1989

Mathematical programming approach to cell formation problems in flexible manufacturing systems.

Raja Gunasingh Kasilingham

University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Recommended Citation
https://scholar.uwindsor.ca/etd/2679

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000ext. 3208.
NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.
MATHEMATICAL PROGRAMMING APPROACH TO CELL FORMATION PROBLEMS IN FLEXIBLE MANUFACTURING SYSTEMS

by

Raja Gunasingh Kasilingam

A Dissertation submitted to the Faculty of Graduate Studies and Research through the Department of Industrial Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy at the University of Windsor

Windsor, Ontario, Canada 1989
The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-54534-8
(c) Raja Gunasingh Kasilingam, 1989

All Rights Reserved
I hereby declare that I am the sole author of this dissertation. I authorize the University of Windsor to lend this dissertation to other institutions or individuals for the purpose of scholarly research.

Raja Gunasingh Kasilingam

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

Raja Gunasingh Kasilingam
The University of Windsor requires the signatures of all persons using or photocopying this dissertation. Please sign below, and give address and date.
ABSTRACT

The objectives of this dissertation are to develop methodologies to solve large instances of the following cell formation problems: (1) the problem of allocating machines to part families, (2) the machine-part grouping problem, and (3) the problem of allocating new parts to existing groups. The problem of allocating machines to part families is formulated as 0-1 integer programming models (Model 1 and Model 2). Two approaches to the machine-part grouping problem are presented. The first approach involves the machine group formation problem (Model 3) first and then considers the problem of allocating parts to machine groups (Model 4). Each of these problems is formulated as a 0-1 linear integer programming problem. The second approach is concerned with simultaneous grouping of parts and machines. This problem is formulated as 0-1 non-linear integer programming models (Model 5 and Model 6). These models consider machine and material handling costs.
restrictions on the number and size of the groups and the number of machines available in each type. An improved formulation of the single function classification model (Model 7) is suggested to develop partallocation schemes to allocate new parts to the existing part families.

Lagrangean relaxation based methods are used to solve the models presented in this dissertation. Alternate relaxations have been compared in terms of the computational time and the quality of bounds obtained. In Models 1 and 2, relaxing the machine group size constraint (Relaxation 2) is found to perform better than relaxing the machine availability constraint. Both the relaxations are relatively insensitive to the initial estimate of the objective function, if underestimated.

In Model 4, relaxing the part group size constraint (Relaxation 2) is found to perform better than relaxing the part allocation constraint and both the relaxations are relatively insensitive to the initial estimate, if underestimated. In all the models (Models 1, 2, 3, and 4), the problem structure does not have any significant effect on
the computational time and the bounds obtained.

An approximate iterative method to solve the simultaneous machine-part grouping models (Models 5 and 6) has been developed and tested using randomly generated problems. The procedure appears to converge almost to the same final solution irrespective of the initial parts allocation chosen. The usefulness of the models to solve large size problems is illustrated using randomly generated problems. An extension of Model 5 to form machine-part groups in the presence of alternate process plans for parts is also presented.
DEDICATION

To my parents

- ix -
ACKNOWLEDGEMENTS

I would like to take this opportunity to express my deep gratitude and thanks to Dr. R.S. Lashkari for his guidance and support that made this dissertation possible. I would also like to express my special thanks to Dr. R.J. Caron, Dr. N. Singh and Dr. S.P. Dutta for serving as committee members. Thanks are extended to Dr. A. Kusiak of the University of Iowa for serving as the external examiner to the dissertation committee. My thanks are due to Jacque and Tom for helping me out in my day to day works.

I wish to acknowledge the financial support provided by NSERC, University of Windsor, and The Ministry of Colleges and Universities (Ontario) during my Ph.D. program.

Finally, my deepest and heartfelt thanks go to my parents whose blessings made my efforts fruitful and to my beloved wife Dhanalakshmi for being an understanding and loving wife which helped me to have an excellent concentration in my research.

-x-
TABLE OF CONTENTS

ABSTRACT vi

DEDICATION ix

ACKNOWLEDGEMENT x

LIST OF TABLES xiv

LIST OF FIGURES xv

NOMENCLATURE xvi

CHAPTER

I INTRODUCTION 1

1.1 Flexible Manufacturing Systems 1

1.2 Problems in Flexible Manufacturing Systems 4

1.2.1 Design Problems 4

1.2.2 Planning Problems 5

1.2.3 Operational Problems 6

1.3 Cell Formation Problems 9

1.3.1 Machine-part Grouping Problem 12

1.3.2 Part Family Formation Problem 15

1.3.3 Machine Grouping Problem 15

1.3.4 Machine Allocation Problem 16

1.3.5 Parts Allocation Problem 17

1.4 Organization of the Dissertation 18

II LITERATURE REVIEW ON CELL FORMATION PROBLEMS 20

2.1 Classification and Coding Based Methods of Part Family Formation 21

2.2 Routing Based Methods of Part Family-Machine Group Formation 23

2.2.1 Evaluative Methods 24

2.2.2 Similarity Coefficient Methods 26
### 2.2.3 Heuristic Methods
2.3 Matrix Based Methods of Machine-part Grouping
   2.3.1 Matrix Manipulation Algorithms
      2.3.1.1 Direct Manipulation Algorithms
      2.3.1.2 Indirect Manipulation Algorithms
   2.3.2 Network Formulation Methods
2.4 Processing Similarity Based Methods of Part Family/Machine Group Formation

### III PROPOSED RESEARCH
3.1 Problem Situation
3.2 Motivation for the Proposed Research
3.3 Objectives of the Proposed Research
3.4 Modeling Approach - A Conceptual Framework
   3.4.1 Mathematical Modelling
   3.4.2 Solution Methodology

### IV DEVELOPMENT OF SIMILARITY/COMPATIBILITY INDICES
4.1 Similarity Index
4.2 Compatibility Indices

### V MACHINE ALLOCATION PROBLEM
5.1 Model 1 - Compatibility Maximization
5.2 Model 2 - Cost Trade-off
5.3 Application of the Formulations
5.4 Solution Procedure for Large Problems
5.5 Computational Experience

### VI MACHINE-PART GROUPING PROBLEM
6.1 Sequential Approach
   6.1.1 Model 3 - Machine Group Formation
   6.1.2 Model 4 - Parts Allocation
   6.1.3 Application of the Formulations
   6.1.4 Solution Procedure for Large Scale Formulations of Model 3
   6.1.5 Computational Experience of Model 3
   6.1.6 Solution Procedure for Large Scale Formulations of Model 4
   6.1.7 Computational Experience of Model 4
6.2 Simultaneous Approach
6.2.1 Model 5 - Compatibility Maximization 124
6.2.2 Model 6 - Cost Trade-off 125
6.2.3 Application of the Formulations 126
6.2.4 Solution Procedure for Large Size Problems 130
6.2.5 Computational Experience 132

VII NEW PARTS ALLOCATION PROBLEM 138
  7.1 Approach to the Problem 138
  7.2 Model 7 - Part Family Identification 140
  7.3 Application of the Formulation 142
  7.4 Some Observations 144

VIII MACHINE-PART GROUPING MODEL - AN EXTENSION 147
  8.1 Formulation of the Model (Model 8) 148
  8.2 Solution Procedure 150
  8.3 Application of the Formulation 151

IX CONCLUSION 156
  9.1 Contributions of the Research 157
  9.2 Directions for Future Research 159

REFERENCES 160

APPENDICES 168

A.1 The Lagrangean Relaxation Method 169
A.2 Linearization Scheme 178
A.3 Program Listings 179

VITA AUCTORIS 204
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table No.</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>An Example of the Number of Common Tools for Different Part/Machine Combinations</td>
</tr>
<tr>
<td>4.2</td>
<td>Example of Compatibility Indices</td>
</tr>
<tr>
<td>5.1</td>
<td>Tooling Requirements of the Parts (for Models 1 and 2)</td>
</tr>
<tr>
<td>5.2</td>
<td>Tools Available on the Machines (for Models 1 and 2)</td>
</tr>
<tr>
<td>5.3</td>
<td>Part Families Composition (for Models 1 and 2)</td>
</tr>
<tr>
<td>5.4</td>
<td>Results of the Example Problem (for Models 1 and 2)</td>
</tr>
<tr>
<td>5.5</td>
<td>Effect of Problem Size</td>
</tr>
<tr>
<td>5.6</td>
<td>Effect of Initial Estimate ($Z_1^*$)</td>
</tr>
<tr>
<td>5.7</td>
<td>Experimental Design for Machine Allocation</td>
</tr>
<tr>
<td>5.8</td>
<td>Effect of Problem Structure</td>
</tr>
<tr>
<td>5.9</td>
<td>Comparison with Optimal Solution</td>
</tr>
<tr>
<td>6.1</td>
<td>Tooling Requirements of the Parts (Models 3 and 4)</td>
</tr>
<tr>
<td>6.2</td>
<td>Tools Available on the Machines (Models 3 and 4)</td>
</tr>
<tr>
<td>6.3</td>
<td>Effect of Problem Size</td>
</tr>
<tr>
<td>6.4</td>
<td>Effect of Initial Estimate ($Z_3^*$)</td>
</tr>
<tr>
<td>6.5</td>
<td>Experimental Design for Machine Grouping</td>
</tr>
<tr>
<td>6.6</td>
<td>Effect of Problem Structure</td>
</tr>
<tr>
<td>6.7</td>
<td>Effect of Problem Size</td>
</tr>
<tr>
<td>6.8</td>
<td>Effect of Initial Estimate ($Z_4^*$)</td>
</tr>
<tr>
<td>6.9</td>
<td>Experimental Design for Parts Allocation</td>
</tr>
<tr>
<td>6.10</td>
<td>Effect of Problem Structure</td>
</tr>
<tr>
<td>6.11</td>
<td>Tooling and Production Requirements of the Parts</td>
</tr>
<tr>
<td>6.12</td>
<td>Details of the Machines Available</td>
</tr>
<tr>
<td>6.13</td>
<td>Results of the Example Problem</td>
</tr>
<tr>
<td>6.14</td>
<td>Schemes for Initial Parts Allocation</td>
</tr>
<tr>
<td>6.15</td>
<td>Effect on the Final Solution</td>
</tr>
<tr>
<td>6.16</td>
<td>Performance on Large Problems</td>
</tr>
<tr>
<td>8.1</td>
<td>Process Plans and Tooling Requirements of the Parts</td>
</tr>
<tr>
<td>8.2</td>
<td>Results of the Example Problem (Model 8)</td>
</tr>
</tbody>
</table>


**LIST OF FIGURES**

<table>
<thead>
<tr>
<th>Figure No.</th>
<th>Description</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Types of Layouts in Manufacturing Systems</td>
<td>10</td>
</tr>
<tr>
<td>1.2</td>
<td>Initial Machine-part Matrix</td>
<td>13</td>
</tr>
<tr>
<td>1.3</td>
<td>Final Machine-part Matrix</td>
<td>14</td>
</tr>
<tr>
<td>3.1</td>
<td>An Overview of the Proposed Research</td>
<td>54</td>
</tr>
</tbody>
</table>
NOMENCLATURE

The following indices are used throughout the dissertation:

- \( i \) - index of part, \( i=1,I \)
- \( j \) - index of machine, \( j=1,J \)
- \( k \) - index of part family/machine group, \( k=1,K \)
- \( p \) - index of process plan for part \( i \), \( p=1,P_i \)

Other notation used is as follows:

- \( a_{i,k} \) = 1 if part \( i \) is in group \( k \)
  = 0 otherwise

- \( A_j \) - number of type \( j \) machines available

- \( b_{j,k} \) = 1 if machine \( j \) is in group \( k \)
  = 0 otherwise

- \( c \) - average cost of an intercell movement

- \( C_{i,j} \) - compatibility index between part \( i \) and machine \( j \)

- \( C_{i,p,j} \) - compatibility index between plan \( p \) of part \( i \) and machine \( j \)

- \( E_{j,l} \) - set of parts requiring both machine \( j \) and machine \( l \)

- \( f_j \) - annual fixed cost rate of machine \( j \)

- \( \xi_{h,k} \) - auxiliary variable in Model 7

- \( G_k \) - limit on the number of machines in group \( k \)

- \( H_k \) - limit on the number of parts assigned to machine group \( k \)
\( I_k \) - set of parts in family \( k \)

\( L \) - size of unit handling load for part \( i \)

\( m_k \) - the minimum range of any family \( k \) (upper limit - lower limit)

\( M_j \) - number of tools available on machine \( j \)

\( N_i \) - number of tools required for part \( i \)

\( N_{ip} \) - number of tools required for plan \( p \) of part \( i \)

\( O_{ij} \) - set of operations of part \( i \) that need machine \( j \)

\( q_i \) - annual production requirements of part \( i \)

\( Q_i \) - set of operations required to be performed on part \( i \)

\( r \) - minimum gap between the lower limit of a part family and the upper limit of the previous family in the number line

\( S_{jl} \) - similarity index between machines \( j \) and \( l \)

\( t_{il} \) - processing time for operation \( l \) of part \( i \)

\( T_{ij} \) - number of common tools between part \( i \) and machine \( j \)

\( T_{ipj} \) - number of common tools between plan \( p \) of part \( i \) and machine \( j \)

\( U = (u_1, u_2, \ldots, u_J) \) - multipliers for Relaxation 1 in Model 1

\( V = (v_1, v_2, \ldots, v_J) \) - multipliers for Relaxation 2 in Model 1
\( \bar{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_j) \) - multipliers for Relaxation in Model 3

\( \bar{\beta} = (\beta_1, \beta_2, \ldots, \beta_1) \) - multipliers for Relaxation 1 in Model 4

\( \bar{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_1) \) - multipliers for Relaxation 2 in Model 4

The decision variables are defined below:

\( B_k \) - upper limit of family \( k \)

\( D_k \) - lower limit of family \( k \)

\( W_{jl} = 1 \) if machines \( j \) and \( l \) are grouped together
\( = 0 \) otherwise

\( X_{ik} = 1 \) if part \( i \) is assigned to machine group \( k \)
\( = 0 \) otherwise

\( X_{ipk} = 1 \) if part \( i \), produced under plan \( p \), is in group \( k \)
\( = 0 \) otherwise

\( Y_j \) - coefficients of machine \( j \) in the parts allocation (weighting) scheme

\( Y_{jk} = 1 \) if machine \( j \) is allocated to group \( k \)
\( = 0 \) otherwise
CHAPTER I

INTRODUCTION

Many production enterprises are presently being affected by structural changes which are caused by factors both internal and external to the organization [7]. The market demands an increasing variety of products and of product variants. The life of many products have a tendency to decrease. More frequent product changes and smaller batch quantities, in conjunction with the desire for shorter order throughput times, require a better organization of production. In recent years, the concept of flexible manufacturing systems has emerged as a viable answer to the problems of low volume, high variety production.

1.1. Flexible Manufacturing Systems (FMS)

An FMS is a manufacturing system consisting of a set of numerically controlled machine tools which perform the operations required to manufacture parts. The machine
tools have automatic tool interchange capabilities and are connected by automatic material handling equipment. The FMS is a result of the evolution of the use of several NC machine tools working independently, into an integrated system of NC machine tools controlled by a computer. Although this evolution occurred within the metal working industry, the same concepts have been employed in other forms of manufacturing.

A typical FMS consists of versatile metal removing machine tools, each of which may have four or five axes of motion. As a result, complex precision operations, which would be difficult to machine manually, can be performed by these machines. Some typical machine tools are lathes, drills and machining centers. The latter are extremely versatile and capable of performing varied operations such as milling, drilling, boring, and tapping.

Several different part types can be processed on an FMS at the same time. It is assumed that there are production orders for a certain number of parts of each part type. The manufacture of a part involves a set of operations to be performed on the part, where each operation may require one
or more cutting tools. Depending on the capability of a machine to load multiple tools, one or more operations of a part may be performed on that machine at a time. The cutting tools required for all the operations performed by a particular machine are stored in its tool magazine. The rapid interchange capability to change tools between the tool magazine and the machine spindle allows a machine to perform several consecutive operations on a complex part with virtually no setup time between the operations.

The parts are transported from machine to machine by a computer controlled material handling system. A pallet and fixture provide the connection between a part and the transport mechanism. Parts requiring refixturing between operations are taken to the refixturing area and placed onto a pallet that contains the necessary fixture. There are other auxiliary equipments in the system such as inspection stations and load/unload stations.

The following are some of the benefits that may be obtained with the installation of an FMS:

* High capital equipment utilization
* Reduced direct labour costs

- 3 -
* Reduced work-in-process inventory and lead time
* Responsiveness to changing production requirements
* Ability to maintain production
* High product quality
* Operational flexibility
* Capacity flexibility

1.2 Problems in Flexible Manufacturing Systems

It is well recognized, both by researchers and practitioners, that the economic benefits of flexible manufacturing systems depend not only on the physical sophistication but also, and equally on the effectiveness of the design, planning and operation of the systems. Of late, this has been the focus of attention of several researchers in the field of Industrial Engineering and applied Operations Research.

1.2.1 Design Problems

The design of FMS is very crucial since it has a great impact on the planning and operational problems that follow. In order to design an efficient FMS, the following problems should be solved [41].

- 4 -
(1) Organizational problem, that is selection of the part families to be manufactured because current FMS technology restricts the shape of parts designed for flexible manufacturing.

(2) Selection of an FMS production system.

(3) Selection of a material-handling system.

(4) Selection of fixtures and pallets.

(5) Selection of an appropriate computer system.

(6) Layout and integration of all the above systems.

Each of these problems is very complex and hence more research is needed to overcome the deficiencies in the design of FMS.

1.2.2 Planning Problems

FMS planning problems [66] are those decisions that have to be made before the FMS can begin to produce parts. Following are the important planning problems:

(1) In some classes of FMSs, grouping of parts and of some of the production system components is desirable to economize on tooling requirements, improve machine utilization and material handling and reduce setup times.
This is known as the grouping problem. Grouping leads to reduced work-in-process inventories and manufacturing lead times. Grouping of parts and other production system components can be performed in an analogous way to the grouping of parts and machines [42].

(2) From the list of part types for which production orders of various sizes are specified, choose a subset of part types for immediate and simultaneous manufacture. This is known as part type selection problem.

(3) Allocate operations and the associated cutting tools for the selected part types among the machines. This is known as operation allocation problem.

1.2.3 Operational Problems

Because FMSs have a high capital cost, a high rate of utilization is necessary to ensure a quick return on investment. The utilization of the FMS, in turn, is enhanced to a great extent if the following operational problems are effectively solved:
(a) Machine loading (capacity balancing) problem

A short-horizon (one month) planning problem for any of the FMSs that result from the grouping of parts and production system components might still be difficult to solve because of its complexity. Such a problem may be solved by applying the two-stage procedure: (1) machine loading, and (2) scheduling, which decomposes the short-horizon planning problem into subproblems which may be solved more easily.

The machine-loading problem in FMSs has been formulated and solved by a relatively large number of authors [38, 39, 62, 64 and 65]. Formulations of the FMS loading problem are dependent on the machine and tool magazine design. The non-linearity in the formulations is due to the design of the tools and tool magazines, since the actual number of slots used depends on the physical placement of the tools in the tool magazine. Currently, many different types of tool magazine are available and, in most of them, the sequence of tools does not influence the capacity of the magazine. This allows much simpler formulation of the FMS loading problem.
(b) Scheduling problem

Considerable effort has been spent in traditional manufacturing systems on solving the part-scheduling problem. In an FMS, however, because of the high cost of components, the part scheduling problem is only one of the scheduling issues. The scheduling problem in FMSs is a multi-dimensional problem involving:

(i) scheduling of parts,

(ii) scheduling of pallets and fixtures,

(iii) scheduling of tools and

(iv) scheduling of the material-handling system (AGVS).

Such a multi-dimensional problem is extremely difficult to solve; it is, therefore, more convenient to consider each of the four subproblems separately, even though each of them can be very difficult to solve in practice.

(c) Control problems

These problems are associated with the continuous monitoring of the system. The following are a few to mention:

(1) determination of policies to handle breakdowns

(2) determination of scheduled, periodic and preventive maintenance policies
(3) determination of in-process and finished goods inspection policies

(4) procedures for tool life and process monitoring

1.3 Cell Formation Problems

Cell formation is an application of the Group Technology (GT) philosophy where the manufacturing system, in total or in part, has been converted into cells. The GT philosophy, very simply put, is to capitalize on similarities in recurring activities. GT can be applied to manufacturing systems in three different ways: The simplest application, common in batch manufacturing environments, is to informally rely on part similarities to gain setup efficiencies when sequencing jobs at a work center. The second application is to create formal part families, dedicate equipment to these families, but let the equipment remain in its original position. The ultimate GT application in manufacturing is to form manufacturing cells.

Figure 1.1 shows the difference among the arrangement of machines in job shop, flow shop and cellular type of systems [50]. The cells are designed to allow for
Fig. 11 Types of Layouts in Manufacturing Systems
efficient flow of parts through the manufacturing system.

The aims of cell formation are to reduce setup times (by using part family tooling and family sequencing), to reduce flow times (by reducing setup times, move times, wait times for moves, and the use of small transfer batches) and, therefore, to reduce inventories and market response times.

The important problems encountered in cell formation are (1) Machine-part grouping problem, (2) Part family formation problem, (3) Machine group formation problem, (4) Machine allocation problem, and (5) Parts allocation problem. Considerable research has gone into this area and several procedures have been developed over the years (see Chapter II). Of particular interest is the development of procedures that can generate solutions with consideration to stated design goals. Desirable design goals could, for example, be the minimization of load imbalances, minimization of costs, minimization of intercell and intra-cell movements, etc.
1.3.1 Machine-part Grouping Problem

This is concerned with the problem of producing parts with similar processing requirements in machine groups. Each machine group consists of dissimilar types of machines (preferably located close to one another) which possess specific manufacturing capabilities to produce one or more part families. This provides an opportunity to reduce setup times, thus allowing manufacturers to reduce lot sizes, trim work-in process inventories, and shorten manufacturing lead times. The problem of machine-part grouping is illustrated using the following example.

Consider a manufacturing system consisting of four machines and producing five parts. The routing for the set of parts is given in the form of a machine-part matrix as shown in Fig. 1.2. In Fig. 1.2 the entries along each column represent the set of machines required to perform the corresponding operation on the parts. Rearranging the rows and columns of the matrix in Fig. 1.2 results in a modified machine-part matrix as shown in Fig.1.3. Two machine groups and part families are visible in this matrix.
<table>
<thead>
<tr>
<th>Part number</th>
<th>Machine number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>1 1 1 1 1</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>1 1 1 1 1</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>1 1 1 1 1</td>
<td>1 2 3 4 5</td>
</tr>
</tbody>
</table>

Fig. 1.2 Initial Machine-Part Matrix
<table>
<thead>
<tr>
<th>Part Family 1</th>
<th>Part Family 2</th>
<th>Machine Number</th>
<th>Machine Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 1.3 Final Machine-Part Matrix
1.3.2 Part Family Formation Problem

A part family can be described as a collection of similar parts. Generally, the parts in one family would have similar geometrical shapes, and/or require similar machining requirements. Usually, part families are formed in one of two ways, (a) the part family consists of parts which are similar in shape within a certain dimensional range, and have most or perhaps all machining requirements in common (e.g. [29] and [30]), or (b) the part family consists of parts of dissimilar geometry, but which have some operations in common (e.g. [13]). The design of part families is usually the first step in cell formation. This alone does not help to achieve the desired objectives of cell formation, unless machines are grouped to manufacture one or more part families.

1.3.3 Machine Group Formation Problem

This is concerned with the problem of grouping machines into cells. Each cell consists of dissimilar types of machines to efficiently produce a family of parts requiring almost similar machining operations. Designing machine groups
essentially means recognizing and using the relationship between machines. This relationship is defined in terms of the parts that have to be processed on these machines. Machine routings and production requirements of the parts are usually the input information needed to form machine groups. Once machines are grouped into cells parts are allocated to the cells and the cells are evaluated on factors such as machine utilization.

1.3.4 Machine Allocation Problem

This is an important problem encountered in the planning stage of a cellular manufacturing system as its implications affect any efforts to economize on tooling requirements and improve machine utilization and materials handling. In many firms, the design of part families is done based on geometric features. The geometric feature based grouping has been mainly a part of design standardization effort for the various shapes of the parts. The concept has recently been considered in the computer aided process planning area where an attempt to relate the processing steps to geometric features is made to develop computerized systems for generating process
plans [11,67]. Hence, the problem is to allocate machines to part families in order to meet some objectives, for instance, minimizing material movement.

1.3.5 Parts Allocation Problem

This problem could arise in two ways: (i) machines have been grouped into cells based on their capabilities to process the parts; and the problem is to allocate parts to the machine groups; (ii) as market needs change, new parts are introduced into the system which have to be manufactured [73]. The allocation of new parts to appropriate machine groups must be done without disrupting the existing configuration. Also, it is helpful to know beforehand if the new part(s) could be manufactured within the existing machine cells design. This leads to other decisions such as redesigning or subcontracting of the part, or expanding the production facility.

