PART FAMILY MACHINE GROUP FORMATION PROBLEM IN CELLULAR MANUFACTURING SYSTEMS

A THESIS
SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL
ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCES
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Levent Kandiller
April, 1989

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Levent Onur(Principal Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Mustafa Akgül

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Ömer S Benli

Assoc. Prof. Ömer Benli

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Cemal Dinçer

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Sinan Kayalıgil

Approved for the Institute of Engineering and Sciences:

M. Baray

Prof. Mehmet Baray
Director of Institute of Engineering and Sciences

ABSTRACT

PART FAMILY MACHINE GROUP FORMATION PROBLEM IN CELLULAR MANUFACTURING SYSTEMS

Levent Kandiller
M.S. in Industrial Engineering
Supervisor: Asst. Prof. Levent Onur
April, 1989

The first and the most important stage in the design of Cellular Manufacturing (CM) systems is the Part Family Machine Group Formation (PF/MG-F) problem. In this thesis, different approaches to the PF/MG-F problem are discussed. Initially, the design process of CM systems is overviewed. Heuristic techniques developed for the PF/MG-F problem are classified in a general framework. The PF/MG-F problem is defined and some efficiency indices designed to evaluate the PF/MG-F techniques are presented. One of the efficiency indices evaluates the inter-cell flows and inner-cell densities while another one measures the within-cell work-load balances. Another index measures the under-utilization levels of machines. A number of the most promising PF/MG-F techniques are selected for detailed analysis. These selected techniques are evaluated and compared in terms of the efficiency measures by employing randomly generated test problems. Finally, further research areas are addressed.

Keywords: Cellular Manufacturing Systems, Group Technology, Clustering.

ÖZET

HÜCRESEL İMALAT SİSTEMLERİNDE PARÇA SINIFLARINI VE TEZGAH GRUPLARINI BELİRLEME PROBLEMİ

Levent Kandiller
Endüstri Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticisi: Y. Doç. Levent Onur
Nisan, 1989

Hücresel imalat sistemlerinin tasarımındaki ilk ve en önemli aşama, parça sınıflarının ve tezgah gruplarının belirlenmesidir. Bu tezde, hücre tipi imalat sistemlerinin tasarım problemine yönelik değişik yaklaşımlar tartılışılmaktadır. İlk olarak hücresel imalat sistemlerinin tasarımı problemi ana hatlarıyla ele alınmıştır. Bu problemi çözümlemek için geliştirilen sezgisel yöntemler genel bir çerçeve içinde sınıflandırılmıştır. Hangi tasarım daha iyidir sorusunu yanıtlamak için bazı ölçütler geliştirilmiştir. Birinci ölçüt, imalat hücreleri arasındaki etkileşimleri ve imalat hücrelerinin yoğunluklarını göz önüne alan bileşik yeterlilik ölçüsüdür. Geliştirilen ikinci ölçüt hücre içi yük dengelerini içermektedir. Son ölçüt ise tezgahların atıl kapasite değerlerini kapsamaktadır. İncelemeye değer görülen altı değişik teknik tanıtılmıştır. Sözü edilen altı teknik, üretilen test problemleri ile geliştirilen yeterlilik ölçütleri bazında karşılaştırılmıştır. Son olarak yakın gelecekte yapılması düşünülen çalışmalara değinilmiştir.

Anahtar kelimeler : Hücre Tipi İmalat Sistemleri, Grup Teknolojisi, Öbekleme.

To my parents,

ACKNOWLEDGEMENT

I am indepted to Asst. Prof. Levent Onur for his supervision, suggestions, and encouragement throughout the development of this thesis. I am grateful to Assoc. Prof. Mustafa Akgül, Assoc. Prof. Ömer Benli, Asst. Prof. Cemal Dinçer, and Assoc. Prof. Sinan Kayalıgil for their valuable comments.

I would like to extend my deepest gratitude and thanks to my parents for their morale support, encouragement, especially at times of despair and hardship. It is to them that this study is affectionally dedicated, without whom it would not be possible.

I wish to express my appreciation to Deniz Gözükara, Gündüz Cöner and other BCC personel for their help. I would like to offer my sincere thanks to my friends Oğuzhan Aygün, Emel and Oğuz Erciyas who provided morale support and encouragement.

TABLE OF CONTENTS

1 INTRODUCTION 1

2 OVERVIEW OF PF/MG-F TECHNIQUES 5

3 ANALYSIS OF THE PF/MG-F PROBLEM 12

3.1 Definition of the PF/MG-F Problem 12

3.2 Efficiency Measures 17

4 SELECTED PF/MG-F TECHNIQUES 23

4.1 Combinatorial Grouping 25

4.1.1 Description of COMBGR 25

4.1.2 Modifications and Extensions 29

4.1.3 *Algorithm* {COMBGR} 32

4.2 Modified Rank Order Clustering 32

4.2.1 Description of MODROC 33

4.2.2 Modifications and Extensions 36

4.2.3 *Algorithm* {MODROC} 38

4.3 Machine Clustering Using Similarity Coefficients 39

4.3.1 Description of MACE 40

4.3.2 Modifications and Extensions 42

4.3.3 *Algorithm* {MACE} 46

4.4 Within-cell Utilization Based Clustering 47

4.4.1 Description of WUBC 47

4.4.2 Modifications and Extensions 50

4.4.3 *Algorithm* {WUBC} 52

4.5 Cost Analysis Algorithm 54

4.5.1 Description of CAA 55

4.5.2 Modifications and Extensions 56

4.5.3 *Algorithm* {CAA} 59

4.6 Seed Clustering 60

4.6.1 Description of ZODIAC 60

4.6.2 Modifications 67

4.6.3 *Algorithm* {ZODIAC} 68

5 EXPERIMENT **71**

5.1 Problem Generator 71

5.1.1 Generating Machine-Part Incidence Matrices 72

5.1.2 Generating Work-Load Matrices 75

5.1.3 *Procedure* {Generator} 79

5.2 Design of Experiment 80

5.3 Results and Discussion 82

5.4 Comparison 93

6 SUMMARY, CONCLUSIONS, SUGGESTIONS 96

6.1 Summary 96

6.2 Conclusions 97

6.3 Suggestions For Further Research 99

APPENDIX A 102

A.1 Generator 103

A.2 COMBGR 108

A.3 MODROC 126

A.4 MACE 141

A.5 WUBC 151

A.6 CAA 162

A.7 ZODIAC 173

APPENDIX B 190

REFERENCES 191

LIST OF FIGURES

2.1 A taxonomy of PF/MG-F techniques. 6

3.1 Part routings, graphs and incidence matrices: An example. . 13

3.2 An example of PF/MG-F having exceptional elements. 15

3.3 An example of work-load matrix employed in PF/MG-F. . . . 16

3.4 An example showing the effects of extensions in the PF/MG-F definition. 18

3.5 Some examples of grouping efficiency values 22

4.1 A ROC iteration. 34

4.2 Diagonal block identification. 37

4.3 Accommodation. 65

5.1 Effect of the clumpiness parameter on incidence matrices. . . 76

5.2 Main clumpiness effects on grouping efficiencies. 85

5.3 Main effects of the shape parameters on work-load balances. . 86

5.4 Main effects of the shape parameters on under-utilizations. . 87

LIST OF TABLES

4.1 Hosts, guests and their inter-relationships. 26

4.2 An effect of the modification in the merging process. 30

4.3 Effect of different similarity coefficients on MACE solutions. . · 44

4.4 An example of blocking machine assignment 46

4.5 A comparison of alternative operating rules. 52

4.6 Effect of CAF in WUBC solutions. 53

4.7 An example of cell expansion in CAA. 57

4.8 Effect of CAF on CAA solutions. 58

4.9 Effect of CSUL on CAA solutions. 59

4.10 An example of artificial seeds. 63

4.11 An example of ideal seeds. 64

5.1 Effect of shape parameters on inner-cell and off-diagonal densities. 74

5.2 Factors of the experiment. 80

5.3 Values of fine-tuning parameters of the selected techniques. . 82

5.4 Main effects of the insensitive factors. 83

5.5 Superior PF/MG-F technique(s) among the selected ones in terms of grouping efficiency. 94

5.6 Superior PF/MG-F technique(s) among the selected ones in terms of work-load balance and under-utilization measures. . 95

1. INTRODUCTION

Accelerated growth rates in manufacturing technologies and the production of many new and diverse products call for the constant improvement in production philosophies. As the scope of manufacturing activities has evolved, **Cellular Manufacturing (CM)** has gained considerable attention from both industry and academia. This growing interest is partly due to the role of CM as a base for integrating Computer-Aided-Design (CAD) and Computer-Aided-Manufacturing (CAM) systems. Successful Japanese adoption of manufacturing cells in realizing Just-In-Time (JIT) systems has increased the premise of CM in planning integrated systems.

Group Technology (GT) is a philosophy that capitalizes on similar and recurrent activities by bringing together and organizing common concepts, principles, problems and tasks to improve productivity. CM is an application of the GT philosophy to production environments. CM seeks to rationalize small- and medium-size batch production by identifying and clustering together related parts and dedicated machines such that design and manufacturing functions can take advantage of their similarities.

Manufacturing cells consist of a collection of dissimilar machines to process a specific family of parts. This physical arrangement of a discrete parts manufacturing shop differs significantly from a job shop or a flow shop layout. Job shops contain general purpose machines located by function to gain flexibility. Long and variable unit production times, large number of setups together with long and variable setup times, and high in-process inventory levels to provide for large product variety are among the characteristics of job shop systems. Moreover, the workforce is highly skilled to operate general purpose machinery in a job shop environment. On the contrary, flow shop systems are characterized by means of special purpose single-function machines organized in manufacturing lines, short lead times, low in-process inventory levels, and high production rates. However, flow shop systems are vulnerable to machine breakdowns and changes in product design. Despite short and constant unit production times, the applicability of flow shops to small or medium volume manufacturing cannot be economically justified [4,18,19,45].

CM has various advantages over job shop type systems. Application of CM reduces material handling, frequency and duration of setups, in-process inventories, lead times, and cost of tooling relative to job shop systems [4,18,19,37]. Increased operator mobility and responsibility within a team work improves human relations and job satisfaction in a CM environment. Product quality is also improved and amount of rework is reduced by direct involvement of cell workers to quality control activities. In addition, capacity planning and material planning and control are simplified together with reduced expediting [19,20,21,45].

On the other hand, CM has some drawbacks. Reducing shop flexibility is the most significant disadvantage. Cell lives depend on changes in product demand and product mix. Transformation into CM systems might require additional investment in equipment through machine duplication. Moreover, rearrangement of facilities demands both time and money. Furthermore, implementation of a CM system may lead to reduction in machine utilizations because of machine duplications and the elimination of non-cell parts. Distortion in flows and performance can occur if some of the non-cell parts are assigned to the cells. Existence of such non-cell parts may increase job flowtimes and tardiness. Lastly, manufacturing cells are sensitive to machine breakdowns and they need higher emphasis on maintenance activities.

CM is usually introduced in a traditional job shop environment by rearranging the existing equipment and/or by acquiring new equipment. Such a structural transformation in manufacturing processes affects all functional areas of the entire organization. In designing an appropriate CM system, it is essential to characterize the *decisions* to be made and the related *criteria* for evaluation.

Decisions related to the design of CM systems can be divided into two categories, these being: structural decisions and operational decisions [43]. *Structural decisions* involve selection of part families, selection of machine types and number of machines, determination of part routings, identification of manufacturing cells, type and number of material handling equipment, specifications of operators, tools and fixtures in each cell, and layout type of both the cells and the shop. *Operational decisions* include detailed job designs, organization of supervisory and support personnel, inspection and maintenance procedures, cost control and incentive systems, design or modification of production planning and control procedures, and reorganization of hardware and software of information system.

Criteria for evaluating alternatives can be grouped into two contradictory classes: system structure criteria, and criteria for performance evaluation. *System structure criteria* contain equipment relocation costs, extra investment requirements, cell flexibility, number and sizes of cells, floor space requirements, existence of intra- and/or inter-cell movements of parts, operators and materials, extent to which parts are completed in their assigned cells and the ratio of the parts handled by cells to total amount of parts in the original shop. *Performance evaluations* can be system oriented as well as job oriented. Equipment and labor utilizations, level of work-in-process inventories, queue lengths, setup times and load balances are system oriented measures. Some job oriented criteria are job output rates, waiting times, transportation times, lateness, rework and scrap rates.

Although there is no exact decision sequence in the design of CM systems, there exists a tendency that structural decisions precede operational decisions. Since CM adoption is usually performed in a job shop environment, selection of parts, machine tools and the related routing is apparent. Cell formation is the first, and the most important phase of the design process. This initial decision influences all other decisions involved in the design of CM systems. During this stage, machine groups of functionally dissimilar types are placed together and are dedicated to the manufacture of a specific range of parts. Consequently, associated cell properties of suggested machine clusters are evaluated in this stage. This critical step in the design of a CM system is entitled as **Part Family Machine Group Formation (PF/MG-F)** problem.

The PF/MG-F problem in designing CM systems was introduced by Burbidge [5] during early 1970s. The PF/MG-F problem is an area in which much research has been conducted since Burbidge's pioneering work. At least in an abstract form, the PF/MG-F problem can be somewhat well structured. Unfortunately, the PF/MG-F problem belongs to the $\mathcal{NP}$-Complete class [29,2]. Therefore a large number of PF/MG-F heuristics has been designed for obtaining an applicable PF/MG-F solution. These techniques can be of valuable assistance in the design process of CM systems.

This thesis involves an analysis of the state-of-art PF/MG-F techniques. Specifically, six such promising techniques have been investigated in detail. These techniques have been modified, and possible extensions have been made. Moreover, the six PF/MG-F techniques have been compared according to three performance indices by means of artificially generated test problems. In the following chapter, the related literature will be reviewed. The PF/MG-F problem, and the efficiency measures designed to evaluate the PF/MG-F solutions will be investigated in the third chapter. The fourth chapter will consist of the detailed analyses of the selected techniques. The experimentation where comparisons are made will be described and the results obtained will be discussed in the fifth chapter. Finally, conclusions and suggestions for further research will be addressed.

2. OVERVIEW OF PF/MG-F TECHNIQUES

The PF/MG-F problem has been analyzed extensively in the literature. A number of classification schemes has been proposed [3,24,43]. A framework for cell formation is developed in this study by differentiating between descriptive and analytical techniques (see Figure 2.1). In the second level of the suggested classification scheme, PF/MG-F approaches are divided into subclasses according to their focus of interest. These subclasses depend on whether the grouping techniques employ part or machine characteristics or both.

Descriptive techniques include both non-algorithmic techniques and evaluative methods. Some of the techniques that form part families by making use of *part attributes* are as follows: visual examination of parts spectrum, part family identification by part name or part function, clustering major components that exist in a product structure of an assembly, and analysis of similarities in part codes generated by any classification and coding system.

Another class of descriptive techniques employ *machine attributes*. De Beer and De Witte [15] visually examined the matrices that they constructed from routing information. In this technique, machines are grouped by considering divisibility of machines of the same type. Parts are allocated to machine groups by taking the divisibility of manufacturing operations into account.

The third class of descriptive techniques employ both *part and machine attributes* simultaneously. Production Flow Analysis and Component Flow Analysis are two famous techniques in this subcategory. Burbidge [5], in his pioneering work described Production Flow Analysis as being the formation of cells and the assignment of families through a progressive analysis

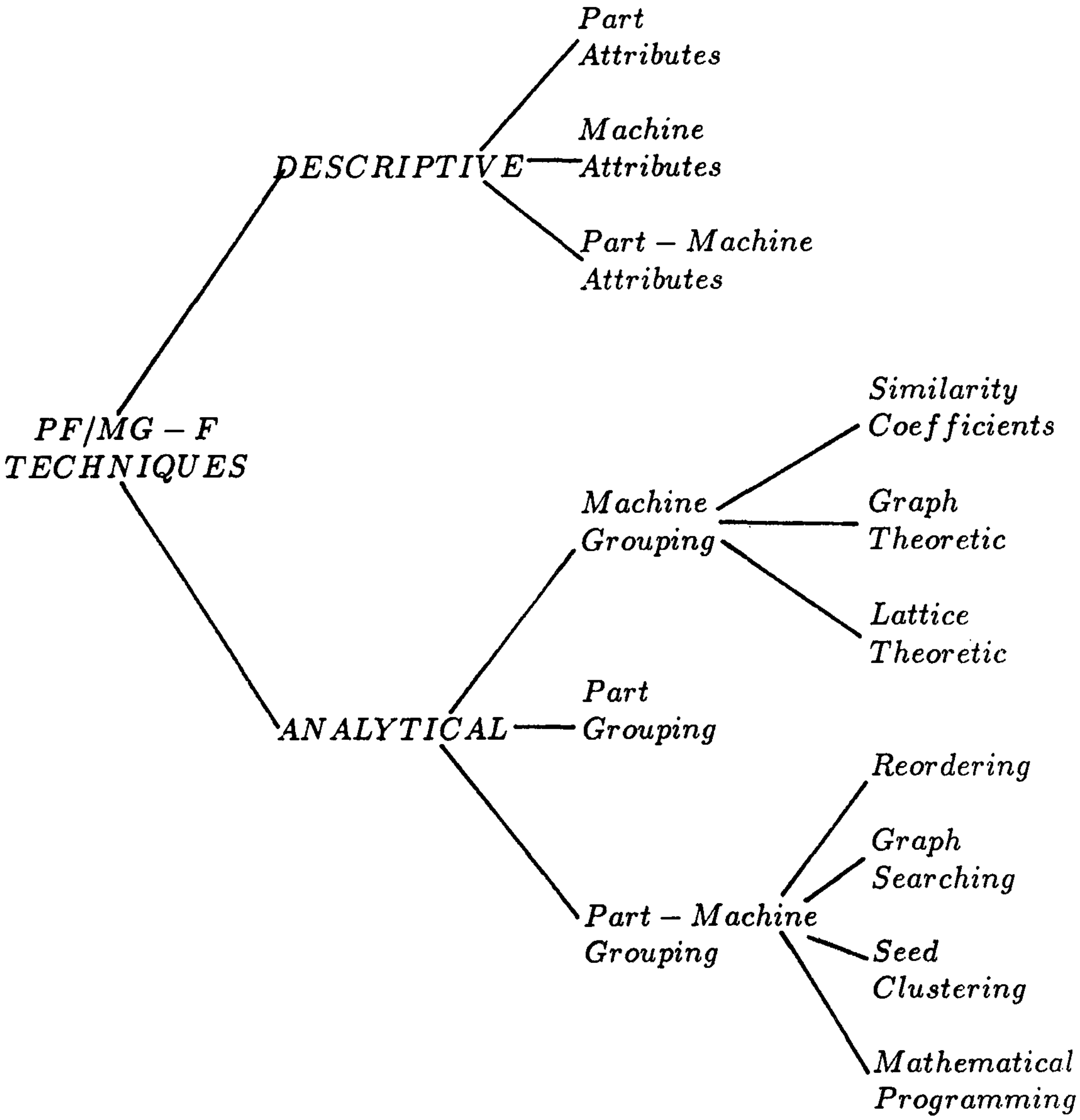


Figure 2.1: A taxonomy of PF/MG-F techniques.

of 'route cards'. Production Flow Analysis consists of a series of subjective evaluations [6]. Factory flow analysis is performed to reduce the number of unnecessary interdepartmental routings of parts. Consequently, departmental flow analysis is carried out. It consists of group analysis and line analysis. Group analysis is concerned with the identification of cells while line analysis attempts to adjust the flow patterns and manufacturing loads. El-Essawy and Torrance [17] suggested Component Flow Analysis to start with the entire part mix instead of dividing the shop into departments. This manual technique is quite similar to Burbidge's work.

Most of the work done in the PF/MG-F problem has been by means of **analytical techniques**. The majority of analytical techniques follow the steps described below. After part and machine populations are selected for possible cellular manufacturing, routings are determined. Candidate cells are identified and part families and machine clusters are assigned to each other to form the candidate cells. Thereafter, the candidate cells are evaluated against various performance criteria. Based on the evaluation, either the candidate cells are established or the preceding steps are repeated. Analytical techniques can be further divided into machine grouping, part grouping and part-machine grouping techniques. *Machine grouping* methods initially identify machine clusters and then assign parts to these clusters. Algorithms of this nature contain similarity coefficient methods, graph-theoretic and lattice-theoretic combinatorial algorithms.

Similarity coefficient is a measure of similarity between each pair of machines, and shows the degree to which the same set of parts can be processed on both machines. The concept of similarity coefficient in PF/MG-F problem was first introduced by McAuley [31]. He developed a procedure, Single Linkage Cluster Analysis, which makes use of a Jaccard's similarity coefficient [38]. This similarity measure is defined for each machine pair to be the number of parts routed through both machines divided by the number of parts processed on at least one of the machines. Single Linkage Cluster Analysis groups machines if their similarity coefficients are greater than a prespecified value. After all machines are clustered, parts are allocated to machine clusters by examining their routings. De Beer and De Witte [15] discussed that Jaccard's similarity coefficient fails when one of the machines process a larger number of parts than the other. De Witte [16] introduced three similarity coefficients by assigning priorities to machines based on their availabilities. He proposed a hierarchical clustering procedure using these coefficients. Waghodekar and Sahu [42] suggested an algorithm called MACE based on the similarity coefficients of the product type. This algorithm is one of the techniques that will be analyzed in more detail.

Rajagopalan and Batra [39] described the first graph-theoretic method for PF/MG-F. The vertices of the graph correspond to machines and arc weights are Jaccard's similarity coefficients. A clique is a maximal complete subgraph. Their algorithm uses cliques of the machine-graph as candidate clusters of machines. Rajagopalan and Batra defined a threshold value to reduce the number of cliques in the graph, and discussed a procedure for selecting this threshold value. Arcs having weights less than the selected value are eliminated from the machine-graph. After parts are assigned to candidate machine clusters, the resultant cells are evaluated.

