(19,3,0)	(7,1,1)								
	M5	1	(2,5,0)									
	M6	0										
C2	M1	0										
	M2	1	(12,1,0)									
	M3	0										
	M4	1	(2,2,0)									
	M5	0										
	M6	0										
C3	M1	2	(10,2,0)	(2,3,1)	(1,1,2)	(2,4,3)	(5,2,4)	(1,2,5)	(13,1,6)	(1,4,7)	(14,2,8)	(2,6,9)
			(13,4,10)	(14,3,11)	(14,4,12)	(14,5,13)	(14,6,14)	(2,7,15)	(13,5,16)	(1,5,17)	(7,3,18)	(2,8,19)
	M2	1	(19,1,0)	(13,2,1)	(5,1,2)	(5,5,3)	(19,2,4)	(2,1,5)	(6,1,6)	(14,1,7)	(5,7,8)	(7,2,9)
	M3	1	(13,6,0)	(5,3,1)	(13,7,2)	(1,3,3)	(5,8,4)					
	M4	1	(5,4,0)									
	M5	1	(5,6,0)									
	M6	1	(10,1,0)	(13,3,1)								

Table 4.6- Part-cell assignment for period 2

Cell	Machine Type	Machine Qty	(Part, Operation, Position)										
C1	M1	3	(25,1,0)	(16,1,1)	(13,1,2)	(11,1,3)	(13,4,4)	(17,1,5)	(17,2,6)	(4,1,7)	(15,4,8)	(9,2,9)	(18,2,10)
			(15,6,11)	(18,4,12)	(9,6,13)	(25,2,14)	(25,3,15)	(18,5,16)	(5,2,17)	(9,7,18)	(4,2,19)	(3,2,20)	(4,3,21)
			(13,5,22)	(16,2,23)	(3,5,24)	(3,6,25)	(18,6,26)	(17,3,27)	(3,7,28)	(17,4,29)	(25,4,30)	(18,8,31)	(11,8,32)
	M2	2	(9,3,0)	(15,2,1)	(12,1,2)	(18,1,3)	(9,4,4)	(5,1,5)	(16,5,6)	(17,5,7)	(9,5,8)	(5,5,9)	(18,7,10)
			(19,1,11)	(13,2,12)	(11,3,13)	(19,2,14)	(17,6,15)	(15,3,16)	(5,7,17)				
	M3	1	(11,2,0)	(11,4,1)	(4,4,2)	(15,1,3)	(13,6,4)	(5,3,5)	(11,6,6)	(11,7,7)	(5,8,8)	(13,7,9)	(15,7,10)
			(3,1,11)	(3,3,12)									
	M4	1	(5,4,0)	(19,3,1)	(3,4,2)	(17,7,3)	(9,1,4)						
	M5	1	(5,6,0)										
	M6	1	(16,4,0)	(13,3,1)	(18,3,2)								
C2	M1	0											
	M2	0											
	M3	0											
	M4	0											
	M5	1	(15,5,0)										
	M6	0											
C3	M1	0											
	M2	0											
	M3	1	(12,2,0)										
	M4	0											
	M5	0											
	M6	1	(16,3,0)	(11,5,1)									

4.3.6 Computational Performance

In order to test the functionality and evaluate the computational performance of the GA based heuristic, different problem sizes were chosen to include 3, 4, 5, 6, 7, 8 parts with 3 and 4 and 5 operations, the total number of the problem instances being 90. Since the only two varying factors in our test problems are the number of parts and number of the corresponding operations and since a combination of the two would affect the computation effort, we have considered a size index as follows: Size index= No. of parts x No of operations.

We therefore categorized test problems as 9 categories in terms of their size indices namely 9, 12, 15, 16, 18, 20, 21, 24 and 25 as shown in Figure 1.Test problems with higher size indices proved to be too complex for LINGO to provide a feasible solution in reasonable amount of time and therefore had to be excluded from experimentation. Input data consisting MCIM and corresponding setup times which were generated by pseudo random function in C++ programming language to maintain a random and unbiased distribution of the input data for the problem instances. Table A.4.4 and A.4.5 (Appendix), show a sample of the input data for MCIM and sequence-dependent setup times for one of the test problems. For the sake of simplicity and without loss of generality, cost coefficients and all other parameters are all set to unity. Moreover, the cell upper bound and lower bound on the number of machines i.e. constraint (16) has been relaxed. In all test problems, parts are selected to be grouped with three machine types in three cells to minimize the overall cost of setup, machine utilization and intercellular movement. These problem instances were run on LINGO 9 (LINDO Systems Incorporation) on a 2 MHZ PC. The setup times were generated pseudo randomly within the range of 10 and 40 minutes while the MCIM is formed by randomly picked binary variables. As for the GA based heuristic, the termination criteria was set at a total of 12000 generations. The best objective values from the GA were picked after forty consecutive rounds of full rejuvenation of the program and the computation time was measured against the generation in which the best objective was achieved for the first time since the start. Tables A4.6.1, A4.6.2 and A4.6.3 (Appendix) show the results from GA based heuristic in comparison with LINGO. The average relative gap shown in Table 4.6 as Ave. Gap has been calculated as follows:

Ave. Gap = Σ[(GA Value – LINGO Value)/LINGO Value]/ number of datasets

In all instances where an optimal solution was found by LINGO within an hour or more of computation time, GA-based heuristic also hit the optimum in a short amount of time, thence relative gap being zero. In over 95% of instances where feasible solution was found by LINGO within an hour of computation time, the GA-based heuristic has equalled or outperformed LINGO. The average relative gap, however, has always been in favour of the GA-based heuristic as it is either zero or negative. Wherever LINGO could not provide a feasible solution even within an hour of computation time, the corresponding GA value has also been excluded from the calculation of average gap.

Figure 4.8 depicts the average computation time of each test problem via GA-based heuristic, versus the size index of the test problems. As shown, there is a sudden jump in the average computation time from size index 9 to 12. This is due to the fact that size index 9 represents 3 parts and 3 operations which together with 3 machines and 3 cells, provides fast finding of the optimum solutions. The optimum value of the objective function in this unique size index is also much lower than the neighbouring size indices since for example trivial solutions could happen when each operation of each part is assigned to one machine in each cell, which in turn eliminates the setup time effect and reduces the objective value drastically. The rest of the data points indicate an upward trend with slight fluctuations stemming from the sets and the inherent randomness associated with genetic algorithm as well as with the randomly generated data sets.

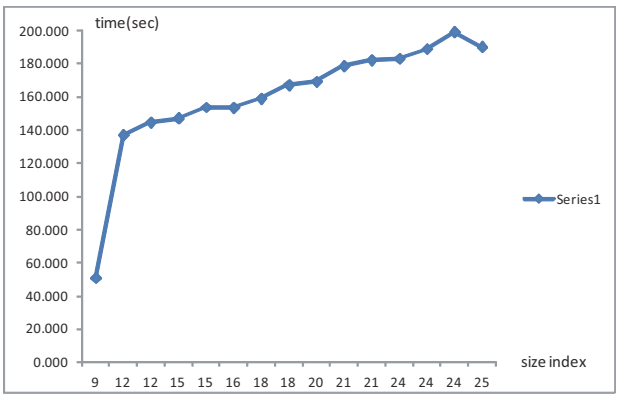

Figure 4.8- Computation time vs. size index

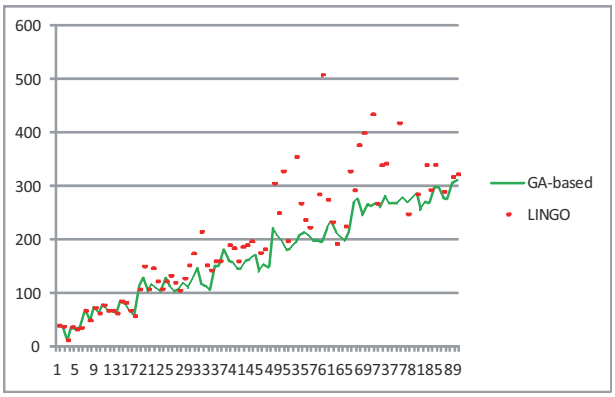

Figure 4.9 -Accuracy vs. size index

In Figure 4.9, the objective values of all data sets for all size indices from GA-based heuristic and LINGO have been plotted. Since LINGO did not provide feasible solutions for some instances and as a result its line graph would be discontinuous, it has been shown as scatter chart. As shown in the graph, in smaller sizes both LINGO and GA-based heuristic perform equally good, hitting the optimum values whenever obtained. While for higher size indices GA-based heuristic has mostly provided lower average objective values for each size index.

4.4 Discussion and Summary

The contrast between economic order quantity (EOQ), and the current model is worth to be noted. In EOQ, the (economic) batch size for each part is calculated independent of those of the others, based on its setup time and cost. The current model though has a fundamental difference with EOQ condition since it corresponds to repetitive manufacturing in closed job shop, therefore the batch size for each part is calculated in conjunction with those of other parts on a common basis namely the product basket. While in EOQ, the number of times that each part batch is processed could be different from those of other parts, in our model, batches of all parts would be processed by a common number of times which is the number of the product baskets in the planning period.

Since the LIP model turns to be NP-hard in its strong sense, branch and bound or branch and cut algorithm is not an efficient algorithm when the problem instances grow in size. Our experimentation with commercial software packages, LINGO in our case, also confirms the above statement. Therefore a problem specific heuristic based on GA was improvised to obtain near optimal solution for real size problems in a reasonably short amount of time. A numerical example of real life scale for 25 parts and maximum of nine operations per part was solved to test the functionality of the GA- based heuristic. The computation performance of the heuristic was successful as examined versus LINGO.

CHAPTER 5

MULTI-PERIOD SOLUTION: DYNAMIC PROGRAMMING –BASED HEURISTIC

In Chapter 4 a GA-based heuristic was improvised that would solve the single period mathematical model and provide solutions with acceptable accuracy in reasonably short amount of time. In transition from single period to multi-period problems, however, the solution space drastically increases which makes it difficult for the GA-based heuristic to explore the solution space in reasonably short amount of time and find near optimal multi-period solutions with acceptable accuracy. In order to address this shortcoming of the GA- based heuristic, we introduce a dynamic programming based heuristic that would use the single period results from GA-based heuristic at different stages of the solution process and recombine them into one multi-period solution.

The inspiration for our multi-period heuristic originates in Chen (1992) in which the author uses demands for parts of one period with machine configurations of other periods provided that this arrangement is feasible (Chen, 1992). The author solves a secondary problem which is a result of decomposition of the main multi-period planning to *n* single period problems, *n* being the total number of planning periods. With the assumption that the machine configuration of one period might be feasible for another period, the author then determines the best combination of demand structure and machine configuration of single–period solutions which would serve as the best multi-period solution among all considered combinations.

There are two major differences however, between our problem and that solved in Chen (1992). First is that in Chen (1992), the author uses a branch and bound algorithm to obtain optimal solutions for the LIP sub problems, while computation burden of our model is too intense to be solved for optimal solutions by branch and bound approach. Second is that the assumption that the machine configuration of one period may prove feasible for other periods as in Chen (1992), deems unlikely in a real manufacturing situation. Besides, checking the relative feasibility of the solutions between the periods may be quite a tedious and time consuming job when the number of the parts and operations are rather large. These two major differences between the two problems would propose a different heuristic that will be discussed in detail in the following section.