However, the problems of cell formation are not as simple as they are explained above. It is not always possible to identify disjoint cells. It can happen that a part may have to visit more than one cell before completing its
processing. This happens when the machines allocated to a family of parts are not capable of processing the parts completely, leading to intercell movements. To reduce the cost of intercell movements, the obvious method is to duplicate the appropriate machines. This may not be possible due to the restriction on the availability of machines and the high cost of the machines. Another problem is the restriction on the number of cells and the number of parts and machines in a cell. These restrictions are often imposed due to supervisory and control purposes. Also, the matrix representation of the processing requirements assume that each operation of a part is restricted to one machine, which is not true in practice.

1.4 Organization of the Dissertation

The dissertation is organized and presented in nine chapters as follows. In the first chapter, an introduction to flexible manufacturing systems and cell formation problems is given. A complete review of the existing literature on cell formation is presented in Chapter II. Motivation for the proposed research, objectives of the proposed research and a conceptual
framework of the methodology are given in Chapter III. A complete description of the development of the similarity and the compatibility indices are presented in Chapter IV. The mathematical programming formulations of the machine allocation problem and the computational results are presented in Chapter V. The mathematical programming formulations of the machine-part grouping problem using the sequential and simultaneous approaches and the computational results are presented in Chapter VI. Approaches to solve the new parts allocation problem are presented in Chapter VII. An extension of the machine-part grouping model to account for the presence of alternate process plans for parts is given in Chapter VIII. The contributions of the dissertation and the directions for future research are given in the last Chapter.
CHAPTER II

LITERATURE REVIEW

The cell formation problems in flexible manufacturing systems have been discussed quite extensively in the literature. A number of procedures both descriptive as well as analytical are reported in the literature. Chu and Pan [14] provided a state-of-the-art review on the use of clustering techniques in cell formation. The problems associated with these techniques and the research directions are presented in this paper. Wemmerlov and Hyer [72] proposed a framework for structuring the cell formation problems. Four basic solution approaches are identified, which are based on the fundamental ways part families and machine groups are matched within the cell formation process. The framework is then used to classify descriptive and analytic procedures available for the cell formation problems.

- 20 -
This Chapter provides a comprehensive review of the various approaches that have been adopted in an attempt to solve these problems. Efforts have been made to classify and review the literature based on the input information used to solve the cell formation problems. This is based on the type of input used to model and solve the problems. On this basis, the available literature can be classified as follows:

(i) Classification and coding based methods
(ii) Routing based methods
(iii) Machine-component matrix based methods
(iv) Processing similarity based methods

A detailed review of the literature including the type of modelling and solution approaches used is given in the following sections.

2.1 Classification and Coding Based Methods of Part Family Formation

In the classification and coding based methods each part has a certain code (e.g. opitz code [52]) depending on its geometric and technological attributes and part families are formed so that parts with similar attributes are
grouped into a family. This is done manually or using analytical methods which define a distance between two parts based on their codes.

Kusiak [40] proposed the application of hierarchical clustering algorithm to form part families. A distance matrix is calculated from the parts code using suitable distance metrics. The problem is also modelled as a p-median problem and solved using subgradient algorithm [43]. Gongaware and Ham [29] and Han and Ham [30] reported multi-objective clustering techniques to form part families based on similarity vectors as well as part codes. Hyer and Wemmerlov [31] discussed the structures, applications and implementation of the GT oriented coding systems.

Dutta et al. [19] presented a methodology using a coding system derived from the Opitz code to form design and tooling families. Each part is represented by an N-vector (N = total number of features) and each element takes on a value of 1 or 0 depending upon whether the feature is present or not. Similarity coefficients
are calculated as a function of the similarity values between all possible pairs of features present in both parts.

2.2 Routing Based Methods of Part Family/Machine Group Formation

The input information in these methods is usually in the form of machine routings for each part. The analysis of the routing information is done manually (evaluative methods), analytically (similarity coefficient methods) or heuristically. The evaluative method requires a series of evaluations to be made by designer, more or less calling upon his ability to recognize a pattern. The similarity coefficient approach is drawn directly from the field of numerical taxonomy. The basis of this method is to define a similarity measure between machines (parts) and then to group the machines (parts) into families based on their similarity measurements. Some of the first few methods reported in the literature fall into this category. These methods can be further classified as follows:
(i) Evaluative methods (ii) Similarity coefficient methods
(iii) Heuristic methods

2.2.1 Evaluative Methods

The concept of production flow analysis (PFA) was first introduced by Burbidge [4]. The main feature of PFA is that it involves the systematic listing of the components in various ways, in the expectation that groups of machines and components may be found by careful inspection. Component flow analysis (CFA) was proposed by El-Essawy and Torrance [21]. It is almost similar to PFA. Burbidge [5] reported a method called Nuclear synthesis which is based on selecting machines used by a few components as starting points for various cells, or nuclei. The next machine is allocated on the basis that it has the smallest number of components left unassigned to a group. Once the Nuclear synthesis is completed, these nuclei are modified.

Purcheck has adopted a set-theoretic approach throughout. In his earliest paper [53], he suggested the use of union operations to build up supersets of machines and components which can be represented as a path along the
edges of a lattice diagram. The lattice diagram grows exponentially as the set is enlarged and hence its usefulness is limited to being an illustrative device. The superset approach is, by its nature, an explicit evaluative method and suffers from all the difficulties of such methods.

De Beer et al. [16] suggested a modified form of PFA. An important aspect of this approach is the development of a method of cell formation based on an analysis of operation routings and the divisibility of operations between machines and hence between cells, this divisibility being governed by the number of machines of the required types that are available for specific operations. Burbidge [6] described how PFA could be carried out manually. An attempt has been made by De Beer and De Witte [17] to extend the basic approach of PFA to consider explicitly the question of machine duplication and different characteristics of the machines. This method has been termed production flow synthesis.
2.2.2 Similarity Coefficient Methods

This approach was first suggested by McAuley [48] to form machine groups. The similarity measurement used is the coefficient of jaccard which is defined as the number of components which visit both machines, divided by the number of components which visit at least one of the machines. In grouping machines, McAuley [48] used single linkage cluster analysis. The main disadvantage of this method is that while two clusters may be linked by this technique on the basis of a single link, many of the members of the two clusters may be quite far away from each other in terms of similarity.

Carrie [7] used numerical taxonomy for finding part families. Similarity co-efficients for every pair of parts are calculated. Single linkage cluster analysis is used and machine loads are evaluated to check if no proposed grouping violates any restrictions on machine availability.

Rajagopalan and Batra [57] developed a graph theoretic method which uses cliques of the machine-graph as a means
for grouping machines. The vertices of this graph are machines and the arcs are Jaccard similarity coefficients. The main disadvantage of this approach is that due to the high density of the graph, a large number of cliques is involved and many of the cliques are not vertex disjoint.

De Witte [18] suggested the use of three similarity coefficients based on routing and machining times to indicate the interdependence of machine types. Based on similarity, the method first allocates machine types to cells and then components are allocated to cells. The number of machines required in each type is determined based on the workload.

Waghodekar and Sahu [70] presented a heuristic, called MACE, based on the similarity coefficient of the product type. This heuristic groups machines based on similarity coefficient and inter-cell moves. It is shown that this method yields less number of exceptional elements than methods using additive type similarity coefficients.
Faber and Carter [23] developed an alternate measure to define the similarity between two machines based on the number of parts processed by both the machines. Based on a specified threshold value and the similarity indices, a machine-machine matrix is formed, which is represented as a graph. A polynomially bounded graph theoretic algorithm which optimally finds the densest subgraph of a graph is proposed to find the machine groups.

Steudel and Ballakur [68] defined a new similarity measure, called cell bond strength which depends upon part routing and production requirements. They suggested a two stage dynamic programming heuristic which first determines the optimum chain so that the sum of the bonds among the machines is maximised and then partitions the chain to form machine groups.

Choobineh [13] proposed a two-stage procedure for the design of a cellular manufacturing system. Clustering techniques, with a new similarity measure based on the manufacturing operations and their sequences, are used to form the part families. Then an integer programming model is proposed.
which will specify the type and the number of machines in each cell and the assignment of part families to cells.

2.2.3 Heuristic Methods

A classification scheme which combines machine requirements and machine sequences by coding them in the form of strings of letters and digits has also been proposed by Purchek [54]. The main drawback is that the code lengths tend to be very long, and in most cases would be difficult to handle effectively. Various mathematical programming formulations have been suggested by Purchek [55]. However, the constraint matrices and the objective functions are rarely adequately described.

Lemoine and Mutel [47] presented a dynamic clustering technique for machine grouping. Their non-hierarchical cluster analysis procedure is based on the minimization of statistical distances with loads and capacities of machines as weights.

Purchek [56], tackles the problem of machine-component grouping in a different fashion. Group formation is done
in terms of minimum differences between masters and maximum combination of masters. A master is defined as a unique or most complex part that has to be processed in one cell. The heuristic proceeds by computing the master sets and their differences. Then the corresponding work load is computed and the combination of master sets is revised, if required based on certain acceptability tests.

Rodriguez and Adaniya [59] suggested an interactive procedure to obtain the number of cells and machines allocated to each cell such that there is a balance between the average setup costs and inventory holding costs. The procedure uses ROC algorithm to generate block diagonal matrix. Then it schedules the products using the concept of economic lotsizing and checks for feasibility, machine utilisation etc.

Askin and Subramanian [1] developed a heuristic procedure to form groups which considers costs of work-in-process and cycle inventory, intra-cell movements and setup, variable processing and fixed machine costs. The heuristic first
reorders parts based on machine routing and then attempts to combine parts to reduce machine requirements. Finally, groups are combined where economic benefits of utilization offset those of set-up, work-in-process and material movement.

Ballakur and Steudel [2] suggested a heuristic procedure considering several practical criteria such as within cell machine utilization, work load fractions etc. A two stage clustering procedure is used to form machine component groups. In the first stage, an ‘admit or reject’ decision is made for each work centre based on the work load fractions. Work load fraction is a function of the work contents of the parts assigned to the cell. The actual number as well as the assignment of the machines to a workcentre are made at the second stage.

Meenakshisundaram and Fu [50] proposed a methodology based on the hospitality and flexibility relationships advocated by Purcheck [55]. From the route sheets of the parts, each part is labeled either as a host or guest. A host difference table is formed for the hosts and an integer
programming model is developed based on the host differences. This model forms as few cells as possible by combining the hosts so that the cost of the combination is minimized.

Wu, et al. [74] developed a Syntactic pattern recognition method to form machine cells by the classification of machining sequences. Each machine is given a unique code and for each component, a string represents the machining sequence. Minimum spanning tree concept is used to form the clusters based on the distances between component strings. The composition of machines to form cells is done using grammatic inference. They also suggested that a new part can be classified as belonging to a particular group if its route string (machining sequence) is recognized by the inferred grammar of the group. This is not realistic, since the machining sequence represents a fixed routing for the part, while the allocation must consider the processing capabilities of the machine cells and the processing requirements of the new part.
Co and Araar [15] presented a three-stage procedure for configuring machines into manufacturing cells, and assigning the cells to process specific sets of parts. First, operations are assigned, with the objective of minimizing the deviation between available capacity and the workload assigned to each machine. This results in a machine-part matrix, which is then manipulated using an extension of King’s algorithm to form machine-part groups. Then a direct search algorithm is used to determine the number of cells, and the composition of each cell.

2.3 Matrix Based Methods of Machine-Part Grouping

In the machine-part matrix each element \( a_{ij} \) of the matrix takes on a value of 1 or 0 depending upon whether machine \( i \) is needed to process part \( j \) or not. Machine-part groups are formed by the direct or indirect manipulation of the matrix using some algorithms or by some other methods which use the information contained in the matrix in a different form.

A great majority of the literature available on cell formation problems falls into this group. These methods
can be further classified into the following groups:

(i) Matrix manipulation algorithms (ii) Network formulation methods.

2.3.1 Matrix Manipulation Algorithms

One group of algorithms (Direct manipulation algorithms) successively rearranges the rows and columns of the machine-part matrix (A) to form the machine-part groups. The other group of algorithms (Indirect manipulation algorithms) work on the matrix (A*A^T) to reorder the rows and columns of the matrix.

2.3.1.1 Direct Manipulation Algorithms

King [33,34] suggested an algorithm, known as rank order clustering algorithm (ROC) which rearranges the columns and rows of the matrix A in the descending order using binary ranking. King [33] suggested a relaxation procedure which determines the number of machines to be duplicated to eliminate bottleneck as well as their disposition in the final matrix. This procedure, however, greatly increases the dimension of the matrix since it begins by assuming a relaxation of one machine to one part. King and Nakornchai [35] suggested ROC2 which sorts
several rows and columns at the same time instead of element by element as done by ROC. ROC2 is computationally more efficient. They also suggested a method which handles bottleneck machines by ignoring them. This method is interactive and the diagonal pattern and number of groups change according to the number of machines duplicated and the number of duplications of each machine. However, it does not suggest a method to choose the right number of duplications based on some quantifiable measures. Chan and Milner [8] suggested a method known as direct clustering algorithm which progressively restructures the matrix A until there is no more improvement due to restructuring.

Chandrasekaran and Rajagopalan [10] proposed a method known as MODROC, which is a modification to the ROC algorithm. MODROC removes the deficiencies of the ROC algorithm to a great extent and enables the identification of bottleneck machines objectively. This method uses ROC algorithm in conjunction with a block and slice method and a hierarchical clustering method.
Seifoddini and Wolfe [61] suggested a method in which machine-part grouping is formed based on machine-part matrix using average linkage cluster analysis and then machine duplication starts with the machine having the largest number of intercell moves and continues until no machine generates more inter cell moves than a specified threshold value. It does not consider the cost of machine duplication. Also, the computation of intercell moves does not consider production requirements of parts.

Khator and Irani [32] defined occupancy values for each part which depends upon the number of parts and machines and routing. They used a heuristic procedure, which progressively develops block diagonalization on the machine-part matrix using the occupancy values to form the groups.

Kusiak and Chow [45] developed two efficient algorithms to solve the machine-part grouping problem. A cluster identification algorithm suggested to form the groups from matrix (A) is reported to be the most efficient algorithm.
developed today. A cost analysis algorithm is developed to solve the augmented formulation of the problem, which associates cost with part and limits the number of machines in each cell.

Kusiak [44] proposed a generalised GT concept, based on the presence of different process plans for each part. Based on this, the machine-part matrix is modified to include the process plans for each part. This improves the diagonal structure of the final matrix and hence reduces the number of bottleneck machines. The problem is also modelled as a 0-1 integer programming model.

2.3.1.2 Indirect Manipulation Algorithms

These algorithms operate on the matrix $B (= A^T A)$ to reorder rows and on the matrix $B^T (= A^T A)$ to reorder the columns. McCormick, et al. [49] developed an algorithm known as bond energy algorithm. These methods seek to determine a permutation of the rows and columns in which a performance measure, in this case, the sum of the products of adjacent elements is maximised.
Slagle et al. [63] suggested a clustering and data-reorganizing algorithm known as shortest spanning path algorithm based on the concept of the shortest spanning path of a graph. The method is very much similar to that of McCormick et al. [49] excepting the performance measure used in the reorganization. Bhat and Haupt [3] developed an efficient clustering algorithm known as matching algorithm on the concept of matching between two rows (columns). This method is computationally efficient, as the change in the number of matchings need not be recomputed for each possible arrangement, whenever a row is rearranged.

All the above methods first place a row arbitrarily and then select a row arbitrarily from the remaining rows. This is placed in all possible positions with respect to the already placed row and the performance measure is computed for each position. The selected row is placed in the position which maximizes the performance measure. This is repeated until all the rows are placed. The above procedure is carried out on columns as
well. Kusiak [43] suggested a method, known as rank energy algorithm. This algorithm orders the rows and columns of the matrices $B$ and $B^T$ based on their weights. This is superior to the existing algorithms of this type and generates solutions of acceptable quality.

2.3.2 Network Formulation Methods

These formulations represent the machine-part matrix in the form of a bipartite graph and use network decomposition methods or some other heuristic methods to form machine-part groupings. Kumar et al. [36] and Kusiak et al. [42] presented a network formulation of the grouping problem. The groups are formed by optimally decomposing the network into subgraphs. The decomposition problem is approximated as a quadratic assignment problem which is solved in two phases. In phase I, the problem is approximated by an easily solved linear transportation problem to get the initial solution. This solution is improved iteratively by an interchange heuristic to get the local optimum.
Kumar and Vannelli [37] considered balancing the capacity of each of the cells and subcontracting costs of each part. A heuristic method is suggested to solve the problem and identify the bottleneck machines. Vannelli and Kumar [69] showed that the problem of identifying the minimal number of bottleneck machines is equivalent to finding the minimal cut nodes of a graph. A heuristic based on dynamic programming approach is developed to solve the problem.

Chandrasekaran and Rajagopalan [9] formulated the grouping problem as a bipartite graph and developed an expression for the upper limit to the number of groups. Using this, a non-hierarchical clustering method is developed to form groups. In order to improve the utilization and inter-cell movements, an ideal seed method, which reorders rows and columns is used to improve the groups. Ideal seeds are centroids of the imaginary perfect groups with essentially the same block diagonal structure as the groups formed initially. A criterion called, grouping efficiency, which is
defined as the weighted average of utilization and intercell movement is developed to compare alternate solutions.

2.4 Processing Similarity Based Methods of Part Family/Machine Group Formation

Processing similarity based methods, group parts and machines based on their similarity in processing requirements. There are not many such methods reported in the literature. The few that are reported define processing similarity based on the tooling requirements. These methods form groups using mathematical programming formulations and/or iterative heuristic procedures. Grouping based on processing similarity seems to be realistic since it allows an operation of a part to be performed in any of the machines that has the required tooling.

The methods reported in the literature [12,20,46 and 58], identify processing requirements in terms of individual tooling for different operations. So the objective is to have similar tooling requirements for all the parts in a
particular family. If all the parts require similar toolings, the disturbance in production due to setups will be minimum. A coefficient for the tooling non-homogeneity between two parts is then defined [46] as the ratio of the number of dissimilar toolings to the total number of toolings required.

An iterative heuristic procedure was suggested by Dutta et al. [20] to solve the part families problem based on tooling similarity. A coefficient for the tooling non-homogeneity between two parts is defined as the ratio of the number of dissimilar toolings to the total number of toolings required. The heuristic starts with a random initial configuration and continually seeks a rearrangement (based on the concept of single reallocation) that strictly lowers the overall dissimilarity. The procedure terminates when no more improvement is possible. A non-linear fractional programming model was suggested by Lashkari et al. [46] to form part families. After linearizing the objective function, a parametric search principle is used to solve the linear fractional 0-1
programming problem. However, the solution procedure suggested is not computationally efficient.

Chen et al., [12] formulated the problem of grouping parts and tools as a 0-1 linear integer program. The formulation takes the set of part types and the required tools as inputs, and divides them into clusters of part types and tools through decomposition of the part-tool matrix using a lagrangean relaxation based solution procedure.

Raja Gunasingh and Lashkari [58] proposed a methodology to allocate machines to part families based on the tooling requirements of the parts and the toolings available on the machines. Two 0-1 integer programming formulations are proposed. These formulations take into account the limitations on the number of machines in a group and the number of machines available of a particular type, cost of intercell movement, and the cost of machine duplication.

A review of the reported literature indicates the following:

(1) Most of the methodologies suggested to solve the cell formation problems are based on a fixed routing for each
part which is given in the form of a machine sequence, or a machine-part matrix.

(2) Some of these methodologies are not suitable for large size problems and the suitability of other methodologies to solve large size problems is not known.

(3) There is no published literature available on machine allocation and parts allocation problems.

(4) Methods are not available to solve the machine grouping problem and the machine-part grouping problem based on the processing requirements of the parts.

(5) Reported methods on parts grouping based on processing requirements become intractable when the problem size becomes large.
CHAPTER III

PROPOSED RESEARCH

The proposed research attempts to develop (1) mathematical models of the following cell formation problems: (a) machine allocation problem, (b) machine-part grouping problem, and (c) new parts allocation problem; and (2) methodologies to solve large scale occurrences of these models.

3.1 Problem Situation

The cell formation problem is one of the most important planning problems in flexible manufacturing system, which in turn has a tremendous impact on the scheduling and control policies of the system. Machine utilization, material handling cost and capital investment depend on the physical configuration of the system. Recent investigations of cellular manufacturing systems in the U.S. industry indicate that the operation of such systems is economically justifiable, since the achieved benefits greatly exceed costs [13]. This shows a strong acceptance and belief in

- 45 -
cellular manufacturing and appears to validate it as a highly profitable venture.

For reasons such as design standardization and computerized process planning, some firms use geometric-features-based grouping to design part families [67]. These firms face the problem of allocating machines to part families in order to economize on tooling requirements and to improve machine utilization and material handling (machine allocation problem).

Most of the modern manufacturing plants have NC / CNC machines located randomly within the factory. Although the individual machines may be very efficient, the way in which they are placed in the system may result in low utilization levels. Most of these plants are planning, or are in the process of reorganizing the system, to gain manufacturing flexibility in order to remain in business. In this situation, since the NC / CNC machines are already available, there is an opportunity to reorganize the system into an independent cellular manufacturing system. Once such a strategic decision is taken, it becomes

- 46 -
imperative to analyse the part range under consideration to form part families, and group the available machines to produce one or more families of parts (machine-part grouping problem).

As market needs change, new parts are introduced which have to be manufactured. A recent survey [73] of 32 US firms operating one or more manufacturing cells indicates that the percentage of new parts average 11.1%, with a range of 5 to 25%. The corresponding figures measured in part volume are 8.1%, 2.0%, and 20%, respectively. The allocation of the new parts to appropriate machine groups must be achieved without disrupting the existing configuration, otherwise increased setup time and material handling may result (new parts allocation problem).

3.2 Motivation for the Proposed Research

From the review of the literature, it can be seen that almost all the work done in this area assumes that each operation of a part is restricted to one machine. This is usually expressed in the form of routing/machine-part matrix. This assumption is not realistic, since in
practice an operation of a particular part may be performed on alternate machines, which have the required capability to perform that operation.

A number of formulations and algorithms have been developed based on the above mentioned assumption. These include matrix manipulation methods, manual evaluative methods, integer programming methods, network modelling heuristics, set theoretic and graph theoretic methods. A few of the formulations take into account some of the physical constraints on the system such as limits on cell size and the number of cells in a heuristic manner under the restrictive assumption mentioned above. A different approach to part family formation, which has been suggested recently, relies on the tooling requirements of the parts as the basis of grouping [20]. This allows for the possibility of an operation of a part to be performed on alternate machines (assuming the required tools are available on more than one machine). However, the models developed based on this approach become intractable when the problem becomes large.
Although methods are available to form part families and to group machines, there is no published literature available on allocating machines (parts) to part families (machine groups) once part families (machine groups) are formed. The problem of allocating new parts to existing groups has not been explicitly discussed in the literature; it has only been referred to in a few papers [19 and 74].

Hence, further research efforts are needed in the following directions:

(1) Development of mathematical models of cell formation problems taking into consideration the physical constraints of the system (such as restrictions on the number of machines in a group, the number of parts in a group, and the available number of machines of a particular type), as well as processing requirements, production requirements, and processing times; and (2) Development of methodologies to solve large scale cell formation problems efficiently.

The following cell formation problems are addressed in this research:
(1) Machine allocation problem,
(2) Machine-part grouping problem, and
(3) New parts allocation problem.

The problem of part family formation based on the processing requirements of the parts is not addressed in this research due to the following reasons: (a) The design of part families is only a means; eventually machines have to be allocated to process the part families; (b) The machine-part grouping problem can be solved without forming the part families; (c) It is easier to form the machine groups first and then allocate parts to machine groups, rather than to form part families first and then allocate machines to part families. This is primarily due to the size of the grouping problem, since the number of parts is usually much higher than the number of machines in any manufacturing system; and (d) Part families based on tooling requirements can be identified once the machine-part grouping problem is solved.

3.3 Objectives of the Proposed Research

The major objectives of the dissertation are as follows:
to develop mathematical models of the three above-mentioned cell formation problems based on the tooling requirements of the parts and the tools available on the machines.

(ii) to develop indices to define the "similarity" between two machines, or the "compatibility" between a machine and a part, which are used in the formulation of the machine allocation and machine-part grouping problem.

(iii) to develop efficient solution procedures for large instances of the above mentioned cell formation problems.

3.4 Modeling Approach - A Conceptual Framework

In this research, a methodology is proposed to model the cell formation problem on the basis of the capabilities of the machines to process the parts in the system. The processing requirements of the parts are related to their tooling requirements, and the machine capabilities are expressed in terms of the tool availabilities on each machine. Indices are developed to define the compatibility of a part with a machine and to define
the similarity between machines based on the tooling requirements of the parts, toolings available on the machine, and processing times. Subsequently, mathematical programming formulations of the cell formation problems are developed and solved using Lagrangean relaxation methods.

3.4.1 Mathematical Modelling

The mathematical modelling of the machine-part grouping problem may be approached from three viewpoints [72]: (a) The first approach considers the machine group formation first, to be followed by the allocation of parts to machine groups; (b) The second approach considers the problem of simultaneous grouping of parts and machines; and (c) The third approach considers the part family formation first, to be followed by the allocation of machines to part families [13]. Here the machine-part grouping problem is approached from the first two viewpoints. The third approach is not investigated due to the reasons outlined in section 3.2.