Lattice-theoretic combinatorial approaches constitute the last type of machine grouping techniques reported in the PF/MG-F literature. Purcheck [34] applied a logical division scheme and discussed the combinatorial characteristics of the grouping problem. Moreover, he designed a technique based on an initial clustering of machines by means of host-guest relationships [33,35]. Hosts are the parts whose routing codes include codes of the remaining parts (guests). A production line is assumed to be materialized for each host. Guests can be processed in one of the lines characterized by the related hosts. Machines in each hypothetical production line form an initial machine cluster, defining a candidate cell. These production lines are hierarchically joined until the original job shop is obtained by considering all possible mergings of candidate cells. This technique will also be analyzed in more detail. Recently, Vakharia and Wemmerlöv [40] have extended the idea of combinatorial grouping to cover operation sequences of parts where a part assignment scheme has been proposed.

Part grouping techniques identify part families prior to machine assignments in forming candidate cells. Cluster Analysis is such an approach designed by Carrie [8]. Routing information is used to construct a similarity matrix representing the degree to which pairs of parts are processed on the same set of machines. This similarity looks alike the coefficients described in the machine grouping subcategory. The algorithm identifies a specific family as a collection of parts having high similarity coefficients between each other. Initial families are identified as a set of parts having higher similarities than a prespecified minimum acceptable level of similarity. Remaining parts are added into the initial families by means of successive decreases in this threshold value. After all parts are grouped into families, machine requirements of each part cluster are calculated. Machine loads are then used to determine the final form of each manufacturing cell dedicated to a specific part family.

Part-machine grouping involves quite a number of techniques that identify manufacturing cells by means of simultaneous and/or subsequent treatment of both parts and machines. Routing information is usually the only relation considered as a base for integrated part family identification and machine grouping. Part-machine grouping techniques can be classified further into reordering, graph searching, seed clustering and mathematical programming techniques according to how they generate the PF/MG-F solutions.

McCormick, Schweitzer and White [32] introduced the reordering concept of machine-part incidence matrix. Rows of the machine-part incidence matrix correspond to machines whereas columns correspond to parts. Each element of the incidence matrix is 'one' if there exists a routing relation between the associated column and row, otherwise it is 'zero'. Mc Cormick et al. developed the Bond Energy Algorithm which tries to increase the total bond energy of the matrix. Bond energy of two adjacent binary vectors is defined as their inner product. Bond energy of the incidence matrix is the sum of the bond energies of all columns and rows. The algorithm permutes rows and columns to obtain mutually exclusive clusters of 'ones' in the matrix, if they exist. King [23] suggested the Rank Order Clustering algorithm which reads each row or column as a binary word. Consequently, integer equivalents of binary words are calculated. Rows and columns are reordered successively in descending order of integer equivalents. Those iterations are terminated when no change is encountered. King and Nakornchai [24] modified this algorithm by utilizing a new data structure and a sorting mechanism. Later, Chandrasekharan and Rajagopalan [11] modified the Rank Order Clustering algorithm. They used King's iterations twice to obtain an incidence matrix containing a rectangular block of 'ones' at its top-left corner. This rectangular block represents a candidate cell. The corresponding columns of the candidate cell are eliminated from the incidence matrix and the procedure is initiated again. Consequently, these candidate cells are successively merged by means of a Jaccard's similarity coefficient. This algorithm will also be examined in more detail. Askin and Subramanian [1] added cost based criteria to tune the solutions obtained from King's algorithm. Direct Clustering Algorithm by Chan and Milner [9] is another reordering algorithm. It generates partial solutions for a subset of parts by decomposing the incidence matrix. Each family found in the previous iteration is considered as a super-part in the current iteration. In this manner, iterations are carried out until all parts are grouped into families.

Graph searching algorithms select a key machine or part according to a prespecified criterion. A bipartite graph generated by parts and machines is breadth-first searched by taking the key as root. Each search is performed to identify a candidate cell. A 'yes/no' decision of whether to include the node that characterizes a machine type or a part is made at each visit of the search. If the decision is 'yes', then the related node is added to the candidate cell. Vertices corresponding to parts and machines identified in the candidate cell are eliminated from the graph as soon as the search is terminated. Ballakur and Steudel [3] chose the key to be one of the machines. The criterion in selecting the key machine is the maximum work load fraction value. Machines are assigned to the candidate cell according to within-cell utilizations. If the visited node represents a part, the assignment of this part is based on the number of possible within-cell operations. This method will be analyzed in more detail. Chow and Kusiak [27] selected the part with the maximum subcontracting cost as the key. During the search, the machines are always assigned to the cell. The decision for a part is 'yes' if another search rooted at that part which results in all 'yes' decisions does not increase the cell size more than a prespecified value. This technique will also be examined in detail. Kusiak and İbrahim [28] developed a knowledge based system for PF/MG-F problem which uses the algorithm developed by Chow and Kusiak. Vannelli and Kumar [41] suggested another technique using a search mechanism as engine. Their criteria for obtaining mutually exclusive manufacturing cells are cell sizes, number of cells and machine duplication costs.

Seed clustering techniques use various kinds of seeds. A seed is a binary vector. For each candidate cell there is a corresponding seed acting as a nucleus for clustering. If parts (machines) are being clustered, sizes of seeds are equal to number of machines (parts). A part (machine) is assigned to a certain cell if the distance between the corresponding part (machine) vector in the incidence matrix and the dedicated seed is minimal among all seeds. Chandrasekharan and Rajagopalan [10] developed such a seed clustering algorithm, ZODIAC, by modifying MacQueen's k-means method [30]. They suggested an upper bound on the number of possible candidate cells and applied the absolute value (d-1) metric for distances. After generating the required number of seeds, parts and machines are grouped independently into equal number of clusters. These independent clusters are assigned to each other, giving rise to ideal seeds. Further clustering is done by using these ideal seeds to determine the final PF/MG-F solution. This method will also be discussed in detail.

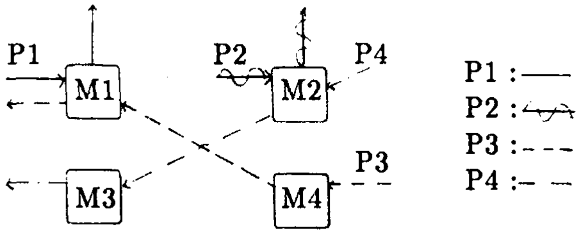
Mathematical Programming techniques employ solution procedures to mathematical model formulations of the entire PF/MG-F problem or an embedded subproblem. Kumar, Kusiak and Vannelli [25] formulated the overall problem as an optimal k-decomposition model of weighted networks. They approximated this quadratic assignment problem by a two-phase procedure. Once cell sizes are fixed, the resultant linear transportation problems for each cell are easily solved. Initially, an intermediate PF/MG-F solution is generated by solving the transportation problems successively. Consequently, an improvement of this intermediate solution is attempted. In addition, Kumar et al. derived bounds on the optimal solution. Later, Kusiak [26] discussed a generalized PF/MG-F concept based on the creation of multiple process plans for one part. An integer programming model was formulated. Co and Araar [14] used mathematical programming to assign operations of parts to machines with the objective of maximizing machine utilizations. They formulated a 0-1 integer programming model to assign jobs to individual machines of the same type. The objective function of the formulation is based on the minimization of the maximum deviation of assigned workload and the available capacity of each machine type. The solution is translated into a machine/part incidence matrix where a search procedure is used to identify the final cells obtained from Rank Order Clustering. Choobineh [13] proposed a linear integer program that considers the economics of production in cells.

3. ANALYSIS OF THE PF/MG-F PROBLEM

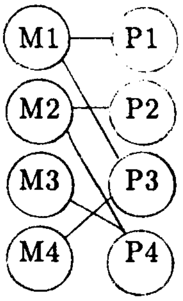
The PF/MG-F problem involves the identification of manufacturing cells formed by clusters of functionally dissimilar machines and the assignment of parts to one of these cells. Cells formed by such machine clusters are dedicated to specific part families based on routing information. In this chapter, the PF/MG-F problem is defined by means of graph theoretical terms. Consequently, efficiency measures designed to evaluate the performances of the PF/MG-F solutions are presented.

3.1 Definition of the PF/MG-F Problem

In any manufacturing environment, machines and parts are related to each other via part routings. This relationship can be abstracted into a bipartite graph. Parts and machine types can be represented as vertices of two distinguished sets. A routing relationship between any part-machine pair is represented by an edge between the corresponding vertices of the graph. A small manufacturing environment having four machine types and four different parts is illustrated in Figure 3.1. The bipartite graph in part (b) of the figure is derived from the process flow chart given in part (a). Each graph can be analyzed by means of incidence matrices. An associated node-to-node incidence matrix is also included in part (b) of the figure. Rows of the incidence matrix are dedicated to machine types, whereas columns correspond to parts. If a routing relationship exists between a certain part–machine pair, the corresponding element in the incidence matrix takes the value of 'one'. Otherwise the corresponding entry is 'zero'. One drawback of this representation is that operation sequences are not considered.

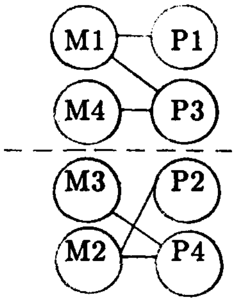


(a) Part Routings



$$\begin{array}{c|cccc|}
 & P1 & P2 & P3 & P4 \\
M1 & 1 & & 1 & \\
M2 & & 1 & & 1 \\
M3 & & & & 1 \\
M4 & & & 1 &
\end{array}$$

(b) Graph and Incidence Matrix (before clustering)



$$\begin{array}{c|cc|cc|}
 & P1 & P3 & P2 & P4 \\
M1 & 1 & 1 & & \\
M4 & & 1 & & \\
\cline{4-5}
M3 & & & & 1 \\
M2 & & & 1 & 1
\end{array}$$

(c) Graph and Incidence Matrix (after clustering)

Figure 3.1: Part routings, graphs and incidence matrices: An example.

Each PF/MG-F solution produces different representation of the same bipartite graph. It is observed from Figure 3.1-c that the incidence matrix contains two diagonal blocks characterized by 'ones'. The existence of such diagonal blocks indicates that the graph can be decomposed into disconnected subgraphs each of which corresponds to a diagonal block. Each diagonal block of 'ones', or the corresponding subgraph identifies a manufacturing cell. One such cell is formed by grouping machines of type-one and -four in order to process both part-one and part-three.

The PF/MG-F problem can be defined as *permuting columns and rows of the incidence matrix so that a block-diagonal structure is obtained.* Desirable PF/MG-F solutions are the ones in which all parts complete all of their manufacturing operations in their assigned cell. For such solutions, there is no inter-cell movement of parts. *Exceptional elements* are the entries of the incidence matrix that do not belong to any block-diagonal structure preventing the solution from being a desirable one. Unfortunately, the majority of the PF/MG-F solutions contain exceptional elements as shown in Figure 3.2. The PF/MG-F problem can alternatively be defined as *the reordering of the incidence matrix so that a minimum number of exceptional elements are obtained, provided that a block-diagonal structure exists.*

The PF/MG-F problem is a *clustering* problem. A clustering problem is defined on a data array $(a_{i,j})(i \in T, j \in P)$ where $a_{i,j}$ measures the strength of the relationship between elements $i \in T, j \in P$. A clustering of the array is obtained by permuting its rows and columns, and should identify subsets of T that are strongly related to subsets of P. In the case of PF/MG-F, T is the set of machine types, P is the set of parts, and $(a_{i,j})$ is the machine-part incidence matrix. Mc Cormick et al. [32] proposed a measure of effectiveness (ME) to convert the clustering problem into an optimization problem. ME is the sum of all products of horizontally and vertically adjacent elements in the array. The clustering problem is to find permutations of rows and columns of $(a_{i,j})$ maximizing ME. In general, the clustering problem for a p-dimensional array can be stated as p Traveling Salesman Problems (TSPs) [29]. Therefore, the PF/MG-F problem is as hard as solving two TSPs in terms of the computational complexity. It follows that *the PF/MG-F problem is $\mathcal{NP}$-Complete.*

It is possible to use work-load matrices instead of incidence matrices in

$$\begin{array}{r|cccccccccccccccc|}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\
1 & & & & & 1 & & & & & 1 & 1 & & & & & 1 \\
2 & & 1 & & 1 & & & & & 1 & & & & & & & \\
3 & 1 & & & & & & & & & & 1 & & & & & 1 \\
4 & & & & & & 1 & & & & & 1 & & 1 & 1 & 1 & \\
5 & & & & 1 & & & & 1 & & & & & & & 1 & \\
6 & & & 1 & & & & 1 & & & & & 1 & 1 & 1 & & \\
7 & & 1 & & 1 & & & & 1 & 1 & & & & 1 & & & \\
8 & 1 & & 1 & & & & & 1 & & 1 & & & & & & 1 \\
9 & & & & & 1 & & & & & 1 & 1 & & & & 1 & 1 \\
10 & & & & & & 1 & & & & & & 1 & 1 & 1 & 1 &
\end{array}$$

(a) Incidence Matrix : Before clustering

$$\begin{array}{r|cccccccccccccccc|}
 & 4 & 8 & 2 & 9 & 16 & 5 & 11 & 1 & 10 & 12 & 6 & 13 & 3 & 15 & 14 & 7 \\
5 & 1 & 1 & 0 & 0 & & & & & & & & & & + & & \\
2 & 1 & 0 & 1 & 1 & & & & & & & & & & & & \\
7 & 1 & 1 & 1 & 1 & & & & & & & & + & & & & \\
3 & & & & & 1 & 0 & 1 & 1 & 0 & & & & & & & \\
9 & & & & & 1 & 1 & 1 & 0 & 1 & & & & & + & & \\
8 & & + & & & 1 & 0 & 0 & 1 & 1 & & & & + & & & \\
1 & & & & & 1 & 1 & 1 & 0 & 1 & & & & & & & \\
6 & & & & & & & & & & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\
4 & & & & & & & + & & & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
10 & & & & & & & & & & 1 & 1 & 1 & 0 & 1 & 1 & 0
\end{array}$$

(b) Incidence Matrix : After clustering
(+):Exceptional element

Figure 3.2: An example of PF/MG-F having exceptional elements.

$$\begin{array}{r|cccccccccccccccc|}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\
\hline
1 & & & & & 05 & & & & & 46 & 37 & & & & & 51 \\
2 & & 61 & & 12 & & & & & 71 & & & & & & & \\
3 & 57 & & & & & & & & & & 49 & & & & & 55 \\
4 & & & & & & 39 & & & & & 24 & & 65 & 54 & 26 & \\
5 & & & & 27 & & & & 13 & & & & & & & 19 & \\
6 & & & 08 & & & & 09 & & & & & 45 & 81 & 37 & & \\
7 & & 31 & & 86 & & & & 17 & 51 & & & & 71 & & & \\
8 & 41 & & 12 & & & & & 71 & & 66 & & & & & & 47 \\
9 & & & & & 07 & & & & & 48 & 09 & & & & 64 & 26 \\
10 & & & & & & 54 & & & & & & 11 & 18 & 33 & 42 & \\
\hline
\end{array}$$

(a) Work–Load Matrix : Before clustering

$$\begin{array}{r|cccccccccccccccc|}
 & 4 & 8 & 2 & 9 & 16 & 5 & 11 & 1 & 10 & 12 & 6 & 13 & 3 & 15 & 14 & 7 \\
\hline
5 & 27 & 13 & 00 & 00 & & & & & & & & & & 19 & & \\
2 & 12 & 00 & 61 & 71 & & & & & & & & & & & & \\
7 & 86 & 17 & 31 & 51 & & & & & & & & 71 & & & & \\
3 & & & & & 55 & 00 & 49 & 57 & 00 & & & & & & & \\
9 & & & & & 26 & 07 & 09 & 00 & 48 & & & & & 64 & & \\
8 & & 71 & & & 47 & 00 & 00 & 41 & 66 & & & & 12 & & & \\
1 & & & & & 51 & 05 & 37 & 00 & 46 & & & & & & & \\
6 & & & & & & & & & & 45 & 00 & 81 & 08 & 00 & 37 & 09 \\
4 & & & & & & & 24 & & & 00 & 39 & 65 & 00 & 26 & 54 & 00 \\
10 & & & & & & & & & & 11 & 54 & 18 & 00 & 42 & 33 & 00 \\
\hline
\end{array}$$

(b) Work–Load Matrix : After Clustering

Figure 3.3: An example of work-load matrix employed in PF/MG-F.

solving PF/MG-F problems. Each entry of a work-load matrix represents the machine fraction of the associated machine type. Machine fraction for any operation of any part is defined as the percentage of machine capacity allocated to this operation. All information contained in incidence matrices can also be obtained from work-load matrices (see Figures 3.2, 3.3). Moreover, the sum of elements of specific row of a work-load matrix indicates the number of machines desired of that type. For example, there should be at least three machines of type-eight as indicated in Figure 3.3, since the sum turns out to be 2.37 machines.

Working with individual machines instead of machine types gives an opportunity to eliminate some of the exceptional elements. Because of the slack in machine requirements of type-eight, the exceptional element created by part-eight can be eliminated by allocating a machine to this operation. An addition of a machine of type-eight to the first cell would eliminate the exceptional element due to part-eight and therefore improve the current solution in Figure 3.3. Application of the same analogy to other exceptional elements can lead to a better solution. The result of applying this approach to the example given by Figure 3.3 is shown in Figure 3.4-a.

Finally, routing alternatives may exist among parts. By considering alternative routings, further improvements in the PF/MG-F solution could occur. For the example considered, if it is possible to use machine type-nine instead of -eight in processing part-three, the corresponding exceptional element would be eliminated. In addition, if part-fifteen was rerouted to machine type-seven instead of type-five, the desirable block-diagonal solution with no exceptional elements would be obtained (Figure 3.4-b).

3.2 Efficiency Measures

One of the important issues involved in the design of CM systems is the evaluation of PF/MG-F solutions. Although there have been quite a number of techniques developed, the evaluation of PF/MG-F solutions has remained somewhat qualitative such as measuring cell independence or flexibility [2,10,43,44,45]. Some of the common quantitative efficiency measures are the number of inter- and inner-cell moves, number and cost of duplicated equipment, number of parts removed from the system, and machine utilizations [1,3,10,11,25,26,27,28,34,35,40,41,42]. In particular, the PF/MG-F techniques have usually been compared to each other by counting the exceptional elements generated in the solutions [3,10,42,43]. The majority of the suggested efficiency measures lack a quantitative standard for systematically comparing different solutions of the same PF/MG-F problem. However, Chandrasekharan and Rajagopalan [10,12] reported an interesting criterion for measuring clustering efficiency. This criterion weighs the concentration of 'ones' in the diagonal blocks of the incidence matrix and the number of exceptional elements in off-diagonal area. A modification of this criterion and some other measures defined in this section will be used in the comparison study.

$$\begin{array}{r|cccccccccccccccc|}
 & 4 & 8 & 2 & 9 & 16 & 5 & 11 & 1 & 10 & 12 & 6 & 13 & 3 & 15 & 14 & 7 \\
\hline
5a & 27 & 13 & 00 & 00 & & & & & & & & & & 19 & & \\
2a & 06 & 00 & 30 & 36 & & & & & & & & & & & & \\
2b & 06 & 00 & 31 & 35 & & & & & & & & & & & & \\
7a & 43 & 09 & 15 & 26 & & & & & & & & & & & & \\
7b & 43 & 08 & 16 & 25 & & & & & & & & & & & & \\
8a & 00 & 71 & 00 & 00 & & & & & & & & & 04 & & & \\
3a & & & & & 27 & 00 & 25 & 28 & 00 & & & & & & & \\
3b & & & & & 28 & 00 & 24 & 29 & 00 & & & & & & & \\
9a & & & & & 26 & 07 & 09 & 00 & 48 & & & & & & & \\
8b & & & & & 24 & 00 & 00 & 20 & 33 & & & & 04 & & & \\
8c & & & & & 23 & 00 & 00 & 21 & 44 & & & & 04 & & & \\
1a & & & & & 26 & 02 & 19 & 00 & 23 & & & & & & & \\
1b & & & & & 25 & 03 & 18 & 00 & 23 & & & & & & & \\
4a & & & & & 00 & 00 & 24 & 00 & 00 & & & & & & & \\
6a & & & & & & & & & & 23 & 00 & 40 & 04 & 00 & 19 & 04 \\
6b & & & & & & & & & & 22 & 00 & 41 & 04 & 00 & 18 & 05 \\
4b & & & & & & & & & & 00 & 19 & 33 & 00 & 13 & 27 & 00 \\
4c & & & & & & & & & & 00 & 20 & 32 & 00 & 13 & 27 & 00 \\
10a & & & & & & & & & & 06 & 27 & 09 & 00 & 21 & 16 & 00 \\
10b & & & & & & & & & & 05 & 27 & 09 & 00 & 21 & 17 & 00 \\
7c & & & & & & & & & & 00 & 00 & 72 & 00 & 00 & 00 & 00 \\
9b & & & & & & & & & & 00 & 00 & 00 & 00 & 64 & 00 & 00
\end{array}$$

(a) Work-Load Matrix : Using number of physical machines

$$\begin{array}{r|cccccccccccccccc|}
 & 4 & 8 & 2 & 9 & 16 & 5 & 11 & 1 & 10 & 12 & 6 & 13 & 3 & 15 & 14 & 7 \\
\hline
5a & 27 & 13 & 00 & 00 & & & & & & & & & & & & \\
2a & 06 & 00 & 30 & 36 & & & & & & & & & & & & \\
2b & 06 & 00 & 31 & 35 & & & & & & & & & & & & \\
7a & 43 & 09 & 15 & 26 & & & & & & & & & & & & \\
7b & 43 & 08 & 16 & 25 & & & & & & & & & & & & \\
8a & 00 & 71 & 00 & 00 & & & & & & & & & & & & \\
3a & & & & & 27 & 00 & 25 & 28 & 00 & & & & & & & \\
3b & & & & & 28 & 00 & 24 & 29 & 00 & & & & & & & \\
9a & & & & & 26 & 07 & 09 & 00 & 48 & & & & & & & \\
8b & & & & & 24 & 00 & 00 & 20 & 33 & & & & & & & \\
8c & & & & & 23 & 00 & 00 & 21 & 44 & & & & & & & \\
1a & & & & & 26 & 02 & 19 & 00 & 23 & & & & & & & \\
1b & & & & & 25 & 03 & 18 & 00 & 23 & & & & & & & \\
4a & & & & & 00 & 00 & 24 & 00 & 00 & & & & & & & \\
6a & & & & & & & & & & 23 & 00 & 40 & 04 & 00 & 19 & 04 \\
6b & & & & & & & & & & 22 & 00 & 41 & 04 & 00 & 18 & 05 \\
4b & & & & & & & & & & 00 & 19 & 33 & 00 & 13 & 27 & 00 \\
4c & & & & & & & & & & 00 & 20 & 32 & 00 & 13 & 27 & 00 \\
10a & & & & & & & & & & 06 & 27 & 09 & 00 & 21 & 16 & 00 \\
10b & & & & & & & & & & 05 & 27 & 09 & 00 & 21 & 17 & 00 \\
7c & & & & & & & & & & 00 & 00 & 72 & 00 & 19 & 00 & 00 \\
9b & & & & & & & & & & 00 & 00 & 00 & 12 & 64 & 00 & 00
\end{array}$$

(b) Work-Load Matrix : Employing routing alternatives
part-3 : (T6–T9) & part-15 : (T4–T7–T9–T10)

Figure 3.4: An example showing the effects of extensions in the PF/MG-F definition.