5.1 Decomposition of the global multi-period model

Consider the main multi-period integer programming model introduced in Chapter 3. Careful examination of the global model suggests that the multi-period model can be totally decomposed to secondary single period sub-problems by removing the transitional elements which interact between periods. Mathematical formulation of the sub-problem SP(t), for each period it would therefore be as follows:

Minimize $$SP(t) = \sum_{i=1}^{I}\sum_{j=1}^{J_i}\sum_{p=1}^{P}\sum_{k=1}^{K}\sum_{l=1}^{L} B(t).\ \lambda_{ijpi'\ j'\ (p+1)kl}(t)\cdot\ S_{ij\,i'j'k}\cdot C_k$$

$$+\sum_{l=1}^{L}\sum_{i=1}^{I}\sum_{j=1}^{J_i}\Psi_i\cdot\eta_i(t).\mu_{ijl}(t)+\sum_{k=1}^{K}\sum_{l=1}^{L}(\Omega_k^{+}\cdot y_{kl}^{+}(t)+\Omega_k^{-}\cdot y_{kl}^{-}(t))$$

Subject to :

$$\lambda_{ijpi'\,j'(p+1)kl}(t) \geq x_{ijpkl}(t) + x_{i'\,j'(p+1)\,kl}(t)) - 1$$

$$\mu_{ijl}(t) \geq z_{ijl}(t) - z_{i(j+1)l}(t)$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J_i} x_{ijpkl}(t) \leq 1$$

$$\sum_{l=1}^{L}\sum_{p=1}^{P}\sum_{k=1}^{K} x_{ijpkl}(t) = 1$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J_i} x_{ijpkl}(t) \geq \sum_{i=1}^{I}\sum_{j=1}^{J_i} x_{ij(p+1)kl}(t)$$

$$\zeta_{ijp(j+1)p'\,kl}(t) \leq \frac{1}{2}(x_{ijpkl}\ \ (t) + x_{i(j+1)p'\ kl}(t))$$

$$\zeta_{ijp(j+1)p'\,kl}(t) \geq x_{ijpkl}(t) + x_{i(j+1)p'\ kl}(t) - 1$$

$$\zeta_{ijp(j+1)p'\,kl}(t)\cdot p' + (1 - \zeta_{ijp(j+1)p'\,kl}(t)).p \geq p$$

$$\sum_{k=1}^{K}\sum_{p=1}^{P} x_{ijpkl}\ \ (t) = z_{ijl}(t)$$

$$\sum_{k=1}^{K} \beta_{ijkl}(t) = z_{ijl}(t)$$

$$x_{ijpkl}(t) \leq y_{kl}(t)$$

$$MIN_l \leq \sum_{k=1}^{K} y_{kl}(t) \leq\ MAX_l$$

$$\Gamma_k\cdot y_{kl}(t) \geq \sum_{i=1}^{I}\sum_{j=1}^{J_i}\theta_{ijk}\cdot\eta_i(t).\beta_{ijkl}(t)$$

$$x_{ijpkl}(t), z_{ijl}(t), \lambda_{ijp\,i'\,j'(p+1)kl}(t), \mu_{ijl}(t), \zeta_{ijp(j+1)p'\,kl}(t) \in \{0,1\}$$

$$y_{kl}(t), y^{+}_{kl}(t), y^{-}_{kl}(t) \in \{0,1,2,...\}$$

For a description of the cost terms in the objective function and the constraints in the above single-period problem please refer to those provided in Chapter 3 for the multi-period global model.

5.2 Application of dynamic programming to sub-problems

SP(t*)* is then solved using the GA-based heuristic introduced in Chapter 4, to provide near optimal single-period solutions for different periods. Recalling the GA-based heuristic introduced in Chapter 4, after every certain number of generations g_{max}, a full rejuvenation will take place which we call it a round. In each round $r=1,..,R$, the best solution $G^*(r)$ of that round is recorded. After R rounds, the heuristic terminates and a set of R best solutions will be formed. The total number of generations involved will be Rxg_{max}. Figure 5.1 clarifies the above procedure. Different near optimal solutions in each period obtained at different generations by GA-based heuristic, can combine with those of neighbouring periods to form a multi-period solution while there is a reconfiguration cost associated with each match up of the solution of the neighbouring periods. Searching for the best combination of feasible solutions of different periods would be equivalent to solving a dynamic programming problem exemplified in Figure 5.2. In Figure 5.2 the R best solutions recorded for each sub-problem have been supplemented by a second index t, to account for the period which the sub-problem is representing. Therefore the notation $G^*(r,t)$ represents r^{th} best feasible solution recorded for period t.

Rounds*(r)*		
1	*2*	*3..... R*
1	1	1.....1
2	2	2.....2
.	.	.....
.	.	.....
.	.	.....
g_{max}	g_{max}	g_{max}.....g_{max}

Figure 5.1- R best solutions in GA-based heuristic

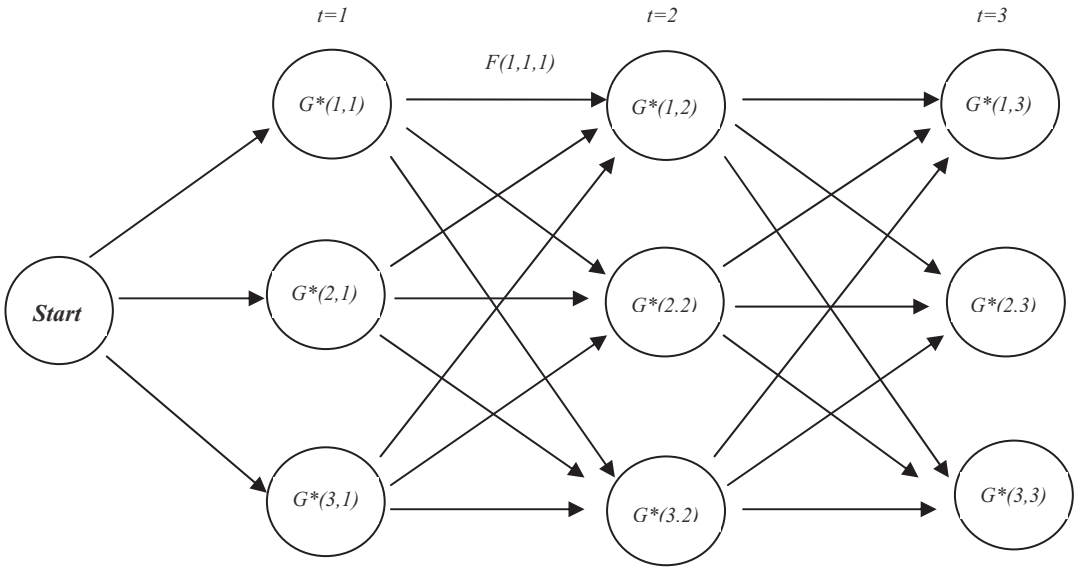

Figure 5.2 -Dynamic programming network representing the heuristic

The graph of the network corresponding to any pair of immediately linked stages *t* and *t+1* is fully connected. $F(t,r,r')$ is the machine reconfiguration cost associated with transition from the best solution in round *r* in period *t* to the best solution in round *r'* in period *t+1*. Ω_k^+ and Ω_k^- are the machine installation and removal unit costs, respectively and $y_{kl}(t)$, is the number of the machine type k in cell *l* in the beginning of period *t* all as explained in the mathematical model in Chapter 3. Eq. (5.1) represents the corresponding cost involved in transition from period *t* to period *t+1*:

$$F(r,r',t)=\sum_{k=1}^{K}\Omega_k^+\sum_{l=1}^{L} max\{\ y_{kl}(t+1)-y_{kl}(t),\ 0\ \}+\sum_{k=1}^{K}\Omega_k^-\sum_{l=1}^{L} max\{\ (y_{kl}(t)-y_{kl}(t+1),\ 0\ \} \quad (5.1)$$

Let *U*(r,t-1)* be the best policy corresponding to node *(r, t-1)* of the network, then the mathematical form of the corresponding dynamic programming problem can be expressed by the backward recursive function (Eq.5.2). This equation affirms that the best policy *U*(r,t-1),* is calculated by finding the minimum of the summation of the best policy of each state (solution) at stage (period) *t, U*(r', t),* the corresponding single-period overall cost of each state (solution), *G*(r, t-1),* and the cost associated with transition from state (solution) *r* at *t-1* to state (solution) *r'* at stage(period) *t,* $F(r,r',t-1)$.

$$U^*(r,t-1)=\min_{r\in\{1\ldots R\}}\ \{\ U^*(r',t)+F(r,r',t-1)+G^*(r',t-1)\ \} \quad (5.2)$$

5.2.1 Dynamic Programming-based heuristic

Step 1 -Solve the secondary problems for each planning period t by the GA-based heuristic

Step 2-Get R best feasible solutions for each period from GA-base heuristic

Step 3-Form the corresponding network consisting of *T* stages and *R* states in each stage

Step 4-Associate transition cost to each arrow in the network

Step 5-Solve the dynamic programming problem represented by network in Step 3

Step 6-Consider the optimal policy determined by dynamic programming as the solution for the multi-period global problem.

5.3 Illustrative Example

The original problem for this example is the one initially introduced in Chapter 4. However the part demand and reconfiguration unit costs have been changed and three periods have been considered as shown in Tables 5.1 and 5.2. First the corresponding secondary single period problems for 3 time periods are solved by the GA-based heuristic, the results of which have been shown in Tables 5.3 and 5. 4. In order to find the best solution for the three-period planning horizon, the dynamic programming-based heuristic is applied as explained in section 1 and 2 above. For each sub-problem solved by GA-based heuristic, the best feasible solutions in each round in terms of objective function value is chosen. In this example twenty best feasible solutions for each of the three periods were selected. These feasible solutions form the states of each stage or period. There will be a cost associated with transiting from any solution in period *t* to any solution in period *t+1*.These transition costs are basically the costs incurred in reconfiguring the machine configuration of the corresponding cells of each period to those of the immediate next period. Finding the best solution for a three period planning

horizon, among the existing feasible solutions, is then equivalent to finding the best policy for a dynamic programming network.

Following the heuristic discussed in section 2 above, the best multi-period solution for three periods is obtained by the combination of solution number 2 in period 1 followed by solution number 19 in period 2, followed by solution number 5 in period 3 as highlighted in Tables A.5.3 and A.5.4 (Appendix). It is worth noting that the elected solution for each period by the DP-based heuristic, is not necessarily the lowest objective function value found by GA-based heuristic in that period. For example, as can be seen in Table A.45.3, the objective function value for solution number 2 for the first period, i.e. 19211, does not have the lowest objective value among all 20 values obtained by GA-based heuristic solved for a single period solution for the first period, as for example node or solution 12, has the lowest overall single –period cost i.e.19085.8, however the best policy starts with solution 2 in period 1 and not solution 12. It goes without saying that although the selected solutions in each period are just suboptimal with regard to LIP sub problems, the obtained multi-period solution however, is an optimal solution of the dynamic programming procedure, hence the optimal policy.

Table 5.1-Product mix demand of 25 parts for three periods: *t=1, t=2, t=3*

t=1: 1200,1000,0,0,3200,1500,1680,2200,0,980,0,740,1800,860,0,0,0,0,1200,1520,2200,0,0,0,0

t=2:
0,0,1480,1800,2800,0,0,0,1780,0,900,1350,1400,0,2500,960,1000,1700,2000,1800,0,0,0,1890,1400

t=3:
0,1200,1600,2200,1700,0,0,960,0,900,1350,0,0,2500,0,1000,1700,1000,1800,0,0,580,0,1320,0

Table 5.2-Reconfiguration unit costs in all three periods

	M1	M2	M3	M4	M5	M6
Removal unit cost	475,	320,	540,	750,	290,	480
Installation unit cost	675,	400,	540,	850,	380,	480

Robustness of the result for the illustrative example 5.3 is examined through statistics regarding the best objective values, shown in Tables A.5.3 and A.5.4 in the Appendix, obtained by the single period GA-based heuristic for periods 1 and 2, as indicated in Table 5.3.