The machine group formation and parts allocation problems (approach 1) are formulated as 0-1 integer programming
models. The simultaneous grouping problem (approach 2) is formulated as non-linear 0-1 integer programming models. The machine allocation problem is formulated as 0-1 integer programming models. The problem of allocating new parts can be solved using parts allocation model (from approach 1 for machine-part grouping problem) or the part family identification model, which is an improved formulation of the single function classification model [27]. The detailed formulations are given in the subsequent Chapters. An overview of the problems addressed in this research and the mathematical models developed is given in Fig. 3.1.

3.4.2 Solution Methodology

All the formulations presented here are in the form of linear/non-linear 0-1 integer programming models. Non-linear 0-1 integer programs are converted to linear 0-1 integer programming models using the linearization scheme suggested by Glover and Woolsey [28]. The machine allocation and the machine-part grouping models presented here can be used effectively to solve the corresponding cell formation problems in small to medium sized manufacturing
Fig. 3.1 An Overview of the Proposed Research
systems using commercially available integer programming (IP) codes. The part family identification model can be solved using commercially available mixed-integer programming (MIP) codes, since the number of 0-1 integer variables in the model is not large.

For large problems, approximate solution procedures based on the concept of Lagrangean relaxation are developed (see [24], [51], and Appendix A.1). It is observed that the Lagrangean relaxations of all the models presented in this research have the integrality property. Thus, the upper bound on the objective function value obtained using the Lagrangean relaxation based solution procedures will only be as good as that obtained by the linear programming relaxation [24]. However, it is well known that the Lagrangean relaxation based solution procedures are more powerful than the solution procedures available for solving the linear programming relaxations, especially when the problem size is large [22].

Finally, the 0-1 integer programming formulations developed for the illustrative examples are solved using the SAS/OR
software package [60]; the Lagrangean relaxation based solution procedures, developed for the larger instances of the formulations, are coded in FORTRAN and run on an IBM 4381 computer.
CHAPTER IV

DEVELOPMENT OF SIMILARITY/COMPATIBILITY INDICES

The development of Similarity and Compatibility indices are presented in this chapter. The indices are illustrated using some examples.

4.1 Similarity Index

A common approach for grouping machines is to identify and exploit their similarities in processing the parts. The similarity between machines \( j \) and \( l \) is expressed in terms of their capabilities in processing a set of parts that need both machines. The capability of a machine is based on the tools available to it and the tooling requirements of the parts. Thus, the similarity between any two machines \( j \) and \( l \) may be defined as follows:

\[
S_{jl} = \frac{\sum_{i \in E_{jl}} [T_{ij} + T_{il}]}{\sum_{i} [T_{ij} + T_{il}]} \tag{4.1}
\]
The following example illustrates the above definition of the similarity index. Table 4.1 shows the number of common tools for a hypothetical 3-machine/5-part problem.

For this example, we have

\[ E_{12} = \{3,5\} \quad E_{13} = \{1,3,4\} \quad E_{23} = \{2,3\} \]

\[ T_{11} = 3 \quad T_{21} = 0 \quad T_{31} = 2 \quad T_{41} = 4 \quad T_{51} = 1 \]
\[ T_{12} = 0 \quad T_{22} = 4 \quad T_{32} = 1 \quad T_{42} = 0 \quad T_{52} = 6 \]
\[ T_{13} = 2 \quad T_{23} = 1 \quad T_{33} = 5 \quad T_{43} = 2 \quad T_{53} = 0 \]

Therefore,

\[ S_{12} = \frac{(2+1) + (1+6)}{(3+2+4+1) + (4+1+6)} = \frac{10}{21} = 0.476 \]

\[ S_{13} = \frac{(3+2+4) + (2+5+2)}{(3+2+4+1) + (2+1+5+2)} = \frac{18}{20} = 0.9 \]

\[ S_{23} = \frac{(4+1) + (1+5)}{(4+1+6) + (2+1+5+2)} = \frac{11}{21} = 0.524 \]

4.2 Compatibility Indices

The development of the indices to define the compatibility of a machine with a part is explained below.
Table 4.1 An Example of the Number of Common Tools for Different Part/Machine Combinations

<table>
<thead>
<tr>
<th>Machine, j</th>
<th>Part, i</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>3</td>
<td>-</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>4</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>
(a) The compatibility index based on tooling requirements is defined as:

\[ C_{ij}^1 = \frac{T_{ij}}{\min\{N_i, M_j\}} \]  \hspace{1cm} (4.2)

This index expresses the compatibility of a machine with a part in terms of processing requirements. A large number of common tools between a part and a machine implies less intercell movement if the part and the machine are placed in the same group.

(b) Alternatively, the compatibility index may be based on tooling needs and processing times, as defined below:

\[ C_{ij}^2 = \frac{\sum_{l \in o_{ij}} t_{il}}{\sum_{r \in q_i} t_{ir}} \]  \hspace{1cm} (4.3)

This index expresses the compatibility of a machine with a part in terms of material movement as well as the utilization of the machine. The longer the processing times of a part on a machine, the higher the utilization of that machine, if the part and the machine are placed in the same group.
The following example illustrates the above definitions of compatibility indices. Table 4.2 illustrates the data pertaining to two parts and one machine.

In this example,

\[ M_1 = 5 \quad N_1 = 3 \quad N_2 = 4 \]

\[ T_{11} = 2 \quad T_{21} = 2 \]

\[ 0_{11} = \{2,3\} \quad 0_{21} = \{1,2\} \quad Q_1 = Q_2 = \{1,2,3,4\} \]

Thus we have,

\[
C_{11}^1 = \frac{2}{\min\{5,3\}} = \frac{2}{3} = 0.67
\]

\[
C_{21}^1 = \frac{2}{\min\{5,4\}} = \frac{2}{4} = 0.50
\]

Using the alternative measure of compatibility, we have:

\[
C_{11}^2 = \frac{3 + 6}{4 + 3 + 6 + 2} = 0.6
\]

\[
C_{21}^2 = \frac{2 + 4}{2 + 4 + 5 + 4} = 0.4
\]
<table>
<thead>
<tr>
<th>Part</th>
<th>Operation number</th>
<th>Tools available on machine 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 2 3 4</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>T6 T1 T2 T6</td>
<td>T1</td>
</tr>
<tr>
<td></td>
<td>4 3 6 2</td>
<td>T2</td>
</tr>
<tr>
<td></td>
<td>operation time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(min.)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>T5 T4 T8 T7</td>
<td>T4</td>
</tr>
<tr>
<td></td>
<td>2 4 5 4</td>
<td>T5</td>
</tr>
<tr>
<td></td>
<td>tools needed</td>
<td></td>
</tr>
<tr>
<td></td>
<td>operation time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(min.)</td>
<td></td>
</tr>
</tbody>
</table>
The similarity index discussed in this chapter forms the basis of modelling the machine grouping problem. The concept of compatibility index is used to model the machine allocation problem and the machine-part grouping problem.
CHAPTER V

THE MACHINE ALLOCATION PROBLEM

The allocation of machines to part families is an important problem encountered in the planning stage of a cellular manufacturing system as its implications affect any efforts to economize on tooling requirements, improve machine utilization and material handling, and reduce set-up times, which ultimately leads to reduced work-in-process inventories and manufacturing lead times.

There could be many objectives of allocating machines to part families. Among the possibilities are: (i) to minimize the cost of material movement, (ii) to seek a trade off between the cost of duplicating machines and the cost of intercell movement, (iii) to provide maximum compatibility between parts and machines, and (iv) to minimize the work load imbalance among the machine groups.
The methodology proposed in this Chapter allocates the machines on the basis of their capabilities to process the parts under consideration. The processing requirements of the parts are related to their tooling needs, and the machine capabilities are expressed in terms of the tool availabilities on each machine. Using the indices of compatibility between the machines and the parts, two 0-1 integer programming formulations of the machine allocation problem are developed. The first formulation allocates the machines in such a way that the sum of compatibility indices of the machines and the parts in all the groups is maximized. The second formulation seeks a trade-off between the cost of duplicating the machines and the cost of intercell movement. These formulations take into account the physical constraints on the system such as limitations on the number of machines in a cell, and the available number of machines of each type.

The machine allocation models presented in this chapter are formulated under the following assumptions:

1. Part families are known.
2. Information regarding tooling requirements, processing times and production quantities for each part are available.

3. Information regarding the toolings available on each machine, machine cost, and the cost of material movement are available.

5.1 Model 1: Compatibility Maximization

The objective function of this model is formulated in order to maximize the sum of the compatibility indices of all machines and parts in all groups.

\[ \text{Max } Z_1 = \sum_{i} \sum_{j} \sum_{k} Y_{jk} a_{ik} C_{ij} \tag{5.1} \]

where \( C_{ij} \) is defined by either equation (4.2) or equation (4.3). The constraints of the model are as follows:

\[ \sum_{j} Y_{jk} \leq G_k \quad \forall k \tag{5.2} \]

\[ \sum_{k} Y_{jk} \leq A_j \quad \forall j \tag{5.3} \]

\[ Y_{jk} \in \{0,1\} \quad \forall (j,k) \tag{5.4} \]
There may be some restrictions on the number of machines in a group in order to have good control and supervision over the individual groups. This restriction is represented by constraint (5.2). In practice, there are restrictions on the availability of each type of machine in the system; this is represented by constraint (5.3). The last constraint ensures integrality of the decision variables.

5.2 Model 2: Cost Trade-off

The objective function of this model seeks a trade-off between the cost of duplicating the machines and the cost of intercell movement. Intercell movement arises if a part in group \( k \) requires a tool which is available only on a machine in group \( r \), \( r \neq k \). Thus, we have:

\[
\text{Min } Z_2 = \sum_{j} \sum_{k} f_{j} Y_{j,k} - \sum_{i} \sum_{j} \sum_{k} a_{i,k} T_{i,j,c} q_{i} / L_{i} \tag{5.5}
\]

The first term in the objective function represents the cost of allocating a machine to a group, whereas the second term represents the resulting savings in the cost of intercell movement. It is noted that if part \( i \) and machine
j are placed in different groups, then $T_{ij}(q_i/L_i)$ represents the upper limit on the number of intercell movements that will arise, and $T_{ij}c q_i/L_i$ is the corresponding average cost. The assumption here is that the operations of part $i$, requiring the common tools between that part and machine $j$ are not necessarily performed all at once. This implies that $T_{ij}(q_i/L_i)$ is the maximum amount of intercell movements to be expected. On the other hand, if all the operations are performed at once the number of intercell movements is only $(q_i/L_i)$. The objective function in (5.5) overestimates the cost of intercell movements; the actual cost depends on the scheduling rule used.

The constraints of the model are the same as in Model 1; that is equations 5.2 to 5.4 are applicable here. However, due to the structure of the objective function, this model may not allocate a machine to any group if it is not cost effective, i.e., if the difference between the savings in the cost of material movement and the cost of allocating that machine is negative. The effect of this is that, for some parts some operations which need that machine are
not performed. Hence, an additional constraint is needed to ensure that at least one machine of each type is available in some group:

\[ \sum_{k} Y_{jk} \geq 1 \quad \forall j \quad (5.6) \]

5.3 Application of the Formulations

The application of the machine allocation models (Model 1 and Model 2) is illustrated using a numerical example. A manufacturing system with 25 parts and 10 machines is considered. The tooling and the production requirements of the parts are given in Table 5.1. The toolings available on the machines, the annual fixed cost rate of the machines, and the number of machines of each type are given in Table 5.2. It is assumed that the average cost of an intercell movement is $5, and that the size of the unit handling load is 5 for all the parts. The parts have been grouped into three families, based on their similarity in tooling requirements. Table 5.3 represents the composition of the three families. The example problem was formulated and solved using Model 1 and Model 2, and the resulting solutions are given in Table 5.4.
<table>
<thead>
<tr>
<th>Part</th>
<th>Annual demand</th>
<th>Tools required (tool codes assumed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>600</td>
<td>A01, D01, D02, D03, H01, H02, M01, M02, M03, M04</td>
</tr>
<tr>
<td>2</td>
<td>800</td>
<td>A02, A03, A04, B01, C01, C02, M01, M05, R01, R02, R03, R04</td>
</tr>
<tr>
<td>3</td>
<td>1800</td>
<td>A01, B01, B02, B03, B04, E02, E03, E04, F01, F02, F03, F04, G05, R02, R03, R04, R06</td>
</tr>
<tr>
<td>4</td>
<td>1200</td>
<td>B02, D01, D02, D03, D04, H01, H02, H03, M01, M02</td>
</tr>
<tr>
<td>5</td>
<td>1300</td>
<td>B01, E02, E03, E04, E05, F04, F05, G01, G02, H03, R01, R03, R04</td>
</tr>
<tr>
<td>6</td>
<td>900</td>
<td>A01, A02, A05, A06, C04, M02, M03, M04, R05, R06</td>
</tr>
<tr>
<td>7</td>
<td>700</td>
<td>A02, A04, A06, C02, C03, C04, G01, G01, M01, M03, M05, M05, R02, R04, R05, R06</td>
</tr>
<tr>
<td>8</td>
<td>1600</td>
<td>B01, B03, E05, E06, R01, R02, R04</td>
</tr>
<tr>
<td>9</td>
<td>1200</td>
<td>A01, B02, B03, B05, B06, E01, E03, E05, F05, G01, G02, R01, R02</td>
</tr>
<tr>
<td>10</td>
<td>1100</td>
<td>D01, D03, D04, E07, H01, H02, H03, H04, M02, M03</td>
</tr>
<tr>
<td>11</td>
<td>1400</td>
<td>A01, A03, A06, C02, C03, C04, G06, M04, M05, R01, R04, R06</td>
</tr>
<tr>
<td>12</td>
<td>800</td>
<td>A01, D01, D02, H01, H02, M01, M02, M03</td>
</tr>
<tr>
<td>13</td>
<td>1300</td>
<td>B01, B03, D01, E01, E03, E05, G01, G03, R01, R02, R04, R06</td>
</tr>
<tr>
<td>14</td>
<td>900</td>
<td>C01, D02, D03, D04, H02, H03, H04, M04, M05</td>
</tr>
<tr>
<td>15</td>
<td>1100</td>
<td>A02, A04, A06, C01, C02, C03, C05, E06, G05, M01, M06, R01, R02, R04, R05</td>
</tr>
<tr>
<td>16</td>
<td>600</td>
<td>A01, A02, C01, C03, C04, E01, M01, M03, M04, M05, R01, R02, R03</td>
</tr>
<tr>
<td>17</td>
<td>1800</td>
<td>A01, D05, D06, F06, H06, H03, H04, M01, M03, M04, M05</td>
</tr>
<tr>
<td>18</td>
<td>1100</td>
<td>B01, C01, E01, E03, E04, E07, F01, F05, G03, G04, G05, G06, R01, R03, R06</td>
</tr>
<tr>
<td>19</td>
<td>1400</td>
<td>B01, B03, B04, C01, C02, E01, F01, F02, F03, H01, R01, R02, R03, R04</td>
</tr>
<tr>
<td>20</td>
<td>800</td>
<td>A05, D03, D04, D05, D06, H01, H02, H04, M04, M05</td>
</tr>
<tr>
<td>21</td>
<td>900</td>
<td>A01, A02, A03, A05, C02, C03, H01, M01, M02, M03, M01, R01, R06</td>
</tr>
<tr>
<td>22</td>
<td>1600</td>
<td>A01, A02, C01, C02, C04, H01, M01, M02, R01, R02, R03</td>
</tr>
<tr>
<td>23</td>
<td>1000</td>
<td>A01, B01, B02, B03, C01, E01, E02, F01, F02, F03, F05, G01, G02, H01, R01, R02, R03</td>
</tr>
<tr>
<td>24</td>
<td>1200</td>
<td>B01, B02, E02, E04, E05, F01, F03, G02, R01, R02, R04, R06</td>
</tr>
<tr>
<td>25</td>
<td>600</td>
<td>B01, B03, B04, C01, D02, E02, E03, F01, F03, F04, G01, G02, G04, R01, R03</td>
</tr>
</tbody>
</table>
### Table 5.2 Tools Available on the Machines
(for Models 1 and 2)

<table>
<thead>
<tr>
<th>Number</th>
<th>Machine available</th>
<th>Annual fixed cost rate</th>
<th>Tools available</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>10,000</td>
<td>A01, A02, A03, A04, A05, A06</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>8,000</td>
<td>B01, B02, B03, B04, B05, B06</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>6,000</td>
<td>C01, C02, C03, C04, C05</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>7,000</td>
<td>D01, D02, D03, D04, D05, D06</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>8,500</td>
<td>E01, E02, E03, E04, E05, E06, E07</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>12,000</td>
<td>F01, F02, F03, F04, F05, F06</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>11,000</td>
<td>G01, G02, G03, G04, G05, G06</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>9,000</td>
<td>H01, H02, H03, H04</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>10,500</td>
<td>M01, M02, M03, M04, M05</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>11,000</td>
<td>R01, R02, R03, R04, R05, R06</td>
</tr>
</tbody>
</table>
Table 5.3  Part Families Composition (for Models 1 and 2)

<table>
<thead>
<tr>
<th>Family</th>
<th>Parts</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3, 5, 8, 9, 13, 18, 19, 23, 24, and 25</td>
</tr>
<tr>
<td>2</td>
<td>2, 6, 7, 11, 15, 16, 21, and 22</td>
</tr>
<tr>
<td>3</td>
<td>1, 4, 10, 12, 14, 17, and 20</td>
</tr>
</tbody>
</table>
Table 5.4 Results of the Example Problem  
(for Models 1 and 2)

<table>
<thead>
<tr>
<th>Model</th>
<th>Machines allocated to</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>family 1</td>
</tr>
<tr>
<td>Model 1</td>
<td>1,2,5,6 &amp; 10</td>
</tr>
<tr>
<td>Model 2</td>
<td>2,5,6,7 &amp; 10</td>
</tr>
</tbody>
</table>
5.4 Solution Procedure for Large Problems

A Lagrangean relaxation based solution procedure has been developed to solve large instances of the machine allocation models presented in Section 5.2 [24]. The solution procedure developed for Model 1 (referred to as Model from now on) and the computational results are presented in this Section. The procedure to solve Model 2 is essentially the same, noting that constraint (5.6) of Model 2 is to be taken into account while making the Lagrangean solution feasible.

To motivate the development of the Lagrangean relaxation procedure, consider the objective functions of the above models which can be further simplified as follows:

Let,

\[ d_{jk} = \sum_i a_{ik} c_{ij} \]

and,

\[ e_{jk} = r_j - \sum_i T_{ij} a_{ik} c q_i / L_i \]

Then,

\[ Z_1 = \text{Max} \sum \sum Y_{jk} d_{jk} \]

- 74 -
\[ Z_2 = \min \sum \sum Y_{jk} e_{jk}. \]

As stated previously, the procedure is developed considering \( Z_1 \) only. The structure of \( Z_2 \) is essentially the same as that of \( Z_1 \), and thus the same procedure could be utilized. Now, considering \( Z_1 \), two relaxations are possible for this Model. The first one is obtained by relaxing the machine availability constraint, equation (5.3), and the second is obtained by relaxing the machine group size constraint, equation (5.2).

**Relaxation 1 (LRMA1).** We define the following problem:

\[ Z_{D11} (U) = \max \sum \sum Y_{jk} d_{jk} - \sum_{j,k} u_j (\sum_{j,k} Y_{jk} - A_j) \]

subject to constraints (5.2), (5.4)

and \( U \geq 0 \)

which may be rewritten as:

\[ \max \sum \sum (d_{jk} - u_j) Y_{jk} + \sum u_j A_j \]

subject to constraints (5.2), (5.4)

and \( U \geq 0 \)

**Relaxation 2 (LRMA2).** The problem is defined as:

\[ Z_{D12} (V) = \max \sum \sum Y_{jk} d_{jk} - \sum_{k,j} v_k (\sum_{j,k} Y_{jk} - G_k) \]
subject to constraints (5.3), (5.4) and $V \geq 0$

which may be rewritten as:

$$\text{Max} \sum \sum (d_{jk} - v_k) Y_{jk} + \sum v_k G_k$$

subject to constraints (5.3), (5.4) and $V \geq 0$

For a given value of the Lagrangean multipliers $U$, Relaxation 1 can be easily solved by simple ranking, i.e., determining the maximum $(d_{jk} - u_j)$ for each $k$ and setting the associated $Y_{jk} = 1$. This is repeated until $G_k$ $Y_{jk}$'s are set equal to 1. The remaining $Y_{jk}$'s are then set to zero [24]. Similarly, for a given value of the Lagrangean multipliers $V$, Relaxation 2 can be solved as follows. Determine the first $A_j$ largest $(d_{jk} - v_k)$ for each $j$ and set the corresponding $Y_{jk}$'s equal to 1 and the remaining $Y_{jk}$'s to zero [24].

The best choice for $U$ would be an optimal solution to the following problem:

$$Z_{D11} = \text{Min}_U Z_{D11}(U)$$

- 76 -
$Z_{D_{11}}$ provides a tight upperbound on $Z_1$. Similarly, the best choice for $V$ would be an optimal solution to the following problem:

$$Z_{D_{12}} = \min_V Z_{D_{12}}(V)$$

$Z_{D_{12}}$ provides a tight upperbound on $Z_1$. The functions $Z_{D_{11}}(U)$ and $Z_{D_{12}}(V)$ have all the nice properties, like continuity and convexity, except one - differentiability. Of all the methods available to solve these types of problems, the subgradient method is easy to program and has performed well on many practical problems (see Appendix A.1, and [24]). Hence it is used to update the multipliers in solving the above problems.

**Relaxation 1.** The subgradient method to update the multipliers is explained here. Given an initial value $U^0$, a sequence $\{ U^p \}$ is generated by the following rule:

$$u_{j}^{p+1} = u_{j}^{p} + t^{p} \left( \sum_{k} Y_{j}^{p} - A_{j} \right) \quad \forall j,$$

where $Y_{j}^{p}$ is an optimal solution to LRMA1 problem (i.e., $Z_{D_{11}}(U^{p})$), $t^{p}$ is a positive scalar step size, and $p$ is the
iteration number. The formula used to calculate the step size is:

\[ t_p = \frac{\lambda_p \left( Z_{D11}(U^p) - Z_1^* \right)}{\| \sum_{j_1} y_{j_k}^p - \Lambda_j \|^2} \]

where \( \lambda_p \) is a scalar satisfying \( 0 < \lambda_p \leq 2 \), and \( Z_1^* \) is the initial estimate on \( Z_{D11} \). The sequence \( \langle \lambda_p \rangle \) is determined by setting \( \lambda_0 = 2 \) and halving it whenever \( Z_{D11}(U^p) \) has failed to decrease in some fixed number of iterations. The method is terminated upon reaching prespecified iteration limit, or when the step size \( t_p \) almost equals zero.

**Relaxation 2.** In the case of LRMA2, the subgradient method is similarly applied; the updating formulae are:

\[ v_{k+1}^p = v_k^p + t_p \left( \sum_{j_k} y_{j_k}^p - G_k \right) \quad \forall k, \]

and

\[ t_p = \frac{\lambda_p \left( Z_{D12}(V^p) - Z_1^* \right)}{\| \sum_{j_k} y_{j_k}^p - G_k \|^2} \]

- 78 -
The resulting solutions of Relaxations 1 and 2 may or may not be feasible to the original problem. If they are not feasible, a feasible solution is obtained by suitably tinkering with the infeasible solution. The following method is used to obtain a feasible solution [24].

**Relaxation 1:**

Let \( \hat{Y}_{jk} \) denote an optimal solution to Relaxation 1. In Relaxation 1, the constraints \( \sum_k Y_{jk} \leq A_j, j=1,J \) are dualized and may be violated. Partition \( J \) into two subsets defined by

\[
S_1 = \{ j : \sum_k \hat{Y}_{jk} \leq A_j \} \\
S_2 = \{ j : \sum_k \hat{Y}_{jk} > A_j \}
\]

The constraints of the Model which are violated by \( \hat{Y}_{jk} \) correspond to \( j \in S_2 \). Hence \( \hat{Y}_{jk} \) has to be modified so that these constraints are satisfied. For all \( j \in S_2 \), simply remove machine \( j \) from \( \sum_k \hat{Y}_{jk} - A_j \) groups. The rule is to remove the machine from group \( \hat{k} \), where \( d_{jk} \hat{k} = \min \{ d_{jk} \} \) and check if now \( j \in S_1 \). If not, the procedure is repeated.
until feasibility is reached.

Relaxation 2:

In the case of Relaxation 2, the procedure to find a feasible solution is as follows. Let

\[ S_3 = \{ k \mid \sum_{j} \hat{Y}_{jk} \leq G_k \} \]
\[ S_4 = \{ k \mid \sum_{j} \hat{Y}_{jk} > G_k \} \]
\[ S_5 = \{ j \mid \sum_{k} \hat{Y}_{jk} = 0 \} \]

For all \( k \in S_4 \), simply remove \( \left( \sum_{j} \hat{Y}_{jk} - G_k \right) \) machines from group \( k \). The rule is, from group \( k \), remove machine \( j \), where \( d_{jk}^\wedge = \min \{ d_{jk} \} \) and continue removing machines until feasibility is restored. This procedure may lead to a solution where a machine is not allocated to any cell. Hence for each \( j \in S_5 \), insert machine \( j \) in the first \( A_j \) groups without violating the group size constraints.