The efficiency of each PF/MG-F solution can be measured by its work-load matrix. Three efficiency indices suggested for evaluating the PF/MG-F solutions are reported in this section. The first measure is the grouping efficiency which penalizes exceptional elements and considers inner-cell densities. This measure is a modified and extended version of what Chandrasekharan and Rajagopalan reported, whereas the remaining two are developed originally. The second measure is concerned with the inner-cell load balances. The last measure focuses on under-utilizations of individual machines. Discussion of these efficiency measures require some notation and definitions:

i : machine type index ($i = 1, \ldots, T$),
j : part index ($j = 1, \ldots, P$),
k : cell index ($k = 1, \ldots, K$),

$CM(k)$: index set of machine types that are assigned to cell k,
$CP(k)$: index set of parts that are assigned to cell k,
$AC(j)$: cell index to which part j is assigned,

$N_{k,i}$: number of physical machines of type i in cell k,
S_k : number of types of machines in cell k,

DC_i : annual depreciation cost of a machine of type i [\$/machine-year],
$WL_{i,j}$: annual work-load of machine type i induced by part j [number of machines],
$TU_{k,i}$: total usage of machines of type i in cell k [number of machines],

$TWLC_j$: total work-load cost of part j [\$],

$$TWLC_j \doteq \sum_{i=1}^{T} WL_{i,j} \times DC_i$$

WCC_j : work-load cost of part j in its assigned cell [\$],

$$WCC_j \doteq \sum_{i \in CM(AC(j))} WL_{i,j} \times DC_i$$

$WLCE_j$: work-load cost of exceptional elements belonging to part j [\$],

$$WLCE_j \doteq TWLC_j - WCC_j$$

$FP_{i,j}$: field potential value of clustering both part j and machine type i into the same cell,

$$FP_{i,j} \doteq TWLC_j \times DC_i \times N_{AC(j),i}$$

$AP_{i,j}$: Assignment potential of clustering both part j and machine type i into the same cell,

$$AP_{i,j} \doteq \begin{cases} FP_{i,j} & \text{if } WL_{i,j} > 0, \\ 0 & \text{if } WL_{i,j} = 0. \end{cases}$$

$MWL_{k,i}$: mean workload of machines of type i assigned to cell k,

$$MWL_{k,i} \doteq \frac{1}{N_{k,i}} \times \sum_{j \in CP(k)} WL_{i,j}$$

MCL_k : mean cell-load in cell k,

$$MCL_k \doteq \frac{1}{S_k} \times \sum_{i \in CM(k)} MWL_{k,i}$$

$UU_{k,i}$: total under-utilization of machine type i in cell k,

$$UU_{k,i} \doteq N_{k,i} - TU_{k,i}$$

Grouping efficiency is a combined measure made up of two parts. The first part of the grouping efficiency is a measure of inter-cell flows created by exceptional elements. The cost of any exceptional element depends on the work-load of the operations to be completed outside the cell and the annual depreciation costs of the associated machines. Inter-cell flow efficiency, μ_1, is defined as the normalized cost of all exceptional elements:

$$\mu_1 \doteq 1 - \frac{\sum_{j=1}^{P} WLCE_j}{\sum_{j=1}^{P} TWLC_j} \tag{1}$$

The second part of the grouping efficiency is a weighed estimate of the inner-cell densities. Each block-diagonal entry in the incidence matrix is weighed by multiplying the column and row weights. The row weights are the annual depreciation costs of the corresponding machines. The column weights identify the annual cost of the machine requirements for the corresponding part. In this manner, a potential field is defined for each diagonal block representing a manufacturing cell. The density of a cell is the normalized total potential in this field. Inner-cell efficiency, μ_2, of any PF/MG-F solution is defined as:

$$\mu_2 \doteq \frac{\sum_{j=1}^{P} \sum_{i \in CM(AC(j))} AP_{i,j}}{\sum_{j=1}^{P} \sum_{i \in CM(AC(j))} FP_{i,j}} \tag{2}$$

Grouping efficiency, μ, is obtained by the convex combination of inter-cell flow and inner-cell efficiencies:

$$\mu = \alpha \times \mu_1 + (1 - \alpha) \times \mu_2, \quad \alpha \in [0, 1] \tag{3}$$

The parameter α can be interpreted as an indication of whether inner-cell efficiencies or inter-cell flows are more important to the decision maker. A large value of α gives more weight to exceptional elements. As α approaches unity, there becomes a tendency to eliminate all exceptional elements and the PF/MG-F solution usually is a job shop system identified as the existence of single cell. On the other hand, a very small α value indicates that inner-cell efficiency is more important than inter-cell flow efficiency, which leads to small-sized cells. Since Cellular Manufacturing lies in between, moderate values of α are suggested. Some examples with different values of grouping efficiency are illustrated in Figure 3.5. These are simple cases where parts, machines, and operations are assumed to be the same.

Work-load balance measure, β, is the second efficiency measure used in this study. It shows the degree of machine load balance in each cell. If all machines in each cell are evenly loaded, then the work-load index will take on a value which is very close to one. This efficiency measure is defined as the weighed sum of square of the deviations between the mean cell load and individual machine loads in each cell:

$$\beta \doteq 1 - \sqrt{\frac{\sum_{k=1}^{K} \sum_{i \in CM(k)} (MWL_{k,i} - MCL_k)^2 \times N_{k,i} \times DC_i}{\sum_{k=1}^{K} \sum_{i \in CM(k)} N_{k,i} \times DC_i}} \tag{4}$$

The last efficiency measure, γ, shows the *under-utilization* levels of the

$$\begin{bmatrix} + & + & + & + & + & + \\ + & + & + & + & + & + \\ + & + & + & + & + & + \\ + & + & + & + & + & + \\ + & + & + & + & + & + \end{bmatrix}$$

$(a)\mu_1 = 1, \mu_2 = 1,$
$K = 1$

$$\begin{bmatrix} + & & + & & + & \\ & + & & + & & + \\ + & & + & & + & \\ & + & & + & & + \\ + & & + & & + & \end{bmatrix}$$

$(b)\mu_1 = 1, \mu_2 = 1/2,$
$K = 1$

$$\begin{bmatrix} + & + & + & & + & \\ + & + & + & & & \\ & & & + & + & + \\ & + & & + & + & + \\ & & & + & + & + \end{bmatrix}$$

$(c)\mu_1 = 15/17, \mu_2 = 1,$
$K = 2$

$$\begin{bmatrix} + & + & + & & & \\ + & + & + & & & \\ & & & + & + & + \\ & & & + & + & + \\ & & & + & + & + \end{bmatrix}$$

$(d)\mu_1 = 1, \mu_2 = 1,$
$K = 2$

$$\begin{bmatrix} + & + & & & & \\ + & + & + & & & \\ & & & + & + & + \\ & & & + & + & + \\ & & & & + & + \end{bmatrix}$$

$(e)\mu_1 = 1, \mu_2 = 13/15,$
$K = 2$

$$\begin{bmatrix} + & + & & & + & \\ + & + & + & & & \\ & & & + & + & + \\ & + & & + & + & + \\ & & & & + & + \end{bmatrix}$$

$(f)\mu_1 = 13/15, \mu_2 = 13/15,$
$K = 2$

Figure 3.5: Some examples of grouping efficiency values

individual machines in terms of depreciation costs. Individual machine utilizations are multiplied by the associated annual depreciation costs and normalized:

$$\gamma \doteq \frac{\sum_{k=1}^{K} \sum_{i \in CM(k)} UU_{k,i} \times DC_i}{\sum_{k=1}^{K} \sum_{i \in CM(k)} DC_i} \tag{5}$$

This efficiency measure considers only the machines that are assigned to cells. Exceptional elements can be taken care by one of two approaches, these being: extra investment, and subcontracting. There is a trade-off involved in deciding which approach to take. In other words, either the associated machine is duplicated to remove the adverse effects of exceptional elements, or the corresponding operation is subcontracted. In the former case, the total number of machines of the associated type is increased by one. This leads to an increase in both of the numerator and denominator of (5) by an amount equal to the annual depreciation cost of that machine type. In the latter case, the work-load of the associated machine type is reduced, causing the numerator to increase again.

4. SELECTED PF/MG-F TECHNIQUES

A variety of PF/MG-F techniques which was summarized in the previous section has been reported in the literature. Some of them were developed by academic researchers while others emerged as a result of practical applications. The techniques which were developed earlier are not mathematically oriented. Implementation of these techniques on the computing environment is usually more difficult than mathematically oriented ones. Most of the techniques proposed for the PF/MG-F problem do not consider any performance criterion during their algorithmic process. The value of the solutions generated by such techniques are limited because of the absence of the consideration of performance criteria that are important in driving to satisfactory PF/MG-F solutions. Moreover, most procedures designed to form manufacturing cells generate different solutions to the same PF/MG-F problem stated in different forms of input.

A subset of the PF/MG-F techniques was selected for a detailed analysis and this selection was based on the following criterion. Either the final versions of well-known PF/MG-F techniques or the promising, recently developed techniques are chosen for further analysis and mutual comparison. These techniques are lattice–theoretic *comb*inatorial *gr*ouping (COMBGR) [34,33,35], *mod*ified *r*ank *o*rder *c*lustering (MODROC) [23,24,10], *m*achine–*c*omponent *ce*ll formation (MACE) [42], and *w*ithin-cell *u*tilization *b*ased *c*lustering (WUBC) [3], *z*ero-*o*ne *d*ata – *i*deal-*s*eed *a*lgorithm for *c*lustering (ZODIAC) [11,12], and *c*ost *a*nalysis *a*lgorithm (CAA) [27].

The selected PF/MG-F techniques are all analytical techniques. These techniques require routing information between machine types and parts. All solutions generated by any of the selected techniques are independent of any special block-diagonal structure embedded in input. All of the selected techniques generate unique solutions to the same problem fed in different input formats. Moreover, they are computationally efficient. All of the selected techniques neglect possible routing alternatives and also do not consider operation sequences. If it is known a priori that a solution with a perfect block-diagonal structure is possible, then all of the selected techniques will generate this ideal PF/MG-F solution with no exceptional elements.

COMBGR and MACE are techniques that consider only the machine grouping problem. Subsequent to the identification of machine clusters, parts are assigned to the cells. However, part and machine assignments are considered simultaneously and/or subsequently in MODROC, WUBC, ZODIAC and CAA. MODROC is a reordering method whereas WUBC and CAA are graph searching algorithms. ZODIAC is the only seed clustering technique reported in the literature.

COMBGR, MODROC and MACE are hierarchical techniques. In the first stage, candidate cells are generated. Candidate cells are subsequently merged. In the last stage, the final PF/MG-F solution is obtained. Therefore, the candidate cell formation decision of the first stage directly affects the final solution. If a machine-part pair is clustered in the same candidate cell, its assignment in the final solution will also be the same cell. On the other hand, the remaining selected techniques are non-hierarchical methods where cells of the final solution are not characterized totally by intermediate steps. In these techniques, once a machine-part pair is grouped in a previous iteration, it is still possible to reassign the part or the machine to a different cell.

These techniques are analyzed in detail in subsequent sections. Various modifications and extensions that improved the performance of each technique were also realized. Subsequent to presenting a specific technique, modifications and extensions are highlighted. Consequently, each technique is summarized in an algorithmic way. The steps of each technique are illustrated by the same example as given in Appendix A.

4.1 Combinatorial Grouping

COMBGR is a lattice-theoretic hierarchical grouping algorithm developed by Purcheck [34]. The basic advantage of COMBGR is that it generates PF/MG-F solutions without any exceptional elements. However, proposed cells are relatively large leading to almost the same drawbacks encountered in job shop systems.

4.1.1 Description of COMBGR

COMBGR algorithm divides parts into two classes as hosts or guests. Initially, candidate cells are identified in such a way that each host represents a candidate cell. Subsequently, candidate cells are merged successively until the original job shop is obtained. Each merge iteration creates an alternative PF/MG-F solution for the decision maker.

First of all, some terms that will be extensively used in describing the COMBGR technique need to be defined. *Routing code* is the ordered index set of machine types corresponding to all of the operations of a part. If a part has the operation sequence given as "$a \mapsto b \mapsto c \mapsto a \mapsto d \mapsto b \mapsto e$", its routing code is '*abcde*'. A ***host*** is defined as the part having a routing code that cannot be included in the codes of other parts as a subset. Hosts constitute the minimal independent set in terms of routing codes of all parts. The set of routing codes of the hosts contains as subsets all the routing codes of all parts in the analysis. ***Guests*** are the parts whose routing codes are contained in at least one host parts' code. The index set of guests that a host can hold is termed as ***hospitality***, and the index set of hosts in which a guest can be included is called ***flexibility***. These definitions are illustrated by an example given in Table 4.1. For instance, part-1 is a host having hospitality towards parts -3, -5 and -6. Conversely, part-5 is a guest having flexibility between parts -1 and -2.

Parts are initially sorted by size and ordered by code significance to determine the hosts and guests. The *size* of a part is defined as the number of elements in the associated routing code. The *code significance* is the integer equivalent of the binary expression of part codes representing the machines

Size	Part #	Routing Code	Significance	Status	Relation
5	7	*bdefg*	122	Host	9,8,10
	2	*acefg*	117	Host	9,4,10,5
	1	*abcde*	31	Host	6,3,5
4	6	*bcde*	30	Guest	1
3	9	*efg*	112	Guest	7,2
	4	*afg*	97	Guest	2
	8	*def*	56	Guest	7
	3	*ade*	25	Guest	1
2	10	*fg*	96	Guest	7,2
	5	*ac*	5	Guest	2,1

Table 4.1: Hosts, guests and their inter-relationships.

used. If a part uses a machine, the corresponding element in the binary expression is 'one', otherwise it is 'zero'. For example, the code significance of part-7 is 122:

$$122 = 0 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 + 1 \times 2^4 + 1 \times 2^5 + 1 \times 2^6.$$

The following approach leads to the identification of the hosts and guests: A host can include a potential guest only if it has a larger size and greater code significance. Sorting by size and ordering by code significance reduces a $P \times P$ comparison matrix to a $h \times (P - h)$ comparison matrix, where h represents the total number of hosts and P denotes the total number of parts. The number of comparisons reduces from $1/2 \times P \times (P - 1)$ to less than $h \times (P - h) + 1/2 \times h \times (h - 1)$ comparisons.

Each host identifies a candidate cell where the machines of the cell are determined from the host part's routing code. An infinite number of machines of the same type is assumed at this stage. After the candidate cells are formed, they are merged in order to reduce the total machine requirements. This merging is based on the machine-differences between the cells. *Minimal machine-difference* between a pair of cells is defined as the minimal set of machines that do not belong to both of the cells. For each cell pair, there are two sets of uncommon machines. The size of the smaller set is the minimal machine-difference. Minimal machine-differences between cells can be calculated as set differences from routing codes. For instance, machines of type a and c should be added into the cell characterized by host-part-7 in order to merge with host-part-2. As can be observed from Table 4.1, host-part-1 can also be included in the cell formed by host-parts -2 and -7 with no additional machine requirement. Hence, adding machines of type a and c to host-part-7 creates an opportunity of merging host-parts -1 and -2 at the same time. This defines a chain of hosts that can be merged together. The *set combination size* of the chain generated in this case has a value of three. The hosts acting as a nuclei for a merging operation is said to have *forward* relationships. A forward relationship is characterized by the index set of the hosts that can be merged into the host in consideration. Conversely, the index set of hosts into which a specific host can be merged is called an *inverse* relationship. Forward and inverse relationships between the hosts are analogous to hospitality and flexibility relationships between the parts. In the above case, host-2 and host-1 have inverse relationships whereas host-7 has a forward relationship.

All cells can be divided into five classes according to their forward and/or inverse relationships [35]. *Urgent* cells are the ones having single inverse relationships. Each such cell can only be merged into the cell defined by its inverse relationship. *Passive* cells are the cells that have only inverse relationships. They can be merged into other cells specified in their inverse relationships but no other cell can be merged into them. *Active* cells are the cells having no inverse relations. They act as a nucleus in merging operations. *Neutral* cells are identified as the cells with both forward and inverse relationships. *Notional* cells are characterized by having no relationships. Priorities are assigned to these classes of cells where urgent cells have the lowest *priority ranking* while notional cells have the highest.

The basic starting criterion in merging candidate cells is to look at the maximum set combination sizes. The cells that are associated with the maximum set combination size are merged together. Each merging of the candidate cells may destroy other chains of host parts. For instance consider the following host chains having set combination sizes of four:

$$I : 1-2-3-4, \qquad II : 2-8-5-6, \qquad III : 7-8-5-6.$$

If the second alternative is selected, the first and third chains will be destroyed because of cell-2 and cells -8, -5, and 6. Clearly, a better way is to choose the first and third chains instead of the second. Selection among the chains having the maximum set size combination is based on the total priority rankings of the cells in the chains. The notional cells are the ones that contain no forward or inverse relationships. If they are among the selected cells to be merged, they will not ruin other chains. Thus, they have the highest priority ranking. On the other hand, urgent cells have to be merged with the cells in their inverse relationships. This necessity may cause an elimination of a number of merging alternatives containing the cells specified in the inverse relationships of the urgent cells. Hence, the urgent cells have the lowest priority ranking.

The set combination sizes of the candidate cell chains are examined. The chain with the maximum set combination size is selected for merging. The existing ties are handled by means of the total priority rankings of the cells that make up the chain. This heuristic rule attempts to destroy a minimum number of chains due to a certain selection. After each merging operation, the inter-relationships and the priorities of the left-over cells are updated. Another cell chain with the maximum set combination size is chosen to be merged. The ties are again broken by means of the total priority rankings of the chains. The chain with the highest total priority ranking is selected from the chains of the same size. Thereafter, the inverse-forward relationships of the remaining cells are again updated and another merging operation is performed. This merging process is terminated when all hosts are merged into a number of super-hosts.

The objective of the above selection scheme is to reduce the number of super-hosts as much as possible. The primary rule based on selecting the chain with the maximum set combination size tries to increase the number of joined cells at each merging operation. The secondary rule based on selecting the one with the highest total priority ranking tries to increase the number of remaining cells. This merging process is analogous to the number-theoretic integer-partitioning scheme. This scheme decomposes a given positive integer. This integer is expressed by a combination of smaller integers such that the sum of the coefficients in the expression is held at a minimum level by starting with the highest possible integer. The analogy between this partitioning scheme and the merging process can be illustrated as:

$$h = n_M \times M + n_{M-1} \times (M-1) + \cdots + n_1 \quad \Longrightarrow \quad s = n_M + n_{M-1} + \cdots + n_1,$$

where h : total number of hosts,

s : total number of super-hosts,
M : maximum possible set combination size,
n_i : number of feasible chains having set combination size of i.

COMBGR assumes infinite number of machines of the same type at the initial stage. At each COMBGR iteration, the machine requirement of the current solution is reduced. Thus, machine requirements of the PF/MG-F solutions at the end of each iteration monotonically decreases while the number of cells reduces by merging hosts into superhosts. The iterations continue until the original job shop having the minimum machine requirement among all solutions is reached.

4.1.2 Modifications and Extensions

Purcheck proposed a lower bound (two) on the set combination sizes which guarantees merging of each candidate cell with another. He suggested that n_1 should be zero, possibly to increase the rate of convergence to the original job shop. This places an artificial condition on the notional cells at the beginning of each merging iteration. Besides, it is sometimes beneficial to keep some candidate cells as they are. For instance, consider the following two cases:

$$case-1: \quad h = 9 = 2\times 3 + 1\times 2 + 1 \ \Rightarrow \ s = 4 \ ,$$

$$case-2: \quad h = 9 = 1\times 3 + 3\times 2 + 0 \ \Rightarrow \ s = 4 \ .$$

In both cases there will be four cells in the next iteration. However, the former may lead to better PF/MG-F proposals in the succeeding stages than the latter, because of the hierarchical nature of COMBGR. In particular, a superior solution to Purcheck's example in [35] can be obtained just by allowing n_1 to take positive values (see Table 4.2).