Table 5.3-Robustness table for the numerical example results

	Average (Best objectives)	Max	Min	Std	Std/Ave
Period 1	22170.38	26893.6	19085	2202.89	0.099
Period 2	26862.9	26862.9	26862.9	0	0

5.4 Comparing the two heuristics for multi-period problem

5.4.1 Comparing the DP-based heuristic with LINGO

In this section we have considered 12 problem instances with 3 parts and 3 and 4 operation for each part.For simplicity, except for setup times, all other parameters are considered unity.Setup times are random numbers of a uniform function from the closed interval [10,40], generated pseudo randomly by C++ srand function . The data for MCIM for different test problems are also generated pseudo randomly from [0,1]. The demand for the periods one, two and three for each part are 1, 2 and 3 respectively.First each test problem has been solved for the three periods.Then the dynamic programming based

heuristic introduced in section 4, has applied to the solutions of three periods to obtain the multi-period solution.Finally the multi-period solution has been comared to that provided by LINGO.As shown in Table 5.4, the Multi-period solution by DP outperforms those of the LINGO in terms of objective value and the computation time. In one case where both DP and LINGo have the same Objective vlaue (84*) the result is in fact optimal.

Table 5.4- Comparing the DP-based heuritic with LINGO

	Period 1-GA		Period 2-GA		Period 3-GA		Multi-period DP		Multi-period LINGO	
	Best obj.	Time(sec)	Best obj.	Time(sec)	Best obj.	Time(sec)	Best obj.	Time(sec)	Best obj.	Time(sec)
3 parts, 3 operations	39	*52*	51	*56.25*	63	*56.953*	171	165.223	190	*3600*
	37	*56.859*	49	*54.313*	61	*60.359*	165	171.555	168	*3600*
	11	*51.015*	22	*51.235*	33	*51.593*	84	165.218	84*	*202*
	36	*54.297*	47	*56.984*	58	*60.578*	159	171.982	180	*3600*
	32	*52.297*	43	*49.907*	54	*51.972*	147	154.307	161	*3600*
	35	*52.844*	46	*54.625*	57	*54.297*	156	161.841	161	*3600*
3 parts, 4 operations	68	*59.665*	85	*59.578*	102	*65.766*	279	185.126	548	*3600*
	49	*57.516*	65	*52.141*	81	*63.485*	220	173.25	394	*3600*
	72	*56.39*	87	*62.203*	102	*61*	285	179.717	375	*3600*
	62	*58.922*	79	*58.922*	96	*60.5*	261	178.471	379	*3600*
	79	*59.984*	96	*60.984*	112	*60.062*	311	181.133	566	*3600*
	68	*58.343*	83	*58.297*	98	*59.125*	273	175.861	382	*3600*

* Optimal result by LINGO

5.4.2 Comparing the DP-based heuristics with multi-period GA

In this section a series of thirteen problems of different scales have been solved by a multi-period feature of GA-based heuristic and by dynamic programming-based approach where the multi-peiod problem is decomposed to single period problems first, then solved by single-period GA-based heuristic where its best feasible solutions are then recombined into multi-period through a dynamic programming setting.The experiment covers a range of problems from 7 parts to 16. A two period problem has been considered

for the comparison between the two heuristics. Since the multi-period solution found by DP-based heuristic is merely dependent on the value of the best feasible single-period solutions and the reconfiguration cost between the two solutions of consecutive periods, it will not be adversely affected by the number of the periods involved and maintains its robustness. In case of multiple GA-based heuristic however, the number of the periods will drastically enlarge the solution space and therefore affect the solutions obtained. Because of this, if the multi-period GA-based solution performs poorly with respect to DP-based heuristic in a two-period setting, adding more periods will not help the GA-based heuristic either.

Table 5.5and 5.6 show a sample of main parameters for one of the problems with 12 parts. As a sample,Tables A.5.5, A.5.6 and A.5.7 in the Appendix, indicate the MCIM, processing times and setup times for problem with 12 parts, 3 operations and 6 machines in Table 5.7.

The dynamic programming is an efficient algorithm that provides an optimal solution based on the input data. This heuristic uses sub-optimal solutions from different periods and combine them by charging the due transition costs that would inevitably incurr in a multi-period setting.

The dynamic programming based heuristic consistenly provides better results in terms of overall cost and computation time when compared with the multi-period feature of the GA-based heuristic. As indicated in Table 5.7, the multi-period GA-based heuristic is at par with DP-based for a medium scale problem with 7 parts and 4 operations but for larger problems it lagged behind the DP-based heuristic in terms of cost and corresponding computation time during which the cost value was achieved.

Table 5.5-Product mix/demand of twelve parts for two periods

t=1: 1200,1000,0,0,3200,1500,1680,2200,0,980,0,740
t=2: 0,0,1480,1800,2800,0,0,0,1780,0,900,1350

Table 5.6-Reconfiguration unit costs in all periods for twelve part problem

	M1	M2	M3	M4
Reconf_unit[K]	75	100	140	90
Machunitcost[K]	1250	1180	1000	1120
Sunitcost[K]	40	35	38	35
Capacity	1200	1200	1200	1350
Material handling unit	(0.5,0.5,0.5,1,0.5,1,0.5,1.5,1,0.5,0.5,0.5)			

Table 5 .7 - Comparing DP-based with Multi-Period GA-based

Part, operation, machine	Period 1 cost($)	Period 2 cost($)	Re-configuration cost($)	Dynamic programming cost ($)	Period 1 time (sec)	Period 2 time (sec)	Transition DP time(sec)	Dynamic prog. time(sec)	GA multi-period cost ($)	GA multi-period time(sec)
7,4,6	7631.33	3550	200	11381.3	73.032	73.547	0.016	146.595	11381.3	178
7,5,6	11164.7	7751.33	275	19191	92.157	78.641	0.031	170.829	23959.7	243.18
8,5,6	11350.4	7811.83	275	19437.2	58.727	82.109	0.016	140.852	24249.8	269.171
9,4,6	11080	8864.52	245	20189.5	56.314	80.234	0.015	136.563	22180.7	250.922
10,3,6	4313.3	5676.67	35	10340	49.162	73.656	0.016	122.834	10461.9	200.297
10,4,6	11471	7236	165	18872	59.268	70.75	0.015	130.033	22181.8	263.532
11,3,6	9051.35	7760.5	0	16811.8	55.923	67.734	0.016	123.673	19170.3	238.157
12,3,6	9500	5413.67	365	15278.7	56.497	69.859	0.015	126.371	20267.7	249.438
13,2,6	10124.4	4847.72	290	15262.1	49.563	74.281	0.015	123.859	17659.9	220.891
15,2,6	8923.42	8221.25	0	17144.7	54.889	74.813	0.016	129.718	19299.8	233
15,3,6	13066.5	11197.4	305	24568.9	65.65	73.125	0.031	138.806	29611.9	319.859
15,4,6	13810.2	16169.3	260	30239.5	71.905	77.219	0.016	149.14	36451.4	409.578
16,2,6	10741.5	7074.63	210	18026.1	56.599	70.093	0.015	126.707	20443.2	240.641

5.5 Discussion and Summary

In this Chapter, a dynamic programming based heuristic was introduced for transition from single-period solutions to multi-period solution. The decomposition of the global model of the current research problem provides an opportunity to divide the main problem to single-period problems. Given the fact that for transition from the machine-cell configuration in one period to that in another period a reconfiguration cost is incurred, based on the current configuration of the concerning periods, any chain of feasible solutions in different periods can be considered to represent a feasible multi-period solution. Finding the best multi-period solution would then turn to be one of finding the optimal policy in a dynamic programming problem.

While the devised problem specific GA-based heuristic has proven satisfactory in comparison with commercial software package based on B&B algorithm, namely LINGO 9, the multi-period solution is not as satisfactory, especially when the problem size increases. The introduced DP heuristic can remedy this shortcoming by decomposing the model into smaller single period problems to be solved by GA-based heuristic and then combine them using dynamic programming approach to find the best chain of solutions that would represent the multi-period solution. The DP-based heuristic has been compared with both LINGO and multi-period GA as depicted in Tables 5.4 and 5.7 respectively, where DP-based approach consistently shows better results in terms of both best objective value and computation time.

The solution found by DP-based heuristic is more robust in terms of sensitivity to the size of the problem and outperforms the multi-period solutions provided by both LINGO and

GA-based heuristic. In the DP-based heuristic, the sub-optimal solutions of the GA-based heuristic are combined by adding the cost of reconfiguring the machine-cell from one period to the next period to the overall single-period cost of the two periods. Since, the GA-based heuristic has performed satisfactorily enough in providing sub-optimal single-period solutions, DP-based heuristic utilizes a set of best feasible solutions for each period to combine with those of the next period in the manner explained above. The best optimal policy is represented by the chain of certain best feasible solutions of each period that collectively represent the lowest overall cost including reconfiguration costs. It is worth noting that if the solutions in each period are optimal or the same, then this approach would not be needed, as any combination of solutions of each period will provide the optimal or the same multi-period solution.

CHAPTER 6

CONCLUSIONS AND FUTURE RESEARCH

6.1 Conclusions

While research in cellular manufacturing has been abundant, there are very few publications in which sequence-dependent setup time has been involved in the process of cell formation and design of the cellular manufacturing system. Another aspect which has just recently gained some attention is the multi-period planning in a dynamic cellular manufacturing system.

The primary goal of this research was to develop a design methodology that takes into consideration the changeover time between different parts processed in the manufacturing cells. Since sequence-dependent setup time is the general form of setup time from which other forms can be derived as special cases, the design was based on sequence-dependent setup time. This in turn leads to the design methodology contributing to the production planning process, since the order in which the parts are processed is also recommended through the design process.

In every cellular manufacturing system, when the product mix changes, there is always chance that a redesign of the system might prove necessary, since the grouping of parts in the machine cells might not be optimal anymore. In these cases, given the conditions, a reconfiguration of the machine-cells might be necessary in order to maintain an optimal status in terms of the relevant production costs. Of course it goes without saying that the cost of system reconfiguration shall also be taken into consideration when calculating the

relevant production costs. In a dynamic environment where the product mix might change several times during the year, a flexible design system should anticipate and take the changes into consideration in advance, so that optimization of the system design is considered over the entire span of planning horizon rather than a myopic approach. This constitutes the secondary goal of the current design methodology which is a multi-period system design optimization. In response to the above requirements, an integer programming mathematical global model was developed. The model however is NP-hard in its strong sense due to the presence of sequence-dependent term which generates a massive amount of integer variables even for moderately sized problem instances.

Obviously the routine optimization software packages available in the market which are deploying branch and bound and or branch and cut algorithms may not solve real life problems in a reasonably short of amount of time when these algorithms are proven not to be efficient ones.

This led to the development of heuristics to handle real life manufacturing problems in a reasonably short amount of time. In order to address this issue, a step by step approach towards tackling the multi-period global was considered. First the heuristic was developed based on GA platform for a special case of the global mathematical model which simplified the model to a less complex one. The purpose of this heuristic was to gain insight in solving the global model and to see whether the adopted approach was promising. The performance of this simplified heuristic was evaluated against commercial software. In 80% of the 36 test problems, the heuristic was equally or more successful than the software. In the next phase a comprehensive GA-based heuristic was developed for which shortcomings and deficiencies of the simplified heuristic were

remedied. The principal goal of the comprehensive GA-based heuristic was to handle the single-period global model. However, the heuristic was developed in a way that could tackle multi-period as well. The performance of the comprehensive GA-based heuristic was evaluated against commercial optimization software package by solving 90 single period test problems where the GA-based heuristic was equally or more successful than the software in 95% of the tests.

Finally the last step was to tackle the multi-period global model. A decomposition of the global model was considered which would decompose the original model to single-period sub-problems. The single-period problems were then solved through comprehensive GA-based heuristic and the best feasible solutions obtained will be prospective solutions for the final multi-period solution. To obtain the multi-period solution however, the candidate solutions are recombined in a dynamic programming setting where the objective is to find that best combination of single-period solutions of different periods that represent the minimum overall cost including that of reconfiguration. The dynamic programming based heuristic consists of six steps as follows:

1-Solve the secondary problems for each planning period t by the GA-based heuristic

2-Get R best feasible solutions for each period from GA-base heuristic

3-Form the corresponding dynamic programming network consisting of T stages and R states in each stage

4-Associate transition cost with each arrow in the network

5-Solve the dynamic programming problem represented by network in Step 3

6-Consider the optimal policy determined by dynamic programming as the solution for the multi-period global problem.