In the case of Model 2, for each \( j \in S_5 \), insert machine \( j \) in at most \( A_j \) groups for which the allocation is cost effective, without violating the group size constraints. If
the allocation is not cost effective for any of the groups i.e. the annual cost rate of a machine is higher than the intercell movement cost), then insert machine \( j \) into the most attractive group (say \( k \); \( d_{jk} = \max (d_{jk}) \)). This avoids violating the constraint (5.6) of Model 2.

5.5 Computational Experience

The Lagrangean relaxation procedure developed in the previous section was applied to a number of large, randomly generated machine allocation problems (Model 1). In this section, the computational results are presented. An IBM random number generator subroutine called RAND was used to generate the various parameters of the test problems used in this study.

(a) The effect of problem size. In the first set of experiments, the effect of the problem size on the performance of the two relaxations was analyzed. Four problems of widely different sizes were generated. The number of machines and the number of machine groups in each case are different, and only one unit of each machine type is assumed to be available. The machine group size in each case is assumed not to exceed the total number of machines
divided by the number of groups. The values of $d_{jk} = \sum_{i} C_{i,j}$ are real numbers and were generated from the uniform distribution [0,5]. The computational results of this experiment are summarized in Table 5.5.

From Table 5.5, it appears that the time required to solve the machine allocation model using Relaxation 2 is less than that obtained using Relaxation 1. The computational time is higher in the case of Relaxation 1 due to the fact that the size of the subproblems solved is larger than the size of the subproblems of Relaxation 2. In the case of Relaxation 1, the subproblems are of size $J$ (i.e., number of machines), whereas in the case of Relaxation 2, the subproblem size is $K$ (i.e., number of groups). Both Relaxation 1 as well as Relaxation 2 are computationally efficient since even the largest problem took only less than a second to obtain the Lagrangean solution (upper bound) and the feasible solution (lower bound). It is also observed that the deviation between the upper bound and the lower bound as a percentage of the upperbound is zero in the case of Relaxation 2; however, it varies from 6.6% to 16.7% in the
<table>
<thead>
<tr>
<th>Problem No.</th>
<th>Problem size</th>
<th>CPU time (m. secs)</th>
<th>(\frac{100(UB-LB)}{UB})</th>
<th>LB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>50</td>
<td>5</td>
<td>23</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>10</td>
<td>69</td>
<td>49</td>
</tr>
<tr>
<td>3</td>
<td>250</td>
<td>10</td>
<td>186</td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>500</td>
<td>20</td>
<td>683</td>
<td>416</td>
</tr>
</tbody>
</table>

Table 5.5 The Effect of Problem Size
case of Relaxation 1. This indicates that Relaxation 2, in
general, produces tighter bounds than Relaxation 1.
Therefore, the feasible solutions obtained by the Lagrangean
relaxation procedure using Relaxation 2 are, in general,
superior to those obtained using Relaxation 1.

(b) The effect of the initial estimate. In the second set of
experiments, the effect of initial estimate \( Z_1^* \) on the
performance of the two relaxations was studied. Six
different problems were generated in which the the number
of machines, the number of groups, and the limit on the
number of machines in each group were fixed at 100, 10, and
10, respectively. It is assumed that only one unit of each
machine type is available. The \( d_{jk} \) values were generated
from the uniform distribution \([0,5]\). The initial estimate
\( Z_1^* \) in each case is also chosen randomly. The results of
this experiment are given in Table 5.6.

From Table 5.6, it appears that the lower bounds obtained
using Relaxation 1 are relatively insensitive to the initial
estimate. In the case of Relaxation 2, the lower bound
Table 5.6 The Effect of Initial Estimate ($Z_1^*$)

<table>
<thead>
<tr>
<th>$Z_1^*$</th>
<th>LB</th>
<th></th>
<th>100*(UB-LB)/UB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rel 1</td>
<td>Rel 2</td>
<td>Rel 1</td>
</tr>
<tr>
<td>0</td>
<td>78.15</td>
<td>79.99</td>
<td>13.37</td>
</tr>
<tr>
<td>30</td>
<td>78.15</td>
<td>79.99</td>
<td>13.37</td>
</tr>
<tr>
<td>60</td>
<td>78.15</td>
<td>79.99</td>
<td>13.37</td>
</tr>
<tr>
<td>80</td>
<td>78.15</td>
<td>79.99</td>
<td>13.37</td>
</tr>
<tr>
<td>100</td>
<td>78.56</td>
<td>45.12</td>
<td>70.13</td>
</tr>
<tr>
<td>120</td>
<td>78.56</td>
<td>45.12</td>
<td>87.39</td>
</tr>
</tbody>
</table>
appears to be insensitive when $Z_1^*$ is underestimated relative to the lower bound; it is, however, sensitive when it is overestimated. In the case of Relaxation 2, the deviation between the upper bound and the lower bound as a percentage of the upper bound appears to be insensitive to the initial estimate. However, in the case of Relaxation 1, this quantity appears to be insensitive when the initial estimate is underestimated; it is sensitive when it is overestimated.

(c) The effect of the problem structure. This experiment tries to capture the effect of the problem structure on the performance of the two relaxations. The number of machines and the number of groups are fixed at 100 and 10, respectively, and the $d_{jk}$ values are generated, as before, using the uniform distribution $[0,5]$. The available number of machines of each type ($A_j$), and the limit on the number of machines in each group ($G_k$), are the two problem parameters considered in this experiment. Parameter $A_j$ may be constant or varying; similarly, parameter $G_k$ may be constant or varying. When the parameters are constant three levels of their values are considered (low, medium, and high), and
when they are varying, two levels are considered (low and high). When the parameters are varying, the values are obtained using a discrete uniform distribution. Combining the two parameters and their associated levels results in 25 problems, which are solved using both relaxations. The experimental design used to study the effect of the problem structure is summarized in Table 5.7, and the computational results are given in Table 5.8.

From Table 5.8, it can be seen that Relaxation 2 consistently produces better lower bounds than Relaxation 1. As the available number of machines of each type increases, the values of the lower bounds obtained increase. This is due to the fact that parameter $A_j$ directly contributes to the objective function. It appears that an increase in group size $G_k$ (limit on the number of machines in each group) marginally increases the value of the lower bounds obtained. Variability in group size, on the other hand, does not seem to have a significant effect on the lower bounds.

(d) Comparison of the Lagrangean solution with optimal solution. The performance of the two relaxations on the
Table 5.7 Experimental Design for Machine Allocation

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. $A_j$</td>
<td></td>
</tr>
<tr>
<td>(a) Constant</td>
<td>low : 1 med : 3</td>
</tr>
<tr>
<td></td>
<td>high : 6</td>
</tr>
<tr>
<td>(b) Varying</td>
<td>low : [1,2]</td>
</tr>
<tr>
<td></td>
<td>high : [1,6]</td>
</tr>
<tr>
<td>2. $G_k$</td>
<td></td>
</tr>
<tr>
<td>(a) Constant</td>
<td>low : 10 med : 12</td>
</tr>
<tr>
<td></td>
<td>high : 20</td>
</tr>
<tr>
<td>(b) Varying</td>
<td>low : [10,12]</td>
</tr>
<tr>
<td></td>
<td>high : [10,20]</td>
</tr>
</tbody>
</table>
### Table 5.8 The Effect of the Problem Structure

<table>
<thead>
<tr>
<th>$G_k$</th>
<th>10</th>
<th>12</th>
<th>20</th>
<th>[10,12]</th>
<th>[10,20]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_j$</td>
<td>Lower bound</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Rel 1</td>
<td>78.35</td>
<td>61.33</td>
<td>69.31</td>
<td>79.03</td>
</tr>
<tr>
<td></td>
<td>Rel 2</td>
<td>80.28</td>
<td>79.97</td>
<td>79.97</td>
<td>80.03</td>
</tr>
<tr>
<td>3</td>
<td>Rel 1</td>
<td>200.86</td>
<td>215.15</td>
<td>174.15</td>
<td>216.05</td>
</tr>
<tr>
<td></td>
<td>Rel 2</td>
<td>215.32</td>
<td>215.38</td>
<td>215.73</td>
<td>216.12</td>
</tr>
<tr>
<td>6</td>
<td>Rel 1</td>
<td>317.13</td>
<td>285.78</td>
<td>350.83</td>
<td>330.32</td>
</tr>
<tr>
<td></td>
<td>Rel 2</td>
<td>349.57</td>
<td>350.92</td>
<td>350.93</td>
<td>350.83</td>
</tr>
<tr>
<td>[1,2]</td>
<td>Rel 1</td>
<td>78.39</td>
<td>67.31</td>
<td>64.88</td>
<td>78.35</td>
</tr>
<tr>
<td></td>
<td>Rel 2</td>
<td>80.2</td>
<td>80.27</td>
<td>80.28</td>
<td>80.28</td>
</tr>
<tr>
<td>[1,6]</td>
<td>Rel 1</td>
<td>144.36</td>
<td>179.63</td>
<td>166.71</td>
<td>164.79</td>
</tr>
<tr>
<td></td>
<td>Rel 2</td>
<td>159.71</td>
<td>184.79</td>
<td>177.13</td>
<td>179.23</td>
</tr>
</tbody>
</table>
sample problem presented in Section 5.3 is given in Table
5.9. It is observed that Relaxation 2 resulted in optimal
solution for this example for both Model 1 and Model 2;
Relaxation 1, however, resulted in solutions that are within
3.5% of the optimal solutions.

It is noted that any generalization in this regard is too
difficult to make. The Lagrangean relaxation procedure
obtains feasible solutions to very large problems for which
optimal solutions are very difficult, if not impossible, to
arrive at. Thus, it is not possible to provide any
comparisons with optimal solutions in such cases. It is
hoped, however, that the feasible solution is "close" to the
optimal solution.
<table>
<thead>
<tr>
<th>Problem number</th>
<th>CPU time (secs)</th>
<th>Objective function value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SAS/OR</td>
<td>Rel 1</td>
</tr>
<tr>
<td>1 Model 1</td>
<td>10.2</td>
<td>0.03</td>
</tr>
<tr>
<td>2 Model 2</td>
<td>10.2</td>
<td>0.03</td>
</tr>
</tbody>
</table>
CHAPTER VI

MACHINE-PART GROUPING PROBLEM

The machine-part grouping problem is concerned with producing parts with similar processing requirements in machine groups. Each machine group consists of dissimilar types of machines which possess specific capabilities to manufacture one or more part families. This problem may be approached in two ways: (i) the sequential approach which considers the machine group formation first, to be followed by the allocation of parts to machine groups, and (ii) the simultaneous approach which considers the problem of the grouping of parts and machines in one step. The models presented in this Chapter are formulated under the following assumptions:

1. Information regarding tooling requirements, processing times, and production quantities for each part are available.

2. Information regarding the toolings available on each
machine, machine cost, and the cost of material movement are available.

6.1 SEQUENTIAL APPROACH

A sequential modelling approach to the machine-part grouping problem based on the tooling requirements of the parts and the tools available on the machines is presented in this Section. The machine group formation problem is modelled as a 0-1 integer programming model that is similar to the p-median problem [24]. The objective is to maximize the sum of the similarity indices among all the machines in all the groups. The parts allocation problem is then modelled as a special case of the generalized assignment problem, given that the machine groups are already formed. The objective is to maximize the sum of the compatibility indices of all the parts in all groups.

6.1.1 Model 3 - Machine group formation

The machine group formation model takes into consideration the physical constraints of the system such as restrictions on the number of machine groups and the available number of machines of each type. The objective function of the model
is formulated in order to maximize the sum of the similarity indices of all the machines in all the machine groups with their respective group medians:

$$\text{Max } Z_g = \sum_j \sum_l S_{jl} W_{jl}$$  \hspace{1cm} (6.1)

where $S_{jl}$ is defined by Equation (4.1).

The constraints of the model are as follows:

$$\sum_l W_{ll} = K \hspace{1cm} \forall l$$  \hspace{1cm} (6.2)

$$\sum_l W_{jl} \leq A_j \hspace{1cm} \forall j$$  \hspace{1cm} (6.3)

$$W_{jl} \leq W_{ll} \hspace{1cm} \forall (j,l)$$  \hspace{1cm} (6.4)

$$W_{jl} \in \{0,1\} \hspace{1cm} \forall (j,l)$$  \hspace{1cm} (6.5)

Constraint (6.2) ensures that there are exactly $K$ machine groups. Constraint (6.3) imposes the restriction on the available number of machines of each type. Constraint (6.4) ensures that machine $j$ is grouped with machine $l$, only if machine $l$ is chosen as the median of a group. The last constraint imposes the integrality requirement.
6.1.2 Model 4 - Parts allocation

The objective function of the model is formulated in order to maximize the sum of the compatibility indices of all the parts allocated to all the machine groups:

\[
\text{Max } Z_4 = \sum \sum \sum \ X_{i,k} \ b_{j,k} \ c_{i,j} \tag{6.6}
\]

where \( c_{i,j} \) is defined by Equation (4.2) or (4.3).

The constraints of the model are as follows:

\[
\sum X_{i,k} = 1 \quad \forall i \tag{6.7}
\]

\[
\sum X_{i,k} \leq H_k \quad \forall k \tag{6.8}
\]

\[
X_{i,k} \in \{0,1\} \quad \forall (i,k) \tag{6.9}
\]

Constraint (6.7) ensures that a part is allocated to only one machine group. Constraint (6.8) imposes the restriction on the number of parts in each group. The last constraint meets the integrality requirements.

6.1.3 Application of the formulations

The application of the above formulations to the problem of cell formation is illustrated in this section using an
example. A manufacturing system with 25 parts and 7 machines is considered. The tooling requirements of the parts are given in Table 6.1, and the number of available machines of each type, and the tools available on each machine type are given in Table 6.2. It is assumed that the maximum number of parts that may be allocated to any machine group is 10, and the maximum number of machine groups is 3. For this example, the machine group formation problem was formulated and solved using Model 3; the resulting machine groups are as follows:

Group 1: machines 1, 2, 5 and 7
Group 2: machines 4 and 6
Group 3: machines 1, 3 and 7

Using the results of Model 3, the parts allocation problem was formulated and solved using Model 4. Combining the results of Model 4 with those of Model 3, the following machine-part groups are formed:

Group 1: parts 3, 5, 8, 9, 13, 19, 23, 24 and 25 machines 1, 2, 5 and 7

Group 2: parts 1, 4, 10, 12, 14, 17 and 20 machines 4 and 6

Group 3: parts 2, 6, 7, 11, 15, 16, 18, 21 and 22 machines 1, 3 and 7
<table>
<thead>
<tr>
<th>Part #</th>
<th>Tools required (tool codes assumed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A01, D01, D02, D03, H01, H02</td>
</tr>
<tr>
<td>2</td>
<td>A02, A03, A04, B01, C01, C02, R01, R02, R03, R04</td>
</tr>
<tr>
<td>3</td>
<td>A01, B01, B02, B03, B04, E02, E03, E04, R02, R03, R04, R06</td>
</tr>
<tr>
<td>4</td>
<td>B02, D01, D02, D03, D04, H01, H02, H03</td>
</tr>
<tr>
<td>5</td>
<td>B01, E02, E03, E04, E05, H03, R01, R03, R04</td>
</tr>
<tr>
<td>6</td>
<td>A01, A02, A05, A06, C04, R05, R06</td>
</tr>
<tr>
<td>7</td>
<td>A02, A04, A06, C02, C03, C04, C05, R02, R04, R05, R06</td>
</tr>
<tr>
<td>8</td>
<td>B01, B03, E05, E06, R01, R02, R04</td>
</tr>
<tr>
<td>9</td>
<td>A01, B02, B03, B05, B06, E01, E03, E05, E06, E07, R01, R02</td>
</tr>
<tr>
<td>10</td>
<td>D01, D03, D04, E07, H01, H02, H03, H04</td>
</tr>
<tr>
<td>11</td>
<td>A01, A03, A06, C02, C03, C04, G06, R01, R02, R04, R06</td>
</tr>
<tr>
<td>12</td>
<td>A01, D01, D02, H01, H02</td>
</tr>
<tr>
<td>13</td>
<td>B01, B03, D01, E01, E03, E05, R01, R02, R04, R06</td>
</tr>
<tr>
<td>14</td>
<td>C01, D02, D03, D04, H02, H03, H04</td>
</tr>
<tr>
<td>15</td>
<td>A02, A04, A06, C01, C02, C03, C05, E06, R01, R02, R04, R05</td>
</tr>
<tr>
<td>16</td>
<td>A01, A02, C01, C03, C04, E01, R01, R02, R03</td>
</tr>
<tr>
<td>17</td>
<td>A01, D05, D06, H03, H05, H06</td>
</tr>
<tr>
<td>18</td>
<td>B01, C01, E01, E03, E04, E07, R01, R03, R06</td>
</tr>
<tr>
<td>19</td>
<td>B01, B03, B04, C01, C02, E01, H01, R01, R02, R03, R04</td>
</tr>
<tr>
<td>20</td>
<td>A05, D03, D04, D05, D06, H01, H02, H04</td>
</tr>
<tr>
<td>21</td>
<td>A01, A02, A05, C02, C03, H01, R01, R06</td>
</tr>
<tr>
<td>22</td>
<td>A01, A02, C01, C02, C04, H01, R01, R02, R03</td>
</tr>
<tr>
<td>23</td>
<td>A01, B01, B02, B03, C01, E01, E02, H01, R01, R02, R03</td>
</tr>
<tr>
<td>24</td>
<td>B01, B02, E02, E04, E05, R01, R02, R04, R06</td>
</tr>
<tr>
<td>25</td>
<td>B01, B03, B04, C01, D02, E02, E03, R01, R03</td>
</tr>
</tbody>
</table>

- 97 -
<table>
<thead>
<tr>
<th>Machine #</th>
<th>Number of machines available</th>
<th>Tools available</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>A01, A02, A03, A04, A05, A06</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>B01, B02, B03, B04, B05, B06</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>C01, C02, C03, C04, C05</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>D01, D02, D03, D04, D05, D06</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>E01, E02, E03, E04, E05, E06, E07</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>H01, H02, H03, H04</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>R01, R02, R03, R04, R05, R06</td>
</tr>
</tbody>
</table>
6.1.4 Solution Procedure for Large-Scale Formulations of Model 3

The solution procedure developed for solving large scale versions of Model 3 based on the concept of Lagrangean relaxation is presented in this section. A Lagrangean relaxation for this model is obtained by dualizing constraint (6.3).

Relaxed model (LRMG). We define the following problem:

\[
Z_{D3}(\tilde{\alpha}) = \max \sum_j \sum_l \bar{W}_{j,l} s_{j,l} - \sum_j \alpha_j \left( \sum_l \bar{W}_{j,l} - A_j \right)
\]

subject to constraints (6.2), (6.4), (6.5), and \( \tilde{\alpha} \geq 0 \)

which may be rewritten as:

\[
\max \sum_j \sum_l (s_{j,l} - \alpha_j) \bar{W}_{j,l} + \sum_j \alpha_j A_j
\]

subject to constraints (6.2), (6.4), (6.5), and \( \tilde{\alpha} \geq 0 \)

This model has a 0-1 variable upper bound (VUB) structure.

It is observed that the VUB constraints (6.4) and the objective function of the relaxed problem (LRMG) imply that,

\[
\bar{W}_{j,l} = \bar{W}_{l,l} \text{ if } s_{j,l} - \alpha_j \geq 0
\]

\[
= 0 \text{ otherwise}
\]

Hence, defining

- 99 -
\[ \hat{S}_l = \sum \max(0, S_{jl} - \alpha_j), \]

The optimal \( \hat{W}_{ll} \)'s are obtained by solving the following trivial problem [51]:

\[ \max \sum \sum \hat{S}_l \hat{W}_{ll} \]

\[ \sum \hat{W}_{ll} = K \quad \text{(KP)} \]

\[ \hat{W}_{ll} \in \{0, 1\} \quad \forall l \]

The subgradient method is again used to update the Lagrangean multipliers in solving the following problem to find the upper bound on \( Z_{D3} \).

\[ Z_{D3} = \min Z_{D3}(\tilde{\alpha}) \]

The subgradient method used to update the multipliers is outlined below. Given an initial value \( \tilde{\alpha}^0 \), a sequence \( \{\tilde{\alpha}^p\} \) is generated by the rule

\[ \tilde{\alpha}_j^{p+1} = \tilde{\alpha}_j^p + t^p \left( \sum \hat{W}_{lj}^p - A_j \right) \quad \forall j, \]

where \( \hat{W}_{lj}^p \) is an optimal solution to the relaxed model, \( t^p \) is a positive scalar step size, and \( p \) is the iteration number.

The formula used to calculate the step size is as follows:
\[ t^p = \frac{\lambda_p \left( Z_{D3}(\bar{c}^p) - Z_3^* \right)}{\| \sum_l \psi_j l - \lambda_j \|_2} \]

where \( \lambda_p \) is a scalar satisfying \( 0 < \lambda_p \leq 2 \), and \( Z_3^* \) is the initial estimate on \( Z_{D3} \). The sequence \( \{\lambda_p\} \) is determined by setting \( \lambda_o = 2 \) and halving it whenever \( Z_{D3}(\bar{c}) \) has failed to decrease in some fixed number of iterations. The method is terminated upon reaching the prespecified iteration limit, or when the step size almost equals zero.

To obtain a feasible solution from the Lagrangean solution, we proceed as follows. Let \( \hat{W}_{jl} \) denote an optimal solution to the relaxed model. In the relaxed model, the constraints

\[ \sum_l W_{jl} \leq A_j, \ \forall j \]

are dualized and may be violated. Partition \( J \) into two subsets defined by,

\[ S_6 = \{ j \mid \sum_l \hat{W}_{jl} \leq A_j \} \]

\[ S_7 = \{ j \mid \sum_l \hat{W}_{jl} > A_j \} \]

- 101 -
The constraints of the model which are violated by $\hat{W}_{j_1}$ correspond to $j \in S_7$. Hence $\hat{W}_{j_1}$'s have to be modified so that these constraints are satisfied. For all $j \in S_7$, simply remove machine $j$ from $\left( \sum_{l} \hat{W}_{j_l} - A_j \right)$ groups. The rule may be to machine $j$ from the groups that have the least similarity with it.

6.1.5 Computational Experience of Model 3

The computational experience of the Lagrangean relaxation procedure on large randomly generated test problems of Model 3 is presented in this section.

(a) The effect of the problem size. In the first set of experiments, the effect of the problem size on the performance of the Lagrangean relaxation procedure was analyzed. Four problems of widely different sizes were chosen and in each case four problem instances were generated. The number of machines and the number of machine groups in each case are different, and only one unit of each machine type is assumed to be available. The similarity
indices are real numbers and were generated from the uniform distribution [0,1]. The computational results of this experiment are given in Table 6.3, where the entries represent the average over four problem instances.

From Table 6.3, it is observed that the Lagrangean relaxation procedure is very efficient in terms of CPU time; even the largest problem, which involves 500 machines and 20 groups, required only 1.5 seconds to reach a feasible solution. The deviation between the upper bound and the lower bound, as a percentage of the upper bound, is only around 5%, indicating that the bounds are reasonably tight.

(b) The effect of the initial estimate. In the second set of experiments, the effect of the initial estimate ($z^*_3$) on the performance of the Lagrangean relaxation procedure was studied. Six different problems were generated in which the number of machines and the number of groups were fixed at 100 and 10, respectively. It is assumed that only one unit of each machine type is available. The similarity indices were generated from the uniform distribution [0,1]. The initial estimate ($z^*_3$) in each case is also chosen randomly.
<table>
<thead>
<tr>
<th>Problem size</th>
<th>No. of machines</th>
<th>No. of groups</th>
<th>CPU time (m.secs)</th>
<th>100*(UB-LB)/UB</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>5</td>
<td></td>
<td>12.59</td>
<td>5.12</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td></td>
<td>50.98</td>
<td>4.92</td>
</tr>
<tr>
<td>250</td>
<td>10</td>
<td></td>
<td>346.07</td>
<td>5.46</td>
</tr>
<tr>
<td>500</td>
<td>20</td>
<td></td>
<td>1467.97</td>
<td>5.23</td>
</tr>
</tbody>
</table>
The results of this experiment are given in Table 6.4. From Table 6.4, it appears that the lower bound as well as the deviation between the upper bound and the lower bound, as a percentage of the upper bound, are insensitive to the initial estimate.