COMBGR identifies manufacturing cells by generating a hierarchy of machine grouping alternatives. Part assignments are not considered in this technique. The following part assignment scheme is being proposed as an extension. After initial candidate cells are formed, parts are assigned. Clearly, each host part is included in the candidate cell characterized by itself. On the other hand, guests can be assigned to any of the cells determined by their flexibility

FIRST PF/MG-F PROPOSAL (9 cells)

Set Size	Host Combination	Machine Composition	Machine Difference
4	3,4,11,12	abdejkns	c
4	6,20,22,23	adfghl	c
3	2,14,17	acdhklnt	m
3	13,16,21	adfghl	p
2	5,10	abdhjns	l
2	1,18	abdkmnsu	e
2	9,24	abdknr	q
2	7,8	abcdhkn	s
2	15,19	adfks	h

Machine total: 71 Capital cost: $ 539 180

SECOND PF/MG-F PROPOSAL (4 cells)
Machine total: 45 Capital cost: $ 415 430

THIRD PF/MG-F PROPOSAL (3 cells)
Machine total: 30 Capital cost: $ 325 190

LAST PF/MG-F PROPOSAL (Original shop)
Machine total: 19 Capital cost: $ 252 820

b) PURCHECK'S SOLUTION

FIRST PF/MG-F PROPOSAL (9 cells)

Set Size	Host Combination	Machine Composition	Machine Difference
4	3,4,11,12	abdejkns	c
4	6,20,22,23	adfghl	c
3	8,15,19	abdfhks	f
3	2,14,17	acdhklnt	m
3	13,16,21	adfghl	p
2	5,10	abdhjns	l
2	1,18	abdkmnsu	e
2	9,24	abdknr	q
1	7	abcdhkn	–

Machine total: 71 Capital cost: $ 537 180

SECOND PF/MG-F PROPOSAL (8 cells)
Machine total: 65 Capital cost: $ 500 810

THIRD PF/MG-F PROPOSAL (5 cells)
Machine total: 49 Capital cost: $ 432 600

FOURTH PF/MG-F PROPOSAL (4 cells)
Machine total: 40 Capital cost: $ 373.230

FIFTH PF/MG-F PROPOSAL (3 cells)
Machine total: 33 Capital cost: $ 333 460

SIXTH PF/MG-F PROPOSAL (2 cells)
Machine total: 27 Capital cost: $ 299 090

LAST PF/MG-F PROPOSAL (Original shop)
Machine total: 19 Capital cost: $ 252 820

a) SUGGESTED SOLUTION

Table 4.2: An effect of the modification in the merging process.

relationships. The machine-difference set between a guest-part and a cell is defined as the set of machine types that the guest does not use. Each guest part is assigned to a cell based on the machine-difference set size for all cells in the flexibility relation. The guest part is assigned to the cell which has the minimum machine-difference set size. In the subsequent stages of COMBGR, parts of the merged cells are joined. Since COMBGR does not give rise to any exceptional elements, the grouping efficiency defined in the previous chapter depends solely on inner-cell densities. The density of any cell is inversely related to the number of 'zero' entries in the corresponding diagonal block of the incidence matrix. The suggested part assignment procedure assures a minimum number of 'zero' entries at any instant of COMBGR. Therefore, it is the optimal scheme in terms of the grouping efficiency measure.

Initial candidate cells are formed by assuming that infinitely many number of machines of each type exist. Nevertheless, each merging iteration reduces the total machine requirement in the system. These iterations are carried out sequentially until the original job shop is obtained where the total machine requirement is at a minimum level. There is no need to further carry out the merging iterations if at any stage the total machine requirements are less than the value defined by the original shop. At this point the algorithm can be terminated. This is considered as a stopping criterion. Another stopping criterion is the minimum machine-difference. If the minimum machine-difference exceeds a prespecified value, the hierarchical merging of cells is interrupted. Both of the two stopping conditions have been found to select relatively good PF/MG-F solutions among the alternatives generated by merges. Inner-cell densities are decreased at each merging process, because of the extra number of 'zero' entries being introduced into the cell. If the final PF/MG-F solution is determined by the first stopping criterion, it is the optimal in terms of the grouping efficiency measure subject to machine availabilities. If the solution is based on the latter criterion, then further merging of cells will decrease the grouping efficiency value drastically. On the other hand, the PF/MG-F proposal at the instant when the latter criterion is satisfied leads to an increase in extra investment amount because of machine duplications. Hence, the minimum allowable machine-difference should be determined by considering the trade-off between extra investment and the grouping efficiency.

4.1.3 *Algorithm* {COMBGR}

```
S-1. Input part routing codes, number of machines in the job shop, and
     maximum machine-difference limit;

S-2. Compute sizes and code significances of parts,
     Sort parts by size in decreasing order,
     Order parts of the same size by code significance in descending order;

S-3. Find hosts and guests, construct hospitality and flexibility relationships,
     Identify initial candidate cells characterized by hosts,
     Assign parts;

S-4. Calculate size of minimal machine-differences between cells,
     If minimal machine-difference size $>$ maximum machine-difference limit,
     jump to S-6,
     Compute set combination sizes, forward and inverse relations,
     Assign priorities;

S-5. Form super-hosts,
     Calculate total machine requirements,
     If total machine requirements $\leq$ total number of machines, jump to S-6,
     Replace hosts by superhosts,
     Return back to S-4;

S-6. Output the solution,
     Calculate efficiency measures;

end {COMBGR}.
```

4.2 Modified Rank Order Clustering

Rank Order Clustering (ROC) is a reordering PF/MG-F technique introduced by King [23]. A ROC iteration consists of a row reordering followed by a column reordering. ROC iterations are executed sequentially until no change in the incidence matrix is encountered. ROC is the most commonly known PF/MG-F technique. During the first half of this decade, almost all of the new PF/MG-F techniques have been compared with ROC. New techniques were approved only if they generated superior solutions relative to what ROC suggested. King and Nakornchai [24] applied data structures based on sparsity techniques and proposed a sorting mechanism. Their developments lead to an efficient implementation, known as ROC2, of the basic ROC idea. Recently, Chandrasekharan and Rajagopalan [11] proposed a new version, MODROC, which was developed after a series of detailed analyses of PF/MG-F solutions generated by ROC. They modified ROC to overcome the observed limitations.

4.2.1 Description of MODROC

The *rank* of a part or a machine type depends on the value of the integer equivalent of the binary expression of the corresponding column or row in the incidence matrix. The rank of a part or a machine type is determined from a sorted rearrangement of all parts or machine types. The rearrangement is based on the integer equivalents in descending order. The part or machine type with the greatest integer equivalent has the first ranking.

Reordering in a ROC iteration is performed by means of the ranks assigned to each row or each column of the incidence matrix. Reordering is achieved by sorting rows or columns in descending order of ranks. Ranks are recalculated after each reordering operation. Each ROC iteration involves a row and column reordering. An example of a ROC iteration is illustrated in Figure 4.1. Here, integer equivalents are calculated by multiplying the associated row or column with the corresponding weights. For instance, machine type-3 has an integer equivalent of 81 (i.e., $2^0+2^4+2^6$). Machine type-3 in Figure 4.1-a has the first ranking since its integer equivalent is the highest. Ranks are assigned to order the machine type index set. This ordered index set determines the reordered form of the machine types.

As indicated earlier, ROC had been treated as a basis for testing other PF/MG-F techniques and therefore some limitations of ROC are reported [2,42,11]. Some of the limitations are as follows. Given a priori, that a desirable PF/MG-F solution can be obtained for a specific problem, it is not certain that the ROC procedure will produce this solution. In other words, a possible PF/MG-F solution with an ideal block diagonal structure

R. O. C.	P1	P2	P3	P4	P5	P6	P7	Int. Eq.	Rank
M1	0	1	0	1	1	1	0	46	4
M2	1	0	1	0	0	0	0	80	2
M3	1	0	1	0	0	0	1	81	1
M4	0	1	0	1	0	1	0	42	5
M5	1	0	0	0	0	0	1	65	3
Weight	2^6	2^5	2^4	2^3	2^2	2^1	2^0	*Row Reordering*	

a) Beginning of a ROC iteration

R. O. C.	P1	P2	P3	P4	P5	P6	P7	Weight
M3	1	0	1	0	0	0	1	2^4
M2	1	0	1	0	0	0	0	2^3
M5	1	0	0	0	0	0	1	2^2
M1	0	1	0	1	1	1	0	2^1
M4	0	1	0	1	0	1	0	2^0
Int. Eq.	28	3	24	3	2	3	20	*Column Reordering*
Rank	1	4	2	5	7	6	3	

b) Rows are reordered

	P1	P3	P7	P2	P4	P6	P5
M3	1	1	1	0	0	0	0
M2	1	1	0	0	0	0	0
M5	1	0	1	0	0	0	0
M1	0	0	0	1	1	1	1
M4	0	0	0	1	1	1	0

c) Columns are reordered

Figure 4.1: A ROC iteration.

may not be generated as a result of ROC. Moreover, ROC may disarrange an inputed PF/MG-F problem having a block diagonal structure with few exceptional elements. Hence, the final solution obtained from ROC may not necessarily be the best solution. The most important observation is the strong dependence of ROC on the initial arrangement of the incidence matrix. Finally, ROC does not suggest any procedure on how to identify the cells from the final form of the incidence matrix. ROC outputs only the final form of the incidence matrix. One has to extract the PF/MG-F solution from this matrix by identifying the manufacturing cells.

The incidence matrix after two ROC iterations contains a block of 'ones' at its top-left corner. MODROC considers this block as a candidate cell. Chandrasekharan and Rajagopalan [11] proposed a procedure that identifies the initial candidate cells. Subsequent to the identification of a candidate cell, the corresponding columns are sliced away. The parts associated with the deleted columns and their related machines form a candidate cell. Subsequently, another two ROC iterations are carried on the resultant incidence matrix. These iterations continue until the incidence matrix becomes empty. Since the corresponding rows are not sliced away at each iteration, any machine type can be assigned to each cell irrespective of their availability. After all initial candidate cells are identified, they are then merged hierarchically as in COMBGR.

Chandrasekharan and Rajagopalan [11] proposed a Jaccard's similarity coefficient to help the merging process. This similarity coefficient, $s_{i,j}$, is based on the machine contents of the cells. It is defined for a pair of cells as the ratio of the total number of common machines to the size of the machine set of the smaller cell:

$$s_{i,j} = \frac{n(CM_i \bigcap CM_j)}{Min\{n(CM_i), n(CM_j)\}} ,$$

where CM_j : machine type index set of cell j,
$n(CM_j)$: size of the set CM_j.

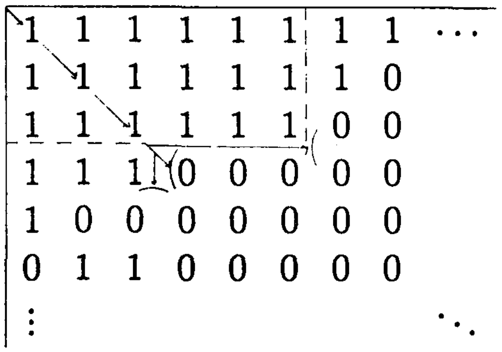
The triangular similarity coefficient matrix is searched to find the pair of cells, say (k, l), with the maximum similarity. Existing ties are broken arbitrarily. Subsequently, cell-k and cell-l are merged and the associated part families are joined. The similarity coefficient matrix is updated, and a new cell pair with the maximum similarity coefficient value is found. The corresponding cells are merged, and so on. The iterative procedure continues until only one cell (i.e., the original job shop) remains.

4.2.2 Modifications and Extensions

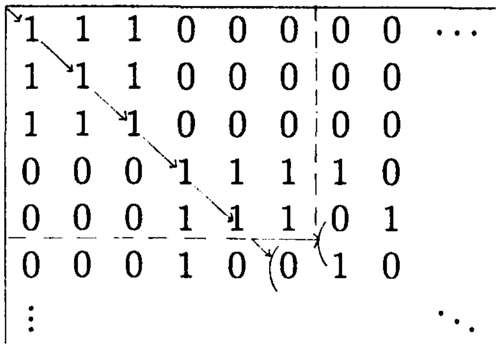
Chandrasekharan and Rajagopalan suggested a procedure for identifying the initial candidate cells. A diagonal search is initiated on the incidence matrix for this purpose, starting from the top-left entry. The search terminates whenever a 'zero' entry is encountered. Thereafter, the rectangular block of 'ones' is identified by checking the row and column of the last diagonal 'one'. This procedure is illustrated in Figure 4.2-a. However, as seen from Figure 4.2-b the above procedure can fail in identifying the diagonal block. During the diagonal search, succeeding elements in the corresponding row and column should also be checked in addition to the diagonal element. The suggested new search continues until at least one of the examined entries has a value of 'zero' in the incidence matrix. This new suggested procedure is illustrated in Figure 4.2-c.

MODROC merges a pair of candidate cells at each merging iteration. In some cases, the resulting big cell attracts other nearby small cells. Hence, MODROC's PF/MG-F solutions usually contain a large cell similar to a job shop together with a number of small sized cells during the final iterations. This aspect reduces the quality of the MODROC solutions.

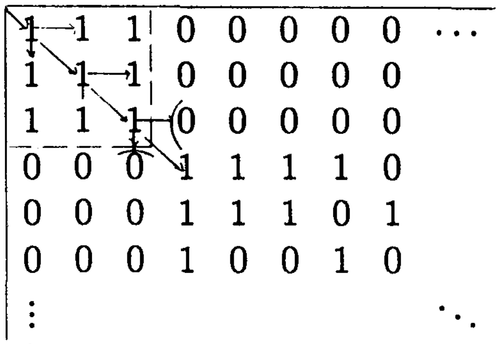
The following merging scheme is being proposed in order to overcome the above drawback and to reduce the total number of merging iterations. All pairs of cells having similarity coefficients greater than a specified value are selected at each merging iteration so that more than one cell pair mergings can take place. The threshold value for similarity coefficients can be determined as a prespecified percent of the maximum similarity. For instance, the threshold value will be 0.60 if the maximum similarity value turns out to be 0.80 and the prespecified percentage is 75. If the number of independent pairs of cells at any instant exceeds another prespecified value, say 20 percent of number of candidate cells, the prespecified percentage is increased. If



a) Chandrasekharan–Rajagopalan's search procedure works



b) Chandrasekharan–Rajagopalan's search procedure fails



c) A suggested new search procedure

Figure 4.2: Diagonal block identification.

there are three independent pairs having similarities more than 0.60 in the above example and the number of candidate cells is ten, then the prespecified percentage is increased sequentially from 75 up to 100 percent to reduce the number of independent pairs to two. After the selected cells are merged, the similarity coefficient matrix is updated and another iteration is started.

Each hierarchical merging iteration generates an alternative PF/MG-F solution. Since the merging iterations in the original version of MODROC continue until a single cell is obtained, the original job shop is the last suggested alternative. However, some of the PF/MG-F proposals are not feasible because a large number of machines of each type could be necessary for the proposal determined by the initial candidate cells. Those alternatives can be eliminated by comparing the total machine requirements with the existing number of machines of each type. Since MODROC operates on an incidence matrix, it implicitly assumes that one unit of an existing machine type can be sufficient if included in the cell. Hence, total machine requirements of the PF/MG-F proposals decrease at each successive merge.

The grouping efficiency measure is suggested as a criterion for selecting the PF/MG-F solution among the proposals generated by MODROC. Only the proposals satisfying the machine requirements constraint are taken into consideration at this stage like in COMBGR. However, computation of the grouping efficiency is not as simple as in COMBGR, since MODROC allows exceptional elements in its PF/MG-F proposals. The grouping efficiency values for all of the feasible proposals are calculated by changing the convex combination coefficient (α) from zero to one with steps of size 0.1. So, MODROC generates at most eleven PF/MG-F alternatives because each alternative may still be optimal with respect to the grouping efficiency computed with different α values. Moreover, MODROC can easily be modified so that its final PF/MG-F solution is the feasible alternative having the maximum grouping efficiency value for a given $\alpha \in [0.0, 1.0]$.

4.2.3 *Algorithm* {MODROC}

S-1. Input incidence matrix, machines in the job shop, lower limit on similarity coefficient, upper limit on number of independent parts, and aspiration level α;

```
S-2. Make two ROC iterations on the incidence matrix;
S-3. Identify the largest top-left block of 'ones',
     Determine the candidate cell,
     Slice the corresponding columns in the incidence matrix,
     If the resultant incidence matrix is not empty, return back to S-2;
S-4. If total machine requirement $\leq$ total number of machines, save the solution,
     If number of cells is equal to one, go to S-6,
     Generate similarity coefficient matrix,
     Choose independent pairs of cells having higher similarities than the lower limit,
     If number of independent pairs is zero, save the solution and go to S-6;
S-5. Merge cells,
     Join part families,
     Return back to S-4;
S-6. Based on $\alpha$, choose the solution with the highest grouping efficiency value among all PF/MG-F proposals,
     Output the solution,
     Calculate efficiency measures;

end { MODROC}.
```

4.3 Machine Clustering Using Similarity Coefficients

MACE is another hierarchical, machine-grouping PF/MG-F technique developed by Waghodekar and Sahu [42]. During the first stage, MACE generates initial machine clusters representing candidate cells. This algorithm brings together all machines of close similarity under one cell by making use of Jaccard's similarity coefficients. Waghodekar and Sahu [42] reported three types of similarity coefficients. The initial candidate cells are subsequently merged into final cells again with the help of similarity coefficients. The last stage of MACE involves part assignments to the final cells.

4.3.1 Description of MACE

Waghodekar and Sahu [42] expressed the closeness between each pair of machine types in three ways. They used Jaccard's similarity coefficient which was introduced in the context of PF/MG-F by Carrie [8]. This similarity coefficient which is of additive type was later used by Rajagopalan and Batra [39]. Waghodekar and Sahu [42] added two new similarity coefficients. One of these similarity coefficients is of multiplicative kind and is based on the total number of parts processed by a pair of machine types. The last suggested similarity coefficient is based on the total flow of common parts processed by a machine type. The three similarity coefficients are defined algebraically as follows:

T : total number of machine types;
P : total number of parts;
IM_i^k : $(i,k)^{th}$ entry of the incidence matrix;
TNC_i : total number of parts using machine i,

$$TNC_i \doteq \sum_{k=1}^{P} IM_i^k;$$

$NCC_{i,j}$: number of common parts using both machine types i and j,

$$NCC_{i,j} \doteq \sum_{k=1}^{P} IM_i^k \times IM_j^k;$$

TFC_i : Total flow of common parts processed by machine i with respect to all other machine types,

$$TFC_i \doteq \sum_{j=1, j\neq i} NCC_{i,j};$$

The similarity coefficient of additive kind between machine types i and j $(SC_{i,j})$ is defined as :

$$SC_{i,j} \doteq \frac{NCC_{i,j}}{TNC_i + TNC_j - NCC_{i,j}}.$$

Similarly, the similarity coefficient of multiplicative kind between machine types i and j $(PSC_{i,j})$ is defined below:

$$PSC_{i,j} \doteq \frac{NCC_{i,j} \times NCC_{i,j}}{TNC_i \times TNC_j}.$$

The last similarity coefficient is of flow kind ($SCTF_{i,j}$) defined for machine types i and j as:

$$SCTF_{i,j} \doteq \frac{NCC_{i,j} \times NCC_{i,j}}{TFC_i \times TFC_j}.$$

The similarity coefficient value between two machine types represents the bilateral closeness. After calculating the similarity coefficient values between all machine types, the pair with the maximum similarity value is selected. If there exists a tie in selecting the closest machine type pair in terms of any one of the similarity coefficients, a secondary search is performed. This search is based on the number of machine types that are close to each pair. The pair with the maximum number of machine types that are close in terms of similarity values is selected as a base for the next candidate cell being formed. The remaining ties are broken arbitrarily.

The selected pair identifies a candidate cell. Machine types that are close to the pair are included in the candidate cell. For a machine type to be admitted into the cell, it should have a high similarity coefficient value corresponding to each machine element of the pair. The machine types that are included in the cell because of closeness to the pair are termed as 'satellite' machine types of the pair. The selected pair and its 'satellites' form the candidate cell. Thereafter, another pair having the highest similarity value among the remaining machine types is selected. Another candidate cell is formed by the selected pair and its 'satellite' machine types. This candidate cell formation procedure terminates when there are no more machine types left.

During the next stage, inter-cell flows are calculated to compute the flow similarity coefficients ($SCTF_{i,j}$) between each pair of candidate cells. Here, the part composition of each candidate cell is found by taking the union of the related part sets of the included machine types. In other words, a candidate cell-part incidence matrix is generated from the machine-part incidence according to the machine contents of the candidate cells:

$$CIM_k^j \doteq \begin{cases} 0 & \text{if } \forall i \in CM(k), \quad IM_i^j = 0, \\ 1 & \text{otherwise;} \end{cases}$$

where IM_i^j: $(i,j)^{th}$ entry in the machine-part incidence matrix,

CIM_k^j: $(k,j)^{th}$ entry in the cell-part incidence matrix,

$CM(k)$: the machine type index set of the k^{th} candidate cell.

A transformation from the machine-part incidence matrix to a candidate cell-part incidence matrix is accomplished by this way. Thereafter, the similarity coefficient values of all pairs of candidate cells are calculated by replacing IM with CIM in the definitions. Consequently, the above candidate cell formation procedure designed for merging the machine types is repeated for the candidate cells to identify the final PF/MG-F cells.

The last stage of MACE involves part assignments. At this stage, the final cell compositions are determined. The final cells are ordered according to their formation sequence. In other words, the earliest formed cell is ordered first whereas the latest one is ordered last. Each part is assigned to the first encountered cell in the order of formation such that the part in consideration has at least one of its manufacturing operations in the cell:

$$CP(j) = k \Leftrightarrow CIM_k^j = 1,\ CIM_i^j = 0, \forall i = 1,\cdots,k-1\ ,\quad \forall j = 1,\cdots,P\ ;$$

where $CP(j)$ is the index of the cell to which part j is assigned. For instance, a specific part is assigned to the first cell in the order if at least one of its operations can be processed, otherwise the second cell in the order is examined, and so on.