The performance of the heuristic was evaluated via the multi-period feature of the comprehensive GA-based heuristic. The experimentation indicates that dynamic programming based heuristic consistently lower overall cost and computation time when compared with those of a multi-period GA-based heuristic in a series of randomly generated test problems.

6.2 Contributions

The contributions of the current thesis have been outlined as follows:

1. A macro analysis of cellular manufacturing was presented that explained the raison d'etre of CM through of volume-variety curve and dissimilarity index of a system

2. The presumption of setup time reduction in cellular manufacturing was debated and the impact of MCIM or lack thereof in that regard was discussed.

3. The setup time definition was elaborated on and standard definition was suggested and various set up time reduction approaches were assessed against the standardized definition.

4. A multi-period mathematical model was developed that incorporates sequence-dependent setup time in the cell formation process of the design of dynamic cellular manufacturing system

5. A problem specific heuristic was tailor made to solve static sequence-dependent related CMS design problems in a reasonable amount of time.

6. A dynamic programming based heuristic was developed for the purpose of transition from single-period solutions regarding different periods to one overall multi-period solution for the entire planning horizon. The DP- based heuristic was then compared with

multi-period featured GA-based heuristic where it consistently provided better results in terms of the cost and computation time.

6.3 Future Research

Due to the complexity of the current novel topic, the following research was considered to be out of the scope and time limitations of the current thesis, therefore considered for future research.

1-It would be a good practice to examine the impact of the sequence-dependent setup time in the production cost. The cost of an ad hoc order of processing parts in the manufacturing cells can be compared with the optimised order recommended by the model introduced in this thesis to see the corresponding impact and how the difference would vary with respect to various parameters of the model.

2-One natural implication of this thesis is its relationship with scheduling. The order or sequence of parts recommended by the introduced model, would have impact on scheduling criteria. The study of those parameters is out of the scope of this thesis and will be studied later. The scheduling will also affect the amount of the WIP in the system which has to be taken into account in the production planning of the manufacturing systems.

3-Considering meta-heuristics and heuristics other than those introduced and applied in this thesis may prove effective in solving more complex versions of the current model featured by further real manufacturing attributes, as well as providing the opportunity for comparing the performance of the new solution approaches with those used in this thesis.

BIBLIOGRAPHY

Abdelmola, A.L.; Taboun, S.M.; Merchawi, S., "Productivity optimization of cellular manufacturing systems", Computers & Industrial Engineering, 35, 1998

Agarwal, A., "Partitioning bottleneck work center for cellular manufacturing: An integrated performance and cost model", Int. J. Production Economics 111 (2008) 635–647

Agarwal, A.; Sarkis, J.,"A review and analysis of comparative performance studies on functional and cellular manufacturing layouts", Computers & Industrial Engineering, 34, 77-89, 1998

Ahmed, P. ; Tavakkoli-Moghaddam,R. ; Safaei,N. ; "A comparison of heuristic methods for solving a cellular manufacturing model in a dynamic environment", 2004, University of Wolverhampton, Working Paper

Ahmed, P; Tavakkoli-Moghaddam, R.;Safaei, N., "A comparison of heuristic methods for solving a cellular manufacturing model in a dynamic environment" Working Paper, 2004

Allahverdi, A.; Gupta, J.N.D.; Aldowasian, T., " A survey of scheduling research involving setup considerations, OMEGA, International Journal of Management Science, 27,1999

Al-Mubarak, F.; Canel,C.; Khumawala, B.M., " A simulation study of focused cellular manufacturing as an alternative batch-processing layout", Internationl Journal of Production Economics, 2002

Anderson, S.W.," Measuring the impact of product mix heterogeneity on manufacturing overhead cost", Accounting Review 70 (3), 363-387, 1993

Arora, P., K.; Haleem, A.; Singh, M. K., "Cell Formation Techniques – A Study", International Journal of Engineering Science and Technology (IJEST), Vol. 3 No. 2, Feb 2011

Askin,R. G.; Standridge, C. R., "Modelling and Analysis of Manufacturing System",1993 Wiley, New York

Askin, R.; Chu, K., "A Graph Partitioning Procedure for Machine Assignment and Cell formation in Group Technology", 1990, International Journal of Production Research, 28(8), 1555-1572.

Atmani, A.; Lashkari, R.S.; Caron, R.J., " A mathematical programming approach to joint cell formation and operation allocation in cellular manufacturing", International Journal of Production Research, 33, 1995

Atmani, A., "A production planning model for flexible manufacturing systems with setup cost consideration", Computers & Industrial Engineering, 29(1-4), 1995

Balakrishnan, J.; Cheng, C.H. , " Multi-period planning and uncertainty issues in cellular manufacturing: A review and future directions ", European Journal of Operational Research 177 (2007) 281–309

Berry, W.; Cooper, L.; Martha, C., "Manufacturing flexibility: Methods for measuring the impact of product variety on performance in process industries", Journal of Operations Management, 17 (2) 163-187, 1999

Boctor, F.F., "A linear formation of the machine cell formation problem", 1991, International Journal of Production Research, 29(2), 343-356.

Burbidge, J.L., "Production Flow Analysis," The Production Engineer (v50, 1971), p139.

Burbridge, J.L., "Change to group technology: process organization is obsolete", International Journal of Production Research, 30(5), 1992

Burbidge, J.L., ”The first step in planning group technology”, Int. J. Production Economics 43 (1996) 261-266

Burgess, A.G.; Morgan, I.; Vollman, T.E.; “Cellular manufacturing: Its impact on the total factory”, International Journal of Production Research, 31(9), 1993

Cantu-Paz, E.; Goldberg, D.E.; “Efficient parallel genetic algorithms: theory and practice”, Computer Methods in Applied Mechanics and Engineering, 186, 221-238, 2000

Caux, C.; Bruniaux, R.; Pierreval, H., “Cell formation with alternative process plans and machine capacity constraints. A new combined approach for manufacturing cell formation”, International Journal of Production Economics, 64, 2000

Chan H.M.; Milner D.A., (1982), “Direct clustering algorithm for group formation in cellular manufacturing”, Journal of Manufacturing Systems, Vol.1, pp.65-71

Chandrashekharan, M.P.; Rajagopalan, R., 1987, “ZODIAC: An algorithm for concurrent formation of part families and machine cells”, International Journal of Production Research, Vol.25, pp.835-843.

Chandrasekharan, M.P.; Rajagopalan, R., “ Groupability: Analysis of the property of binary data matrices for group technology”, 1989, International Journal of Production Research, 27(6), 1035-1052

Chang, P-T.; Lee, E.S., “A multi-solution method for cell formation exploring practical alternatives in group technology manufacturing” Computers & Mathematics, with application, 40, 2000

Chen, M., “A mathematical programming model for system reconfiguration in a dynamic cellular manufacturing environment”, 1998, Annals of Operations Research 77, 109 – 128

Chen, M., “Computer techniques and applications for the design of optimum cellular manufacturing systems”, Chapter 2, 2004

Cheng, T.C.E; Gupta, J.N.D.; Wang, G., “A review of flowshop scheduling research with setup times, Production and Operations Management, 2000

Child, P.; Diedrichs, R.; Sanders, F.; Wisiniowski, S. ”The management of complexity” Sloan Management Review, SMR forum, 1993

Chu, C-H.; Tsai M., “A comparison of three array-based clustering techniques for manufacturing cell formation” International Journal of Production Research, 28(8), 1990, 1417-1433

Cooper, J.; Hinde, C., “Improving genetic algorithms’ efficiency using intelligent fitness functions”, Proceedings of the 16th international conference on Developments in applied artificial intelligence, pp. 636–643, 2003.

Cox, III, J.F.; Blackstone, J.H.; Spencer, M.S., APICS dictionary, 8t ed., Fall Church, VA, 1995

Damodaran, V.; Lashkari, R.S.; Singh, N., “A production planning model for cellular manufacturing systems with refixturing consideration”, International Journal of Production Research, 30, 1992

De Lit, P.; Falkenauer, E.; Delchambre, A., “Grouping genetic algorithms: an efficient method to solve the cell formation problem”, Mathematics and Computers in Simulation, 51, 257–271, 2000

Defersha, F.M; Chen, M. ,“A linear programming embedded genetic algorithm for an integrated cell formation and lot sizing considering product quality”, European Journal of Operational Research 187 (2008) 46–69

Defersha, F.M; Chen, M., “A comprehensive mathematical model for the design of cellular manufacturing systems”, Int. J. Production Economics 103 (2006) 767–783

Diaz-Diaz, B.; Lozano S.; Racero, J., Guerro F., “Machine cell formation in generalized group technology”, Computers ind. Engng. 41, 227-240, 2001

Elmaraghy H.A.; Gu, P., “Feature based expert parts assignment in cellular manufacturing”, 1998, Journal of Manufacturing, systems, Vol.8, pp.139-144

Evans, D.H., “ Modular design: a special case in nonlinear programming”, Operations Research, 11637-643, 1963

Fernandes, R.; Gouveia, J.B.; Pinho, C., “Product mix strategy and manufacturing flexibility”, Article in Press, 2012

Flynn, B.B.; Jacobs, F.R., “A simulation comparison of group technology with traditional job shop manufacturing, International Journal of Production Research, 24, 1986

Franca, P.M.; Gupta, J.N.D; Mendes, A.S.; Moscato, P.; Veltink, K. J., “Evolutionary algorithms for scheduling a flowshop manufacturing cell with sequence-dependent family setups” Computers & Industrial Engineering 48, 491–506, 2005

Garza, O.; Smuunt, T.L., “Countering the negative impact of intercell flow in cellular manufacturing”, Journal of Operation Management, 10(1), 1991

Gen, M.; Cheng, R., “Genetic Algorithms and Engineering Design”, 1996, Wiley, New York, NY

Ghosh, T.; Sengupta, S.; Chattopadhyay, M.; Dan, P.,K., “ Meta-heuristics in cellular manufacturing: A state-of-the-art review”, International Journal of Industrial Engineering Computations, 2011, 2, 87–122

Goldberg, D.E., “Genetic Algorithms in Search, Optimisation and Machine Learning”, 1989, Addison-Wesley: New York, NY.

Goncalves, J.F; Resende, G.C., “A hybrid genetic algorithm, AT&T Labs Research Technical Report, Oct.2002

Greene, T.J.; Sadowski, R. P , “A Review of Cellular Manufacturing Assumptions, Advantages and Design Techniques”, Journal of Operations Management, Vol. 4. No. 2, February 1984

Harhalakis, G.; Nagi, R.;Proth,J.M., (1990), “An efficient heuristic in manufacturing cell formation for group technology application”, International Journal of Production Research, 28(1) pp.185-198.

Hendizadeh,S.H.; Faramarzi, H; Mansouri,S.A.;J. N.D. Gupta, ElMekkawy,T.Y., “Meta-heuristics for scheduling a flowline manufacturing cell with sequence dependent family setup times”, Int. J. Production Economics 111 (2008) 593–605

Holland, J.H., “Adaptation in natural and artificial systems: An introductory analysis with application to biology, control and artificial intelligence” University of Michigan Press, Ann Arbor, MI., 1975,

Hyer, N.L. ; Wemmerlov, U., "Group Technology in US Manufacturing Industry: A Survey of Current Practices," International Journal of Production Research (v27, n2, 1989), pp1287-1304.

Hyer, N.L., “The potential of group technology for U.S. manufacturing” Journal of Operation Management” , 4(3), 1984

Irizarry, M.D.L.A.; Wilson, J.R.; Trevino, J.; “A flexible simulation tool for manufacturing-cell design, I: model structure, operation and case study”, IIE Transaction 33, 827-836, 2001.