(c) **The effect of the problem structure.** This experiment tries to capture the effect of the problem structure on the performance of the Lagrangean relaxation procedure. The number of machines and the number of groups are fixed at 100 and 10, respectively, and the similarity indices are generated, as before, using the uniform distribution [0,1]. The parameter $A_j$ may be constant or varying. When it is constant, three levels of its values are considered (low, medium, and high), and when it is varying, two levels of its values are considered (low and high). This results in 5 problems, which are solved using Lagrangean relaxation procedure. When the parameter is varying, the values are obtained using a discrete uniform distribution. The experimental design used to study the effect of the problem structure is summarized in Table 6.5, and the computational results are given in Table 6.6.
<table>
<thead>
<tr>
<th>$z^*_3$</th>
<th>Lower bound</th>
<th>$100\times(UB-LB)/UB$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>81.696</td>
<td>4.92</td>
</tr>
<tr>
<td>40</td>
<td>81.696</td>
<td>4.92</td>
</tr>
<tr>
<td>80</td>
<td>81.696</td>
<td>4.92</td>
</tr>
<tr>
<td>120</td>
<td>81.696</td>
<td>4.92</td>
</tr>
<tr>
<td>240</td>
<td>81.696</td>
<td>4.92</td>
</tr>
<tr>
<td>480</td>
<td>81.696</td>
<td>4.92</td>
</tr>
</tbody>
</table>
Table 6.5 Experimental Design for Machine Grouping

<table>
<thead>
<tr>
<th>Parameter ( A_j )</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Constant</td>
<td>low : 1, med : 3</td>
</tr>
<tr>
<td></td>
<td>high : 6</td>
</tr>
<tr>
<td>2. Varying</td>
<td>low : [1,2]</td>
</tr>
<tr>
<td></td>
<td>high : [1,6]</td>
</tr>
</tbody>
</table>
Table 6.6 Effect of Problem Structure

<table>
<thead>
<tr>
<th>No. of machines available</th>
<th>Lower bound</th>
<th>100*(UB-LB)/UB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>81.69</td>
<td>4.92</td>
</tr>
<tr>
<td>3</td>
<td>218.34</td>
<td>4.92</td>
</tr>
<tr>
<td>6</td>
<td>352.62</td>
<td>4.62</td>
</tr>
<tr>
<td>2. Varying</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[1,3]</td>
<td>81.83</td>
<td>5.37</td>
</tr>
<tr>
<td>[1,6]</td>
<td>186.36</td>
<td>5.41</td>
</tr>
</tbody>
</table>
From Table 6.6, it appears that the variation in the parameter $A_j$ does not have any significant effect on the deviation between the upper bound and the lower bound. However, it influences the lower bound since parameter $A_j$ directly contributes to the objective function value.

### 6.1.6 Solution Procedure for Large-Scale Formulations of Model 4

The objective function of the parts allocation model (Model 4) can be further simplified as follows. Let,

$$ h_{i,k} = \sum_j b_{j,k} C_{i,j} $$

Then,

$$ Z_4 = \text{Max} \sum_i \sum_k X_{i,k} h_{i,k} $$

The solution procedure developed for the above model is explained in this section. Two relaxations are possible for this model. The first one is obtained by relaxing the part allocation constraint (6.7) and the second one by relaxing the group size constraint (6.8).

**Relaxation 1 (LRPA1).** Define the following problem:

$$ Z_{D41}(\beta) = \text{Max} \sum_i \sum_k X_{i,k} h_{i,k} - \sum_k \beta \left( \sum_i X_{i,k} - 1 \right) $$

- 109 -
subject to constraints (6.8), (6.9), and \( \beta \) unrestricted

which may be rewritten as:

\[
\text{Max } \sum \sum (h_{i,k} - \beta_i) X_{i,k} + \sum \beta_i
\]

subject to constraints (6.8), (6.9), and \( \beta \) unrestricted

**Relaxation 2 (LRPA2).** Define the problem as:

\[
Z_{D42}(\gamma) = \text{Max } \sum \sum X_{i,k} h_{i,k} - \sum \gamma_k (\sum X_{i,k} - H_k)
\]

subject to constraints (6.7), (6.9),

and \( \gamma \geq 0 \)

which may be rewritten as:

\[
\text{Max } \sum \sum (h_{i,k} - \gamma_k) X_{i,k} + \sum \gamma_k H_k
\]

subject to constraints (6.7), (6.9),

and \( \gamma \geq 0 \)

For a given value of the Lagrangean multipliers \( \beta \),

Relaxation 1 can be easily solved by determining the first \( H_k \) largest \( (h_{i,k} - \beta_i) \) coefficients for each \( k \), and setting the associated \( X_{i,k} = 1 \). The remaining \( X_{i,k} \)'s are then set to zero [24]. Similarly, for a given value of the Lagrangean multipliers \( \gamma \), Relaxation 2 can be solved as follows. Determine the maximum \( (h_{i,k} - \gamma_k) \) for each \( i \), and set the correspond-
After setting $X_{ik} = 1$ and the remaining $X_{ik}$'s to zero [24].

The subgradient method is again used to update the Lagrange multipliers in solving the following problems:

$$Z_{D41} = \min_{\beta} Z_{D41}(\beta)$$

$$Z_{D42} = \min_{\gamma} Z_{D42}(\gamma)$$

**Relaxation 1.** The subgradient method to update the multipliers $\beta$ is explained below. Given an initial value $\beta^0$, a sequence $\{\beta^p\}$ is generated by the rule

$$\beta_{i}^{p+1} = \beta_{i}^{p} + t^p \left( \sum_{k} X_{ik}^p - 1 \right) \quad \forall i,$$

where $X_{ik}^p$ is an optimal solution to LRPA1, and $t^p$ is a positive scalar step size. The formula used to calculate the step size is:

$$t^p = \frac{\lambda \left( Z_{D41}(\beta^p) - Z_4^* \right)}{\left\| \sum_{k} X_{ik}^p - 1 \right\|^2}$$
where $\lambda_p$ is a scalar satisfying $0 < \lambda_p \leq 2$, and $z^*_4$ is the initial estimate on $Z_{D44}$. The sequence $\{\lambda_p\}$ is determined by setting $\lambda_0 = 2$ and halving it whenever $Z_{D44}(\lambda)$ has failed to decrease in some fixed number of iterations. The method is terminated upon reaching the prespecified iterations limit, or when the step size becomes negligibly small.

The resulting solution may or may not be feasible, however, to the original problem. To obtain a feasible solution from the Lagrangean solution, we proceed as follows. Let $\hat{\chi}_{ik}$ denote an optimal solution to LRPA1. In LRPA1, the constraints

$$\sum_k x_{ik} = 1, \ i = 1, I$$

are dualized and may be violated. Partition $I$ into three subsets defined by

$$S_8 = \{i \mid \sum_k \hat{\chi}_{ik} < 1\}$$

$$S_9 = \{i \mid \sum_k \hat{\chi}_{ik} > 1\}$$

$$S_{10} = \{i \mid \sum_k \hat{\chi}_{ik} = 1\}$$
The constraints of the model which are violated by \( \hat{x}_{i,k} \) correspond to \( i \in S_8 \cup S_9 \). Hence, \( \hat{x}_{i,k} \) has to be modified so that these constraints are satisfied. For all \( i \in S_9 \), simply remove part \( i \) from \( (\sum_{k} \hat{x}_{i,k} - 1) \) groups. The rule is to remove the part from group \( \hat{k} \), where \( h_{i,k}^\hat{=} = \min \{ h_{i,k} \} \) and check if now \( i \in S_{10} \). If not, the procedure is repeated until feasibility is reached. For each part \( i \in S_8 \), insert part \( i \) into group \( \hat{k} \), where \( h_{i,k}^\hat{=} = \max \{ h_{i,k} \} \), without violating the group size constraints.

**Relaxation 2.** In the case of Relaxation 2 (LRPA2), for the subgradient method we have,

\[
\gamma^{p+1}_k = \gamma^p_k + t^p \left( \sum_i X^p_{i,k} - H_k \right) \quad \forall k,
\]

and

\[
t^p = \frac{\lambda_p \left( Z_{D42}(\gamma^p) - Z^*_4 \right)}{\left\| \sum_i X^p_{i,k} - H_k \right\|_2^2}
\]

The procedure to find a feasible solution is as follows.

Let,
\[ S_{11} = \{ k \mid \sum_i \hat{X}_{i,k} > H_k \} \]
\[ S_{12} = \{ i \mid \sum_k \hat{X}_{i,k} = 0 \} \]

For all \( k \in S_{11} \), simply remove \( \left( \sum_i \hat{X}_{i,k} - H_k \right) \) parts from group \( k \). The rule may be to remove the part corresponding to the least \( h_{i,k} \) and continue until feasibility is restored.

This procedure may lead to a solution where a part is not allocated to any of the groups. Hence for each part \( i \in S_{12} \), insert the part into group \( \hat{k} \), where \( h_{i,\hat{k}} = \max \{ h_{i,k} \} \), without violating the group size restrictions.

### 6.1.7 Computational Experience of Model 4

The computational experience of the two Lagrangean relaxations on large, randomly generated versions of Model 4 is presented in this section.

(a) **The effect of the problem size.** In the first set of experiments, the effect of the problem size on the performance of the two relaxations was analyzed. Seven problems of widely different sizes were generated. The number of parts and the number of part groups in each case are different. The part group size in each case is assumed
not to exceed the total number of parts divided by the
number of groups. The values of \( h_{ik} = \sum b_{jk} c_{ij} \) are real
numbers were generated from the uniform distribution \([0,10]\).
The results of this experiment are given in Table 6.7.

From Table 6.7, it appears that the time required to solve
the parts allocation model using Relaxation 2 is less than
that obtained using Relaxation 1. The CPU time is higher in
the case of Relaxation 1 due to the fact that the size of
the subproblems solved is larger than that of the
subproblems of Relaxation 2. In the case of Relaxation 1,
the subproblems are of size \( I \) (i.e., number of parts),
whereas in the case of Relaxation 2, the subproblem size is
only \( K \) (i.e., number of groups). Both Relaxation 1 as well as
Relaxation 2 appear to be computationally efficient since
even the largest problem, involving 2000 parts and 20
groups, took less than 3 seconds to obtain the Lagrangean
solution (upper bound) and the feasible solution (lower
bound). It is also observed that the deviation between the
upper bound and the lower bound as a percentage of upper
bound is zero in the case of Relaxation 2; however, it
varies from 3.6% to 49.9% in the case of Relaxation 1. This
<table>
<thead>
<tr>
<th>No.</th>
<th>No. of parts</th>
<th>No. of groups</th>
<th>Group size</th>
<th>CPU time (m. secs)</th>
<th>$100 \times (UB-LB)/UB$</th>
<th>LB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>50</td>
<td>5</td>
<td>10</td>
<td>16, 13</td>
<td>9.39, 0.0</td>
<td>35.49, 37.42</td>
</tr>
<tr>
<td>2</td>
<td>500</td>
<td>10</td>
<td>50</td>
<td>349, 239</td>
<td>6.12, 0.0</td>
<td>399.55, 400.72</td>
</tr>
<tr>
<td>3</td>
<td>500</td>
<td>20</td>
<td>25</td>
<td>673, 426</td>
<td>49.9, 0.0</td>
<td>425.25, 425.88</td>
</tr>
<tr>
<td>4</td>
<td>1000</td>
<td>10</td>
<td>100</td>
<td>676, 476</td>
<td>6.1, 0.0</td>
<td>799.33, 811.1</td>
</tr>
<tr>
<td>5</td>
<td>1000</td>
<td>20</td>
<td>50</td>
<td>1356, 876</td>
<td>49.9, 0.0</td>
<td>859.55, 850.67</td>
</tr>
<tr>
<td>6</td>
<td>2000</td>
<td>10</td>
<td>200</td>
<td>1406, 953</td>
<td>5.8, 0.0</td>
<td>1602.13, 1613.63</td>
</tr>
<tr>
<td>7</td>
<td>2000</td>
<td>20</td>
<td>100</td>
<td>2679, 1699</td>
<td>3.6, 0.0</td>
<td>1687.22, 1709.19</td>
</tr>
</tbody>
</table>
indicates that Relaxation 2 produces tighter bounds than Relaxation 1. Relaxation 2 also results in lower bounds that are superior to those of Relaxation 1.

(b) The effect of the initial estimate. In the second set of experiments, the effect of the initial estimate \( Z^*_4 \) on the performance of the two relaxations was studied. Six different problems were generated in which the number of parts, the number of groups, and the limit on the number of parts in each group were fixed at 1000, 20, and 50, respectively. The \( h_{ik} \) values were generated from the uniform distribution \([0,10]\). The initial estimate \( Z^*_4 \) in each case is also chosen randomly. The results of this experiment are given in Table 6.8.

From Table 6.8, it appears that the lower bounds obtained in the case of both Relaxation 1 as well as Relaxation 2 are relatively insensitive to the initial estimate. In the case of Relaxation 1, the deviation between the upper bound and the lower bound, as a percentage of the upper bound, appears to be substantial, when \( Z^*_4 \) is overestimated relative to the
Table 6.8. Effect of Initial Estimate ($Z_4^*$)

<table>
<thead>
<tr>
<th>$Z_4^*$</th>
<th>LB</th>
<th>100*(UB-LB)/UB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rel 1</td>
<td>Rel 2</td>
</tr>
<tr>
<td>0</td>
<td>842.06</td>
<td>850.66</td>
</tr>
<tr>
<td>300</td>
<td>842.06</td>
<td>850.66</td>
</tr>
<tr>
<td>600</td>
<td>842.06</td>
<td>850.66</td>
</tr>
<tr>
<td>800</td>
<td>842.06</td>
<td>850.66</td>
</tr>
<tr>
<td>1000</td>
<td>840.17</td>
<td>850.66</td>
</tr>
<tr>
<td>1200</td>
<td>841.19</td>
<td>850.66</td>
</tr>
</tbody>
</table>
lower bound; the deviation, however, decreases when the value of $Z^*_1$ is underestimated.

(c) The effect of the problem structure. This experiment tries to capture the effect of the problem structure on the performance of the two relaxations. The number of parts and the number of groups were fixed at 100 and 5, respectively, and the $h_{i,k}$ values were generated, as before, using the uniform distribution $[0,10]$. The parameter $H_k$ is considered in this experiment; it may be constant or varying. When it is constant, three levels of its values are considered (low, medium, and high), and when it is varying, two levels of its values (low and high) are considered. This results in 5 problems, which are solved using both relaxations. When it is varying, the values are obtained using a discrete uniform distribution. The experimental design used to study the effect of the problem structure is summarized in Table 6.9, and the computational results are given in Table 6.10.
Table 6.9 Experimental Design for Parts Allocation

<table>
<thead>
<tr>
<th>$H_k$</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Constant</td>
<td>low ($g_1$): 20 Med ($g_2$): 24 high ($g_3$): 20</td>
</tr>
<tr>
<td>2. Varying</td>
<td>low ($h_1$): [20, 24] high ($h_2$): [20, 40]</td>
</tr>
</tbody>
</table>
Table 6.10 Effect of Problem Structure

<table>
<thead>
<tr>
<th>$H_k$</th>
<th>LB</th>
<th>100*(UB-LB)/UB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rel 1</td>
<td>Rel 2</td>
</tr>
<tr>
<td>g1</td>
<td>74.04</td>
<td>74.8</td>
</tr>
<tr>
<td>g2</td>
<td>74.6</td>
<td>74.83</td>
</tr>
<tr>
<td>g3</td>
<td>76.1</td>
<td>76.28</td>
</tr>
<tr>
<td>h1</td>
<td>75.2</td>
<td>75.32</td>
</tr>
<tr>
<td>h2</td>
<td>77.2</td>
<td>78.18</td>
</tr>
</tbody>
</table>
From Table 6.10, it can be seen that, in all cases, Relaxation 2 produces slightly better lower bounds than does Relaxation 1. As \( H_k \) increases, the value of the lower bounds obtained increase. This may be due to the fact that this is a maximization problem subject to capacity constraints. Hence, when the capacity is relaxed, it results in an improved solution. In the case of Relaxation 1, the change in group size appears to have some influence on the deviation between the upper bound and the lower bound; but it does not seem to have any pattern.

6.2 SIMULTANEOUS APPROACH

A methodology to simultaneously form the machine-part groups on the basis of the processing requirements of the parts and the processing capabilities of the available machines is presented in this Section. Two non-linear 0-1 integer programming formulations of the machine-part grouping problem are presented. The objective of the first model is to group the parts and machines in such a way that the sum of the compatibility indices of the machines and parts in all the groups is maximized. The objective of the second model is to form the groups while seeking a trade-off
between the cost of duplicating machines and the cost of intercell movement. These formulations take into account the physical constraints on the system such as, limitations on the number of machines and parts in a group, the number of groups, and the available number of machines of each type.

### 6.2.1 Model 5 - Compatibility Maximization

The objective function of this model is formulated in order to maximize the sum of the compatibility indices of all the machines and parts in all the groups:

\[
\text{Max } Z_5 = \sum_i \sum_j \sum_k X_{i,k} Y_{j,k} C_{i,j}
\]  

(6.10)

where \(C_{i,j}\) is defined by either equation (4.2) or (4.3).

The constraints of the model are as follows:

\[
\sum_k X_{i,k} = 1 \quad \forall i
\]  

(6.11)

\[
\sum_j Y_{j,k} \leq G_k \quad \forall k
\]  

(6.12)

\[
\sum_i X_{i,k} \leq H_k \quad \forall k
\]  

(6.13)

\[
\sum_k Y_{j,k} \leq A_j \quad \forall j
\]  

(6.14)

\[
X_{i,k}, Y_{j,k} \in \{0, 1\} \quad \forall (i, j, k)
\]  

(6.15)
Constraint (6.11) ensures that a part is allocated to only one family. There may be some restrictions on the number of parts and the number of machines in a group in order to have good control and supervision over the individual groups. These restrictions are represented by constraints (6.12) and (6.13). In practice, there are restrictions on the availability of each type of machine in the system; this is represented by constraint (6.14). The last constraint ensures integrality of the decision variables.

6.2.2 Model 6 - Cost Trade-off

The objective function of this model seeks a trade-off between the cost of duplicating the machines and the cost of intercell movement. Intercell movement arises if a part in group k requires a tool which is available only on a machine in group r, r≠k. Thus, we have:

$$\text{Min } Z_6 = \sum_{j} \sum_{k} f_{jk} Y_{jk} - \sum_{i} \sum_{j} X_{ik} Y_{jk} T_{ij} c q_i / L_i$$  \hspace{1cm} (6.16)

The constraints of this Model include that of Model 5 (i.e. equations (6.11) to (6.15)) and the following constraint.
\[
\sum_{k} Y_{jk} \geq 1 \quad \forall j \quad (6.17)
\]

For an explanation of the objective function (6.16) and the constraint (6.17) refer to Section 5.2.

6.2.3 Application of the Formulations

The application of the machine–part grouping models (Model 5 and Model 6) is illustrated using an example problem. A manufacturing system with 12 parts and 6 machines is considered. The tooling and the production requirement of the parts are given in Table 6.11. The toolings available on the machines, the annual fixed cost rate of the machines, and the available number of machines of each type are given in Table 6.12. The parts and the machines are to be grouped into three cells. The maximum number of parts in any group is assumed to be 6. The maximum number of machines is 4 in the first group, and 2 each in the other two groups. The average cost of an intercell movement is assumed to be $5 and the size of the unit handling load is assumed to be 5 for all the parts.

The example problem has been formulated and solved using both Model 5 and Model 6 after linearizing (see Appendix
<table>
<thead>
<tr>
<th>Part</th>
<th>Annual demand</th>
<th>Tools required (tool codes assumed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>500</td>
<td>B01, C02, D03, F02, F03</td>
</tr>
<tr>
<td>2</td>
<td>1800</td>
<td>B05, B06, E01, E03, E05</td>
</tr>
<tr>
<td>3</td>
<td>1500</td>
<td>B03, D01, E05, F01, F06</td>
</tr>
<tr>
<td>4</td>
<td>1200</td>
<td>B01, B02, C01, D02</td>
</tr>
<tr>
<td>5</td>
<td>2000</td>
<td>A01, A05, A06, F01, F07, F08</td>
</tr>
<tr>
<td>6</td>
<td>1600</td>
<td>B01, B03, B06, E02, E04, E06</td>
</tr>
<tr>
<td>7</td>
<td>650</td>
<td>A03, B02, B04, C01, F05</td>
</tr>
<tr>
<td>8</td>
<td>1350</td>
<td>C03, E01, E02, E04, E06</td>
</tr>
<tr>
<td>9</td>
<td>1200</td>
<td>A01, A02, A06, D04</td>
</tr>
<tr>
<td>10</td>
<td>1100</td>
<td>B03, C02, D03, F02, F07</td>
</tr>
<tr>
<td>11</td>
<td>900</td>
<td>B03, B04, C01, D02, F02, F03, F04</td>
</tr>
<tr>
<td>12</td>
<td>1650</td>
<td>A03, A04, A06</td>
</tr>
<tr>
<td>Machine</td>
<td>Number available</td>
<td>Annual fixed cost rate</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>10,000</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>8,000</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>6,000</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>7,000</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>9,000</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>9,000</td>
</tr>
</tbody>
</table>
A.2), and the resulting solutions are given in Table 6.13. From Table 6.13, it can be seen that Model 5 uses up all the available machines; however, Model 6 does not utilize all the available machines; only one unit of machine 6 is used. The additional unit is not allocated to any of the groups since the allocation is not cost effective.

6.2.4 Solution Procedure for Large Size Problems

For small problem sizes, commercially available integer programming codes may be used once the models are linearized. In the case of large versions of the models, this approach is not feasible for the reasons that are explained earlier. Instead, an approximate iterative solution procedure has been developed making use of the special structure of the models. This procedure partitions the original, non-linear model into two sub-models, one for machine allocation and the other for parts allocation.

In this Section, the solution procedure developed for Model 5, is presented. The procedure to solve Model 6 is essentially the same, noting that constraint (6.17) of Model 6 is taken into account while making the Lagrangean solution of the machine allocation sub-model feasible.
Table 6.13 Results of the Example Problem

<table>
<thead>
<tr>
<th>Groups</th>
<th>Model 5</th>
<th>Model 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. parts</td>
<td>1,3,4,7,10,11</td>
<td>1,3,4,7,10,11</td>
</tr>
<tr>
<td>machines</td>
<td>2,3,4,6</td>
<td>2,3,4,6</td>
</tr>
<tr>
<td>2. parts</td>
<td>2,6,8</td>
<td>2,6,8</td>
</tr>
<tr>
<td>machines</td>
<td>2,5</td>
<td>2,5</td>
</tr>
<tr>
<td>3. parts</td>
<td>5,9,12</td>
<td>5,9,12</td>
</tr>
<tr>
<td>machines</td>
<td>1,6</td>
<td>1</td>
</tr>
</tbody>
</table>
For a given initial feasible parts allocation \( \bar{x}_{i,k} \), the machine allocation sub-model (MA) is given below:

\[
\text{Max } Z_7 = \sum_i \sum_j \sum_k \bar{x}_{i,k} y_{j,k} c_{i,j} \\
\text{subject to}
\]

\[
\sum_j y_{j,k} \leq G_k \quad \forall k \tag{6.12}
\]

\[
\sum_k y_{j,k} \leq A_j \quad \forall j \tag{6.14}
\]

\[
y_{j,k} \in \{0,1\} \quad \forall (j,k) \tag{6.19}
\]

Let the solution to the machine allocation sub-model be \( \bar{y}_{j,k} \).

The corresponding parts allocation sub-model (PA) is then given below:

\[
\text{Max } Z_8 = \sum_i \sum_j \sum_k \bar{x}_{i,k} \bar{y}_{j,k} c_{i,j} \\
\text{subject to}
\]

\[
\sum_k x_{i,k} = 1 \quad \forall i \tag{6.11}
\]

\[
\sum_i x_{i,k} \leq H_k \quad \forall k \tag{6.13}
\]

\[
x_{i,k} \in \{0,1\} \quad \forall (i,k) \tag{6.21}
\]
Starting with a chosen feasible parts allocation, the machine allocation sub-model (MA) is solved; the results are then used to resolve the parts allocation sub-model (PA), and the procedure is repeated. The iterations continue until sufficient stability in the solutions results. The machine allocation and the parts allocation submodels may be solved using the Lagrangean relaxation methods outlined in Sections 5.4 and 6.1.6, respectively.

6.2.5 Computational Experience

The iterative solution procedure was applied to randomly generated test problems to study: (a) the effect of the initial parts allocation on the final solution, and (b) its performance in the case of large problems. The machine allocation submodel and the parts allocation submodel were solved using the Lagrangean relaxation procedures (Relaxation 2) discussed in Sections 5.4 and 6.1.6, respectively.

(a) The effect of the initial allocation of parts. In the first set of experiments, the effect of the initial allocation of parts was analyzed. The number of parts, the number of machines, and the number of groups were fixed at
12, 6, and 3, respectively, and only one unit of each machine type was assumed to be available. The limit on the number of parts in each group and the number of machines in each group were fixed at 4 and 2, respectively, and the values of the compatibility indices were generated from the uniform distribution [0, 1]. One problem was generated, and was solved 6 times using the iterative procedure, each time with a different initial parts allocation. The 6 different schemes used for the initial allocation of parts are given in Table 6.14. The results of this experiment are given in Table 6.15.

From Table 6.15, it is observed that irrespective of the initial parts allocation scheme the procedure converges almost to the same final solution. However, the time required to solve the problem varies depending upon the scheme chosen. This is due to the variation in the number of iterations needed for the procedure to converge.