4.3.2 Modifications and Extensions

The part assignment scheme suggested by Waghodekar and Sahu [42] may lead to a number of exceptional elements. Moreover, it may also result in considerably low inner-cell densities. Hence, the PF/MG-F solutions generated by MACE have low grouping efficiency values, no matter what aspiration level (α) is specified. A superior part assignment scheme is proposed in order to overcome this handicap. Each part is assigned to the cell that can process the majority of the part's operations. The work-load cost fractions of the parts described in the previous chapter are used as a basis for part assignments. The work-load cost fraction of a part is the percentage of the total part work-load cost that is allocated to its assigned cell. The cell with the highest work-load cost fraction for each part is selected for assignment:

$$\begin{aligned} k \in CP(j) \Leftrightarrow WLCF_j^k &= \tfrac{WCC_j^k}{TWLC_j} \\ &= Max\{WLCF_j^i : i = 1, \cdots, K\}, \quad \forall j = 1, \cdots, P\,; \end{aligned}$$

where $CP(j)$: index set of cells to which part j can be assigned,
$TWLC_j$: total work-load cost of part j,
WCC_j^k : work-load cost of part j in cell k,
$WLCF_j^k$: work-load cost fraction of part j in cell k,
K : number of cells,
P : number of parts,

The measure of closeness related to similarity coefficients was not explicitly and clearly explained by Waghodekar and Sahu [42]. A threshold value for similarity coefficients is being suggested in clustering the machines. Machine types having similarity coefficients smaller than this threshold value are not included in the same candidate cell. After the pair with the maximum similarity is chosen, two passes are made to configure the candidate cell. During the first pass, machine types that have higher similarity coefficients than a prespecified value to both of the selected machine types are added to the candidate cell. This prespecified value depends on the threshold value and the highest similarity coefficient. During the second pass, the threshold value is lowered. The similarities between the remaining machine types to the machine types already assigned are examined. Machine types that have higher similarity coefficients than the threshold value to all machine types belonging to the candidate cell are assigned. If a machine type has a lower similarity value than the threshold value to at least one of the machines in the cell, then it is prevented from being a 'satellite' machine for the selected pair.

The MACE technique generates three PF/MG-F solutions based on the three different similarity coefficients. An initial study is carried out to identify the effects of different similarity coefficients. The performance of the new part assignment scheme is also compared with the one suggested by Waghodekar and Sahu. In this comparative study, three different classes of PF/MG-F problems are considered. The first class of problems are designed specifically such that job shop type solutions will be the best alternative. Conversely, the third class contains the test problems representing near ideal CM situations.

PROBLEM CLASS	SIMILARITY COEFFICIENT	PART ASSIGNMENT SCHEME			
		Old		*New*	
		Inter-cell Efficiency	Inner-cell Efficiency	Inter-cell Efficiency	Inner-cell Efficiency
Job shop like	$SC_{i,j}$	0.2667	0.4566	0.4833	0.7067
	$PSC_{i,j}$	0.2501	0.6804	0.4603	0.7905
	$SCTF_{i,j}$	0.2103	0.8701	0.5067	0.9733
Intermediate	$SC_{i,j}$	0.5017	0.4707	0.5850	0.7117
	$PSC_{i,j}$	0.4304	0.5607	0.5851	0.8433
	$SCTF_{i,j}$	0.2483	0.9067	0.4203	0.9650
Ideal CM like	$SC_{i,j}$	0.7633	0.6367	0.8967	0.8702
	$PSC_{i,j}$	0.7602	0.6367	0.8933	0.8701
	$SCTF_{i,j}$	0.4604	0.8867	0.4935	0.9567

Table 4.3: Effect of different similarity coefficients on MACE solutions.

The second class consists of the PF/MG-F problems that fall in between. This idea is extended and explained in the next chapter. Twelve statistically independent problems are generated for each class. Each assignment scheme is evaluated under different similarity coefficients for all of the test problems in each class. Moreover, the threshold values used in candidate cell formation and final cell formation are varied. The first threshold value is varied four times between 0.05 and 0.20 whereas the second one is varied six times between 0.05 and 0.30. Therefore, a total of 3 × 12 × 2 × 3 × 4 × 6 (=5184) PF/MG-F results is generated in this comparative analysis. The grouping efficiency measure described in the third chapter is used as an evaluation criterion. The results are given in Table 4.3.

The results show that the PF/MG-F solutions are insensitive to the threshold values within the specified limits. However, the results are slightly better if both of the threshold values are fixed at 0.10. The effects of similarity coefficients on the solutions for each part assignment scheme and each class of problems are illustrated in Table 4.3. The new assignment scheme is always superior in terms of the grouping efficiency measure. The use of the similarity coefficients based on flows ($SCTF_{i,j}$) leads to better designs in terms of grouping efficiency in job shop like manufacturing environments. Similarity coefficients of multiplicative type ($PSC_{i,j}$) should be preferred in intermediate manufacturing environments while the similarity coefficient of additive kind ($SC_{i,j}$) gives relatively better PF/MG-F solutions in near ideal CM environments.

Finally, cells with no parts assigned could exist. These cells consist of only machines. This situation was also observed by Waghodekar and Sahu [42]. They termed the machines of such cells as blocking machines and indicated a necessity to perform a secondary analysis. However, they lacked a method to handle such blocking machines. The following procedure is being proposed to fill the underlined gap. Let,

i : blocking machine;

IM_i^j : entry corresponding to machine type i and part j in the incidence matrix;

K : number of cells excluding the ones with blocking machines;

$PC(k)$: index set of parts assigned to cell k;

$n(PC(k))$: total number of parts assigned to cell k;

O_k^i : total number of exceptional elements related to the blocking machine i created by the parts assigned to cell k;

$$O_k^i \doteq \sum_{j \in PC(k)} IM_i^j \ ;$$

E_k^i : total number of exceptional elements created if the blocking machine i is assigned to cell k,

$$E_k^i \doteq \sum_{l=1, l \neq k}^{K} O_l^i \ ;$$

Z_k^i : number of 'zeros' created if the blocking machine i is assigned to cell k,

$$Z_k^i \doteq n(PC(k)) - O_k^i \ .$$

The following procedure is suggested for assigning each blocking machine of type i to a cell. First, the cells with nonblocking machines are ordered in increasing ($Z_k^i + E_k^i$) values. Since assignment of blocking machines to cells should lead to improvements in the grouping efficiency measure, cells with small ($Z_k^i + E_k^i$) values are more attractive, and therefore individual blocking machines are allocated in this order. If the number of blocking machines of the same type is greater than the number of cells, another pass is made. In this manner, the blocking machines are relocated to the cells. For blocking machines of different types, the procedure is repeated. This procedure is illustrated in Table 4.4 for two machines of the same blocking type i whose

CELLS	$IM_i^{CP(j)}$	O_k	E_k	Z_k	$E_k + Z_k$	No
1	1101001	4	15	3	18	1
2	1001101100111001	9	10	7	17	1
3	0011010100101	6	13	7	20	0
4	0000000	0	19	7	26	0

Table 4.4: An example of blocking machine assignment

part encode is given by:

$$IM_i = [1101001|1001101100111001|0011010100101|0000000].$$

The vertical bars in the encode identifies cell boundaries. This procedure attempts to increase the number of eliminated exceptional elements and decrease the number of 'zeros' within each cell at the same time. Therefore, the suggested procedure increases the grouping efficiency.

4.3.3 *Algorithm* {MACE}

S-1. Input similarity coefficient type, threshold value, number of machines in the job shop;

S-2. Compute $NCC_{i,j}, TNC_i, TFC_i$,
Calculate similarity coefficients of the selected type;

S-3. Select the machine pair with the maximum similarity,
Examine the closest machines,
Form a candidate cell;

S-4. Repeat S-3 until no more machine type is left;

S-5. Compute inter-cell flows,
Replace machines by candidate cells,
Calculate similarity coefficients between candidate cells ($SCTF_{k,l}$),
Repeat S-3 until no more candidate cells are left;

S-6. Assign parts,
Check the existence of blocking machines,
If there exists blocking machines, relocate them;

```
S-7. Output the solution,
     Calculate efficiency measures;

end {MACE}.
```

4.4 Within-cell Utilization Based Clustering

WUBC is a graph searching PF/MG-F technique where parts and machine types are considered simultaneously. A number of practical criteria are used in forming cells: these being work-load fractions and within-cell utilizations of machine types, percentage of operations of parts completed within a single cell, and the maximum allowable cell size. Initially, WUBC determines the manufacturing cells by focusing on machine types. This clustering is based on work-loads and cell size restrictions. In the meantime, parts are assigned to cells based on the number of operations that can be processed within the cells.

4.4.1 Description of WUBC

WUBC induces a breadth-first search on the graph generated by the routing relationships between parts and machine types. A *key* machine type is selected as the root in the search. All parts routed through the key machine type are examined. These parts are either admitted to the cell generated by the key machine type or they remain in their previously assigned cells. The parts that were not previously assigned are automatically included in the cell when they are examined for the first time. Consequently, all machine types related to the admitted parts are examined in this process. Machine types are added to the cell if their within-cell work-loads due to the parts already assigned exceed a prespecified level. Upon completion of each search, the required number of machines of each type are allocated to the cell based on the within-cell utilizations. Thereafter, another search is initiated by selecting a new key machine type among the remaining machines. The above procedure is repeated for the new cell. This process continues until no more key machines are available. Finally, a *remainder cell* is formed by bringing all left-over machines together, if they exist. All assigned parts are examined for reassignment to the remainder cell in the case it is not empty. Conversely, if there are unassigned parts, they are included in the remainder cell.

WUBC starts with the selection of a key machine type to initiate the formation process of a cell. The machine type with the highest work-load per machine is chosen as the key:

$$\frac{\sum_{j=1}^{P} WL_{key,j}}{n(key)} = Max \left\{ \frac{\sum_{j=1}^{P} WL_{i,j}}{n(i)} : i = 1, \cdots, T \right\} ,$$

where P : total number of parts,

T : total number of machine types,

$WL_{i,j}$: work-load of machine i due to part j,

$n(i)$: total number of individual machines of type i.

If there are more than one machine type having the same highest work-load per machine, then the one with the lowest index is selected. Later, alternative rules for selecting the key machine type will be investigated. The key machine type is admitted to the cell and added to a first-come-first-served (FCFS) queue. A machine-part graph search rooted at the key machine type is initiated to identify parts that have operations on the key and other machines related to these parts.

All parts having manufacturing operations on the key machine type are examined to determine whether they can be admitted to the cell or not. If the part in consideration has not yet been assigned to a cell, it is admitted to this cell. Otherwise, it is marked if it has more within-cell operations in this cell than in its currently assigned cell. Alternative rules for part assignment will again be discussed in the next subsection.

In the next step of a cell formation iteration, WUBC considers non-key machine types that have routing relationships with the parts assigned to the cell. At this stage, the non-key machine types with at least one available machine are evaluated. If all individual machines were assigned to the previously formed cells, the associated machine type is marked so that it will no longer be considered. *Work-load fractions* (WLF) of the non-key machines are calculated next. WLF of a machine type i in cell k is defined as the ratio of within-cell work-load due to those parts that have already been assigned, to the total load:

$$WLF_{k,i} \doteq \frac{\sum_{j \in CP(k)} WL_{i,j}}{\sum_{j=1}^{P} WL_{i,j}} ,$$

where $CP(k)$ is the index set of parts already assigned to cell k. At this stage, an admit/reject decision is made. A prespecified value is used for this decision. *Cell admission factor* (CAF) is defined as the minimum WLF for any machine type to be admitted to the cell. So, a machine type is included in the cell if it has not been admitted or rejected from this cell and its WLF is greater than or equal to CAF. Otherwise, it is rejected. A rejection prevents admission of the associated machine type to this cell.

All admitted non-key machine types containing only one individual machine are added to the FCFS queue. Then, the key machine type that identified the cell is deleted from the FCFS queue. The next machine type in the FCFS queue is selected as the next root to perform another search on the remaining machine-part graph. So, the parts having operations related to the root are examined for assignment. Consequently, machine types that were not considered previously are scanned. New admitted ones are inserted into the FCFS queue if they consist of single machines. Thereafter, the root machine type (i.e., one at the top of the queue) is deleted from the FCFS queue. Then the next machine type in the queue is selected and this procedure continues until the FCFS queue becomes empty.

Within-cell utilizations (WCU) of the admitted machine types are computed next. WCU of a machine type i for cell k is determined by dividing the within-cell load to the number of individual machines of that type:

$$WCU_{k,i} \doteq \frac{\sum_{j \in CP(k)} WL_{i,j}}{n(i)} , \quad \forall i \in CM(k) ;$$

where $CM(k)$ is the index set of the admitted non-key machine types. The admitted non-key machine types are sorted in decreasing order of WCUs. Their inclusion into the cell is based on a prespecified cell size restriction, called *cell size upper limit* (CSUL). CSUL is the maximum number of machine types that can be assigned to a cell. CSUL is usually set equal to the total number of machines. Starting from the machine type having the maximum WCU, the admitted machine types other than the key are included in the cell until either CSUL is reached, or all are included in the cell. Consequently, all marked parts are examined for assignment. If they still have more within-cell operations in the cell under consideration than their current cells, they are assigned. All marks on the parts violating the above condition are removed. Such parts are not assigned to this cell and they remained in their currently assigned cells.

If the cell contains only one machine type (i.e., the key), it is discarded. That is, the machines of the key type are released. Moreover, the related part assignments are recoiled. Furthermore, this machine type is prevented from being a key in later iterations.

WUBC determines a new key machine type to form the next cell. If all machine types have per machine work-loads less than CAF, no key machine type is found. In this case, all left-over machines are admitted to the remainder cell. Then, the parts that are either unassigned or eligible to reassignment are allocated to the remainder cell.

4.4.2 Modifications and Extensions

When a part is reassigned, work-loads of the related machine types in the part's previous cell are affected. Ballakur and Steudel [3] ignored this fact. After a part changes its cell, work-loads of the corresponding machine types should be updated. A decrease in the work-load of a machine type can lead to a reduction in the number of required machines of that type. For instance, consider a case where there are two machines of a certain type with a WCU of 0.60 each. If a part imposing a load of 0.25 on this machine type is reassigned, one of the two machines could be released. In this case, the WCU of the remaining one is increased to 0.95. Moreover, the total work-load associated with a machine type could reduce to a very low value such that there would be no need to keep this machine type. Therefore, it should be removed from the cell. This might influence the cell-parts still having loads on such removed machine types. These parts that are currently assigned to such cells should be evaluated to determine whether reassignment to other cells would be beneficial. If such a part would have more operations in another cell, it would be reassigned. These new part reassignments could lead to new work-load decreases and machine type releases. Such a cycle could possibly create an empty cell at the end.

Alternative rules for both selecting the key machine type and part assignments are investigated. The rules considered are as follows:

A - Rules for selecting the key machine type

- ($A1$). Work-load per machine,
- ($A2$). Number of routed parts,
- ($A3$). Work-load per machine divided by the number of routed parts [A1/A2].
- ($A4$). Total work-load,
- ($A5$). Total work-load divided by the number of routed parts [A4/A2],
- ($A6$). Work-load cost per machine,
- ($A7$). Total workload cost,
- ($A8$). Annual depreciation cost;

B - Rules for part assignment

- ($B1$). Number of operations,
- ($B2$). Percentage of work-load,
- ($B3$). Percentage of work-load cost.

Ballakur and Steudel [3] proposed the rules $A1, A2, A3$, and $B1$. All of the rules are analyzed by means of 48 randomly generated PF/MG-F problems. For each test problem, nine different CAF levels are used for each rule combination. The resultant PF/MG-F solutions for each combination are compared in terms of the grouping efficiency measure described in the previous chapter. The grouping efficiency value of each rule combination for each problem is determined by taking the maximum value of the grouping efficiencies due to various CAF levels. The main effect of a certain rule is calculated by taking the mean grouping efficiency values of all test problems that utilized this rule. Similarly, the main effect of a certain CAF level is computed by taking the mean of the grouping efficiency values of all test problems that employed this specific CAF level.

The main effects of the key machine type selection and the part assignment rules are presented in Table 4.5. The grouping efficiency values do not change significantly. So, the extra investment amounts due to machine duplications are analyzed in order to evaluate the operating rules. It can be seen from Table 4.5 that the (A4,B2) combination results in considerably

RULE	Inter-cell Efficiency	Inner-cell Efficiency	Extra Investment
A1	0.9543	0.2367	$ 47091
A2	0.9701	0.2341	13277
A3	0.9553	0.2330	59320
A4	0.9685	0.2346	7495
A5	0.9690	0.2338	12295
A6	0.9522	0.2369	64592
A7	0.9700	0.2333	12159
A8	0.9697	0.2332	16604
B1	0.9622	0.2335	95045
B2	0.9633	0.2351	61508
B3	0.9654	0.2348	76280

Table 4.5: A comparison of alternative operating rules.

better PF/MG-F solutions than the other combinations. Moreover, the effect of CAF is also investigated in Table 4.6. The effect of various CAF levels on the grouping efficiencies are small. So, the extra investment amounts are included in Table 4.6 to serve a basis for evaluation. As CAF increases, the solutions tend to result in low inner-cell densities and small number of exceptional elements. In the meantime, increases in CAF reduce the need for extra investment in machine duplications. The results obtained for the ($A4$-$B2$) combination are added to the table to illustrate the significance of cross effects.

4.4.3 *Algorithm* {WUBC}

S-1. Input work-load matrix, cell admission factor, cell size upper limit, number of machines of each type, rule for key machine type selection, rule for part assignments;

S-2. Select key machine type according to the inputed rule,
If there is no key machine type, then go to S-10,
Insert key machine type into the FCFS queue,
Add key machine type into cell;

S-3. Examine all parts routed through the key machine type,
If the examined part is not already assigned, then assign the part;

CAF (%)	ALL COMBINATIONS			(A4-B2) COMBINATION		
	Inter-cell Efficiency	Inner-cell Efficiency	Extra Investment	Inter-cell Efficiency	Inner-cell Efficiency	Extra Investment
10	0.9372	0.2435	$ 102764	0.9534	0.2427	$ 36050
20	0.9533	0.2370	47494	0.9669	0.2347	1580
30	0.9513	0.2373	24945	0.9505	0.2438	0
40	0.9580	0.2362	19402	0.9667	0.2368	0
50	0.9582	0.2357	11320	0.9641	0.2375	0
60	0.9639	0.2311	7237	0.9719	0.2323	0
70	0.9795	0.2309	7809	0.9804	0.2327	0
80	0.9811	0.2310	5788	0.9839	0.2296	0
90	0.9901	0.2272	6075	0.9950	0.2267	0

Table 4.6: Effect of CAF in WUBC solutions.

If the examined part has a higher part assignment value in this cell, then mark the part;

S-4. Evaluate all non-key machine types in the routings of marked or assigned parts,
If non-key type is neither admitted nor rejected and its WLF $\geq$ CAF, then admit the non-key type, otherwise reject;

S-5. Insert all single machine admitted non-key types into the FCFS queue,
Delete the top machine type from the FCFS queue,
If FCFS queue is not empty, then set the new key as the top element of the queue and go to S-3;

S-6. If there is at least one machine type other than the key in the cell, go to S-7,
Erase marks on parts,
Release machines of the key type in this cell,
Prevent this type from being a key in further iterations,
Go to S-2;

S-7. Compute WCU of all admitted machine types due to marked parts,
List admitted machine types in decreasing order of WCU values,
Assign admitted machine types in this order until CSUL is reached;

S-8. Examine all marked parts,
Assign a marked part if it has a higher part assignment value in this

```
      cell;

S-9.  If there is no part assigned to the cell, then discard the cell,
      Otherwise, for assigned marked parts, update the work-loads of the
      corresponding machine types in previous cells, release them if necessary,
      Erase marks on parts,
      Go to S-2;

S-10. Add all left-over machines into the remainder cell,
      Examine all parts for possible reassignment to the remainder cell;

S-11. If there is no part assignment to the remainder cell, then go to S-12,
      For reassigned parts, update work-loads of the corresponding machine
      types in previous cells,
      If there is a machine release, then go to S-10;

S-12. Output solution,
      Calculate efficiency measures;

end {WUBC}.
```

4.5 Cost Analysis Algorithm

CAA is a another graph searching PF/MG-F technique designed by Chow and Kusiak [27]. In this sense, it is quite similar to the WUBC technique. However, CAA focuses on parts in defining the root in the search process. CAA initiates a breadth-first search on the machine-part graph formed by the routing relationships. Each search determines a manufacturing cell. During the search, an admit/reject decision is made for all parts except the root. A rejection causes elimination of the associated part from the analysis. The rejected parts should be subcontracted, since they lead to exceptional elements. A part is admitted to the cell if it does not give rise to an increase in the number of machine types than a prespecified value. The machine types that are used by the admitted parts are added to the cell unless they have not been included previously. The search continues by examining the unassigned parts related to the machine types in the cell. When no part can be assigned to the cell, the next cell is being constructed by taking another key part to be the root. This process continues until all parts are either assigned to a cell or rejected from the analysis.

4.5.1 Description of CAA

CAA makes use of *Cluster Identification Algorithm* (CIA) developed by Chow and Kusiak [27]. CIA is an efficient search algorithm that tries to identify a block-diagonal structure having no exceptional elements in an incidence matrix. CIA breadth-first searches the corresponding machine-part graph to obtain a perfect decomposition. CIA joins the associated cells in case there is an exceptional element. That is, the cell containing the corresponding machine type for an exceptional element is merged with the cell where the part is assigned. Therefore, CIA usually ends up with the same graph or the original shop if applied to PF/MG-F. CIA is used in CAA as a subroutine for detecting the possible increases in machine composition of the cells.

CAA initially selects the key part. The part with the maximum unit subcontracting cost is chosen as the key part to act as a root in the search. CAA assigns the key part to a new cell and adds the machine types that are used for the manufacture of this part into the cell. Thereafter, non-key parts having operations on the admitted machine types are examined. These non-key parts are sorted according to their costs in decreasing order. CIA is executed for each non-key part in the order specified above. A non-key part is assigned to the cell if it does not increase the number of machine types above the prespecified cell size limit (CSUL). Otherwise, it is rejected and discarded from the cell formation process. CAA assumes that each rejected part will be subcontracted and therefore is not considered in later iterations. Then, the next non-key part in the order is checked for assignment by means of another CIA execution, and so on. When CSUL is reached, the cell including the admitted parts and their related machine types is being formed. The remaining parts that are neither assigned nor rejected are considered in the next CAA iterations, and the rejected parts are added to the set of parts to be subcontracted.