Jayakumar, V.; Raju, R., “ An Adaptive Cellular Manufacturing System Design with Routing Flexibility and Dynamic System Reconfiguration”, European Journal of Scientific Research

Vol.47 No.4 (2010), pp.595-611

Jeon, G.; Leep,H.R.; Parsaei, H.R., “A cellular manufacturing system based on new similarity coefficient which considers alternative routes during machine failure”, Computers ind. Engng , Vol. 34, No. I, pp. 21-36, 1998

Kannan, V.R.; Ghosh, S., “Using dynamic cellular manufacturing to simplify scheduling in cell based production systems”, Omega, Int. J. Mgmt Sci. Vol. 23, No. 4, pp. 443~152, 1995

Kaparthi, S. ; Suresh, N.C., “Machine component cell formation in group technology: A neural network approach”, 1992, International Journal of Production Research, 25(6), 1353-1367.

Kaparthi, S.; Suresh, N.C. , "A Neural Network System for Shape-Based Classification and Coding of Rotational Parts", 1991, International Journal of Production Research, 29(9), 1771 -1784.

Khator, S.K.; Irani,S.K, “Cell formation in group technology: A new approach”, 1987, Computers in Industrial Engineering 12(2), 131-142

King J.R., “Machine Component Grouping in Production Flow Analysis: An Approach using Rank Order Clustering Algorithm”, 1980, International Journal of Production Research, Vol. 80. pp. 213-219.

King J.R.; Nakoranchi V., “Machine Component Group formation in Group Technology: Review and Extension”, 1982, International Journal of Production Research , Vol.20, no.5 , pp.117-121

Koren, Y.; Shpitalni, M. ,“Design of reconfigurable manufacturing systems”, Journal of Manufacturing Systems, 2011

Kumar K.R.; Vannelli, A., “Strategic subcontracting for efficient disaggregated manufacturing”, 1987, International Journal of Production Research, Vol.25, no.12, pp. 1715-1721.

Kusiak, A.; Chow W.S., “Efficient solving of the group technology problem”, 1987, Journal of Manufacturing Systems”, Vol.6, pp.117.

Kusiak, A., “Group technology: Models and solution approaches” First Industrial Engineering Research Conference, 1992

Kusiak, A., "The Generalized Group Technology Concept," International Journal of Production Research,1987, v25, n4, 561 - 569.

Kusiak, A., “A knowledge based system for group technology”, 1988, International Journal of Production Research, 26, 887-905.

LaScola Needy, K.; Billo, R. E.; Colosimo Warner, R., “A cost model for the evaluation of alternative cellular manufacturing configurations”, Computers ind. Engng, V 34, No.1, 1998

Lashkari, R.S.; Boparai; R.; Paulo, J., “Towards an integrated model of operation allocation and material handling selection in cellular manufacturing systems”, International Journal of Production Economics, 87, 2004

Lin, S-W.; Ying, K-C.; Lee, Z-J., “Metaheuristics for scheduling a non-permutation flowline manufacturing cell with sequence-dependent family setup times” Computers & Operations Research 36, 1110 – 1121, 2009

Murugan, M.; Selladurai, V., “Manufacturing Cell Design with Reduction in Setup Time through Genetic Algorithm” , 2005, Journal of Theoretical and Applied Information Technology, pp76-97

Mahdavi, I.; Mahadevan, B., “CLASS: An algorithm for cellular manufacturing system and layout design using sequence data”, Robotics and Computer-Integrated Manufacturing 24 , 488–497, 2008

Malakooti, B.; Yang, Z., “Multiple criteria approach and generation of efficient alternatives for machine-part family formation in group technology”, IIE transactions, 34, 2002

McCormick, W.T.; Schweitzer, P.J.; White, T.E., “Problem decomposition and data recognition by a clustering technique”, 1972, Operation Research, Vol.20, pp.993-998.

Molleman, E.; Slomp, J.; Rolefes, S., “ The evolution of a cellular manufacturing system- a longitudinal case study”, International Journal of Production Economics, 75, 05-322, 2002

Moon, C.; Gen, M., “A genetic algorithm-based approach for design of independent manufacturing cells”, International Journal of Production Economics, 60-61, 1999

Morris, J.S.; Tersine, R.J., “A comparison of cell loading practices in group technology”. J. Manuf. Opns Mgmt 2, 299-313, 1989

Morris, J.S.; Tersine, R.J., “A simulation analysis of factors influencing the attractiveness of group technology cellular layouts, Management Science, 36, 1990

Mungwattana, A., “Design of Cellular Manufacturing Systems for dynamic and uncertain production requirements with presence of routing flexibility”, 2000, PhD thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA.

Murugan, M.; V. Selladurai; “Manufacturing cell design with reduction in setup time through genetic algorithm”, 2005, Journal of Theoretical and Applied Information Technology”.

Naderi, B.; Khalili, M.; Tavakkoli-Moghaddam, R., “A hybrid artificial immune algorithm for a realistic variant of job shops to minimize the total completion time”, Computers &Industrial Engineering, 2008

Nair, G.J.; Narendran, T.T., “CASE: A clustering algorithm for cell formation with sequence data”, International Journal of Production research, 36(1), 1998

Nye, T.J.; Jewkes, E.M.; Dilts, D.M., “Optimal investment in setup reduction in manufacturing systems with WIP inventories”, European Journal of Operation Research, 135, 2001, 128-141

Offodile, O.F.; Mehrez, A.; J Grznar, J., "Cellular Manufacturing:A Taxonomic Review Framework", , Journal of Manufacturing Systems, Volume 13, No. 3

Ohta, H.; Nakamura, M., "Cell formation with reduction in setup times", Computers & Industrial Engineering, 42, 2002

Pandremenos, J.; Paralikas, J; Salonitis, K.; Chryssolouris, G. "Modularity concepts for the automotive industry: A critical review", CIRP Journal of Manufacturing Science and Technology 1 (2009) 148–152

Pine, II, J.B., "Mass customization: The new frontier in business competition" , Harvard Business School Press, Cambridge, MA, 1993

Rajamani, D.; Singh, N.; Aneja, P., "A model for cell formation in manufacturing systems with sequence dependence", Int. J. Prod. Res., 30, 1227-1235

Safaei, N.; Tavakkoli-Moghaddam, R., "Integrated multi-period cell formation and subcontracting production planning in dynamic cellular manufacturing", 2009, International Journal of Production Economics.

Salvador, F.; Forza, C.; Rungtusantham, "Modularity, product variety, production volume and component sourcing: theorizing beyond generic prescriptions", 2002, Journal of Operation Management, 20 , 549-575.

Samaddar, S.; Rabinowitz, G.; Mehrez, A., "Resource sharing and scheduling for cyclic production in a computer-integrated manufacturing cell", Computers &Industrial Engineering 36, 1999

Schaller, J.E.; Gupta, J.N.D.; Vakharia, A.J., "Scheduling a flow line manufacturing cell with sequence-dependent family setup times", European Journal of operations Research, 125, 2000

Seifoddini, H.; Djassemi, M., "Merits of the production volume based similarity coefficient in machine cell formation", Journal of manufacturing system, 14(1), 1995

Shambu, G.; Suresh, N. C., “Performance of hybrid manufacturing systems: A computer simulation investigation”, European Journal of Operational Research 120, 2000

Shtub, A., “Modelling group technology cell formation as a generalized assignment problem, “ International Journal of Production Research, 27(5) 775 - 782, 1989

Singh, N., Rajamani, R., “Cellular Manufacturing Systems”, Chapman & Hall, Inc, UK, 1995

Snead, C.S., “Group Technology”, Van Norstrand Reinhold, New York, 1989.

Starr, M.K., “Modular-production: a new concept”, Harvard Business Review, 43 (6), 1965

Suresh, N.C., “Partitioning work centers for group technology: Insights from an analytical model”, Decision Sciences, 22(4), 1991

Suresh, N.C., “Partitioning work centers for group technology: Analytical extension and shop-level simulation investigation, Decision Sciences, 23(2), 1992

Suresh, N.C.; Meredith, J.R., “Coping with the loss of pooling synergy in cellular manufacturing system”, Management Science, 40(4), 1994

Tariq, A., “Operational Design Of A Cellular Manufacturing System”, 2010, PhD Thesis, National University of Science & Technology (NUST)

Tavakkoli-Moghaddam, R.; Aryanezhad, M.B.; Safaei, N; Azaron, A.; “Solving a dynamic cell formation problem using metaheuristics”, Applied Mathematics and Computation 170, 761–780, 2005

Tompkins, J.A.; White, J.A., Bozer; Y.A.; Tanchoco, J.M.A., “Facilities Planning”. Wiley, New York, 2003

Vannelli, A.; Kumar, K. R., 1986, "A method for finding minimal bottleneck cells for grouping part-machine families", International Journal of Production Research, 24(2) 387-400.

Vilas, C.O.Y; Vandael, N. " A cost operations based product heterogeneity index", International Journal of Production Economics, 79 45-55, 2002

Wan, J.; Evers, P.T.; Dresner, M.E., "Too much of a good thing: The impact of product variety on operations and sales performance", Journal of Operations Management 30 (2012) 316–324

Wang, H.; Zhu, X.; Wang, H.; Hu,S.,J.; Lin,Z.; Chen, G.," Multi-objective optimization of product variety and manufacturing complexity in mixed-model assembly systems", Journal of Manufacturing Systems 30 (2011) 16–27

Wemmerlov, U., Hyer, N.L., "Cellular manufacturing in the US industry. A survey of users", International Journal of Production Research, 27(9), 1989

Wu[a], X.; Chu, C-H.; Wang, Y.; Yan,W.; "A genetic algorithm for cellular manufacturing design and layout", European Journal of Operational Research 181, 156–167, 2007

Wu[b], X.; Chu, C-H.; Wang, Y.; Yue, D.; " Genetic algorithms for integrating cell formation with machine layout and scheduling", Computers & Industrial Engineering 53 277–289, 2007

www.alkolas3d.tripod.com

www.cs.kent.ac.uk/people/staff/aaf/ (Natural Computation CO637, Evolutionary Computation II-power point by Alex Freitas)

www.scribd.com/doc/396655/Genetic-Algorithm-Overview

Yang, J., Deane, R.H., “Setup time reduction and competitive advantage in a closed manufacturing cell”, European Journal of Operational Research 69 (3), pp. 413-423, 1993

Yasuda, K.; Yin, Y., “Dissimilarity measure for solving the cell formation problem in cellular manufacturing”, Computers and Industrial Engineering, 39, 2002

Yeniay, O., “Penalty function methods for constrained optimization with Genetic Algorithm” , Mathematical and Computational Applications, Vol. 10, No. 1, pp. 45-56, 2005

APPENDIX

Table A.4.1- MCIM for numerical example in Chapter 4 [machine, part, operation]

1,1,1 1,1,2 1,1,4 1,1,5 1,2,3 1,2,4 1,2,6 1,2,7 1,2,8 1,3,2 1,3,5 1,3,6 1,3,7 1,4,1 1,4,2 1,4,3 1,5,2 1,7,3 1,9,2 1,9,6 1,9,7 1,10,2 1,11,1 1,11,8 1,13,1 1,13,4 1,13,5 1,14,2 1,14,3 1,14,4 1,14,5 1,14,6 1,15,4 1,15,6 1,16,1 1,16,2 1,17,1 1,17,2 1,17,3 1,17,4 1,18,2 1,18,4 1,18,5 1,18,6 1,18,8 1,22,1 1,22,5 1,22,6 1,23,1 1,23,2 1,23,3 1,23,4 1,23,5 1,25,1 1,25,2 1,25,3 1,25,4

2,2,1 2,5,1 2,5,5 2,5,7 2,6,1 2,7,2 2,9,3 2,9,4 2,9,5 2,11,3 2,12,1 2,13,2 2,14,1 2,15,2 2,15,3 2,16,5 2,17,5 2,17,6 2,18,1 2,18,7 2,19,1 2,19,2 2,22,2 2,22,3