(b) Performance on large problems. In the second set of experiments, the performance of the iterative procedure on large problems was studied. The procedure was applied to
<table>
<thead>
<tr>
<th>Scheme</th>
<th>group 1</th>
<th>group 2</th>
<th>group 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1,2,3 and 4</td>
<td>5,6,7 and 8</td>
<td>9,10,11 and 12</td>
</tr>
<tr>
<td>2</td>
<td>1,2,3 and 4</td>
<td>9,10,11 and 12</td>
<td>5,6,7 and 8</td>
</tr>
<tr>
<td>3</td>
<td>5,6,7 and 8</td>
<td>1,2,3 and 4</td>
<td>9,10,11 and 12</td>
</tr>
<tr>
<td>4</td>
<td>5,6,7 and 8</td>
<td>9,10,11 and 12</td>
<td>1,2,3 and 4</td>
</tr>
<tr>
<td>5</td>
<td>9,10,11 and 12</td>
<td>1,2,3 and 4</td>
<td>5,6,7 and 8</td>
</tr>
<tr>
<td>6</td>
<td>9,10,11 and 12</td>
<td>5,6,7 and 8</td>
<td>1,2,3 and 4</td>
</tr>
</tbody>
</table>
Table 6.15 The Effect on Final Solution

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Objective function value</th>
<th>CPU time (m. sec)</th>
<th>No. of iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8.09</td>
<td>59</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>8.09</td>
<td>60</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>8.18</td>
<td>69</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>8.15</td>
<td>56</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>8.15</td>
<td>58</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>8.18</td>
<td>68</td>
<td>4</td>
</tr>
</tbody>
</table>
four randomly generated problems of different sizes. The compatibility indices were generated from the uniform distribution $[0,1]$ and only one unit of each machine type was assumed to be available. The limit on the number of parts in each group was based on the number of parts and the number of groups, and the limit on the number of machines in each group was based on the number of machines and the number of groups. The results of this experiment are given in Table 6.16.

From Table 6.16, it can be seen that the number of iterations and the CPU time increase significantly with the size of the problem. The CPU time required to solve this Model appears to be high; for instance the largest problem, involving 1000 parts and 100 machines, took nearly one hour to reach the final solution. The CPU times are high due to the fact that the procedure involves the formulation of the submodels, as well as computing the corresponding objective function coefficients at every iteration.
Table 6.16 Performance on Large Problems

<table>
<thead>
<tr>
<th>Problem No.</th>
<th>Problem size</th>
<th>No. of iterations</th>
<th>CPU time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>500</td>
<td>25</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>500</td>
<td>50</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>1000</td>
<td>50</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>1000</td>
<td>100</td>
<td>10</td>
</tr>
</tbody>
</table>
CHAPTER VII

NEW PARTS ALLOCATION PROBLEM

The problem of allocating new parts to existing groups can be solved using any one of the following methods:

(a) The parts allocation model discussed in section 6.1.2 could be used to assign new parts to existing groups.

(b) The concept of statistical classification could be used to develop allocation schemes to assign new parts to existing groups.

This Chapter reports on the development of a model for part family identification which is based on statistical classification concepts.

7.1 Approach to the Problem

It is assumed that the family membership for the existing m-dimensional parts is known. The m dimensions may be the attributes that characterize a part. Based on the concept of single function classification [27], a part family
identification model (Model 7) is formulated to develop a weighting scheme as a linear function of the attributes of the parts. Using the linear weighting scheme a part is characterized by a point on the real line; by extension, a part family is represented by an interval on the line. The weights of the attributes and the family membership intervals are obtained from the solution to Model 7. The weighting scheme may then be applied to new parts in order to determine their family membership.

The problem may be recast more formally as follows. Given the families and the set of parts in each family, the objective is to find the functional form of the weighting scheme and the appropriate intervals for each family using the attributes of the parts so that the number of proper allocations is maximized. The proposed procedure mathematically combines the attributes of the parts into a single linear function, which can then be used to appropriately assign new parts to the existing families.
7.2 Model 7 - Part Family Identification

We proceed with the mathematical formulation of the model as follows.

(a) Assuming there are J machines in the cellular manufac-
turing system, we define the attributes of a part i as the number of common tools between the part and each of the J machines, i.e., $T_{ij}, j=1, \ldots, J$. Thus each part is represented by an $j$-dimensional vector $(T_{i1}, T_{i2}, \ldots, T_{ij})$.

(b) Part $i$ which is known to be in family $k$, is properly classified if its functional value falls between the lower ($D_k$) and upper ($B_k$) limits of family $k$. This is ensured by the following set of constraints.

$$\sum_{j} T_{ij} Y_j - \delta \geq D_k \quad i \in I_k, \forall k \quad (7.1)$$

$$\sum_{j} T_{ij} Y_j + \delta \leq B_k \quad i \in I_k, \forall k \quad (7.2)$$

where $\delta$ is a constant which is introduced to ensure that the functional values of the parts fall well within the limits of the interval of family $k$. The $D_k$ and $B_k$ values define $K$ segments on the real line.
(c) The following constraints ensure that the length of the interval for any family \( k \) is at least an amount \( m_k \):

\[
B_k - D_k \geq m_k \quad \forall k \tag{7.3}
\]

(d) Additional constraints are needed to minimize the overlap 'r' between any two consecutive families \( k \) and \( h \) on the real line:

\[
D_k - B_h + M \varepsilon_{hk} \geq r \quad \forall (h,k) \tag{7.4}
\]

\[
D_h - B_k - M (\varepsilon_{hk} - 1) \geq r
\]

where \( M \) is a large positive integer and the auxiliary 0-1 integer variables, \( \varepsilon_{hk} \), are introduced to ensure that only one of the above constraints is met. This set of constraints are needed for all the \( K(K-1)/2 \) combinations of families.

(e) A solution that maximizes the value of \( r \) would minimize the overlap between any two consecutive families; the objective function, therefore, is formulated as follows:

\[
\max Z_0 = r \tag{7.5}
\]

The problem is now to maximize the objective function, equation (7.5), subject to the constraints (7.1) through

- 140 -
This is a mixed integer programming model with $(J+2K+1)$ continuous variables, $K(K-1)/2$ 0-1 integer variables, and $(2I+K^2)$ constraints.

The solution to the model will yield the values of the coefficients of the weighting scheme ($Y_j$'s) and the interval limits of the families ($D_k$'s and $B_k$'s). The resulting weighting scheme is then given as:

$$WS = \sum T_{i,j} Y_j$$

Any new part $x$ may now be assigned to the appropriate family, by identifying the $T_{x,j}$ values and determining the interval in which $WS = \sum T_{x,j} Y_j$ falls.

7.3 Application of the Formulation

The application of Model 7 is illustrated using an example problem. A cellular manufacturing system with 25 parts and 10 machines is considered. The tools available on the machines and the tooling requirements of the parts are given in Tables 5.1 and 5.2, respectively. Based on their tooling similarities, the parts are assumed to have
been grouped into three families as follows.

Family 1: Parts 3, 5, 8, 9, 13, 18, 19, 23, 24, 25

Family 2: Parts 2, 6, 7, 11, 15, 16, 21, 22

Family 3: Parts 1, 4, 10, 12, 14, 17, 20

The problem has the following dimensions:

No. of 0-1 integer variables = 3

No. of real variables = 17

No. of constraints = 59

The problem has been formulated and solved as Model 7 with

\[ M = 100, \ \delta = 1.0, \ m_1 = 1.1, \ m_2 = 1.2, \text{ and } m_3 = 1.3. \] Based on

the \( Y_j \) values in the solution to the model, the resulting

weighting scheme is as follows:

\[ WS = -7.33 T_{i1} -0.113 T_{i2} -2.406 T_{i3} -2.406 T_{i4} +2.406 T_{i5} + \]

\[ 9.737 T_{i5} -1.917 T_{i7} -4.812 T_{i8} -3.609 T_{i9} -8.645 T_{i10} \]

The scheme results in the following three intervals on the

real line corresponding to the three families:

Interval for family 1: [-11.03, 6.827]

Interval for family 2: [-73.932, -58.85]

Interval for family 3: [-39.609, -30.278]
Any new part may now be assigned to the appropriate family using this allocation scheme. For instance, consider a new part (part #26) with the following tooling requirements.

\[ A03, A04, A05, B03, C04, C05, F01, M02, M04, R03, R04, R05, R06 \]

The attributes of the new part (i.e., the number of common tools between the part and the 10 machines) are as follows:

<table>
<thead>
<tr>
<th>j</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_{26,j}$</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

Using the weighting scheme, we obtain:

\[ WS = \sum T_{26,j} Y_j = -58.98, \]

which falls in the interval for family 2, indicating the family membership for part #26.

### 7.4 Some Observations

The weighting (allocation) scheme presented in this Chapter employs the tooling requirements of the parts and the toolings available on the machines to develop family membership intervals on the real line. New parts may then be
allocated to appropriate families if their tooling requirements are known. Should a new part fail to correspond to any of the family intervals, it indicates that the part does not fit into the existing part family configuration, and therefore, redesigning or subcontracting the part may be considered.

In Model 7, the constraints are formulated to ensure proper classification of all the parts, and the objective function is such that it minimizes the overlap between any two consecutive families; however, in the single function classification model, the constraints are formulated to avoid overlap between any two families, and the objective function maximizes the number of proper classifications. In order to achieve this objective, corresponding to each part a 0-1 variable is introduced in constraints (7.1) and (7.2) and the sum of the 0-1 variables corresponding to all the parts is minimized. When a solution can be found with the 0-1 variable corresponding to a part equal to zero, that part is properly classified. In Model 7, the number of 0-1 variables is dependent on the number of families, while in the single
function classification model it is dependent on the number of parts.

The modifications made to the single function classification model reduces the number of 0-1 integer variables in the model to a considerable extent. For instance, if the number of parts, machines, and families are 1000, 100, and 10, respectively, then the single function classification model will result in 1090 0-1 integer variables, 120 continuous variables, and 2145 constraints, while the proposed model will result in only 45 0-1 integer variables, 121 continuous variables and 2100 constraints, making it possible to use commercially available mixed-integer programming codes to solve the model.
CHAPTER VIII

MACHINE-PART GROUPING MODEL - AN EXTENSION

In this chapter, the problem of machine-part grouping in the presence of alternate process plans for parts is addressed. The model developed in Chapter VI (Model 5) to simultaneously form the machine-part groups has been extended to take into account the presence of alternate process plans. In practice, it is possible to have more than one process plan for a part [44]. Each process plan for a part is defined in terms of tooling requirements to manufacture that part. The capability of a machine to process parts is expressed in terms of the tools available on that machine. Since a given tool may be available on more than one machine, an operation of a part may be performed on alternate machines. A non-linear 0-1 integer programming model, based on the compatibility between the process plan of a part and a machine has
been developed to select a process plan for each part and to form the machine-part groups.

8.1 Formulation of the Model (Model 8)

The model is formulated under the following assumptions:

1. For each part, the tooling requirements are known for each process plan.

2. The toolings available on each machine type and the available number of machines of each type are known.

The objective function of the Model is formulated in order to maximize the compatibility indices of all machines and process plans of all parts in all groups.

\[
\text{Max } Z_{10} = \sum_{i} \sum_{j} \sum_{k} X_{ipk} Y_{jk} C_{ipj} \tag{8.1}
\]

where, \( C_{ipj} \) is an index which expresses the compatibility of a machine with a process plan of the part in terms of the processing requirements of the part, and is defined as follows:

\[
C_{ipj} = \frac{T_{ipj}}{\min_{k} \left( \frac{N_{ipj}}{N_{hip}} \right)} \tag{8.2}
\]
A large number of common tools between a process plan of a part and a machine implies less intercell movement if that process plan is selected for the part and if the part and machine are placed in the same group.

The constraints of the model are as follows:

\[ \sum_{p} \sum_{k} X_{i,p,k} = 1 \quad \forall i \quad (8.3) \]

\[ \sum_{i} \sum_{p} X_{i,p,k} \leq H_{k} \quad \forall k \quad (8.4) \]

\[ \sum_{k} Y_{j,k} \leq A_{j} \quad \forall j \quad (8.5) \]

\[ \sum_{j} Y_{j,k} \leq G_{k} \quad \forall k \quad (8.6) \]

\[ X_{i,p,k}, Y_{j,k} \in \{0,1\} \quad \forall (i,j,k,p) \quad (8.7) \]

Constraint (8.3) ensures that only one process plan is selected for each part and a part belongs to only one group. In practice, there are restrictions on the availability of the number of machines of each type in the system; this is represented by constraint (8.5). There may be restrictions on the number of parts and the number of
machines in a group in order to have good control and supervision over the individual groups. These are represented by constraints (8.4) and (8.6), respectively. The last constraint ensures integrality of the decision variables.

8.2 Solution Procedure

For small problem sizes, commercially available integer programming (IP) codes can be used once the models are linearized (see Appendix A.2). In the case of large manufacturing systems, the model may be solved using the iterative procedure discussed in Section 6.2.4, once it is partitioned into the following submodels.

For a given initial feasible parts allocation \( \bar{X}_{ipk} \), the machine allocation model (MA1) is given below:

\[
\text{Max } Z_{11} = \sum_{i} \sum_{p} \sum_{j} \sum_{k} X_{ipk} Y_{jk} G_{lpj} \\
\text{subject to} \\
\sum_{k} Y_{jk} \leq A_j \quad \forall j \\
\sum_{j} Y_{jk} \leq G_k \quad \forall k
\]  
(8.8)  
(8.5)  
(8.6)
\[ Y_{jk} \in \{0,1\} \quad \forall (j,k) \quad (8.9) \]

Let the solution to the machine allocation model be \( Y_{jk} \). The corresponding parts allocation model (PA1) is then formed as follows:

\[
\text{Max } Z_{12} = \sum_{i} \sum_{p} \sum_{j} X_{i,p,k} Y_{jk} C_{i,p,j} \quad (8.10)
\]

subject to

\[
\sum_{p} \sum_{k} X_{i,p,k} = 1 \quad \forall i \quad (8.3)
\]

\[
\sum_{i} \sum_{p} X_{i,p,k} \leq H_{k} \quad \forall k \quad (8.4)
\]

\[
X_{i,p,k} \in \{0,1\} \quad \forall (i,p,k) \quad (8.11)
\]

Models (MA1) and (PA1) may be solved using the Lagrangean relaxation based methods outlined in Sections 5.4 and 6.1.6.

9.3 Application of the Formulation

The application of the formulation is illustrated using an example problem. A manufacturing system with 15 parts, 10 machines, and 3 groups is considered. The process plans for each part and the corresponding tooling requirements are given in Table 8.1. The toolings available on the machines
### Table 8.1 Process Plans and Tooling Requirements of the Parts

<table>
<thead>
<tr>
<th>Part</th>
<th>Process plan number</th>
<th>Tools required (tool codes assumed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>B02, D01, D02, D03, D04, H01, H02, H03, M01, M02</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A01, D01, D02, D03, H01, H02, M01, M02, M03, M04</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>A02, A03, A04, B01, C01, C02, M01, M05, R01, R02, R03, R04</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A01, A02, A05, A06, C04, M02, M03, M04, R05, R06</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>A01, B01, B02, B03, B04, E02, E03, E04, F01, F02, F03, F04, G05, R02, R03, R04, R06</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B01, E02, E03, E04, E05, F04, F05, G01, G02, H03, R01, R03, R04</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B01, B03, E05, E06, F03, F04, F05, G01, R01, R02, R04</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>A02, A04, A06, C02, C03, C04, C05, F01, G01, M01, M03, M05, R02, R04, R05, R06</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>A01, D01, D02, H01, H02, M01, M02, M03</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>D01, D03, D04, E07, H01, H02, H03, H04, M02, M03</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>A01, A03, A06, C02, C03, C04, G06, M04, M05, R01, R02, R04, R06</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>A01, B02, B03, B05, B06, E01, E03, E05, E06, E07, F05, G01, G02, R01, R02</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>B01, C01, E01, E03, E04, E07, F01, F05, G03, G04, G05, G06, R01, R03, R06</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>B01, B03, D01, E01, E03, E05, F01, F03, F04, G01, G03, R01, R02, R04, R06</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>C01, D02, D03, D04, H02, H03, H04, M04, M05</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>A02, A04, A06, C01, C02, C03, C05, E06, G05, M01, M06, R01, R02, R04, R05</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>A01, A02, C01, C03, C04, E01, M01, M03, M04, M05, R01, R02, R03</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>A01, A02, A03, A05, C02, C03, H01, M01, M02, M03, M04, R01, R06</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>A01, B01, B02, B03, C01, E01, E02, F01, F02, F03, F05, G01, G02, H01, R01, R02, R03</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>B01, B03, B04, C01, C02, E01, F01, F02, F03, H01, R01, R02, R03, R04</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>A01, D05, D06, F06, H06, H03, H04, M01, M03, M04, M05</td>
</tr>
</tbody>
</table>

- 151 -
<table>
<thead>
<tr>
<th>Part</th>
<th>Process plan number</th>
<th>Tools required (tool codes assumed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>1</td>
<td>A05, D03, D04, D05, D06, H01, H02, H04, M04, M05</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A01, A02, C01, C02, C04, H01, M01, M02, R01, R02, R03</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>B01, B02, E02, E04, E05, F01, F03, G02, R01, R02, R04, R06</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>B01, B03, B04, C01, D02, E02, E03, F01, F03, F04, G01, G02, G04, R01, R03</td>
</tr>
</tbody>
</table>
and the available number of machines of each type are given in Table 5.2. The number of parts in a group is restricted to 6, and the number of machines to 4. The problem was formulated as Model 8 and solved using the iterative procedure. The resulting machine-part groups and the process plans are given in Table 8.2.
Table 8.2  Results of the Example Problem (Model 8)

<table>
<thead>
<tr>
<th>Group k</th>
<th>Machine</th>
<th>Part i</th>
<th>Process plan p</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1,4,8,9</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2,5,6,7,10</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1,3,9,10</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
CHAPTER IX

CONCLUSIONS

In this work, we have developed mathematical models of the following cell formation problems in flexible manufacturing systems:

(a) Machine allocation problem, (b) Machine-part grouping problem, and (c) New parts allocation problem.

In addition, methodologies have been devised to solve large instances of the models developed.

The problem of allocating machines to part families has been formulated as 0-1 integer programming models. The machine-part grouping problem has been approached from two viewpoints. The first approach considers the machine group formation problem first and then the problem of allocating parts to machine groups. Each of these problems has been formulated as a 0-1 integer programming problem. The second approach considers simultaneous
grouping of parts and machines, and results in non-linear 0-1 integer programming models.

The cell formation models developed take into consideration the physical constraints of the system (such as restrictions on the number of machines in a group, on the number of parts in a group, and on the available number of machines of a particular type), as well as processing requirements, production requirements, and processing times.

An improved formulation of the single function classification model has been used to develop allocation schemes to allocate new parts to the existing part families.

The formulations presented here may also be used under the restrictive assumption that each operation of a part is performed on only one machine, by defining the similarity/compatibility indices appropriately.

9.1 Contributions of the Dissertation

The contributions of the Dissertation towards the cell
formation problems in flexible manufacturing systems are as follows:

(i) Defining a new index to express the "similarity" between two machines on the basis of their capabilities to process the set of parts under consideration.

(ii) Development of indices to express the "compatibility" between a part and a machine on the basis of the processing requirements of the parts and the capabilities of the machines.

(iii) New formulations of the cell formation problems, namely, machine allocation problem, machine-part grouping problem, and parts allocation problem.

(iv) An improved formulation of the single function classification model to develop part allocation scheme.

(v) Development of solution methodologies to solve large scale cell formation problems efficiently.

(vi) An extension of the simultaneous machine-part grouping formulation to account for the presence of alternate process plans for parts.
9.2 Directions for Future Research

(1) The methodology suggested in this research to solve the cell formation problems is based on certain assumptions. Hence, efforts could be made to extend the methodology to consider the following situations:

(a) The number of machine groups is not equal to the number of part families.

(b) Capacity restrictions on the machines to process the parts under consideration.

(c) Other objectives, such as maximizing machine utilization, minimizing tool change over, etc.

(2) The problem of selecting the number and the size of pallets while forming the groups may be an interesting area to investigate.

(3) The problem of process plan selection and machine-part grouping to minimize the total investment and operating cost may be worth investigating.
REFERENCES


(29) T.A. Gongaware and I. Ham, "Cluster analysis applications for GT manufacturing systems", Proceedings: IX North


(50) R. Meenakshisundaram and S. S. Fu, "Group technology cell
formation: some new insights", Computers and Industrial

(51) J. M. Mulvey and H. P. Crowder, "Cluster analysis:
An application of Lagrangean relaxation", Management

(52) H. Opitz, "A Classification System to Describe Work-

(53) G. F. K. Purcheck, "Combinatorial grouping - a lattice
theoretic method for the design of manufacturing sys-

(54) G. F. K. Purcheck, "A mathematical classification as a
basis for the design of Group technology production

(55) G. F. K. Purcheck, "A linear programming method for the
combinatorial grouping of an incomplete power set",

(56) G. F. K. Purcheck, "Machine-component group formation: A
heuristic method for flexible production cells and
FMSs", International Journal of Production Research,

(57) R. Rajagopalan and J. L. Batra, "Design of cellular pro-
duction systems: A graph-theoretic approach", Interna-
tional Journal of Production Research, 13(6), 567-579
(1975).

(58) K. Raja Gunasingh and R. S. Lashkari, "Mathematical mod-
ing of the machine grouping problem in cellular manu-
facturing systems", Proceedings: III International Con-
ference on CAD/CAM/Robotics and The Factories of the

(59) R. Rodriguez and O. Adaniya, "Group Technology Cell
allocation", Proceedings: Fall Industrial Engineering

- 164 -


APPENDICES
Appendix A.1

The Lagrangean Relaxation Method
The Lagrangean Relaxation Method

Lagrangean relaxation is a technique that is suitable for solving large scale mathematical programming problems. It is based upon the observation that many difficult integer programming problems can be modelled as a relatively easy problem complicated by a set of side constraints. To exploit this observation, a Lagrangean problem is created in which the complicating side constraints are replaced with a penalty term in the objective function involving the amount of violation of the constraints. The Lagrangean problem is easy to solve and provides an upper bound (for a maximization problem) on the optimal value of the original problem.

There are three major issues in designing a Lagrangean relaxation based method: (1) selection of constraints that should be relaxed, (2) computation of good multipliers, and (3) computation of good feasible solution to the original problem, given a solution to the relaxed problem.
1.0 Basic Constructions

Consider the following integer program.

\[ Z = \min cx \]
\[ \text{s.t. } Ax = b \]
\[ Dx \leq e \]
\[ x \geq 0 \text{ and integer.} \]

where \( x \) is \( nx1 \), \( b \) is \( mx1 \), \( e \) is \( kx1 \) and all other matrices have conformable dimensions. Let \((LP)\) denote problem \((P)\) with the integrality constraint on \( x \) relaxed, and let \( Z_{LP} \) denote the optimal value of \((LP)\). It is assumed that the constraints of \((P)\) have been partitioned into the two sets \( Ax = b \) and \( Dx \leq e \) so as to make it easy to solve \((easy relative to \((P))\) the Lagrangean problem.

\[ Z_D(u) = \min cx + u(Ax - b) \]
\[ \text{s.t. } Dx \leq e \]
\[ x \geq 0 \text{ and integer} \]

where \( u = (u_1, \ldots, u_m) \) is a vector of Lagrangian multipliers.

Assume that \((P)\) is feasible and that the set \( X = \{ x : Dx \leq e, x \geq 0 \text{ and integer} \} \) of feasible solutions to \((LR_u)\) is finite.

Then \( Z_D(u) \) is finite for all \( u \). It is easy to show that \( Z_D(u) \leq Z \) by assuming an optimal solution \( \hat{x} \) to \((P)\) and observing that
\[ Z_D(u) \leq c^\hat{x} + u(A^\hat{x} - b) = Z \]

The inequality in the above relation follows from the definition of \( Z_D(u) \) and the equality from \( Z = c^\hat{x} \) and \( A^\hat{x} - b = 0 \).

2.0 Determination of Lagrangean Multipliers

The best choice for \( u \) would be an optimal solution to the following dual problem:

\[
Z_D = \max_u Z_D(u) \quad (D)
\]

Most schemes for determining \( u \) have as their objective finding optimal or near optimal solutions to (D). Problem (D) has a number of structural properties that make it feasible to solve. \( Z_D(u) \) is the lower envelope of a finite family of linear functions. The function \( Z_D(u) \) has all the nice properties, like continuity and concavity, that make it easy to solve using hill climbing algorithm, except one - differentiability. The function is non-differentiable at any \( u \) where problem \( (LR_u) \) has multiple optima. Although it is differentiable almost everywhere, it is generally non-differentiable at an optimal point [24].
The following three approaches [24] are popular in solving (D) in Lagrangean relaxation applications: (1) the subgradient method, (2) the boxstep method, and (3) multiplier adjustment methods.

The subgradient method is an adaptation of the gradient method in which gradients are replaced by subgradients [24].

An $m$-vector $y$ is called a subgradient of $Z_D(u)$ at $\tilde{u}$, if it satisfies

$$Z_D(u) \leq Z_D(\tilde{u}) + y(u - \tilde{u}), \text{ for all } u.$$ 

It is apparent that $Z_D(u)$ is subdifferentiable everywhere. The vector $(Ax^k - b)$ is a subgradient at any $u$ for which $x^k$ solves $(LR_u)$. Given an initial value $u_0$, a sequence $(u_k)$ is generated by the rule

$$u_{k+1} = u_k + t_k (Ax^k - b)$$

where $x^k$ is an optimal solution to $(LR_u^k)$ and $t_k$ is a positive scalar step size. The step size used most commonly in practice is

$$t_k = \frac{\lambda_k (Z^* - Z_D(u_k))}{\|Ax^k - b\|^2}$$

- 172 -
where $\lambda_k$ is a scalar satisfying $0 < \lambda_k \leq 2$ and $Z^*$ is an upper bound on $Z_D$. Often the sequence $\lambda_k$ is determined by setting $\lambda_0 = 2$ and halving whenever $Z_D(u)$ has failed to increase in a fixed number of iterations. The sufficient condition for optimal convergence comes from the fundamental theoretical result that

$$Z_D(u_k) \rightarrow Z_D \text{ if } t_k \rightarrow 0 \text{ and } \sum_{i=1}^{k} t_i \rightarrow \infty.$$ 

This rule has performed well empirically, even though it is not guaranteed to satisfy the sufficient condition. Unless an $u_k$ is obtained for which $Z_D(u_k)$ equals the cost of a known feasible solution, there is no way of proving optimality in the subgradient method. To resolve this difficulty, the method is terminated upon reaching an arbitrary iteration limit or when the step size almost equals zero. The choice of step size is an area which is not perfectly understood.