Subsequent to the formation of a cell, CAA selects a new key part among the remaining parts. Thereafter, the remaining machine types having routing relationships with the key part are admitted to the new cell. Consequently, the non-key parts are determined among the remaining parts and they are examined by means of CIA. The remaining machines related to the selected parts are brought together to form this cell. A cell formation iteration is terminated when CSUL is reached. CAA iterations continue until all of the parts are either assigned to a cell or they are rejected.

4.5.2 Modifications and Extensions

Chow and Kusiak [27] use unit subcontracting costs to select the key part. Alternative criteria for selecting a key part such as unit production costs and part flow rates were also suggested. In this study, the work-load costs described in the previous chapter are proposed for determining key parts that might lead to solutions with better efficiency values. It is assumed that subcontracting costs are directly proportional to the manufacturing costs. The part j with the highest total work-load cost $TWLC_j$ is selected as the key in order to include the effect of demand volumes.

CAA expands cells by only examining the costs of the allowable non-key parts. A non-key part is considered to be allowable if it does not increase the cell size above the CSUL. An initial study showed that this orthodox scheme of selecting the part with the maximum cost for expansion can result in inferior PF/MG-F solutions. This scheme usually causes low inner-cell densities and high number of rejected parts. Moreover, this might prevent closer parts from being admitted to the cell as indicated in Table 4.7. If part-1 is assigned to the cell, the other parts are rejected. The gain from expanding the cell by part-1 is 78 whereas the loss is 44+37+22+19=122. Besides, part-1 is not close to the parts in the cell. A better expansion can be realized by rejecting part-1, and assigning other parts to the cell.

The following method is introduced to eliminate such adverse effects from CAA. First, allowable non-key parts are examined for low extra machine requirements. If the number of extra machines needed for a non-key part is less than or equal to cell admission factor (CAF), then the part is considered as a candidate. CAF is defined as the ratio of the maximum number of extra machines that can be assigned, to the current cell size. For instance,

Machine composition of the cell: acdfg Current size of the cell: 5 Cell size upper limit: 9

Part	Machine Composition	Extra Machines	Cost
1	abehi	behi	78
2	abdejk	bejk	44
3	cdek	ek	37
4	abcefj	bej	22
5	adefgj	ej	19

Table 4.7: An example of cell expansion in CAA.

in Table 4.7, the candidate parts for a CAF value of sixty percent are 3, 4, and 5.

Second, the set of extra machine requirements for each candidate part is examined to see whether it contains the set of extra machine requirements of other candidate parts. If there are such set inclusion relationships, then the latter parts can be included in the cell at no cost once the former is assigned. The costs of the latter candidate parts are added to the cost of the former part to determine the actual cost. For instance, it is possible to include part-5 with no additional cost once part-4 is assigned to the cell. In this case, the cost of part-4 is increased to a value of 22+19. The actual cost of the other candidate parts remain unchanged.

Finally, the candidate part having the maximum actual cost is assigned to the cell. Machine composition of the cell is also updated. If other candidate parts can be included into the cell with no additional machine requirements, then they are also assigned to the cell without increasing its size.

CAA operates on the machine-part incidence matrix. Hence, it causes the admission of all machines of a machine type into a single cell. However, if the work-load matrix is used instead of the incidence matrix, individual machines of the same type can be included to more than one cell to provide for more flexibility. This would clearly improve the quality of the PF/MG-F solutions generated by CAA. After each cell is formed, the required number of machines of each type is calculated. If there are remaining machines then

CAF (%)	Inter-cell Efficiency	Inner-cell Efficiency
10	0.2443	0.8952
20	0.3002	0.8460
30	0.3162	0.8043
40	0.3358	0.7311
50	0.3502	0.6574
60	0.3562	0.6466
70	0.3638	0.6060
80	0.3716	0.5875
90	0.3747	0.5853

Table 4.8: Effect of CAF on CAA solutions.

their corresponding types are considered in the formation of future cells. Thus, assignment of the machines of the same type to different cells can prevent the rejection of some parts. So, CAA iterations are carried out until there are no more unassigned machine. At this instant, the unassigned parts are entitled as rejected. This procedure automatically eliminates the need for CIA executions.

The effect of CAF on the PF/MG-F solutions generated by CAA is investigated. The results obtained by testing 92 randomly generated problems are illustrated in Table 4.8. A CAF value of 20 percent seems to be satisfactory, since the increase in inter-cell flow efficiency absorbs the decreasing effect of the inner-cell density. In the same investigation, the effect of CSUL is also analyzed. In order to prevent an implicit rejection of any part, CSUL should be set to a value greater than or equal to the number of machine types used by any part. This specifies a lower bound on CSUL, symbolized as LCSUL. LCSUL can easily be calculated from the incidence matrix. The number of 'ones' in the most dense column is taken as the LCSUL value. In this analysis, CSUL values are varied five times starting from LCSUL. The mean results obtained are given in Table 4.9. It can be observed from the table that it is better to set CSUL at its minimum possible value.

Finally, subcontracting can be eliminated by distributing the rejected parts among the cells. As shown by Tables 4.8 and 4.9, the effect of CAF and CSUL on low inter-cell flow efficiencies is relatively insignificant. CAA

CSUL	Inter-cell Efficiency	Inner-cell Efficiency
LCSUL+0	0.3136	0.7617
LCSUL+2	0.3246	0.7295
LCSUL+4	0.3342	0.6954
LCSUL+6	0.3475	0.6709
LCSUL+8	0.3546	0.6733

Table 4.9: Effect of CSUL on CAA solutions.

suggests the subcontracting of the rejected parts. However, the majority of manufacturing operations of all rejected parts can be processed in any one of the cells formed by CAA. This creates exceptional elements corresponding to the rejected parts. The following assignment scheme for the rejected parts is suggested for increasing the grouping efficiency. The cell having the maximum associated inner-cell work-load is selected for assigning a rejected part:

$$k \in CP(j) \Leftrightarrow WCC_{k,j} = Max\{WCC_{l,j} : l = 1, \cdots, K\} \quad , \forall j \in R;$$

where R : set of rejected parts,
K : total number of cells,
$CP(j)$: index set of cells where part j can be assigned,
$WCC_{k,j}$: workload cost of part j in cell k.

4.5.3 *Algorithm* {CAA}

S-1. Input work-load matrix (WL), cell admission factor (CAF), total workload costs for parts (TWLC), and number of machines of each type,
Compute CSUL;

S-2. Select the key part,
Assign the key part to the cell,
Add all machine types related to the key part into the cell;

S-3. Find candidate parts,
If there is no candidate part, then go to S-5,
Investigate set inclusion relations between extra machine requirement

```
     of candidate parts,
     Update cost of candidate parts;

S-4. Find the candidate part having the maximum cost,
     Expand the cell,
     If CSUL is not reached, then go to S-3;

S-5. Calculate number of machines for each type in the cell,
     If there are left-over machines go to S-2;

S-6. Assign rejected parts;

S-7. Output solution,
     Calculate efficiency measures;

end {CAA}.
```

4.6 Seed Clustering

ZODIAC is the only seed clustering PF/MG-F technique that is analyzed in this study. It was developed by Chandrasekharan and Rajagopalan [10,12]. The parts and machine types are treated independently during the initial phase. Rows of the machine-part incidence matrix represent machine types in binary vector format. Similarly, a binary vector for a specific part is obtained by taking the corresponding column in the incidence matrix. Parts and machine types are clustered separately by means of seeds. After the same number of part and machine clusters are obtained, similarity coefficients between the respective clusters are calculated. To form cells part and machine clusters are assigned to each other by the use of similarity coefficients. Final ZODIAC iterations try to improve the PF/MG-F solution generated at the end of the previous stage.

4.6.1 Description of ZODIAC

In order to follow the steps that ZODIAC undertakes, some concepts are needed. First, the clustering seed is defined and various seeds used in ZODIAC are analyzed. Thereafter, the efficiency variables employed in ZODIAC are introduced. Finally, the overall technique is summarized.

A *seed* is a binary vector of an appropriate dimension. If parts are subjected to clustering, the dimension of seeds is equal to the number of machine types. Conversely, the dimension of seeds used to cluster machine types is set equal to the number of parts. Seeds represent different regions in the simplex defined by a number of binary vectors. The set of binary vectors are either rows or columns of the incidence matrix. Each seed is capable of being a nucleus for clustering. That is each seed can identify a group of parts or machine types. Hence the number of seeds is equal to the current number of clusters at any clustering iteration.

The *data set* consists of binary vectors representing the machine types and/or parts obtained simply by taking the rows and/or columns of the incidence matrix, respectively. A *clustering* iteration groups the binary vectors in the data set into a number of clusters equal to the number of seeds selected. Since the part clustering iteration and the machine clustering iteration are the same, only the part clustering iteration will be introduced. The dimension of the chosen seeds is equal to the number of machine types. Each part's binary vector or equivalently the corresponding column of the incidence matrix is compared to all of the selected seeds. The part under consideration is admitted to the group represented by the closest seed in terms of Minkowsky metric distances [36]. ZODIAC uses d_1 metric for distances. The distance between a column of the incidence matrix and a seed is the total number of distinct elements. So, the following identity characterizes a clustering iteration:

$$\begin{aligned} k \in CP(j) \Leftrightarrow d_1(IM^j, s_k) &= \textstyle\sum_{i=1}^{T} |IM_i^j - s_k(i)| \\ &= Min\{d_1(IM^j, s_l) : l = 1, \cdots, K\}, \quad \forall j = 1, \cdots, P; \end{aligned}$$

where $CP(j)$: index set of clusters that part j can be assigned,

IM^j : column of the incidence matrix representing part j,

IM_i^j : $(i,j)^{th}$ entry of the incidence matrix,

s_k : seed representing cluster k,

$s_k(i)$: i^{th} entry of the seed representing cluster k,

$d_1(\cdot,\cdot)$: distance function in terms of d_1 metric,

T : total number of machine types,

K : total number of current cells,

P : total number of parts.

Upon completion of a clustering iteration, three kinds of groupings may occur. The first type of grouping is called simple part clusters if they contain more than one part. Singleton part clusters include only one part whereas null part clusters do not contain any part. ZODIAC usually eliminates null and singleton clusters.

Chandrasekharan and Rajagopalan [10] reported an upper bound for the number of cells. They stated that there can be no more than K^* manufacturing cells in a PF/MG-F problem with T machine types, P parts and E routing relations. They assumed that there are no alternative routing for any part and considered only incidence matrices where all machine types consist of single machines. K^* is determined based on the following observation from graph theory. In a disconnected bipartite graph with $T + P$ vertices and E edges, the number of components is a maximum when $K^* - 1$ components are single edged. They provided a formula for determining the maximum number of cells:

$$K^* = \left\lceil 0.5 \times \left((T + P - 1) - \sqrt{(T + P - 1)^2 - 4 \times (T \times P - E)} \right) \right\rceil .$$

This upper bound leads to the formation of *artificial seeds*. Artificial seeds are binary vectors chosen such that they represent totally different regions in the simplex. They are not picked from the data set. Initially, the number of artificial seeds K are determined. There are initially K^* artificial seeds to start with. Thereafter, each binary vector representing an artificial seed is divided into K zones. Each artificial seed has 'ones' in its associated zone and 'zeros' outside. An example of artificial seeds generated for both machine types and parts is illustrated in Table 4.10.

Arbitrary seeds are the binary vectors picked from the data set and then updated. MacQueen [30] proposed to choose the first K vectors in the data set. Later, Chandrasekharan and Rajagopalan [10] suggested the last K vectors as arbitrary seeds. Both of the two approaches take the next arbitrary seeds as the updated centroids of the cells formed in the previous clustering iteration. However, when the centroids are updated at the end of each iteration, the arbitrary seeds for the next iteration move towards the interior

PROBLEM INSTANCE	ARTIFICIAL SEEDS	
	Machine types	*Parts*
K=3	[11111 0000 0000]	[111 000 00]
T=8	[00000 1111 0000]	[000 111 00]
P=13	[00000 0000 1111]	[000 000 11]

Table 4.10: An example of artificial seeds.

of the simplex. They start to attract points from the opposite side. If this phenomenon continues, one seed becomes the main centroid of the simplex, attracting all vectors. At the end, this results in a single cluster, the original job shop. Chandrasekharan and Rajagopalan [12] observed this fact and suggested that no points other than the vertices of the simplex need to be considered as arbitrary seeds. They also indicated that updates of the seeds can be done by moving one extreme point of the simplex to another, not by moving towards the interior.

Representative seeds are the binary vectors each of which is picked from one preliminary group of data. Preliminary groups are formed by a clustering iteration that uses the artificial seeds. Representative seeds are always a subset of the actual columns or rows of the incidence matrix. A representative seed is chosen from each preliminary group. No representative seed is selected from singleton clusters. So, all singleton clusters are eliminated automatically.

As soon as the same number of part and machine clusters are obtained, machine-part cluster pairs are formed. *Ideal seeds* for machine clusters are determined from the assigned part clusters, or vice versa. Ideal seed for a part family is a binary vector containing 'ones' in the positions of the machine types that belong to the assigned machine group. For instance, the ideal seeds for the incidence matrix given in Figure 4.3-b are indicated in Table 4.11.

Chandrasekharan and Rajagopalan used a *similarity coefficient* to assign part clusters to machine clusters. The similarity coefficient is defined between a machine cluster and all part clusters as:

$$F_j = q \times \frac{e_i^j}{M_i \times P_j} + (1-q) \times \left(1 - \frac{e_i - e_i^j}{M_i \times (P - P_j)}\right) ,$$

CELL #	COMPOSITION *Machine*	*Part*	IDEAL SEEDS *Machine types*	*Parts*
1	1	1,2,3,4	[1 00 000 0000]	[1111 000 00 0]
2	2,3	5,6,7	[0 11 000 0000]	[0000 111 00 0]
3	4,5,6	8,9	[0 00 111 0000]	[0000 000 11 0]
4	7,8,9,10	10	[0 00 000 1111]	[0000 000 00 1]

Table 4.11: An example of ideal seeds.

where e_i^j : total number of 'ones' in part cluster j of machine cluster i,
e_i : total number of 'ones' in machine cluster i,
M_i : size of machine cluster i,
P_j : size of part cluster j,
P : total number of parts,
q : weighting factor between 0 and 1.

For each machine group, a similarity coefficient value for each part family is computed. The one that gives the highest F_j value is assigned to the machine group. This process is repeated for all machine clusters and the assigned part family is eliminated at each iteration. Consequently, the machine-part incidence matrix is arranged in a block-diagonal form.

The application of this similarity coefficient tries to maximize the *clustering efficiency*. A quantitative scale is used to measure the clustering efficiency, ξ, which is defined in quite a similar way as that of the grouping efficiency described in the previous chapter:

$$\xi \doteq q \times \xi_1 + (1-q) \times \xi_2 \quad ,$$

where ξ_1 : ratio of 'ones' to total number of elements in diagonal blocks,
ξ_2 : ratio of 'zeros' to total number of elements in off-diagonal blocks.

The limitations imposed by the data set are also analyzed by a *limiting efficiency* measure. Three cases for the limiting efficiency are considered. Before describing the limiting efficiency, the concept of *accommodation* needs to be introduced. Accommodation indicates the portion of the incidence matrix area covered by the diagonal-blocks representing the cells. First, machine

$$\underset{a)Maximum}{\begin{bmatrix} 1 & & & & & & & & & \\ & 1 & 1 & & & & & & & \\ & 1 & 1 & & & & & & & \\ & & & 1 & 1 & 1 & & & & \\ & & & 1 & 1 & 1 & & & & \\ & & & 1 & 1 & 1 & & & & \\ & & & & & & 1 & 1 & 1 & 1 \\ & & & & & & 1 & 1 & 1 & 1 \\ & & & & & & 1 & 1 & 1 & 1 \\ & & & & & & 1 & 1 & 1 & 1 \end{bmatrix}} \quad \underset{b)Minimum}{\begin{bmatrix} 1 & 1 & 1 & 1 & & & & & & \\ & & & & 1 & 1 & 1 & & & \\ & & & & 1 & 1 & 1 & & & \\ & & & & & & & 1 & 1 & \\ & & & & & & & 1 & 1 & \\ & & & & & & & 1 & 1 & \\ & & & & & & & & & 1 \\ & & & & & & & & & 1 \\ & & & & & & & & & 1 \\ & & & & & & & & & 1 \end{bmatrix}}$$

Figure 4.3: Accommodation.

clusters and part clusters are ordered in increasing sizes. If they are assigned to each other in this order, the largest possible block-diagonal format would be realized. On the other hand, if the k^{th} largest machine group is assigned to the k^{th} smallest part family, the off-diagonal area would have the largest possible area. These two cases are shown in Figure 4.3. In the former case, accommodation is said to be at the maximum (S_{max}) while accommodation is at the minimum (S_{min}) in the latter case. If the total number of 'ones' (E) in the data set is less than the minimum accommodation, the diagonal blocks can include all 'ones'. Thus, it is hypothetically possible to avoid the exceptional elements. Conversely, exceptional elements are inevitable if E is greater than S_{max}. In the remaining case, if E is between S_{min} and S_{max}, the limiting efficiency is set to unity. Hence, the limiting efficiency, ξ_0, is defined as:

$$\xi_0 \doteq \begin{cases} q \times \frac{E}{S_{min}} + (1-q) & if \;\; E < S_{min}, \\ 1 & if \;\; S_{min} \leq E \leq S_{max}, \\ q + (1-q) \times \frac{P \times T - E}{P \times T - S_{max}} & if \;\; E > S_{max}. \end{cases}$$

The limiting efficiency leads to the definition of *relative efficiency*, ξ_R, as:

$$\xi_R \doteq \frac{\xi}{\xi_0}$$

ZODIAC operates in the following manner. First, ZODIAC generates artificial seeds for part clusters based on the limiting number of cells, K^*. After preliminary groupings of the columns are formed, the number of cells is updated by discarding the null clusters. Thereafter, the representative seeds for parts are chosen, and the columns are reclustered. The number of non-null part clusters is computed. Next, the above iterations are carried out for the rows of the incidence matrix by generating the artificial seeds for machine types. If the number of part clusters is not equal to that of the machine clusters, the number of cells is set to the minimum, and the whole process is repeated. This main loop terminates whenever the number of part and machine clusters are equal to each other.

After equal number of part and machine clusters are obtained, the similarity coefficients are computed. A part cluster is allocated to each machine cluster according to the similarity coefficient values. Thereafter, the corresponding columns and rows of the original incidence matrix are reordered to obtain a block-diagonal structure. In the second stage of the ZODIAC technique, the associated efficiency measures are calculated. If the relative clustering efficiency is extremely good, the PF/MG-F solution found is considered as the final solution. Otherwise, improvements on the current PF/MG-F solution are attempted in the next stage.

In the third stage, ZODIAC reclusters part families and machine groups by employing the ideal seeds. The indicated part and machine clustering is dependent on each other because of the ideal seeds. Initially, ideal seeds for part clusters are generated. After parts are clustered and null clusters are eliminated, the representative seeds for the part clusters are determined. Then, parts are reclustered. Thereafter, the ideal seeds for machine clusters are computed and the above iteration is repeated. The columns and rows of the incidence matrix are reordered according to their new clusters. The calculation of the efficiency values completes an ideal-seed clustering iteration.

If the current relative clustering efficiency values are approximately equal to unity, ZODIAC terminates. If the efficiency values of the previous solution are better than the current values, then the earlier composition is used as the final PF/MG-F solution. If neither the current efficiency values are satisfactory, nor they are relatively inferior as compared to the previous values, another ideal-seed clustering iteration is carried out. At this instant, the smallest cell represented by the smallest diagonal block can optionally be eliminated. This is accomplished by decreasing the number of ideal seeds by one and not choosing an ideal seed from this block. These final iterations are terminated when the relative efficiency approaches unity or it is less than that of the previous iteration. Finally, a part clustering is performed according to the ideal part seeds to adjust the final form of the PF/MG-F solution.

4.6.2 Modifications

Arbitrary choice of a representative seed from each group may not represent the corresponding cluster. Hence, the most dense binary vector in each cluster is selected as the representative seed. Since the number of possible cells is decreased due to singleton and null clusters, the remaining representative seeds can be determined by the following method composed of two passes. The first pass is performed to identify a seed from each non-singleton cluster. During the second pass, a number of secondary representative seeds are selected. The binary vectors that are distant from all seeds are picked until the limit K on the number of cells is reached. This way of fixing the representative seeds gives better results.

In order to eliminate the final adjustment through a part clustering, the following modification in the last ZODIAC stage is proposed. An ideal seed clustering iteration begins with a row clustering instead of a column clustering, followed by another iteration for parts. Moreover, an unequal number of part and machine clusters may exist after an ideal seed clustering iteration. This is due to the elimination of singleton clusters in generating representative seeds. In addition, intermediate PF/MG-F solutions having equal number of part and machine clusters can be obtained during an ideal seed clustering iteration. Some of the intermediate solutions are observed to be superior in terms of the clustering efficiency measure as compared to the final solutions obtained at the end. Furthermore, the elimination of the smallest cell in between two ideal seed clustering iterations might ruin the block-diagonal structure because of the use of representative seeds. Since there is no ideal seed selected for the eliminated block, the parts assigned to that block are distributed to other part clusters by means of initial clustering based on ideal seeds. If any one of the representative seeds that correspond to the remaining part clusters happens to be one of such parts, then the block-diagonal structure is destroyed. This drawback can be handled by machine clustering using ideal seeds at the beginning of an ideal seed clustering iteration, followed by a part clustering operation.