3,1,3 3,3,1 3,3,3 3,4,4 3,5,3 3,5,8 3,11,2 3,11,4 3,11,6 3,11,7 3,12,2 3,13,6 3,13,7 3,15,1 3,15,7

4,2,2 4,3,4 4,5,4 4,7,1 4,9,1 4,17,7 4,19,3

5,2,5 5,5,6 5,15,5 5,22,4

6,10,1 6,11,5 6,13,3 6,16,3 6,16,4 6,18,3

Table A.4.2.1- Operation-level setup times for numerical example in Chapter 4

15	19	26	13	14	12	18	18	17	15	28	21	12	35	27	14	24	34	34	24	22	39	23	30	27	17	28	27	23	30	33
38	12	15	22	30	33	17	12	29	27	25	23	14	23	12	21	31	23	22	25	33	25	21	12	32	12	36	24	20	27	30
13	24	30	20	35	33	11	30	36	27	35	20	30	24	35	34	39	35	36	14	18	32	11	36	38	13	11	24	38	34	39
32	20	37	20	16	20	27	20	15	16	27	23	28	34	19	32	26	25	17	37	24	25	17	29	18	23	30	12	32	13	36
27	11	35	15	14	20	32	26	30	26	11	31	31	27	13	20	10	24	18	23	30	11	12	39	10	23	38	38	35	27	27
10	28	29	33	20	17	24	37	18	33	17	11	30	10	28	13	34	35	10	11	34	35	26	22	20	23	35	16	36	16	37
16	17	37	31	29	31	36	12	17	12	37	16	15	21	13	19	29	15	29	32	16	15	33	22	24	14	34	32	17	39	12
33	37	33	21	35	27	28	35	18	13	12	34	11	11	25	19	15	21	30	16	24	35	23	10	38	29	31	23	24	27	24
16	23	16	35	12	28	21	10	23	31	35	25	12	12	21	17	34	18	22	33	16	23	30	23	15	12	18	20	37	32	12
15	39	21	10	38	15	20	37	19	20	35	13	34	31	20	34	18	15	39	12	25	14	11	13	29	25	38	22	37	39	16
11	17	27	31	39	20	13	24	31	27	35	23	33	32	34	31	32	19	10	38	17	14	35	16	28	20	27	37	34	33	21
23	16	33	20	11	19	10	24	22	39	13	11	24	28	32	10	15	35	19	13	32	18	37	34	38	11	24	33	15	15	12
36	28	39	32	39	15	39	24	25	10	35	18	38	10	11	13	22	32	12	17	20	13	31	13	33	27	30	39	15	25	38
13	37	28	13	17	39	13	19	11	38	12	18	30	21	28	24	18	26	10	39	13	20	10	38	17	16	33	15	16	15	28
23	19	14	38	13	39	17	26	34	25	18	36	11	35	25	21	29	26	24	15	34	12	15	13	12	36	28	32	36	36	23
19	14	25	26	32	35	13	15	26	16	21	14	34	16	32	36	38	32	15	25	31	30	31	25	11	37	26	20	10	32	14
13	13	36	22	38	12	13	34	23	24	33	16	38	34	27	23	29	38	20	30	28	38	30	15	21	34	33	26	13	39	32
31	29	28	32	13	24	10	12	33	31	29	18	19	28	39	11	13	25	11	24	12	25	33	18	19	12	38	27	30	31	34
38	11	39	23	25	19	17	19	38	10	31	24	37	25	25	25	10	19	12	18	12	14	15	20	14	34	20	35	11	11	35
39	20	18	13	18	17	22	26	19	23	28	21	20	23	13	38	24	38	10	11	35	24	31	33	17	33	25	18	15	26	39
10	28	19	23	16	30	16	19	37	38	19	14	31	15	16	16	26	16	26	27	31	24	30	26	31	32	31	14	39	25	32
33	34	11	25	32	23	31	16	38	28	32	27	17	31	32	26	30	34	29	14	16	30	33	36	32	14	35	18	18	31	12
11	34	39	18	12	14	15	12	19	19	38	33	28	16	29	18	26	26	32	33	18	32	16	12	27	35	37	38	12	37	28
30	22	30	28	19	32	28	32	10	35	25	12	37	26	17	18	17	38	31	36	30	19	15	38	29	20	30	20	37	32	15
15	22	21	24	28	26	24	16	18	19	38	13	25	15	11	35	25	18	14	10	15	24	37	22	11	24	19	19	10	13	10
33	15	30	37	26	17	28	13	26	34	34	11	39	35	25	27	24	27	23	24	20	28	38	31	15	12	23	12	14	16	16
26	22	19	10	24	19	23	11	31	19	29	25	35	35	26	20	35	39	34	26	13	34	19	10	25	13	29	31	12	38	38
31	28	26	20	22	34	29	20	12	22	21	25	14	14	24	32	35	30	34	22	11	32	27	17	19	25	25	34	10	33	14
31	32	12	19	28	12	22	37	28	15	28	13	39	14	24	36	39	38	15	26	16	20	37	32	17	12	26	24	37	21	23
12	31	16	33	18	13	22	38	26	20	11	32	16	30	25	14	16	37	19	13	14	27	28	28	11	16	23	38	27	36	11
27	12	23	27	18	35	38	25	38	37	23	23	11	10	28	21	20	19	12	24	39	31	25	27	23	33	14	12	12	10	11
12	27	27	10	22	34	11	10	13	27	23	29	15	19	27	22	30	22	28	16	17	30	18	21	18	23	30	23	26	22	10
12	28	24	31	28	33	27	35	28	31	27	20	34	38	28	31	33	36	27	38	29	35	15	15	36	16	13	30	20	18	28
12	23	30	26	35	20	29	27	32	12	18	22	39	24	24	18	22	17	12	35	18	34	27	33	29	35	31	19	30	22	28
23	31	34	23	21	30	34	10	27	39	30	28	11	34	20	27	39	35	33	28	32	23	32	31	10	34	20	37	35	29	20
35	31	22	20	29	18	19	21	17	23	13	10	27	36	26	30	22	28	39	28	39	19	39	34	22	10	18	17	18	37	33
20	37	16	24	22	21	39	31	26	30	33	13	16	22	22	21	14	13	22	26	20	15	24	29	16	10	28	12	27	14	20

Table A.4.2.2- Operation-level Setup times for numerical example in Chapter 4 (continued)

17 36 10 11 22 29 23 28 25 21 35 16 26 28 18 29 18 17 10 37 39 31 35 18 26 13 29 34 35 13 28
32 33 29 24 38 25 11 13 22 38 21 16 24 16 27 37 29 16 34 21 29 32 26 13 11 15 19 17 10 22 36
11 17 36 32 15 33 34 10 27 15 15 33 31 18 28 28 20 31 37 25 30 15 29 10 39 13 15 24 12 17 22
26 27 32 32 20 11 11 37 25 39 31 38 11 17 38 26 26 29 19 12 16 36 15 28 14 10 13 27 38 33 11
31 22 15 17 36 32 21 26 11 21 15 27 14 28 25 16 14 19 18 24 23 32 25 33 38 33 12 17 10 16 38
10 21 11 19 15 33 26 23 11 35 33 24 30 33 21 37 39 30 23 15 12 19 31 22 16 11 37 20 31 25 12
12 39 30 15 21 38 23 26 38 38 26 31 26 16 27 10 34 29 38 29 24 39 38 27 35 16 34 28 36 18 17
18 10 29 35 36 39 10 20 11 13 35 35 28 20 21 17 27 14 36 27 20 17 34 11 22 11 18 23 35 17 31
34 27 26 38 22 13 30 12 27 18 38 15 39 37 11 23 34 17 31 13 37 31 32 27 15 23 34 34 25 35 21
14 11 22 16 36 23 13 30 10 28 31 13 38 21 16 29 38 10 11 13 29 33 29 23 16 35 15 34 20 25 10
16 15 35 31 38 39 16 32 22 25 10 35 32 35 12 30 16 37 12 18 10 10 26 38 33 10 36 24 33 27 10
35 12 10 27 15 36 11 18 11 18 35 26 35 29 32 26 35 22 33 32 31 22 12 18 18 18 27 34 20 10 15
19 29 35 33 30 29 29 15 21 11 21 32 33 13 29 37 32 13 23 30 14 28 14 23 39 22 34 39 30 15 27
14 17 36 16 33 10 39 30 18 30 31 15 11 10 33 19 25 38 39 36 12 15 37 26 33 28 15 19 13 10 12
15 31 35 27 37 39 24 11 27 18 15 17 38 28 34 32 23 38 24 11 33 30 24 21 27 17 32 11 33 31 23
21 35 31 25 20 25 11 32 22 31 15 25 13 18 19 32 28 34 22 17 15 28 34 10 29 34 14 13 35 18 14
11 30 13 33 24 10 16 17 25 35 16 30 39 18 13 35 37 29 10 14 28 17 34 10 22 13 25 24 31 16 16
34 35 12 29 31 22 34 39 19 23 37 36 15 38 27 38 29 37 37 31 16 35 20 11 13 22 16 30 35 37 14
34 29 10 26 21 31 29 10 34 12 31 19 17 30 39 23 33 33 11 23 15 36 39 25 29 18 28 33 39 35 31
33 26 12 28 28 21 19 18 21 13 13 34 33 39 12 17 22 11 17 16 31 17 33 10 26 20 32 35 36 10 27
22 20 19 15 24 33 32 15 16 20 18 28 21 25 27 35 27 29 22 23 17 32 17 19 13 16 13 28 31 20 36
18 27 36 25 21 32 39 35 36 11 35 14 31 23 38 25 20 29 12 31 23 33 15 22 29 10 24 28 25 17 23
33 30 33 20 23 22 11 31 10 18 21 18 37 22 38 39 22 23 10 22 38 22 15 25 18 11 12 34 39 38 28
32 17 11 27 39 35 10 33 34 10 31 15 13 34 22 13 32 39 34 29 32 14 19 20 34 24 39 28 15 33 32
13 30 11 37 21 29 38 38 33 32 39 13 33 12 18 13 14 20 22 20 21 22 21 30 37 26 36 18 10 15 38
18 29 25 37 13 22 12 12 20 10 12 13 21 36 38 27 17 12 28 27 29 15 19 12 39 11 33 10 18 11 19
32 10 12 14 27 35 28 21 20 10 27 31 16 19 13 20 13 17 23 20 34 13 22 23 11 20 22 27 37 21 13
20 17 20 33 26 33 31 19 15 26 21 15 21 20 29 37 23 10 38 31 35 21 37 11 37 19 34 29 18 30 30
11 38 35 20 28 19 22 38 20 18 15 21 24 17 23 16 31 16 18 30 28 36 25 27 31 30 32 28 23 15 35
16 11 14 19 23 18 13 22 33 13 27 13 10 21 24 18 27 38 39 18 19 17 29 11 19 15 18 19 36 39 33
13 31 14 11 34 35 37 26 33 34 26 33 18 33 34 22 27 38 19 20 11 24 20 37 23 24 19 14 35 36 38
19 28 39 19 27 20 18 14 15 36 19 23 16 22 10 10 16 25 31 26 21 29 37 23 35 30 30 36 25 21 28
12 17 32 34 35 12 32 25 38 26 19 38 33 35 35 10 15 28 11 35 10 37 33 34 21 35 26 24 11 17 20
14 10 26 31 20 25 25 12 35 38 23 12 27 28 15 32 17 32 19 37 36 33 31 35 39 38 19 15 14 19 35
32 25 32 12 13 30 28 27 12 38 13 37 15 14 31 28 32 19 16 38 39 28 11 26 23 37 15 14 11 20 21
12 22 38 25 34 26 27 37 23 24 14 29 21 18 23 38 37 39 17 11 21 14 13 25 12 29 11 15 15 36 27
19 20 22 33 11 19 38 17 18 34 14 14 23 35 18 30 32 29 24 39 38 33 35 10 24 13 15 32 25 17 27