Another class of algorithms [25] for solving (D) is based on applying a variant of simplex method to solve (P). However, this approach is known to converge very slowly and does not produce monotonically increasing lower bounds. Boxstep
method is a modification of the simplex based methods. Beginning at given \( u_0 \), a sequence \( \{u_k\} \) is generated. To obtain \( u_{k+1} \) from \( u_k \), the dual of \( (D) \) is solved with the additional requirement that \( \|u^i - u^i_k\| \leq \delta \) for some fixed positive \( \delta \). Let \( \hat{u}_k \) denote the optimal solution to this problem. If \( \|\hat{u}_k^i - u_k^i\| < \delta \) for all \( i \) then \( \hat{u}_k \) is optimal in \( (D) \). Otherwise set \( u_{k+1} = u_k + t_k(\hat{u}_k - u_k) \), where \( t_k \) is the scalar that solves

\[
\text{Max } Z_D \left( u_k + t_k(\hat{u}_k - u_k) \right).
\]

This line search problem is solved by Fibonacci methods. The Boxstep method (in general the simplex based methods) is harder to program and have not performed quite so well computationally as the subgradient method.

The third approach, multiplier adjustment methods, are specialized algorithms for \( (D) \) that exploit the special structure of a particular formulation [26]. In these methods, a sequence \( u_k \) is generated by the rule \( u_{k+1} = u_k + t_k d_k \), where \( t_k \) is a positive scalar and \( d_k \) is a direction.
To determine $d_k$, a small set of finite primitive directions (S) are defined to evaluate the directional derivative of $Z_D(u)$. Usually, directions in S involve changes in only one or two multipliers. For directions in S, it should be easy to determine the directional derivative of $Z_D(u)$. Directions in S are scanned in a fixed order and $d_k$ is taken to be either the first direction found along which $Z_D(u)$ increases, or the direction of steepest ascent within S. The step size $t_k$ can be chosen either to maximize $Z_D(u_k + t_k d_k)$ or to go to the first point at which the directional derivative changes. If S contains no improving direction, the procedure is terminated. The set S should be manageably small, but still include directions that allow ascent to at least a near optimal solution.

1.3 Selecting a Relaxation

Two properties are important in evaluating a relaxation: the sharpness of the bounds produced and the amount of computation required to obtain these bounds. Usually selecting a relaxation involves a trade-off between these two properties. It is usually possible to at least compare
the bounds and computational requirements for different relaxations of the same model. For a given model, the bound obtained from a relaxation that does not possess integrality property is always better than the bound obtained from a relaxation that possesses integrality property. In the case of a relaxation that possesses the integrality property, the bound will only be as good as the bound obtained using the LP relaxation of the model.

1.4 Feasibility and Optimality

It is possible in the course of solving (D) that a solution to \( LR_u \) will be discovered that is feasible in \( P \). If the dualised constraints are equalities, this solution is also optimal for \( P \). If the dualised constraints are inequalities, a Lagrangean solution may be feasible but non-optimal for \( P \). It is rare that a feasible solution of either type is discovered. On the other hand, it often happens that a solution to \( LR_u \) obtained while optimizing (D) will be nearly feasible for \( P \) and can be made feasible with some judicious tinkering.
APPENDIX A.2

Linearizing Nonlinear Terms
in 0-1 Integer Programs
Linearizing Nonlinear Terms in 0-1 Integer Programs

The linearization method due to Glover and Woolsey [28] is explained below. Consider the product term $X_{ik}Y_{jk}$, where both $X_{ik}$ and $Y_{jk}$ are 0-1 integer variables. Each product term $X_{ik}Y_{jk}$ is now replaced by a continuous linearization variable $Z_{ijk}$ with the addition of the following constraints.

\[
\begin{align*}
X_{ik} + Y_{jk} - Z_{ijk} & \leq 1 \\
Z_{ijk} & \leq X_{ik} \\
Z_{ijk} & \leq Y_{jk} \\
X_{ik}, Y_{jk} & \in \{0, 1\} \quad \text{and} \quad Z_{ijk} \geq 0
\end{align*}
\]

The above constraints force the variable $Z_{ijk}$ to assume 0-1 values. Thus, linearization of each product term results in an additional continuous variable and four additional constraints.
Appendix A.3

Program Listings
PROGRAM LISTINGS

MACHINE ALLOCATION PROBLEM (MODELS 1 AND 2)
(RELAXATION 1)

C * LAGRANGEAN RELAXATION METHOD TO SOLVE THE MACHINE ALLOCATION
C * PROBLEM USING REGULAR SUBGRADIENT
C * CONSTRAINT ON MACHINE AVAILABILITY IS RELAXED
C *
MAIN PROGRAM
SUBROUTINE MALAG(M,N,C,NMA,NMG,KLIM,ALAMDA,KLIN,ZHAT,EPS,JX,VB)
DIMENSION C(50,10), NMA(50), NMG(10), IJX(50,10)
DIMENSION CK(50), AX(50), BX(10), JXX(50), JX(50,10), V(50)
REAL NMA,NMG,NNMA
C READ (5,*) M,N,NNMA,ISEED
C READ (5,*) (NMG(J), J=1,N)
C READ (5,*) (NMA(I), I=1,M)
C DO 5 I=1,M
C 5 READ (5,*) (C(I,J), J=1,N)
C READ (5,*) KLIM, ALAMDA, KLIN, ZHAT, EPS
C CALL GENMAM (M,N,ISEED,NNMA,C,NMA,NMG)
C *
C * INITIALIZE VARIABLES
NKPN=0
IFEAS=0
K=0
C *
C CALL TINIT
C *
C * INITIALIZE LAGRANGEAN MULTIPLIERS
DO 111 I=1,M
V(I)=0.0
111 CONTINUE
115 VP=0.0
VB=0.0
DO 117 I=1,M
VP=VP+V(I)*NMA(I)
117 CONTINUE

C *
C * SOLVE THE LAGRANGEAN PROBLEM BY SOLVING THE SUBPROBLEMS
DO 125 J=1,N
DO 121 I=1,M
CK(I)=-((C(I,J)-V(I))
121 CONTINUE
B=NMG(J)
C *
C * SOLVE THE SUBPROBLEM CORRESPONDING TO 'J'
CALL MAOPT(M,CK,B,ZSTAR,JXX)
NKPN=NKPN+1
VP=VP-ZSTAR
DO 123 I=1,M
JX(I,J)=JXX(I)

- 180 -
CONTINUE

C * CHECK FEASIBILITY OF RELAXED CONSTRAINT (MACHINE AVAILABILITY)
DO 61 I=1,M
    AX(I)=0.
DO 60 J=1,N
    IF(JX(I*J)=EQ.0) GO TO 60
    VB=VB+C(I*J)
    AX(I)=AX(I)+1.0
60 CONTINUE

C * IF CONSTRAINTS ARE FEASIBLE SET FEASIBILITY = 1
IF (AX(I)GT.NMA(I)) GO TO 80
62 CONTINUE

C * UPDATE MULTIPLIERS IF CONSTRAINTS ARE VIOLATED
DO 65 I=1,M
    IF(AX(I)*LT.NMA(I)) GO TO 80
65 CONTINUE
GO TO 993

C * AFTER KLIN ITERATIONS REDUCE ALAMDA BY HALF
K=K+1
IF(K*EQ.KLIN) GO TO 993

C * COMPUTE STEPSIZE
ANORM=0.
DO 91 I=1,M
    ANORM=ANORM+(AX(I)-NMA(I))**2
91 CONTINUE
STPSIZ=ALAMDA*(ZMAT-VP)/ANORM

C * IF(STPSIZ.LE.EPS) GO TO 998
DO 92 J=1,N
    V(J)=V(J)+STPSIZ*(AX(I)-NMA(I))
V(J)=AMAX(V(J),0.0)
92 CONTINUE

C * APPLY THE PROCEDURE TO FIND THE FEASIBLE SOLUTION
GO TO 115

998 IF(IFEAS.EQ.1) GO TO 999

CALL MAFEAS(M,N,AX,BX,NMA,NMG,C,JX,RED)
CALL TUSED(MCP,)

END
WRITE(6,*) (JX(I,J), J=1,N)
GO TO 996
999 CALL TUSED(MCPU)
WRITE(6,*) 'BEST FEASIBLE SOLUTION',
WRITE (6,*) 'MACHINE ALLOCATION'
DO 988 I=1,M
988 WRITE(6,*) (JX(I,J), J=1,N)
C996 WRITE(5,*) 'OBJECTIVE FUNCTION VALUE (ORIGINAL PROBLEM) = ', VB
WRITE(6,*) 'TOTAL TIME TAKEN (MILLI-SECONDS) = ', MCPU
WRITE(6,*) 'TOTAL NO. OF SUBPROBLEMS SOLVED = ', NKP\N
996 RETURN
C STOP
END

C *
SUBROUTINE TO MAKE THE SOLUTION CORRESPONDING TO RELAXATION 1
C
FEASIBLE
SUBROUTINE MAFEAS(M,N,AX,BX,NMA,NMG,C,JX,RED)
DIMENSION C(50,10),JX(50,10),JXX(50)
DIMENSION NMA(50),NMG(10),AX(50),BX(10)
DIMENSION CR(10),IR(10),CL(10)
REAL NMA,NMG
RED=0.
ADD=0.
C *
RECTIFY THE VIOLATED MACHINE AVAILABILITY CONSTRAINTS
DO 10 I=1,M
IF(AX(I)>NMA(I)) GO TO 5
10 GO TO 10
5 DO 15 J=1,N
CL(J)=C(I,J)
15 IR(J)=J
C *
SUBROUTINE TO ARRANGE NUMBERS IN DECREASING ORDER
CALL SVRGP(N,CL,CR,IR)
DO 20 J=1,N
IF(JX(I,IR(J))=EQ.0) GO TO 20
JX(I,IR(J))=0
BX(IR(J))=BX(IR(J))-1.0
RED=RED+CL(IR(J))
C WRITE(5,*) RED,CL(IR(J))
AX(I)=AX(I)-1.0
IF(AX(I)=EQ.NMA(I)) GO TO 10
20 CONTINUE
10 CONTINUE
C *
ALLOCATE MACHINES THAT ARE NOT IN ANY GROUP
DO 40 I=1,M
IF(AX(I)>NMA(I)) GO TO 45
GO TO 40
45 DO 30 J=1,N
IR(J)=J
30 CL(J)=-C(I,J)
CALL SVRGP(N,CL,CR,IR)
DO 55 J=1,N
- 192 -
JX(I*IR(J))=1
IF(BX(IR(J))GE.NMG(IR(J))) GO TO 55
ADD=ADD+CL(IR(J))
WRITE(6,*) ADD,CL(IR(J))
BX(IR(J))=BX(IR(J))+1.0
AX(I)=AX(I)+1
IF(CL(IR(J+1))GT.0) GO TO 40
IF(AX(I)=EQ.NMA(I)) GO TO 40
CONTINUE
CONTINUE
RED=RED+ADD
RETURN
END
MACHINE ALLOCATION PROBLEM (MODELS 1 AND 2)  
(RELAXATION 2)

C LAGRANGIAN RELAXATION METHOD TO SOLVE THE MACHINE ALLOCATION
C PROBLEM USING REGULAR SUBGRADIENT
C ** CONSTRAINT ON NUMBER OF MACHINES IN A GROUP IS RELAXED
C MAIN PROGRAM
DIMENSION C(500,20), NMA(500), NMG(20), IJX(500,20), BX(20)
DIMENSION CK(20), AX(500), JXX(20), JX(500,20), V(20)
REAL NMA,NMG,NNMA
READ (5,*), M,N,NNMA,ISEED
C READ (5,*), (NMG(J), J=1,N)
C READ (5,*), (NMA(I), I=1,M)
DO 5 I=1,M
C 5 READ (5,*), (C(I,J), J=1,N)
C READ (5,*), KLM, ALAMDA, KLIN, ZHAT, EPS
CALL GENMAM(M,N,ISEED,NNMA,C,NMA,NMG)
C * INITIALIZE VARIABLES
NKPN=0
IFEAS=0
K=0
CALL TINIT
DO 111 J=1,N
V(J)=0.0
111 CONTINUE
115 VP=0.0
VB=0.0
DO 117 J=1,N
VP=VP+V(J)*NMG(J)
117 CONTINUE
C * SOLVE THE LAGRANGEAN PROBLEMS BY SOLVING THE SUBPROBLEMS
DO 125 I=1,M
DO 121 J=1,N
CK(J)=-(C(I,J)+V(J))
C WRITE(6,*), C(I,J),CK(J)
121 CONTINUE
125 CONTINUE
C * SOLVE THE SUBPROBLEM CORRESPONDING TO 'I'
B=NMA(I)
CALL MAOPT(N,CK,B,ZSTAR,JX)
NKPN=NKPN+1
VP=VP-ZSTAR
DO 123 J=1,N
JX(I,J)=JXK(J)
123 CONTINUE
125 CONTINUE
SUM=0.
C * CHECK CONSTRAINT FEASIBILITY
DO 61 J=1,N
AX(J)=0.
DO 60 I=1,M
IF(JX(I,J)>EO) GO TO 50
60 CONTINUE
61 CONTINUE
- 194 -
VB=VB+C(I,J)
AX(J)=AX(J)+1.0
60 CONTINUE
SUM = (NMG(J)-AX(J))*V(J) + SUM
61 CONTINUE
DO 62 J=1,N
IF(AX(J).GT.NMG(J)) GO TO 90
62 CONTINUE
IFEAS=1
DO 222 I=1,M
DO 222 J=1,N
222 JX(I,J)=JX(I,J)
DO 65 J=1,N
IF(AX(J).LT.NMG(J)) GO TO 90
65 CONTINUE
GO TO 998
C * UPDATE MULTIPLIERS
80 K=K+1
IF(K.EQ.XLIM) GO TO 999
C * AFTER XLIN ITERATIONS REDUCE XALPHA BY HALF
I1=K/XLIN
IF((K-I1*XLIN).EQ.0) XALPHA=XALPHA/2.
C * COMPUTE STEPSIZE
ANORM=0.
DO 91 J=1,N
ANORM=ANORM+(AX(J)-NMG(J))**2
91 CONTINUE
STPSIZ=XALPHA*(Z-HAT-VP)/ANORM
C IF(STPSIZ.LE.FPS) GO TO 998
DO 92 J=1,N
V(J)=V(J)*STPSIZ*(AX(J)-NMG(J))
V(J)=AMAX1(V(J),0.0)
92 CONTINUE
GO TO 115
998 IF(IIFEAS.EQ.1) GO TO 999
C * APPLY THE PROCEDURE TO FIND THE FEASIBLE SOLUTION
DO 161 I=1,M
BX(I)=0.0
DO 161 J=1,N
IF(JX(I,J).EQ.0) GO TO 161
BX(I)=BX(I)+1.0
161 CONTINUE
CALL MAPEAB(M,N,BX,AX,NMA,NMG,C,JX,RED)
CALL TUSED(MCPU)
VB=VB-RED
WRITE(6,*) ' SOLUTION MADE FEASIBLE '
DO 885 I=1,M
885 WRITE(6,*) (JX(I,J),J=1,N)
GO TO 996
999 CALL TUSED(MCPU)
WRITE(6,*) 'BEST FEASIBLE SOLUTION '
DO 898 I=1,M
898 

- 195 -
WRITE(6,*) (IJK(I,J), J=1,N)
WRITE(6,*) 'OBJECTIVE FUNCTION VALUE (ORIGINAL PROBLEM) = ', VC
WRITE(6,*) 'DEVIATION = ', DEV
WRITE(6,*) 'AX = (AX(J), J=1,N)
WRITE(6,*) 'V(J) = (V(J), J=1,N)
WRITE(6,*) 'TOTAL TIME TAKEN (MILLI-SECONDS) = ', MCPU
WRITE(6,*) 'TOTAL NO. OF SUBPROBLEMS SOLVED = ', NKPN
STOP
END

C ** SUBROUTINE TO MAKE THE SOLUTION CORRESPONDING TO RELAXATION 2
C **
SUBROUTINE MAEAB(N,N,BX,AX,NMA,NMG,C,JX,RED)
DIMENSION AX(500),BX(20),C(500,20),JX(500,20)
DIMENSION CR(500),IR(500),NMG(20),CK(500),NMA(500)
REAL NMA,NMG
RED=0.
ADD=0.
C **
** RECTIFY THE VIOLATED GROUP SIZE CONSTRAINTS
DO 30 J=1,N
5 CK(I)=C(I,J)
IF(BX(J)*GT*NMG(J)) GO TO 10
GO TO 30
10 DO 12 I=1,M
12 IR(I)=I
C **
** SUBROUTINE TO ARRANGE NUMBERS IN DECREASING ORDER
CALL SVRG(N,CK,CR,IR)
DO 15 I=1,N
IF(JX(IR(I),J)*EQ.0) GO TO 15
JX(IR(I),J)=0
AX(IR(I))=AX(IR(I))-1.0
BX(J)=BX(J)-1.0
RED=RED+CK(IR(I))
IF(BX(J)*GT*NMG(J)) GO TO 30
15 CONTINUE
10 CONTINUE
C **
** ALLOCATE THE MACHINES THAT ARE NOT IN ANY GROUP
DO 40 I=1,M
IF(AX(I)*LE.0.5) GO TO 45
GO TO 40
45 DO 60 J=1,N
IR(J)=J
60 CK(J)=-C(I,J)
CALL SVRG(N,CK,CR,IR)
DO 55 J=1,N
IF(BX(IR(J))*GE.*NMG(IR(J))) GO TO 55
JX(I,IR(J))=1
ADD=ADD-CK(IR(J))
BX(IR(J))=BX(IR(J))+1.0
AX(I)=AX(I)+1.0

IF(CK(IR(J+1)) .GT. 0) GO TO 40
IF(A(I) .EQ. NMA(I)) GO TO 40

55 CONTINUE

40 CONTINUE
RED = RED - ADD
RETURN
END
MACHINE ALLOCATION PROBLEM (MODELS 1 AND 2)
(Subroutines common to Relaxations 1 and 2)

C * SUBROUTINE TO SOLVE THE SUBPROBLEM IN MACHINE ALLOCATION
C MODEL FOR A GIVEN MULTIPLIER
SUBROUTINE MAOPT(N,CK,B,ZSTAR,JXX)
DIMENSION CK(50),D(50),JXX(50),CR(50),IR(50)
ZSTAR=0.
DO 10 J=1,N
JXX(J)=0
D(J)=CK(J)
10 IR(J)=J
C * IBM SUBROUTINE TO ARRANGE NUMBERS IN DECENDING ORDER
CALL SVRGP(N,CK,CR,IR)
NM=0
DO 20 J=1,N
JXX(IR(J))=1
NM=NM+1
ZSTAR=ZSTAR+D(IR(J))
IF(NM.GE.B) GO TO 30
20 CONTINUE
30 RETURN
END

C * SUBROUTINE TO GENERATE TEST PROBLEMS FOR THE MACHINE
C ALLOCATION MODEL
SUBROUTINE GENMAM(M,N,ISEED,NNMA,C,NMA,NMG)
DIMENSION C(500,20),NMA(500),NMG(20),R(10000)
REAL NNMA,NMA,NMG
ANMG=0.
C * GENERATE THE NO. OF MACHINES AVAILABLE IN EACH TYPE
DO 15 I=1,M
R(I)=RAND(ISEED)
15 NMA(I)=IFIX(NNMA+5.0*R(I))
C 15 NMA(I)=NNMA
C * GENERATE THE LIMIT ON THE NO. OF MACHINES IN A GROUP
DO 80 J=1,N
80 INMG=INMG+NMA(I)
VAL=FLOAT(INMG/N)
DO 10 J=1,N
R(J)=RAND(ISEED)
10 NMG(J)=IFIX(VAL+1.0*R(J))
C 10 NMG(J)=VAL*2.0
C * GENERATE THE COMPATIBILITY INDICES
DO 20 I=1,M
DO 20 J=1,N
R(J)=RAND(ISEED)
20 C(I,J)=R(J)
RETURN
END
MACHINE GROUP FORMATION PROBLEM (MODEL 3)

C * LAGRANGEAN RELAXATION METHOD TO SOLVE THE MACHINE GROUPING
C PROBLEM USING REGULAR SUBGRADIENT
C *
C MAIN PROGRAM
DIMENSION C(50,50), NMA(50), D(50), IJX(50,50)
DIMENSION CK(50,50), AX(50), JYK(50), JX(50,50), V(50)
REAL NMA*, NNMMA*, NNMAL
READ (5,*) M*, P*, NNMMA*, NNMAL, ISEED
C READ (5,*) (NMA(I), I=1,M)
C DO 5 I=1,M
C 5 READ (5,*) (C(I,J), J=1,M)
C READ (5,*) KLIM*, ALAMDA*, KLIN*, ZHAT*, EPS
C *
C PROBLEM DATA GENERATION
CALL GENMG(M, NNMMA*, NNMAL, ISEED, NMA*, C)
C *
C INITIALIZE VARIABLES
NKPN=0
IFRAS=0
K=0
CALL TINIT
DO 111 I=1,M
V(I)=0.0
111 CONTINUE
115 VP=0.0
VB=0.0
DO 117 I=1,M
VP=VP+V(I)*NMA(I)
117 VP=VP+V(I)*NMA(I)
C *
C SOLVE THE LAGRANGEAN PROBLEM BY SOLVING THE SUBPROBLEMS
DO 121 I=1,M
DO 121 J=1,M
CK(I,J)=(C(I,J)-V(I))
121 CONTINUE
C *
C SOLVE THE SUBPROBLEM
DO 222 J=1,M
D(J)=0.
DO 222 I=1,M
D(J)=D(J)+AMAX1(CK(I,J), 0.0)
DO 333 J=1,M
333 D(J)=-D(J)
CALL MG0PT(M, D*, P*, ZSTAR, JYK)
NKPN=NKPN+1
VP=VP-ZSTAR
C *
C CHECK CONSTRAINT FEASIBILITY
DO 123 J=1,M
DO 123 I=1,M
IF(CF(I,J)·GE.0.) GO TO 122
JX(I,J)=0
GO TO 123
122 JX(I,J)=JYK(J)
123 CONTINUE
DO 60 I=1,M
AX(I)=0.
DO 60 J=1,M
IF(JX(I,J)*EQ.0) GO TO 50
VB=VB+C(I,J)
AX(I)=AX(I)+1.0
60 CONTINUE
DO 62 I=1,M
IF(AX(I)*GT.NMA(I)) GO TO 90
62 CONTINUE
IFEAS=1
DO 65 I=1,M
DO 65 J=1,M
IHX(I,J)=JX(I,J)
VC=VB
65 CONTINUE
GO TO 998
67 DO 75 I=1,M
IF(AX(I)*LT.NMA(I)) GO TO 90
75 CONTINUE
GO TO 998
C * COMPUTE NEW MULTIPLIERS
80 K=K+1
IF(K*GE.KLIN) GO TO 998
C * AFTER KLIN ITERATIONS REDUCE ALAMDA BY HALF
I=K/KLIN
IF((K-I)*KLIN*EQ.0) ALAMDA=ALAMDA/2.
C * COMPUTE STEPSIZE
ANORM=0.
DO 91 I=1,M
ANORM=ANORM+(AX(I)-NMA(I))**2
91 CONTINUE
STPSIZ=ABS(ALAMDA*(ZHT-VP)/ANORM)
IF(STPSIZ.LE.EPS) GO TO 998
DO 92 I=1,M
VI(I)=V(I)+STPSIZ*(AX(I)-NMA(I))
92 CONTINUE
GO TO 115
C * MAKE THE LAGRANGEAN SOLUTION FEASIBLE
998 IF(IFEAS.EQ.1) GO TO 997
CALL MGFEAS(M*AX+NMA+C*JX+RED)
CALL TUSED(MCPU)
VB=VB-RED
WRITE(6,*,'(A)') 'SOLUTION MADE FEASIBLE'
DO 889 I=1,M
WRITE(6,*,'(A)') (JX(I,J), J=1,M)
GO TO 996
997 CALL TUSED(MCPU)
WRITE(6,*,'(A)') 'BEST FEASIBLE SOLUTION'
DO 888 I=1,M
WRITE(6,*,'(A)') (IJX(I,J), J=1,M)
WRITE(6,*,'(A)') 'OBJECTIVE FUNCTION VALUE(ORIGINAL PROBLEM) = ', VC
996 WRITE(6,*,'(A)') 'TOTAL TIME TAKEN (MILLI-SECONDS) = ', MCU
- 190 -
C *  SUBROUTINE TO SOLVE THE SUBPROBLEM IN MACHINE ALLOCATION
C  PROBLEM FOR A GIVEN MULTIPLIER
SUBROUTINE MGDT(M,CK,R,ZSTAR,JXK)
DIMENSION CK(50), C(50), JXK(50), CR(50), IR(50)
ZSTAR=0.
DO 10 J=1,M
   JXK(J)=0
   C(J)=CK(J)
  10 IR(J)=J
C *  IBM SUBROUTINE TO SORT NUMBERS IN DECENDING ORDER
CALL SVRG1(M,CK,CR,IR)
NM=0
DO 20 J=1,M
   JXK(IR(J)) = 1
   NM=NM+1
   ZSTAR=ZSTAR+C(IR(J))
IF(NM.EQ.B) GO TO 30
  20 CONTINUE
  30 RETURN
END
C *  SUBROUTINE TO MAKE THE LAGRANGEAN SOLUTION OF THE MACHINE
C  GROUPING MODEL FEASIBLE
SUBROUTINE MGFEAS(M,AX,NMA,C,JX,RED)
DIMENSION AX(50), C(50,50), NMA(50), JX(50,50)
DIMENSION CR(50), IR(50), CK(50)
REAL NMA, RED=0., ADD=0.
C * RECTIFY VIOLATED MACHINE AVAILABILITY CONSTRAINTS
DO 10 I=1,M
   IF(AX(I)*GT*NMA(I)) GO TO 5
  10 CONTINUE
C *  IBM SUBROUTINE TO SORT NUMBERS IN DECENDING ORDER
CALL SVRG1 (M,CK,CR,IR)
DO 20 J=1,M
   IF(JX(I,IR(J))*EQ.0) GO TO 20
   JX(I,IR(J))=0
   RED=RED+CK(IR(J))
   AX(I)=AX(I)-1.0
   IF(AX(I)*EQ.NMA(I)) GO TO 10
  20 CONTINUE
  10 CONTINUE
DO 40 I=1,M
IF(AX(I)<LT·NM(A(I)) GO TO 45
GO TO 40
45 DO 30 J=1,M
IR(J)=J
30 CK(J)=C(I,J)
CALL SWRGP(M,Ck,Cr,IR)
DO 55 J=1,M
IF(JX(I*IR(J))·EQ·1) GO TO 55
JX(I*IR(J))=1
ADD=ADD·CK(IR(J))
AX(I)=AX(I)+1.0
IF(AX(I)·GE·NM(A(I)) GO TO 40
55 CONTINUE
40 CONTINUE
RED=RED+ADD
RETURN
END