The above handicaps are removed by modifying the steps of final ZODIAC iterations. The initial step of an ideal seed clustering iteration involves a row clustering using ideal seeds for machines followed by a column clustering employing again ideal part seeds. Thereafter, a column clustering is performed using representative part seeds. Then, the number of cells is updated. Consequently, ideal seeds for machines are generated based on the final part clustering operation using representative seeds. Rows of the incidence matrix are also clustered by means of these ideal seeds. If the number of part and machine clusters are the same, the incidence matrix is reordered and the efficiency measures are calculated. Otherwise, the process is repeated for obtaining an equal number of part and machine clusters through a clustering of columns using ideal part seeds. If there are an equal number of part and machine clusters, columns of the incidence matrix are permuted, and the efficiency measures are computed. If the number of part clusters is still not equal to the number of machine clusters, the above steps of the modified ideal seed clustering iteration is repeated for parts. These alternating executions of the above steps continue until an equal number of part and machine clusters are obtained like in the first stage of ZODIAC.

4.6.3 *Algorithm* {ZODIAC}

```
S-1. Input machine-part incidence matrix, weighting factor $q$, threshold value
     for representative seeds,
     Calculate maximum allowable number of cells, $K^*$,
     Set $K \leftarrow K^*$;

S-2. Choose $K$ artificial seeds for columns,
     Cluster columns,
     Choose representative seeds for columns,
     Cluster columns;

S-3. Find number of non-null column clusters, $K_C$,
     Modify $K \leftarrow K_C$,
     Repeat S-2 for rows,
```

```
       Find number of non-null row clusters, $K_R$;

S-4.   Modify $K \leftarrow Min\{K_R, K_C\}$,
       If $K_R \neq K_C$, then go to S-2,
       Reorder rows and columns in the order of cluster membership;

S-5.   Compute similarity coefficients,
       Allocate part clusters to machine clusters,
       Reorder columns according to the new order of clusters;

S-6.   Compute clustering efficiency $\xi$, and relative efficiency $\xi_R$,
       If $\xi_R \cong 1$, then go to S-11,
       Set $\hat{\xi} \leftarrow \xi$;

S-7.   Generate ideal seeds for machine clusters,
       Cluster rows,
       Modify $K \leftarrow K_R$,
       Generate ideal seeds for part clusters,
       Cluster columns,
       Modify $K \leftarrow Min\{K_R, K_C\}$,
       Generate representative seeds for part clusters,
       Cluster columns,
       Modify $K \leftarrow Min\{K_R, K_C\}$,
       Generate ideal seeds for machine clusters,
       Cluster rows;

S-8.   If $K_R \neq K_C$, then go to S-9,
       Generate ideal seeds for part clusters,
       Cluster columns,
       If $K_R \neq K_C$, then go to S-7,
       Go to S-10;

S-9.   Replace columns by rows,
       Repeat S-7,
       If $K_R \neq K_C$, then go to S-7;

S-10.  Reorder columns according to the new order of clusters,
       Compute $\xi$ and $\xi_R$,
       If $\xi_R \cong 1$, then go to S-11,
       If $\xi < \hat{\xi}$, then revert to the earlier grouping and go to S-11,
```

```
        Replace $\hat{\xi} \leftarrow \xi$,
        Liquidate the smallest block (optional),
        Go to S-7;

S-11.   Output solution,
        Calculate efficiency measures;

  end {ZODIAC}.
```

5. EXPERIMENT

Most of the suggested PF/MG-F techniques have usually been illustrated by a limited number of examples. These examples have been designed to show the effectiveness of the individual techniques, and all of the considered PF/MG-F problems have been small sized and easy to follow. Almost all of the authors reported that their proposed PF/MG-F techniques have been implemented satisfactorily. However, the details of the implementations have not been disclosed. Moreover, only a limited number of PF/MG-F techniques has been compared against each other by means of only the example problems used in their analyses.

A comprehensive comparison of the selected PF/MG-F techniques are performed by means of randomly generated test problems. All of the techniques are compared by means of the efficiency measures described in the third chapter. First, the module that generates the test problems for PF/MG-F will be presented in this chapter. Consequently, factors of the experiment will be described. Finally, the results obtained will be presented and discussed in detail.

5.1 Problem Generator

In this study, a random PF/MG-F problem generator used to evaluate and compare the existing PF/MG-F techniques is developed. The problem generator first produces a machine-part incidence matrix. Based on the information from the incidence matrix, operation sequences and standard times for the manufacturing operations of each part are generated. Thereafter, annual depreciation costs and availabilities of machine types, and annual demands of parts are created. This characterizes a work-load matrix. The number of individual machines of each type and total work-load cost of each part are computed from the work-load matrix. Finally, the efficiency measures for the generated shop are calculated. An example of the generation process is illustrated in Appendix A.

5.1.1 Generating Machine-Part Incidence Matrices

The most important stage in generating test problems for PF/MG-F is the creation of machine-part incidence matrices. This process is at the top of the hierarchy of generating an artificial PF/MG-F problem. The majority of the reported PF/MG-F techniques use the information presented in the incidence matrix. The incidence matrix indicates size of the original shop, density of the shop, and place of the generated shop in the manufacturing spectrum.

The *manufacturing spectrum* consists of two ends, ideal job shop and ideal CM shop. Ideal job shop is the manufacturing environment where all parts use all machines. Conversely, ideal CM shop represents the existence of manufacturing cells that are mutually separated and therefore independent. In other words, there exists no inter-cell interactions between the cells of an ideal CM system. Flow shop systems can be thought of as a special subset of ideal CM systems whereas project-oriented systems can be regarded as ideal job shop systems. An ideal job shop system can be characterized by a machine-part incidence matrix consisting of only 'ones'. On the other hand, an incidence matrix with a perfect block-diagonal structure illustrates an ideal CM system. A perfect block-diagonal structure is realized if diagonal blocks contain all 'ones' and no 'zeros' inside, and the reverse in the off-diagonal area.

Four parameters are required to construct a machine-part incidence matrix. The first two are *size* parameters showing the size of the shop to be generated. These parameters are the number of parts and the number of machine types in the system. The size parameters determine the dimension of the incidence matrix. The remaining two are *shape* parameters. The first one is called density, representing the ratio of 'ones' in the incidence matrix to the total area. The other shape parameter is called clumpiness which identifies the location of the generated shop on the manufacturing spectrum. The clumpiness parameter shows the degree of block-diagonalization in the machine-part incidence matrix. This parameter always takes positive values. An ideal job shop can be obtained by specifying the density parameter as one. Ideal CM systems can be generated by setting lower values of the density parameter such as 0.10 and high values of the clumpiness parameter such as 1000.

The detailed analysis of the generation process requires the definition of inner-cell density and off-diagonal density. Let,

T : number of machine types (number of rows of incidence matrix),

P : number of parts (number of columns of incidence matrix),

k : number of imaginary cells (number of possible diagonal blocks in incidence matrix,

c : clumpiness parameter,

d : overall density of the shop (density of incidence matrix),

d_I : inner-cell density of imaginary cells (density of block-diagonal area in incidence matrix),

d_O : density of exceptional elements (density of off-diagonal area in incidence matrix),

$\bar{t}$: mean number of machine types in each imaginary cell $[\bar{t} \doteq T \times d]$,

$\bar{p}$: mean number of parts in each imaginary cell $[\bar{p} \doteq P \times d]$.

The number of 'ones' in the incidence matrix, E, can be calculated in the following two ways:

$$E = T \times P \times d,$$
$$E = (k \times \bar{t} \times \bar{p}) \times d_I + ((T \times P) - (k \times \bar{t} \times \bar{p})) \times d_O.$$

By assuming that the number of imaginary cells is inversely proportional to the overall density, and from the above equations

$$k \doteq \frac{1}{d} \qquad \leadsto \qquad d = d \times d_I + (1 - d) \times d_O.$$

The above equation can be solved for d_O and d_I by parameterizing on c as:

$$d_O = \frac{d}{c}\ , \qquad d_I = 1 - \frac{1-d}{c}.$$

Thus, the clumpiness parameter and the overall density give rise to inner-cell and off-diagonal densities. The effect of the shape parameters on the

c	d	d_O	d_I	c	d	d_O	d_I
1	0.10	0.1000	0.1000	6	0.10	0.0166	0.8500
	0.15	0.1500	0.1500		0.15	0.0250	0.8583
	0.20	0.2000	0.2000		0.20	0.0333	0.8687
2	0.10	0.0500	0.5500	7	0.10	0.0143	0.8714
	0.15	0.0750	0.5750		0.15	0.0214	0.8786
	0.20	0.1000	0.6000		0.20	0.0286	0.8857
3	0.10	0.0333	0.7000	8	0.10	0.0125	0.8875
	0.15	0.0500	0.7167		0.15	0.0188	0.8938
	0.20	0.0667	0.7333		0.20	0.0250	0.9000
4	0.10	0.0250	0.7750	9	0.10	0.0111	0.9000
	0.15	0.0375	0.7875		0.15	0.0167	0.9056
	0.20	0.0500	0.8000		0.20	0.0222	0.9111
5	0.10	0.0200	0.8200	10	0.10	0.0100	0.9100
	0.15	0.0300	0.8300		0.15	0.0150	0.9150
	0.20	0.0400	0.8400		0.20	0.0200	0.9200

Table 5.1: Effect of shape parameters on inner-cell and off-diagonal densities.

densities is illustrated in Table 5.1. It can be observed from the table that the effect of the clumpiness parameter on both of the individual densities is more significant than the effect of the overall density on them.

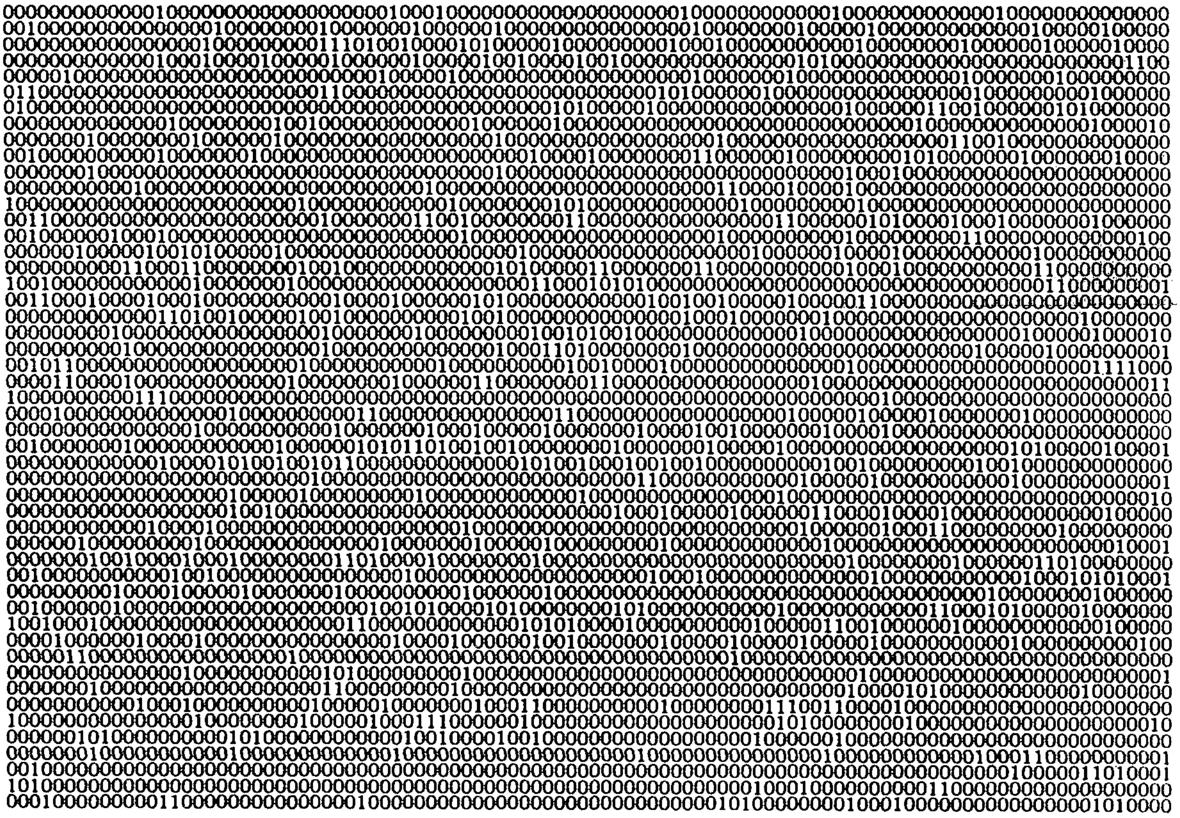
The following procedure is used to generate a machine-part incidence matrix. First, the dimension of the incidence matrix is fixed by the size parameters. Thereafter, the number of imaginary cells (k) is determined by taking the lowest integer value greater than the reciprocal of the density parameter. The number of machine types and parts for the first k-1 imaginary cells are generated from uniform distributions having means $\bar{t}$ and $\bar{p}$, respectively. The composition of the last imaginary cell is determined by the remaining machine types and parts. In addition, inner-cell density and off-diagonal density of the incidence matrix are computed.

In the second stage, the incidence matrix is generated rowwise. For each machine type in the shop, the associated row of the incidence matrix is determined element by element. A $U(0,1)$ random variable is generated for each entry in the corresponding row. If the entry belongs to an imaginary cell, and the generated random number is less than or equal to the inner-cell density (d_I) , it takes a value of 'one'. If the entry falls into the off-diagonal area, the random number is compared with the off-diagonal density (d_O). In this case, the entry is a 'one' only if the generated random number is less than or equal to d_O. If the random number is greater than the associated density in both cases, the entry becomes a 'zero'. After all rows of the incidence matrix are generated, a search on all rows and columns is performed to identify whether there exists a column or row containing only 'zeros'. If this is the case, then the related column or row is regenerated. Finally, the columns and rows of the incidence matrix are permuted randomly. Since all of the selected PF/MG-F techniques do not depend on the initial form of the incidence matrix, the last step can be eliminated with no adverse effect on the final PF/MG-F solutions generated by the selected techniques. Some examples of the generated incidence matrices based on different clumpiness values are given in Figure 5.1.

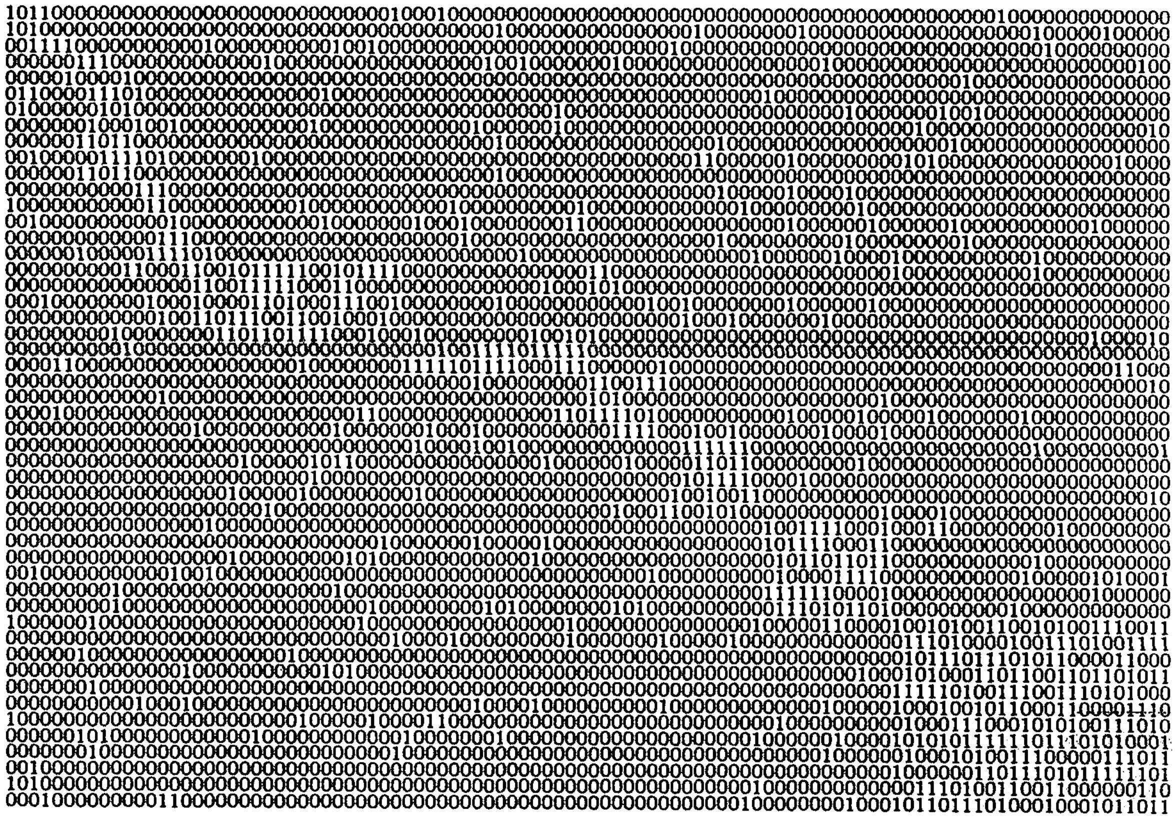
5.1.2 Generating Work-Load Matrices

A work-load matrix corresponding to the machine-part incidence matrix is generated next. In this stage, operation sequences and standard times of parts are determined after processing rates of all machine types are fixed. Consequently, annual demands of parts and annual available capacities of machine types are produced. Thereafter, the work-load matrix is computed. Finally, the number of machines of each type and total work-load costs of each part are calculated from the work-load matrix.

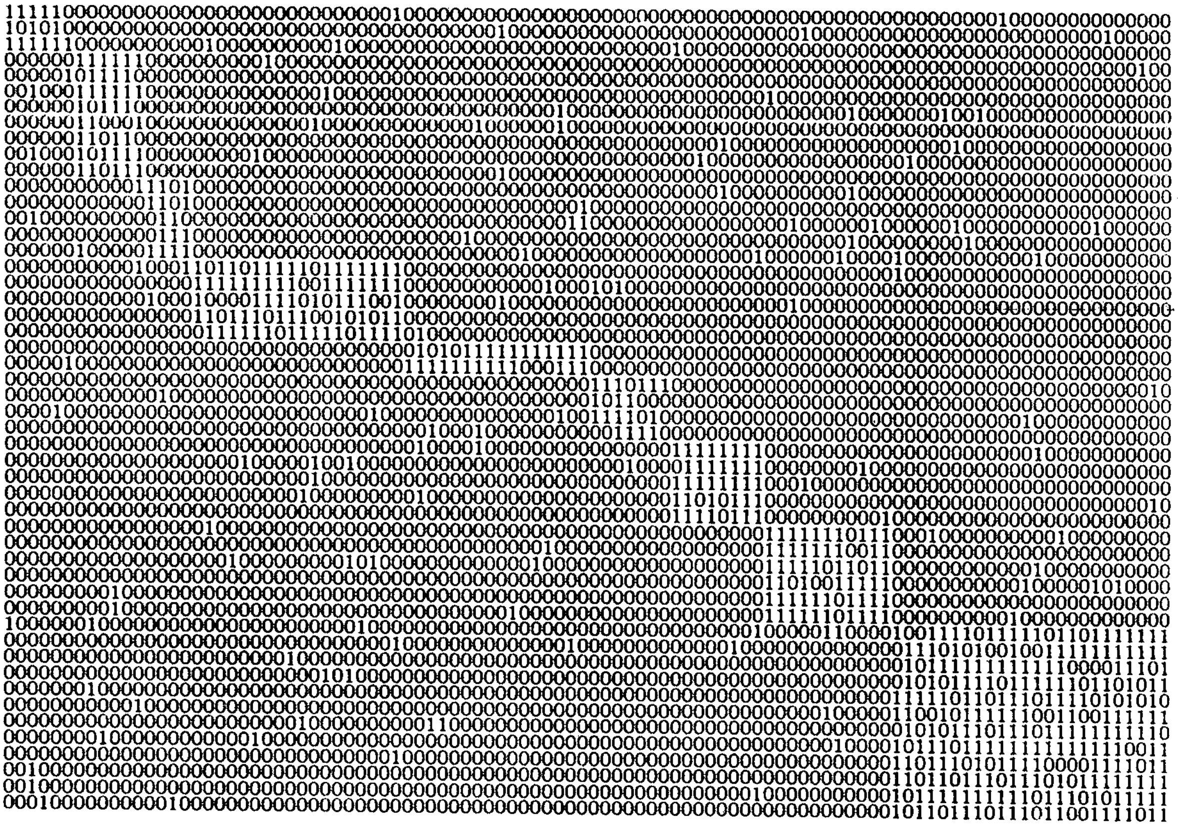
The number of operations of a part is bounded below by the total number of 'ones' (π) in the corresponding column of the incidence matrix. There could be more operations than the number of machine types that a part uses. The total number of operations that a part requires is generated from a uniform distribution between a lower bound set equal to the total number of machine types used to produce the part and a suitable upper bound. The upper bound is specified as $100 + \delta$ percent of the number of machine types on which the part is processed. For example, the number of operations is generated from a uniform distribution between 10 and 15 if the total number of machine types used is 10 and δ is specified to be 50 percent. The operation sequence for each part is next determined so that no consecutive machine types are the



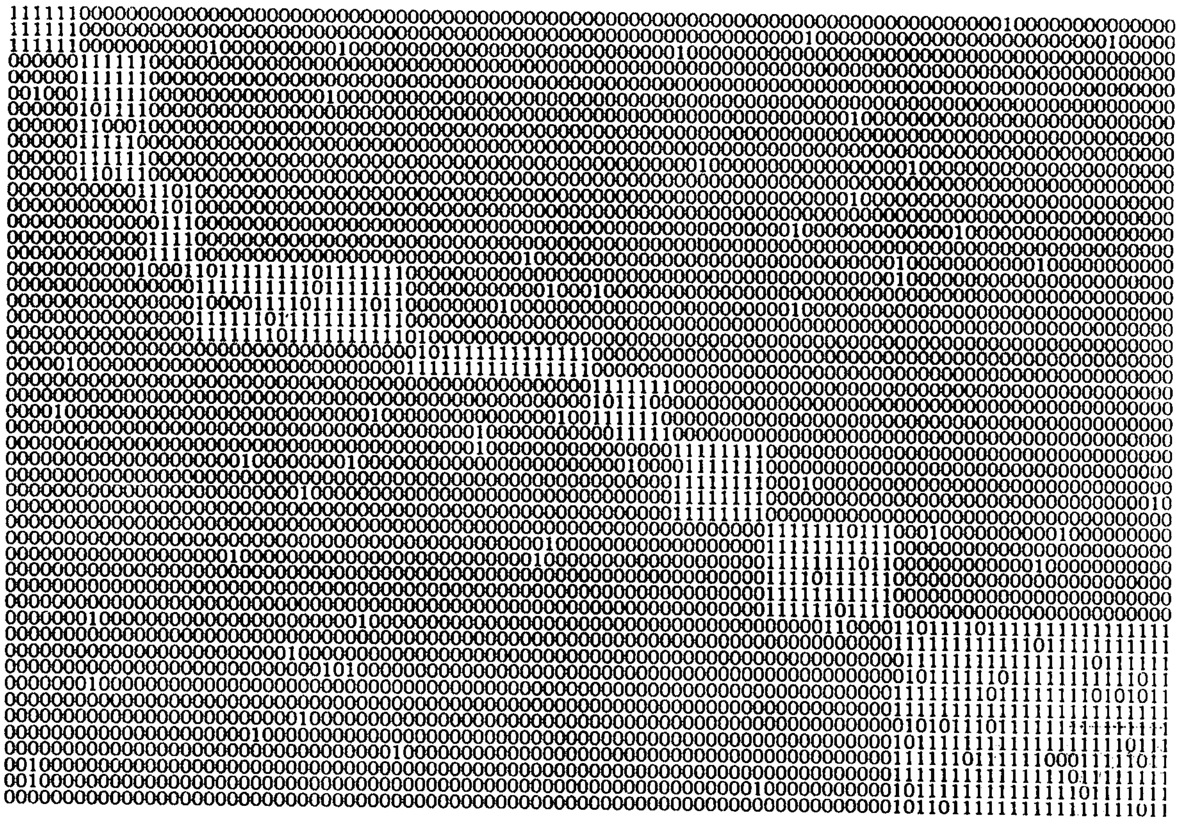
a) Clumpiness=1



b) Clumpiness=2



c) Clumpiness=4



d) Clumpiness=9

Figure 5.1: Effect of the clumpiness parameter on incidence matrices.

same. In other words, "$m1 \mapsto m2 \mapsto m2 \mapsto m1 \mapsto m1 \mapsto m3$" is not a possible operation sequence whereas "$m1 \mapsto m2 \mapsto m1 \mapsto m3$" is possible. Moreover, the standard time for each operation is randomly generated from the stratified exponential distribution with parameter set to the processing rate parameter of the related machine type.