Table A.4.2.3- Operation-level Setup times for numerical example in Chapter 4 (continued)

37 24 24 10 34 11 31 35 24 30 32 38 16 18 31 14 17 26 24 39 28 37 23 38 16 22 18 10 29 18 20
24 29 16 18 22 38 34 27 19 34 10 37 14 18 11 38 32 22 11 24 36 16 39 23 18 31 33 26 24 25 12
12 20 31 10 33 18 27 33 23 11 34 25 36 11 32 11 29 31 28 30 34 27 37 14 11 11 39 38 31 30 13
32 15 10 23 22 32 16 23 33 36 31 15 28 32 31 19 23 39 17 32 31 23 22 20 29 12 17 34 35 10 23
22 29 27 18 20 21 21 24 17 34 13 29 19 26 17 36 32 10 39 36 12 33 10 13 25 11 39 16 17 19 22
37 35 36 18 22 14 26 23 20 19 21 28 37 16 18 24 10 13 17 39 16 12 22 37 23 20 27 33 20 37 13
36 34 16 27 24 20 21 12 34 24 36 33 14 25 20 12 20 31 16 29 33 37 19 11 25 32 14 34 21 13 24
26 18 38 17 30 33 24 34 36 17 11 38 11 12 21 32 13 21 21 31 10 38 39 21 10 29 37 17 34 15 11
36 37 18 23 19 33 21 23 29 16 33 27 25 32 20 37 11 37 10 16 37 11 37 25 14 38 21 17 25 12 21
27 35 33 11 10 10 10 35 30 30 37 28 36 14 29 21 24 28 10 38 19 29 27 39 19 39 16 10 16 33 38
20 33 36 39 14 12 10 34 34 33 36 39 11 28 39 21 31 18 26 13 25 30 19 21 10 28 22 32 26 18 19
17 37 38 19 13 30 35 16 13 19 27 17 14 23 25 28 23 33 13 36 11 30 34 17 24 32 17 21 35 17 12
32 27 30 23 16 30 29 28 29 26 14 10 32 19 10 13 32 32 34 27 27 34 15 23 31 23 13 24 23 25 12
21 16 39 38 28 24 21 14 15 27 33 20 12 17 17 37 10 12 20 30 28 36 14 25 30 21 35 37 33 22 27
19 11 21 34 10 23 33 31 35 18 13 20 35 21 10 22 25 31 30 34 20 16 14 38 19 35 26 32 12 12 21
27 24 17 38 16 20 30 12 21 13 19 32 34 31 25 24 10 36 36 18 11 13 23 31 30 25 38 39 23 38 39
18 34 13 29 39 16 39 17 37 22 25 22 23 32 28 33 33 28 15 37 18 28 21 23 25 31 32 10 28 26 25
35 33 19 25 13 15 24 28 25 10 33 35 30 28 34 23 35 28 21 12 36 10 22 22 31 16 25 20 21 17 37
13 28 13 15 21 23 32 21 29 24 32 33 30 11 24 38 36 36 11 23 18 27 29 34 14 23 25 34 24 22 27
27 25 26 26 39 30 37 30 19 18 16 17 13 32 10 11 15 33 35 26 15 14 16 28 31 15 18 15 33 13 32
28 17 16 39 36 34 26 23 32 21 30 35 28 37 20 10 13 17 33 14 16 32 10 31 31 16 15 23 34 29 35
33 34 23 37 27 30 39 31 15 34 10 30 27 21 12 13 27 17 13 23 23 29 11 25 38 38 18 10 12 24 14
23 33 16 27 26 26 21 14 18 31 16 35 29 16 38 17 11 19 21 26 27 26 33 17 21 32 37 19 24 27 11
38 22 34 38 18 11 16 22 20 14 28 35 19 14 34 14 29 19 14 35 33 32 31 17 34 33 11 29 30 37 33
38 14 22 26 19 38 27 37 14 18 27 18 23 12 31 19 33 37 34 34 14 13 15 31 13 12 28 20 29 20 13
14 29 14 24 23 19 38 25 13 38 31 32 23 38 19 28 15 31 10 36 21 18 11 28 27 27 12 38 12 33 16
26 29 36 25 31 33 18 30 37 26 11 36 20 23 21 30 14 33 11 35 38 39 25 14 25 27 24 24 16 17 27
25 32 16 38 38 19 31 19 28 30 35 10 12 39 36 20 22 32 29 32 16 17 13 36 30 35 22 21 17 21 31
25 23 22 16 36 18 21 35 28 36 19 34 16 26 12 38 28 31 36 18 27 38 29 17 19 30 25 28 32 26 19
37 31 37 25 21 36 24 34 39 19 36 21 25 35 36 30 37 15 10 17 39 15 36 28 25 13 17 30 27 15 22
33 10 14 29 38 25 18 30 20 29 37 27 34 14 27 25 29 22 13 12 32 33 31 29 17 33 20 14 23 17 38
10 18 38 11 22 27 24 19 11 29 33 16 22 30 37 22 20 10 25 23 35 17 27 24 34 23 13 28 16 14 14
25 29 32 24 34 22 34 32 15 23 22 33 24 20 22 23 27 12 20 17
37 26 21 37 35 26 17 39 10 18 35 37 14 32 15 36 15 28 18 12 38 15 32 17 26 27 23 23 15 36 19
14 34 15 39 26 39 22 39 35 26 31 37 28 33 37 32 36 19 11 10 10 14 35 26 23 36 34 16 27 26 23
39 21 24 17 35 20 20 35 15 28 28 26 19 18 31 25 36 20 36 30 34 17 28 16 34 30 33 34 11 30 20

Table A.4.2.4- Operation-level Setup times for numerical example in Chapter 4 (continued)

25 16 14 10 36 39 26 32 19 31 12 16 24 17 30 18 22 14 11 20 30 23 26 11 17 39 18 30 32 29 18
39 17 34 17 17 25 23 29 31 32 29 16 21 31 34 19 38 36 39 17 14 38 12 15 18 12 35 16 31 19 33
27 11 33 12 30 16 28 29 19 33 29 11 31 13 24 18 22 14 19 21 14 33 29 34 29 35 32 11 26 27 12
35 11 28 30 37 35 27 24 22 39 31 28 24 34 28 11 32 38 34 16 17 33 11 25 28 35 34 32 16 26 34
16 21 24 30 34 36 36 17 15 18 11 31 35 31 11 29 10 37 16 33 24 34 15 27 12 26 15 16 32 39 26
25 13 22 21 26 33 14 39 30 35 26 15 30 36 35 23 23 22 36 36 10 26 32 10 30 35 32 20 15 26 31
20 34 39 24 30 12 16 26 34 30 27 32 26 26 38 15 31 10 10 12 11 13 18 24 13 25 10 38 37 30 35
33 25 16 33 20 22 39 29 33 27 29 13 39 23 20 12 35 11 31 38 23 23 37 14 16 27 31 29 24 16 11
12 28 32 25 27 27 38 34 32 30 26 25 20 35 16 20 37 12 14 14 24 20 25 20 16 16 22 16 23 31 27
35 25 35 34 28 29 39 38 21 12 17 37 19 22 21 38 32 19 27 29 19 20 10 36 14 32 35 23 25 21 39
27 18 14 29 21 29 28 27 19 36 39 25 16 34 17 24 14 38 16 35 31 24 29 28 28 37 13 24 35 27 17
35 31 13 13 37 29 16 35 23 14 10 32 24 39 18 10 24 12 26 11 35 36 11 30 35 30 34 38 17 30 38
11 37 32 28 30 21 11 19 21 30 33 12 13 12 32 28 11 38 17 35 36 12 26 14 37 14 33 17 20 15 34
23 37 13 28 30 25 36 38 22 26 28 11 32 16 16 12 14 33 10 12 10 15 11 10 35

38 10 11 33 28 32 18 14 15 17 12 33 28 39 31 16 22 12 31 23 18 35 36 16 38 19 33 35 27 26 19
15 27 37 30 10 22 25 29 13 10 23 17 16 21 25 22 25 19 15 33 15 35 23 36 26 18 25 32 21 30 32
16 18 39 26 12 15 28 38 36 26 21 36 27 11 22 25 22 14 19 32 17 10 30 14 19 20 18 31 30 10 21
17 23 35 26 29 25 30 16 36 32 13 26 37 15 32 15 36 16 36 17 38 36 14 15 25 18 10 20 34 14 22
22 18 16 25 10 39 29 29 13 18 37 31 25 31 29 39 18 15 12 13 21 20 27 33 38 38 15 29 26 39 33
21 31 39 11 32 34 26 18 30 38 39 35 20 14 16 28 36 37 33 33 11 27 13 11 23 14 32 37 17 33 13
36 25 35 33 16 20 19 35 36 33 23 32 39 19 16 33 35 25 31 34 17 31 37 29 29 26 25 12 32 33 17
30 39 17 36 19 25 16 19 29 33 15 18 31 38 19 16 21 28 14 35 13 24 36

34 11 25 32 11 17 35 24 38 33 28 25 22 13 31 23 22 10 30 10 18 35 12 38 18 37 31 10 28 26 13
12 32 32 17 20 18 24 37 14 13 14

34 14 26 10 25 15 33 37 12 21 39 10 18 28 38 20 29 31 26 27

28 16 15 29 38 20 25 34 26 37 36 24 10 11 18 18 14 29 14 36 14 20 39 12 19 11 22 14 13 29

Table A.4.3-Processing times for the operations of numerical example in Chapter 4

[machine, part, operation, time]

0,0,0,6 0,0,1,2 0,0,3,1 0,0,4,1 0,1,2,1 0,1,3,6 0,1,5,1 0,1,6,3 0,1,7,4 0,2,1,2 0,2,4,6 0,2,5,6 0,2,6,1 0,3,0,5 0,3,1,6 0,3,2,6 0,4,1,3 0,6,2,3 0,8,1,3 0,8,5,4 0,8,6,2 0,9,1,6 0,10,0,1 0,10,7,4 0,12,0,4 0,12,3,4 0,12,4,2 0,13,1,5 0,13,2,2 0,13,3,3 0,13,4,2 0,13,5,5 0,14,3,5 0,14,5,4 0,15,0,4 0,15,1,5 0,16,0,3 0,16,1,6 0,16,2,1 0,16,3,6 0,17,1,6 0,17,3,3 0,17,4,4 0,17,5,5 0,17,7,1 0,21,0,6 0,21,4,2 0,21,5,3 0,22,0,2 0,22,1,2 0,22,2,3 0,22,3,2 0,22,4,1 0,24,0,5 0,24,1,4 0,24,2,6 0,24,3,4
1,1,0,2 1,4,0,5 1,4,4,2 1,4,6,2 1,5,0,2 1,6,1,2 1,8,2,1 1,8,3,2 1,8,4,2 1,10,2,6 1,11,0,4 1,12,1,5 1,13,0,3 1,14,1,2 1,14,2,3 1,15,4,5 1,16,4,6 1,16,5,1 1,17,0,1 1,17,6,2 1,18,0,3 1,18,1,2 1,21,1,3 1,21,2,6
2,0,2,6 2,2,0,2 2,2,2,4 2,3,3,2 2,4,2,1 2,4,7,3 2,10,1,3 2,10,3,6 2,10,5,6 2,10,6,1 2,11,1,2 2,12,5,5 2,12,6,1 2,14,0,5 2,14,6,2
3,1,1,5 3,2,3,1 3,4,3,3 3,6,0,2 3,8,0,2 3,16,6,4 3,18,2,3
4,1,4,2 4,4,5,3 4,14,4,3 4,21,3,6
5,9,0,1 5,10,4,4 5,12,2,4 5,15,2,3 5,15,3,1 5,17,2,3

Table A.4.4- MCIM for 3 machine, 3 parts, 4 operations- computational performance

1,2,2 1,2,3 1,2,4 1,3,2 1,3,3 1,3,4
2,1,1 2,1,2 2,1,4 2,2,1
3,1,3 3,3,1

Table A.4.5- Setup time for 3 machine, 3 parts, 4 operations- computational performance