C * SUBROUTINE TO GENERATE TEST PROBLEMS FOR MACHINE
C GROUPING MODEL
SUBROUTINE GENMG(M,NMAU,NNMAL,ISEED,NMA,SA)
DIMENSION S(50,50),NM(50),R(2500)
REAL NM(50),NMA,NNMAL

C * GENERATE THE NO. OF MACHINES AVAILABLE IN EACH TYPE
DO 15 I=1,M
R(I)=RAND(ISEED)
15 NMA(I)=IFIX(NNMAL+(NMMAU-NNMAL)·R(I))
C * GENERATE SIMILARITY INDICES
DO 20 I=1,M
DO 20 J=1,M
R(J)=RAND(ISEED)
20 S(I,J)=R(J)
RETURN
END
PARTS ALLOCATION PROBLEM (MODEL 4)
(RELAXATION 1)

C * LAGRANGEAN RELAXATION METHOD TO SOLVE THE PARTS ALLOCATION
C * PROBLEM USING REGULAR SUBGRADIENT
C * CONSTRAINT "EACH PART CAN BE IN ONLY ONE GROUP" IS RELAXED
C *
C * MAIN PROGRAM
C SUBROUTINE PALAG(M,N,C,NPG,KLIM,ALAMDA,KLIN,ZHAT,JX,VB)
C DIMENSION CK(2000), AX(2000), V(20), BX(20)
C REAL NPG
C READ (5,*) M, N, ISEED
C READ (5,*) (NPG(J), J=1,N)
C DO 5 I=1,M
C 5 READ (5,*) (C(I,J), J=1,N)
C READ (5,*) KLIM, ALAMDA, KLIN, ZHAT
C VAL=FLOAT(M/N)+2
C *
C GENERATE TEST PROBLEMS
C CALL GENPAM(M,N,ISEED,VAL,C,NPG)
C *
C INITIALIZE VARIABLES
C NKPN=0
C IPFAS=0
C K=0
C CALL TINIT
C *
C SET INITIAL MULTIPLIERS
C DO 111 I=1,M
C V(I)=0.0
C 111 CONTINUE
C VP=0.0
C VB=0.0
C DO 117 I=1,M
C VP=VP-V(I)
C 117 CONTINUE
C *
C SOLVE THE LAGRANGEAN PROBLEM BY SOLVING THE SUBPROBLEMS
C DO 125 J=1,N
C DO 121 I=1,M
C CK(I)=-(C(I,J)+V(I))
C 121 CONTINUE
C *
C SOLVE THE SUBPROBLEM CORRESPONDING TO 'J'
C B=NPG(J)
C CALL PAOPT(M,CK,B,ZSTAR,JX)
C NKPN=NKPN+1
C VP=VP-ZSTAR
C DO 123 I=1,M
C JX(I,J)=JXK(I)
C 123 CONTINUE
C *
C CHECK CONSTRAINT FEASIBILITY
C DO 61 I=1,M
C AX(I)=0.0
C DO 60 J=1,N
C IF(JX(I,J)<-EO.0) GO TO 60
C 60 CONTINUE
C
VB=VB+C(I,J)
AX(I)=AX(I)+1.0
60 CONTINUE
61 CONTINUE
DO 62 I=1,M
   IF(AX(I)*NE.1.) GO TO 80
62 CONTINUE
C ** NO LAGRANGEAN DUALITY GAP
   IPEAS=1
   GO TO 997
C * UPDATE MULTIPLIERS
80 K=K+1
   IF(K.EQ.KLIM) GO TO 998
C ** AFTER KLIN ITERATIONS REDUCE ALAMDA BY HALF
   I1=K/KLIM
   IF((K-I1*KLIN)*EQ.0) ALAMDA=ALAMDA/2.
C * COMPUTE STEPSIZE
   ANORM=0.
   DO 91 I=1,M
      ANORM=ANORM+(AX(I)-1.)**2
91 CONTINUE
   STPSIZ=ALAMDA*(ZHAT-VP)/ANORM
C IF(STPSIZ.LE.EPS) GO TO 999
   DO 92 I=1,M
      V(I)=V(I)+STPSIZ*(AX(I)-1.)
92 CONTINUE
   GO TO 115
C 993 DO 161 J=1,N
   BX(J)=0
   DO 161 I=1,M
      IF(JX(I+J)*EQ.0) GO TO 161
      BX(J)=BX(J)+1.0
161 CONTINUE
   DO 887 I=1,M
887 WRITE(6,*)(JX(I,J), J=1,N)
C * APPLY THE PROCEDURE TO MAKE THE SOLUTION FEASIBLE
   CALL PAPFAS(M,N,AX,BX,NPG,C,JX,RED)
   CALL TUSED(MCPU)
   WRITE (6,*),' SOLUTION MADE FEASIBLE '
   VB=VB-RED
   GO TO 996
997 WRITE(6,*),' FEASIBLE AND OPTIMAL SOLUTION'
   CALL TUSED(MCPU)
996 WRITE(6,*),' OBJECTIVE FUNCTION VALUE(ORIGINAL PROBLEM) = ', VB
   WRITE(6,*),' TOTAL TIME TAKEN (MILLI-SECONDS) = ', MCPU
   WRITE(6,*),' TOTAL NO. OF SUBPROBLEMS SOLVED = ', NKPN
   DEV=(VP-VB)*100.00/VP
   WRITE(6,*),' PARTS ALLOCATION'
   DO 888 I=1,M
888 WRITE(6,*)(JX(I,J), J=1,N)
RETURN
C STOP
END

C * SUBROUTINE TO MAKE THE SOLUTION CORRESPONDING TO RELAXATION 1
C * FEASIBLE
SUBROUTINE PAFEAS(M,N,AX,BX,NPG,C,JX,RED)
DIMENSION CR(20),IR(20),NPG(20),CK(20)
REAL NPG
RED=0.
ADD=0.

C * RECTIFY THE VIOLATED PART ALLOCATION CONSTRAINTS
DO 30 I=1,M
   DO 5 J=1,N
      5 CK(J)=C(I,J)
      IF(AX(I)*GT.1.0) GO TO 10
      GO TO 30
   10 DO 12 J=1,N
    12 IR(J)=J
   CALL SVRGP(N,CK,CR,IR)
   DO 15 J=1,N
      IF(JX(I,IR(J))*EQ.0) GO TO 15
      JX(I,IR(J))=0
      BX(IR(J))=BX(IR(J))-1.0
      AX(I)=AX(I)-1.0
      RED=RED+CK(IR(J))
      IF(AX(I)*EQ.1.0) GO TO 30
    15 CONTINUE
   30 CONTINUE
   DO 40 I=1,M
      IF(AX(I)*LE.0.5) GO TO 20
      GO TO 40
   20 DO 22 J=1,N
    22 CK(J)=-C(I,J)
   CALL SVRGP(N,CK,CR,IR)
   DO 32 J=1,N
      IF(BX(IR(J))*GE.NPG(IR(J))) GO TO 32
      JX(I,IR(J))=1
      BX(IR(J))=BX(IR(J))+1.0
      ADD=ADD-CK(IR(J))
   32 CONTINUE
   40 CONTINUE
   RED=RED-ADD
RETURN
END
PARTS ALLOCATION PROBLEM (MODEL 4)
(RELAXATION 2)

C * LAGRANGEAN RELAXATION METHOD TO SOLVE THE PARTS ALLOCATION
C * PROBLEM USING REGULAR SUBGRADIENT
C * THE CONSTRAINT ON NUMBER OF PARTS IN A GROUP IS RELAXED
C *
MAIN PROGRAM
REAL NPG
READ (5,*), M,N,I,SEED
READ (5,*) (NPG(J), J=1,N)
DO 5 I=1,M
5 READ (5,*) (C(I,J), J=1,N)
READ (5,*) KLM, KLIM, ZHAT, EPS
IVAL=FLOAT(M/N)
C *
CALL GENPAM(M*N,IVAL,I,SEED,C,NPG)
C *
INITIALIZE VARIABLES
NKPN=0
IFEAS=0
K=0
CALL TINIT
DO 111 J=1,N
V(J)=0.0
111 CONTINUE
115 VP=0.0
VB=0.0
DO 117 J=1,N
VP=VP-V(J)*NPG(J)
117 VP=VP-ZSTAR
DO 121 J=1,N
CK(J)=-(C(I,J)+V(J))
121 CONTINUE
C *
SOLVE THE SUBPROBLEM CORRESPONDING TO PART *I*
B=1
CALL PAOPT(N,CK,B,ZSTAR,JXX)
NKPN=NKPN+1
VP=VP-ZSTAR
DO 123 J=1,N
JX(I,J)=JXX(J)
123 CONTINUE
125 CONTINUE
SUM=0.
DO 61 J=1,N
AX(J)=0.
DO 60 I=1,M
IF(JX(I,J) EQ 0) GO TO 60
VB=VB+C(I,J)
60 CONTINUE
61 CONTINUE
AX(J) = AX(J) + 1.0
60 CONTINUE
SUM = (NPG(J) - AX(J)) * V(J) + SUM
61 CONTINUE
DO 62 J = 1, N
C ** FEASIBILITY CHECKING
IF (AX(J) GT NPG(J)) GO TO 90
62 CONTINUE
IFEAS = 1
DO 222 I = 1, M
DO 222 J = 1, N
222 IX(I, J) = JX(I, J)
DO 65 J = 1, N
IF (AX(J) LT NPG(J)) GO TO 90
65 CONTINUE
GO TO 998
C * UPDATE MULTIPLIERS
80 K = K + 1
IF (K EQ KLIN) GO TO 998
C ** AFTER KLIN ITERATIONS REDUCE ALAMDA BY HALF
I1 = K / KLIN
IF ((K - I1 * KLIN) EQ 0) ALAMDA = ALAMDA / 2.
C * COMPUTE STEP SIZE
ANORM = 0.
DO 91 J = 1, N
ANORM = ANORM + (AX(J) - NPG(J)) ** 2
91 CONTINUE
STPSIZ = ALAMDA * (ZHT - VP) / ANORM
C IF (STPSIZ LE EPS) GO TO 993
DO 92 J = 1, N
V(J) = V(J) + STPSIZ * (AX(J) - NPG(J))
92 CONTINUE
GO TO 115
998 IF (IFEAS EQ 1) GO TO 999
WRITE (6, *) ' SOLUTION MADE FEASIBLE '
C * APPLY THE PROCEDURE TO FIND THE FEASIBLE SOLUTION
DO 161 I = 1, M
RX(I) = 0.0
DO 161 J = 1, N
IF (JX(I, J) EQ 0) GO TO 161
RX(I) = RX(I) + 1.0
161 CONTINUE
DO 885 I = 1, M
895 WRITE (6, *) (JX(I, J), J = 1, N)
CALL PAFEAB(M, N, RX, AX, NPG, C, JX, RED)
VB = VB - RED
CALL TUSED(MCPU)
WRITE (6, *) ' PARTS ALLOCATION '
DO 886 I = 1, M
886 WRITE (6, *) (JX(I, J), J = 1, N)
GO TO 396
999 CALL TUSED(MCPU)
WRITE (6, *) ' BEST FEASIBLE SOLUTION '
WRITE(*,*) * PARTS ALLOCATION*
DO 998 I=1,M
998 WRITE(*,*) (IJX(I,J), J=1,N)
WRITE(*,*) * OBJECTIVE FUNCTION VALUE(ORIGINAL PROBLEM) = '*, VB
WRITE(*,*) * TOTAL TIME TAKEN (MILLI-SECONDS) = '*, MCPU
WRITE(*,*) * TOTAL NO. OF SUBPROBLEMS SOLVED = '*, NKPN
STOP
END

C * SUBROUTINE TO MAKE THE SOLUTION CORRESPONDING TO RELAXATION 2
C * FEASIBLE
SUBROUTINE PAFeAB(M,N,AX,BX,NPG,C,JX,RED)
REAL NPG
RED=0.
ADD=0.

C * RECTIFY THE VIOLATED GROUP SIZE CONSTRAINTS
DO 30 J=1,N
DO 7 I=1,M
7 CL(I)=C(I,J)
IF(BX(J)*GT*NPG(J)) GO TO 10
GO TO 30
10 DO 12 I=1,M
12 IR(I)=I
CALL SVRGPN(CL,CR,IR)
DO 15 I=1,N
IF(JX(IR(I),J)*EQ.0) GO TO 15
JX(IR(I),J)=0
AX(IR(I))=AX(IR(I))-1.0
BX(J)=BX(J)-1.0
RED=RED+CL(IR(I))
IF(BX(J)*LE*NPG(J)) GO TO 30
CONTINUE
30 CONTINUE
DO 40 I=1,M
IF(AX(I)*LE.0.5) GO TO 45
GO TO 40
40 CONTINUE
DO 60 J=1,N
IR(J)=J
60 CL(J)=-C(I,J)
CALL SVRGPN(N,CL,CR,IR)
DO 55 J=1,N
IF(BX(IR(J))*GE.*NPG(IR(J))) GO TO 55
JX(I,IR(J))=1
ADD=ADD+CL(IR(J))
BX(IR(J))=BX(IR(J))+1.0
AX(I)=AX(I)+1.0
GO TO 40
55 CONTINUE
40 CONTINUE
RETURN
END
PARTS ALLOCATION PROBLEM (MODEL 4)
(Subroutines common to Relaxations 1 and 2)

C * SUBROUTINE TO SOLVE THE SUBPROBLEM IN PARTS ALLOCATION
C PROBLEM FOR A GIVEN MULTIPLIER
SUBROUTINE PAOPT(M,CK,B,ZSTAR,JXK)
DIMENSION CK(500), C(500), JXK(500), CR(500), IR(500)
ZSTAR=0.
DO 10 J=1,M
JXK(J)=0
C(J)=CK(J)
10 IR(J)=J
C * SORT IN THE DECENDING ORDER
CALL SVRGP (M,CK,CR,IR)
NM=0
DO 20 J=1,M
JXK(IR(J)) = 1
NM=NM+1
ZSTAR=ZSTAR+C(IR(J))
IF(NM.GE.3) GO TO 30
20 CONTINUE
30 RETURN
END

C * SUBROUTINE TO GENERATE TEST PROBLEMS FOR PARTS
C * ALLOCATION MODEL
SUBROUTINE GENPAM(M,N,ISRED,IVAL,C,NPG)
DIMENSION C(2000,20), NPG(20), R(40000)
REAL NPG
C * GENERATE LIMIT ON THE NUMBER OF PARTS IN EACH GROUP
DO 10 J=1,N
R(J)=RAND(ISRED)
10 NPG(J)= IFIX(IVAL+2.0*IVAL*R(J))
C 10 NPG(J)= IVAL*2.0
C * GENERATE COMPATIBILITY INDICES
DO 20 I=1,M
DO 20 J=1,N
R(J)=RAND(ISRED)
20 CI(J)=ABS(R(J))
RETURN
END
MACHINE-PART GROUPING PROBLEM (MODELS 5 AND 6)

C * ITERATIVE PROCEDURE TO SOLVE THE SIMULTANEOUS MACHINE-PART
C GROUPING MODEL
DIMENSION NMA(500),NMG(20),NPX(2000,20),IY(500,20)
DIMENSION C(2000,500),CM(500,20),CP(2000,20)
REAL NPG,NMA,NMG,NNMA,NNMG,NPG

C * READ CORE INPUT INFORMATION
READ (5,*) KLIM,ALAMDA,KLIN,ZHAT,EPS
READ (5,*) NPM,NMF,ISEED,EPSLON
READ (5,*) NNMA,NNGM,NPG

C READ(5,*) (NMA(J),J=1,NM)
C READ(5,*) (NMG(K),K=1,NP)
C READ(5,*) (NPX(K),K=1,NP)
C DO 5 I=1,NP
C5 READ(5,*) (C(I,J),J=1,NM)
C * GENERATE TEST PROBLEMS
CALL GENNSM (NPM,NMF,NM,NNMA,NM,NMG,NPG,ISEED,C,NNMA,NMG,NPG)
CALL TINIT

C ** MAKE THE INITIAL PARTS ALLOCATION
KK=0
KL=1
DO 38 I=1,NP
IX(I,K)=1
KK=KK+1
IF (KK.LT.((NP/NF)+1)) GO TO 38
KK=0
KL=KL+1
CONTINUE
38 ITRN=1
C * COMPUTE THE CORRESPONDING OBJECTIVE FUNCTION COEFFICIENTS IN THE
C MACHINE ALLOCATION MODEL
39 DO 40 J=1,NM
40 CM(I,J)=0
DO 40 I=1,NP
CM(I,K)=CM(I,K)+C(I,J)*IX(I,K)
C * SOLVE THE MACHINE ALLOCATION MODEL USING LAGRANGEAN RELAXATION
CALL MALAG(NPM,NMF,CM,NMA,NMG,KLIM,ALAMDA,KLIN,ZHAT,EPS,IY,OBJM)
C * COMPUTE THE CORRESPONDING OBJECTIVE FUNCTION COEFFICIENTS IN THE
C PARTS ALLOCATION MODEL
45 DO 45 I=1,NP
45 CP(I,K)=CP(I,K)+C(I,J)*IY(J,K)
C * SOLVE THE CORRESPONDING PARTS ALLOCATION MODEL
CALL PLAG(NPM,NMF,CM,NMG,KLIM,ALAMDA,KLIN,ZHAT,IX,OBJP)
WRITE(5,*) 'OBJM OBJP* OBJMOBJP)
IF (ABS(OBJM-OBJP)*LE.EPSLON) GO TO 100
ITRN=ITRN+1
IF (ITRN.EQ.50) GO TO 100

- 200 -
GO TO 39
C ** PRINT THE RESULTS
100 CALL TUSED(MCPU)
   WRITE(6,*) 'NO. OF ITERATIONS = ', ITRN
   WRITE(6,*) 'CPU TIME (M.sec) = ', MCPU
   WRITE(6,*) 'OBJECTIVE VALUE = ', OBJM
   WRITE(6,*) 'PARTS ASSIGNMENT'
   DO 50 I=1,NP
   WRITE(6,*) (IX(I,K),K=1,NP)
   WRITE(6,*) 'MACHINE ALLOCATION'
   DO 52 J=1,NM
   WRITE(6,*) (IJ(J,K),K=1,NP)
   STOP
END

C * SUBROUTINE TO GENERATE TEST PROBLEMS FOR THE SIMULTANEOUS
C MACHINE-PART GROUPING MODEL
C SUBROUTINE GENSMG(M,N,NF,NNMA,NNMG,NNPG,ISEED,C,NMA,NMG,NPG)
C DIMENSION C(20,500),NPG(20),R(100),NMG(20),NMA(500)
C REAL NMA,NMG,NPG,NNMA,NNMG,NNPG
C *
C GENERATE THE NO. OF MACHINES AVAILABLE IN EACH MACHINE TYPE
C DO 10 J=1,N
10 NMA(J)=NNMA
C *
C GENERATE THE LIMIT ON THE NO. OF MACHINES AND THE NO. OF PARTS
C IN EACH GROUP
C DO 12 K=1,NF
12 NMG(K)=NNMG
   NPG(K)=NNPG
C *
C GENERATE COMPATIBILITY INDICES
C DO 20 I=1,M
   DO 20 J=1,N
20   R(J)=RAND(ISEED)
   C(I,J)=5.0*R(J)
RETURN
END
COMPUTATION OF SIMILARITY INDICES (Section 4.1)

C * PROGRAM TO CALCULATE THE SIMILARITY INDEX BETWEEN TWO
C MACHINES
C DIMENSION ANCT(30,15), SUM(15,15), S(15,15), TOT(15)
C *
C READ THE NO. OF PARTS AND THE NO. OF MACHINES
C READ(5,*) P, M
C *
C READ THE NO. OF COMMON TOOLS BETWEEN A PART AND A MACHINE
C DO 10 I=1,P
10 READ(5,*) (ANCT(I,J), J=1,M)
   DO 15 J=1,M
   DO 15 K=1,M
   TOT(J)=0
   SUM(J,K)=0
   DO 20 J=1,M
   DO 20 K=1,P
   TOT(J)=ANCT(I,J)+TOT(J)

C * COMPUTE THE NO. OF PARTS COMMON TO BOTH MACHINES J AND K
C DO 30 J=1,M
C DO 30 K=1,M
C DO 30 I=1,P
   IF((ANCT(I,J).GT.0).AND.(ANCT(I,K).GT.0)) GO TO 25
   GO TO 30
25 SUM(J,K)=SUM(J,K)+ANCT(I,J)+ANCT(I,K)
30 CONTINUE
C DO 40 J=1,M
C DO 40 K=1,M
40 S(J,K)=SUM(J,K)/(TOT(J)+TOT(K))
C DO 50 J=1,M
50 WRITE(6,*) (S(J,K), K=1,M)
C STOP
C END
COMPUTATION OF COMPATIBILITY INDICES (Section 4.2)

C * PROGRAM TO COMPUTE THE COMPATIBILITY INDEX BETWEEN A PART AND A MACHINE
C
DIMENSION ANM(20), ANP(50), ANCT(50, 20), C(40, 20)
C *
READ THE NUMBER OF PARTS AND THE NO. OF MACHINES
READ (5, *) P, M
C *
READ THE NO. OF TOOLS REQUIRED FOR EACH PART
READ (5, *) (ANP(I), I=1, P)
C *
READ THE NO. OF TOOLS AVAILABLE ON EACH MACHINE
READ (5, *) (ANM(J), J=1, M)
C *
READ THE NO. OF COMMON TOOLS BETWEEN A PART AND A MACHINE
DO 10 I=1, P
10 READ(5, *) (ANCT(I, J), J=1, M)
DO 20 I=1, P
20 DO J=1, M
20 C(I, J)=ANCT(I, J)/AMIN1(ANM(J), ANP(I))
DO 30 I=1, P
30 WRITR(6, 5) (C(I, J), J=1, M)
5 FORMAT(10(F5.3, 2X))
STOP
END
VITA AUCTORIS

1958  Born in Tuticorin, India on the 28th of April

1974  Completed High school education from N.L.C. Boys High School, Neyveli, India

1975  Completed Pre-university education from St. Joseph’s College, Trichy, India

1980  Graduated from Regional Engineering College, Trichy, India with a Bachelor’s degree (Honours) in Industrial Engineering

1980–83  Worked as an Industrial Engineer in the Management Services Department of Southern Petrochemical Industries Corporation Ltd., Tuticorin, India

1984  Graduated from the Asian Institute of Technology, Bangkok, Thailand with a Master’s degree in Industrial Engineering and Management

1984–85  Worked as a Consultant with Besant Raj Consultants (P) Ltd., Madras, India

1985–86  Worked as a Consultant with Royal Industries (Thailand) Ltd., Bangkok, Thailand

1989  Currently a candidate for the degree of Doctor of Philosophy in Industrial Engineering at the University of Windsor, Windsor, Canada