31 34 12 13 32 32 31 36 18 15 27 16 21 28 19 32 27 29 35 34 33 31 12 13 13 24 31 11 33 38
27 24 32 37 37 29 33 31 29 38 26 15
30 22

Table A.4.6.1 Comparison of objective values and computation times between GA-based and LINGO for 90 data sets

	GA		LINGO			GA		LINGO	
	Best obj	Time(sec)	Best.obj	Time(sec)		Best obj	Time(sec)	Best.obj	Time(sec)
3 parts	3 operations				4 parts	4 operations			
*	39	53.422	39	95		123	155.731	174	3600
*	37	48.687	37	20		145	154.345	**	3600
*	11	48.531	11	1		119	152.224	214	3600
*	36	48.484	36	24		113	151.965	152	3600
*	32	52.906	32	52		105	152.710	143	3600
*	35	52.703	35	32		150	152..622	160	3600
	Ave	50.789	Ave.Gap	0.00		Ave	153.395	Ave.Gap	-0.26
3 parts	4 opertions				3 parts	6 operations			
	68	142.459	68	3600		150	167.279	160	3600
	48	141.368	49	3600		181	168.229	**	3600
	72	142.705	72	3600		160	170.145	190	3600
	62	145.457	62	3600		158	166.634	184	3600
	78	150.536	78	3600		146	166.339	159	3600
	68	145.055	68	3600		145	163.795	186	3600
	Ave	144.597	Ave.Gap	0.00		Ave	167.070	Ave.Gap	-0.13
4 parts	3 operations				6 parts	3 operations			
*	67	139.906	67	7218		161	159.810	190	3600
*	63	138.773	63	18034		164	160.691	196	3600
	85	138.951	85	3600		170	156.988	**	3600
	81	136.877	82	3600		141	156.837	175	3600
	68	133.581	67	3600		154	161.577	183	3600
*	58	133.326	58	1782		147	157.527	**	3600
	Ave	136.902	Ave.Gap	0.00		Ave	158.905	Ave.Gap	-0.55

*Optimal solution was found by LINGO

**LINGO did not provide a feasible solution within 1 hour of computation

Table A.4.6.2 Comparison of objective values and computation times between GA-based and LINGO for 90 data sets

	GA		LINGO			GA		LINGO	
	Best obj	**Time(sec)**	**Best.obj**	**Time(sec)**		**Best obj**	**Time(sec)**	**Best.obj**	**Time(sec)**
3 parts	5 operations				**5 parts**	4 operations			
	112	*155.815*	106	*3600*		220	*175.283*	305	*3600*
	128	*155.982*	150	*3600*		205	*169.309*	250	*3600*
	102	*153.716*	107	*3600*		198	*168.138*	328	*3600*
	115	*154.225*	146	*3600*		180	*167.281*	198	*3600*
	108	*151.414*	121	*3600*		184	*168.437*	**	*3600*
	104	*150.432*	107	*3600*		193	166.137	354	*3600*
	Ave	*153.597*	**Ave.Gap**	*-0.08*		**Ave**	169.098	**Ave.Gap**	*-0.28*
5 part	3 operations				**3 parts**	7 operations			
	127	*149.027*	122	*3600*		207	*181.312*	267	*3600*
	114	*148.191*	133	*3600*		212	*180.451*	236	*3600*
	103	*145.001*	119	*3600*		209	*181.979*	222	*3600*
	107	*146.082*	105	*3600*		199	*188.153*	**	*3600*
	119	*146.170*	127	*3600*		198	*181.941*	285	*3600*
	110	*147.152*	151	*3600*		195	*179.070*	506	*3600*
	Ave	146.937	**Ave.Gap**	*-0.09*		**Ave**	182.151	**Ave.Gap**	*-0.26*
7 parts	3 operations				**6 parts**	4 operations			
	220	*197.477*	275	*3600*		269	183.678	292	*3600*
	232	*173.412*	233	*3600*		276	184.134	376	*3600*
	212	*180.901*	**193**	*3600*		246	179.688	399	*3600*
	206	*177.366*	**	*3600*		265	187.370	**	*3600*
	199	*172.240*	225	*3600*		262	181.198	434	*3600*
	212	*170.283*	327	*3600*		267	182.177	267	*3600*
	Ave	178.613	**Ave.Gap**	*-0.11*		**Ave**	183.041	**Ave.Gap**	*-0.22*

Table A.4.6.3 Comparison of objective values and computation times between GA-based and LINGO for 90 data sets (continued)

	GA		LINGO	
	Best obj	**Time(sec)**	**Best.obj**	**Time(sec)**
3 parts	8 operations			
	261	*199.777*	339	*3600*
	281	*199.369*	342	*3600*
	267	*199.051*	**	*3600*
	268	*201.626*	**	*3600*
	268	*197.129*	418	*3600*
	279	*196.948*	**	*3600*
	Ave	198.983	**Ave.Gap**	*-0.13*
8 parts	3 operations			
	267	*189.219*	**248**	*3600*
	276	*189.119*	**	*3600*
	286	*190.806*	**285**	*3600*
	255	*190.148*	**	*3600*
	270	*184.773*	339	*3600*
	269	*188.202*	292	*3600*
	Ave	188.711	**Ave.Gap**	*-0.05*
5 parts	5 operations			
	297	*193.051*	340	3600
	299	*193.325*	**	3600
	279	*190.818*	289	3600
	276	*190.696*	**	3600
	305	*187.949*	317	3600
	310	*183.953*	323	3600
	Ave	189.965	**Ave.Gap**	*-0.06*

Table A.5.3-Results of dynamic programming based heuristic (period 3 and 2)

Best policy	Origin node	Destination node	Origin cost	Destination cost	Transition cost
68227.8	0	5	44501.6	21426.2	2300
71340.1	1	14	43965.3	25614.8	1760
57330.6	2	5	35104.4	21426.2	800
56952.4	3	5	34246.2	21426.2	1280
57436.2	4	5	34860	21426.2	1150
68842.8	5	5	40426.6	21426.2	6990
60161.4	6	5	37460.2	21426.2	1275
65806.3	7	5	42465.1	21426.2	1915
58561.3	8	5	35375.1	21426.2	1760
62818.4	9	5	40032.2	21426.2	1360
65507.2	10	5	42456	21426.2	1625
59698.9	11	5	36912.7	21426.2	1360
61971.5	12	5	39185.3	21426.2	1360
60853.6	13	5	37877.4	21426.2	1550
51727.6	14	5	28436.4	21426.2	1865
68807.6	15	5	44041.4	21426.2	3340
66426.8	16	5	42720.6	21426.2	2280
58384.1	17	5	35677.9	21426.2	1280
62764.2	18	5	38983	21426.2	2355
48609.1	**19**	**5**	**26862.9**	**21426.2**	**320**

Table A.5.4-Results of dynamic programming based heuristic (period 2 and 1)

Best policy	Origin node	Destination node	Origin cost	Destination cost	Transition cost
76894.1 0		19	25230	26862.9	3055
73134.1 1		19	21740	26862.9	2785
69185.1 2		**19**	**19211**	**26862.9**	**1365**
77293.1 3		19	21618.6	26862.9	7065
75544.1 4		19	24709.6	26862.9	2225
73501.1 5		19	22207.4	26862.9	2685
70767.7 6		19	20413.6	26862.9	1745
84063.1 7		19	26893.6	26862.9	8560
73223.1 8		19	22788.6	26862.9	1825
74198.1 9		19	22343.6	26862.9	3245
74962.1 10		19	22418.2	26862.9	3935
69614.9 11		19	19390.8	26862.9	1615
69249.9 12		19	19085.8	26862.9	1555
75644.1 13		19	24560	26862.9	2475
69940.5 14		19	19506.4	26862.9	1825
74538.1 15		19	22203.6	26862.9	3725
80111.1 16		19	24812.2	26862.9	6690
70547.3 17		19	20193.2	26862.9	1745
76855.1 18		19	21650.8	26862.9	6595
73715.1 19		19	22430.6	26862.9	2675

Table A.5.5 -MCIM for 12 parts , 3 operations
[machine, part, operation]

```
1,1,1 1,1,2 1,1,3 1,2,1 1,2,2 1,2,3 1,3,1 1,3,3 1,4,2 1,5,2 1,6,3 1,7,1
1,8,2 1,8,3 1,9,2 1,10,2 1,11,1 1,11,2 1,11,3 1,12,3
2,3,2 2,4,3 2,5,1 2,6,2 2,7,2 2,7,3 2,9,3
3,1,1 3,4,1 3,5,3 3,9,1 3,10,3 3,12,1 3,12,2
4,6,1
5,8,1
6,10,1
```

Table A.5.6 - Processing times for 12 part, 3operations, 3 machines
[machine, part, operation, time]

```
0,0,0,2 0,0,1,5 0,0,2,5 0,1,0,6 0,1,1,3 0,1,2,1 0,2,0,1 0,2,2,5 0,3,1,5
0,4,1,3 0,5,2,5 0,6,0,5 0,7,1,3 0,7,2,4 0,8,1,3 0,9,1,4 0,10,0,5
0,10,1,3 0,10,2,1 0,11,2,4
1,2,1,4 1,3,2,3 1,4,0,4 1,5,1,6 1,6,1,1 1,6,2,5 1,8,2,1
2,0,0,3 2,3,0,5 2,4,2,3 2,8,0,6 2,9,2,5 2,11,0,1 2,11,1,4
3,5,0,3
4,7,0,2
5,9,0,6
```

Table A.5.7 - setup times for 12 part, 3operations, 3 machines

```
38 18 22 19 21 13 15 29 28 34 10 37 36 33 16 21 15 24 32 10 19 27 13 37
22 26 10 21 16 15 27 25 14 21 32 30 20 31 14 26 20 37 21 17 37 17 27 13
13 15 39 29 18 21 28 12 26 36 20 13 18 10 21 22 15 20 39 34 27 18 13 35
31 12 30 11 36 24 20 16 31 28 19 28 14 11 34 33 29 28 18 10 18 17 27 18
23 28 23 27 31 10 27 23 14 39 36 15 31 30 39 29 36 38 13 24 38 34 39 29
12 25 35 13 13 23 37 14 33 18 10 28 28 10 36 18 21 39 18 29 17 22 22 38
32 18 39 20 27 18 21 25 38 26 31 12 34 12 25 28 36 32 36 15 33 29 22 34
16 21 28 12 31 11 39 27 16 32 39 15 12 30 30 13 29 21 28 31 19 25 23 32
35 12 35 38 26 37 37 32 12 39 34 21 39 36 19 38 22 15 35 24 29 11 12 35
10 38 13 39 13 29 26 17 29 29 17 16 29 23 15 37 26 36 35 18 32 15 34 34
11 36 11 36 33 33 35 25 23 22 18 12 15 33 16 11 38 16 32 11 14 16 12 29
11 25 30 23 36 24 39 22 19 13 24 34 30 15 19 26 13 14 12 18 18 17 15 28
21 12 35 27 14 24 34 34 24 22 39 23 30 27 17 28 27 23 30 33 38 12 15 22
30 33 17 12 29 27 25 23 14 23 12 21 31 23 22 25 33 25 21 12 32 12 36 24
20 27 30 13 24 30 20 35 33 11 30 36 27 35 20 30 24 35 34 39 35 36 14 18
32 11 36 38 13 11 24 38 34 39 32 20 37 20 16 20 27 20 15 16
27 23 28 34 19 32 26 25 17 37 24 25 17 29 18 23 30 12 32 13 36 27 11 35
15 14 20 32 26 30 26 11 31 31 27 13 20 10 24 18 23 30
11 12 39 10 23 38 38 35 27 27 10 28 29 33 20 17 24 37 18 33 17 11 30 10
28 13 34 35 10 11 34 35 26 22 20 23 35 16 36 16 37 16
```