The work-load of a part on a machine type is defined as the number of of machines that the part uses within a year. Annual demands of all parts and the available capacity of single individual machines for all types are calculated to aid the computation of the work-loads. Annual part demands are assumed to be uniformly distributed. Nevertheless, any other kind of distribution can easily be incorporated. Annual capacity of each machine type on the other hand, is assumed to depend only on the availability of workers. If it is assumed that there is a fixed commitment of working hours in a year, this leads to constant annual capacities for machine types.

After annual part demands and machine availabilities are determined, the work-load matrix is generated. An element of the work-load matrix takes a positive value if and only if the corresponding entry in the incidence matrix is 'one'. Elementary machine fraction calculations are carried out for each machine type in the operation sequence of each part in order to determine the values of the positive entries of the work-load matrix. If a machine type is repeated in an operation sequence, the corresponding machine fractions are simply added to compute the total work-load of the part on this machine type. Any 'zero' entry in the incidence matrix indicates that no operation takes place on the corresponding machine type.

The number of machines available of each type is computed by adding the work-load matrix values of the corresponding row. The smallest integer greater than or equal to the calculated value specifies the available number of individual machines of the associated type. Moreover, annual depreciation costs of machine types are generated uniformly. It is again possible to use any other distribution for this purpose. By using the annual depreciation costs and information in the work-load matrix, the efficiency indices described in the third chapter can be calculated. At this stage, the incidence matrix corresponds to a job shop environment. Thus, there is no exceptional element and therefore the inter-cell flow efficiency is always unity. Furthermore, the under-utilization index has the highest value since both machine duplications and work-load loss due to exceptional elements do not exist.

5.1.3 *Procedure* {Generator}

```
S-1. Input number of machine types, number of parts, density, and clumpiness,
     Input random number seeds,
     Input limits of uniform distributions for number of parts and number
     of machine types in each imaginary cell, processing rates for machines,
     demands for parts, depreciation costs for machines,
     Input upper end percentage $\delta$ for number of operations of parts, annual
     machine capacities;

S-2. Compute inner-cell and off-diagonal densities,
     Generate incidence matrix rowwise,
     Check for empty row and/or column,
     If there exists empty row or column, regenerate it,
     Permute rows of the incidence matrix randomly,
     Permute columns of the incidence matrix randomly;

S-3. Determine number of operations of parts,
     Determine machine type sequence for parts,
     Compute standard times for operations,
     Compute annual demands for parts,
     Generate work-load matrix;

S-4. Calculate number of machines of each type,
     Determine annual depreciation costs for machines,
     Compute efficiency measures,
     Output the generated shop;

end {Generator}.
```

FACTORS		VALUES		#
Size Parameters	Number of machine types	$T = 50$		1
	Number of parts	$P = 100, 150$		2
Shape Parameters	Density	$d = .10, .15, .20$		3
	Clumpiness	$c = 1, 2, 4, 9$		4
Distribution Parameters	Number of machine types in imaginary cells	$U[0.4 \times \bar{t}; 1.6 \times \bar{t}]$		1
	Number of parts in imaginary cells	$U[0.6 \times \bar{p}; 1.8 \times \bar{p}]$		1
	Number of operations	$U[\pi; 0.33 \times \pi]$		1
	Machine rates	$U[2; 11]$		1
	Annual part demands	High variance	$U[50; 4050]$	2
		Low variance	$U[1050; 3050]$	
	Annual machine depreciations $	High variance	$U[100; 100100]$	2
		Low variance	$U[25050; 75050]$	
Other	Annual machine availabilities	120000 min.		1
Total number of combinations				96
Total number of statistical independent replications				10
Total number of selected PF/MG-F techniques				6

Table 5.2: Factors of the experiment.

5.2 Design of Experiment

Performances of the selected PF/MG-F techniques are tested under artificially but randomly generated conditions. In order to reduce the size of the experiment, some input parameters of the generator are fixed. In this study, the effects of the variable input parameters on the selected PF/MG-F techniques are investigated.

The approach used in this study is to design a factorial experiment. The factors involved in the experiment are the number of parts, density of the shop, clumpiness, demand of parts, and machine depreciation costs. The other parameters influencing the PF/MG-F solutions are fixed. These are the number of machine types, annual availability of machines, the distribution parameters of number of machine types and parts in imaginary cells, number of operations of parts, and processing rates of machines. A summary of the factors and their levels in the experimentation is presented in Table 5.2.

Two levels for the number of parts are set to evaluate the behavior of the selected PF/MG-F techniques under different problem sizes. The main motivation behind taking more than one value for the number of parts is to investigate how a specific cell system will be affected if some new parts are introduced into the system. Both of the shape parameters determine the type of the shop being generated. The densities selected in this study are based on the results of example problems tested and specific implementations reported in the literature. Densities between 0.10 and 0.20 are sufficient for the incidence matrices used in these kind of experiments. So, levels of the density factor include both boundary values and the mean value. Small values for the clumpiness parameter substantially change the shape of incidence matrices. However, as the clumpiness value increases, a decrease in the marginal changes of the shape of the incidence matrices is observed. Because of this reason, four levels of the clumpiness factor is utilized, these being 1, 2, 4, and 9. The effect of different clumpiness levels on an example problem with a density of 0.10 can be seen in Figure 5.1. Finally, both demand and depreciation cost factors are analyzed at two levels, high variance and low variance. Different variabilities among machine types and parts are evaluated by this means.

Ten different PF/MG-F problems are generated for each combination of the factors. These problems are statistically independent from each other. Statistical independence is achieved by using a different seed for each problem generated.

Each of the selected techniques requires setting some specific parameters before they can be applied to problems. Therefore the performances of the six selected PF/MG-F techniques under different scenarios are measured and compared. The best fine-tuned values of these parameters are illustrated in Table 5.3. Sensitivities of these fine-tuning parameters are evaluated by a number of sample runs.

The evaluation consists of analyzing each technique under 96 different scenarios. For each scenario, ten statistically independent PF/MG-F problems are generated. Hence, the experimentation consists of a total of 5760 runs:

$$6 \times 10 \times 2 \times 3 \times 4 \times 2 \times 2 = 5760.$$

ALGORITHM	FACTOR	VALUE
COMBGR	Maximum machine difference limit	7
MODROC	Lower limit on similarities	0.75
	Upper limit on independent pairs	5
MACE	Threshold values	0.10
	Similarity (job shop like)	*SCTF*
	Similarity (intermediate)	*PSC*
	Similarity (ideal CM like)	*SC*
ZODIAC	Weighting factor	0.50
	Threshold value	7
WUBC	Cell admission factor	0.60
	Cell size upper limit	50
	Key machine selection rule	A4
	Part assignment rule	B2
CAA	Cell admission factor	0.20
	Extra factor on cell size limit	0

Table 5.3: Values of fine-tuning parameters of the selected techniques.

5.3 Results and Discussion

Performances of the six selected PF/MG-F techniques are analyzed in terms of five factors. The three efficiency indices described in the third chapter are computed for each combination of these factors. At any instant, only one factor level is changed while the others are held fixed. Furthermore, ten different PF/MG-F problems are tested for each combination. All of the efficiency values of the solutions generated by each PF/MG-F technique for each test problem combination are determined by simply taking the averages of the ten replications. Thereafter, for each technique, the efficiency values of each level of each factor are calculated by taking the mean of the values of all combinations. So, the main effects of all factors are determined for all of the selected PF/MG-F techniques.

The main effects indicate that some factors are quite sensitive whereas the others are not. The shape parameters are found to be the most sensitive factors. Different levels of density and clumpiness factors generate different PF/MG-F solutions for all of the selected techniques. For this reason, a two-way analysis for the cross effects is performed on the results. On the other hand, the selected PF/MG-F techniques are insensitive to the annual

PF/MG-F TECHNIQUE	EFFICIENCY MEASURE	# OF PARTS		DEMAND		DEPREC.	
		100	150	H.Var.	L.Var.	H.Var.	L.Var.
COMBGR	Inter-cell flow	100.00	100.00	100.00	100.00	100.00	100.00
	Inner-cell density	31.37	31.48	31.50	31.35	31.65	31.21
	Work-load balance	72.54	73.50	72.67	73.37	73.02	73.02
	Under-utilization	62.12	58.71	60.71	60.12	60.44	60.40
MODROC	Inter-cell flow	68.67	77.46	73.04	73.08	73.08	73.04
	Inner-cell density	46.46	49.54	47.88	48.12	48.04	47.96
	Work-load balance	53.69	53.25	53.62	53.31	53.48	53.46
	Under-utilization	85.29	87.58	86.25	86.63	86.42	86.46
WUBC	Inter-cell flow	95.60	96.42	96.00	96.02	96.02	96.00
	Inner-cell density	22.35	23.44	22.96	22.83	23.08	22.71
	Work-load balance	84.58	88.08	86.21	86.46	86.33	86.33
	Under-utilization	51.79	52.69	52.08	52.40	52.25	52.23
CAA	Inter-cell flow	74.37	78.15	76.31	76.21	76.46	76.06
	Inner-cell density	57.60	59.27	58.94	57.94	58.79	58.08
	Work-load balance	78.52	78.33	78.35	78.50	78.44	78.42
	Under-utilization	57.12	57.04	57.10	57.06	57.23	56.94
ZODIAC	Inter-cell flow	62.44	66.33	64.40	64.37	64.37	64.40
	Inner-cell density	74.42	73.58	74.04	73.96	74.21	73.79
	Work-load balance	85.60	87.75	86.50	86.85	86.67	86.69
	Under-utilization	58.56	56.48	57.60	57.44	57.52	57.52
MACE	Inter-cell flow	65.02	64.85	64.92	64.96	66.04	63.83
	Inner-cell density	81.42	81.27	81.42	81.27	81.63	81.06
	Work-load balance	84.35	86.15	85.21	85.29	85.00	85.50
	Under-utilization	57.96	55.67	57.08	56.54	56.42	57.21

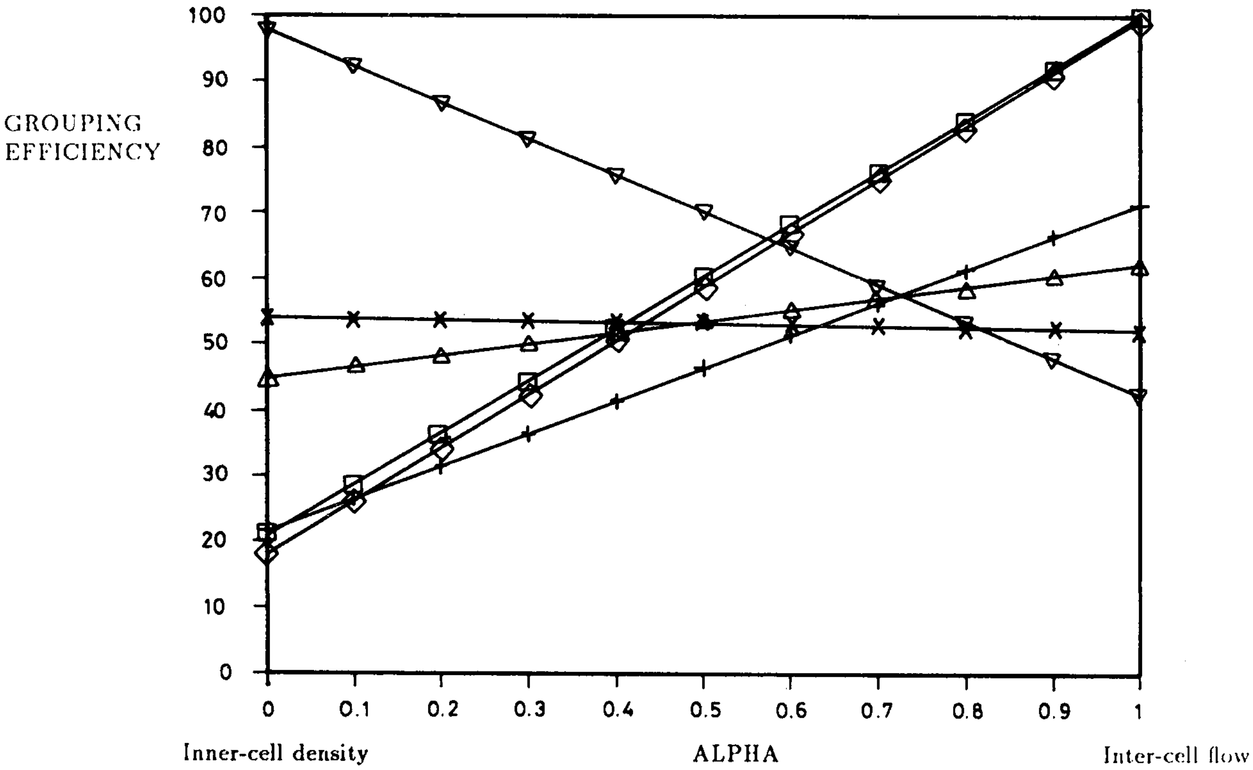
Table 5.4: Main effects of the insensitive factors.

demand variations of parts and the annual depreciation cost variations of machine types with respect to all efficiency measures. Moreover, increasing the number of parts from 100 to 150 changes the results slightly. The effect of the number of parts on the PF/MG-F solutions is considered to be so little that no further analysis on this factor is made. The main effects of the insensitive factors are given in Table 5.4.

As the number of parts is increased, the best improvement in under-utilizations is achieved by COMBGR. MODROC is the most adversely affected PF/MG-F technique by this factor. Both inter-cell flow and inner-cell density measures increase with an increase in the number of parts. The effect of increasing the number of parts on the MODROC's solutions is also significant in terms of work-load balances. When the number of parts is increased to 150, MODROC creates solutions with higher under-utilization values as compared to the case when the number of parts is 100. The WUBC technique leads to better PF/MG-F solutions in terms of the work-load balance efficiency when the number of parts is increased. CAA generates superior solutions with respect to the grouping efficiency if the number of parts is held at 150 instead of 100. However, this increase in the grouping efficiencies is small as compared to that of MODROC's. Application of ZODIAC to the PF/MG-F problems with small number of parts causes inter-cell densities to increase. Higher values for the number of parts make ZODIAC generate solutions with better inter-cell flow efficiencies and work-load balance measures. The solutions created by MACE have better work-load balances and under-utilizations when the number of parts is increased from 100 to 150. It is observed from Table 5.4 that the overall effect of an increase in the number of parts is insignificant.

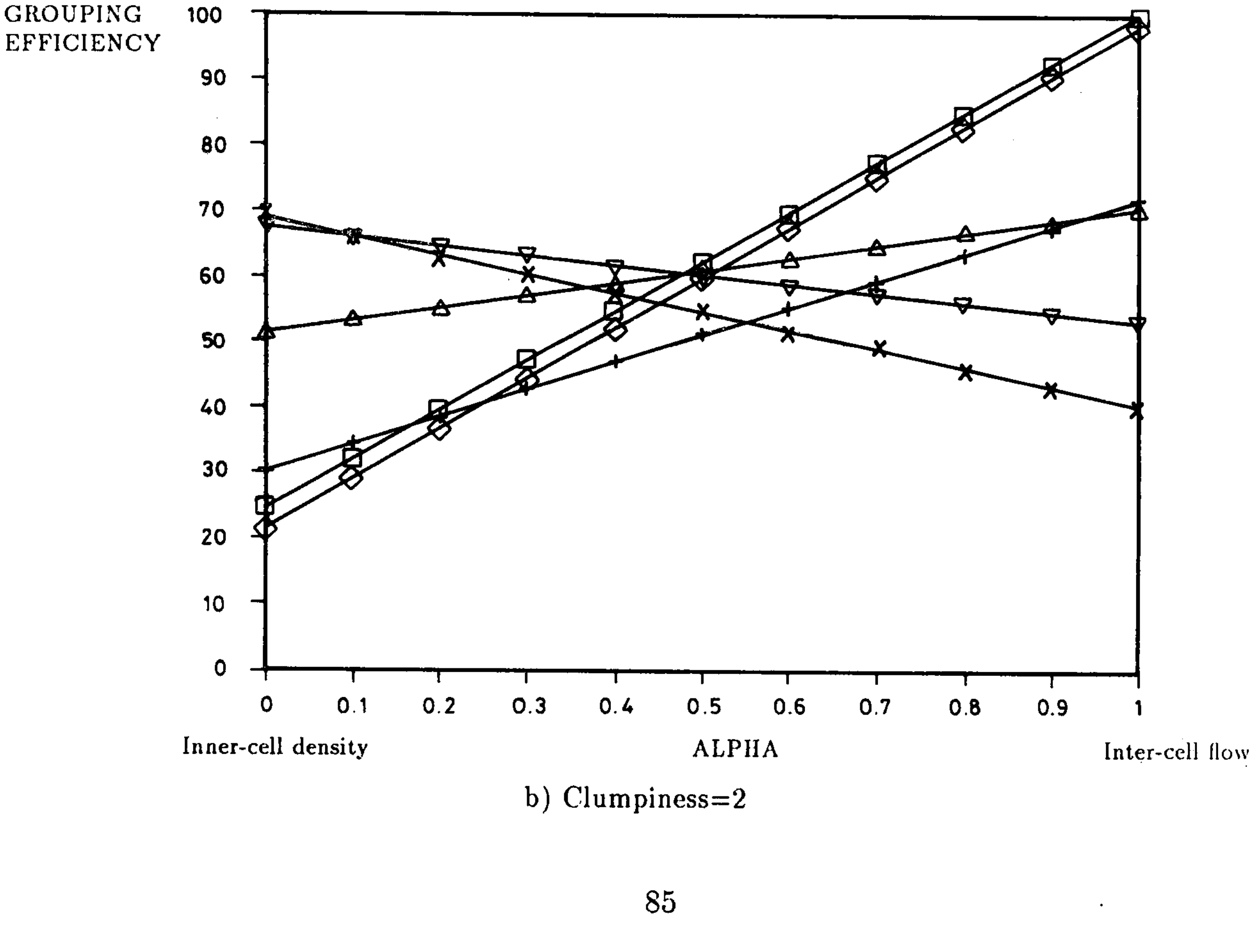
The shape parameters of the test problems significantly affect all of the performance measures for all of the selected PF/MG-F techniques. The effect of clumpiness usually dominates the effect of density in terms of the grouping efficiency. The main clumpiness effect on the grouping efficiency measure is illustrated in Figure 5.2. The two-way effects of the shape parameters on the efficiency measures are included in Appendix B. The effect of clumpiness usually dominates the effect of density. In addition, the main effects of both of the shape parameters on the work-load balance and under-utilization values are presented in Figures 5.3 and 5.4, respectively.

Since COMBGR generates PF/MG-F solutions with no exceptional elements, the associated inter-cell flow efficiencies take a value of 100. Because COMBGR tends to generate a small number of large sized cells each of which has the job shop characteristics, the inner-cell density measures are low. Hence, the grouping efficiency graphs based on different aspiration levels between zero and one is increasing. The inner-cell density measure improves as both clumpiness and density are increased. For instance, an average inner-cell density of 17 is obtained for a density of 0.10 and a clumpiness of 1, whereas it is about 56 when the density is 0.20 and the clumpiness is 9.

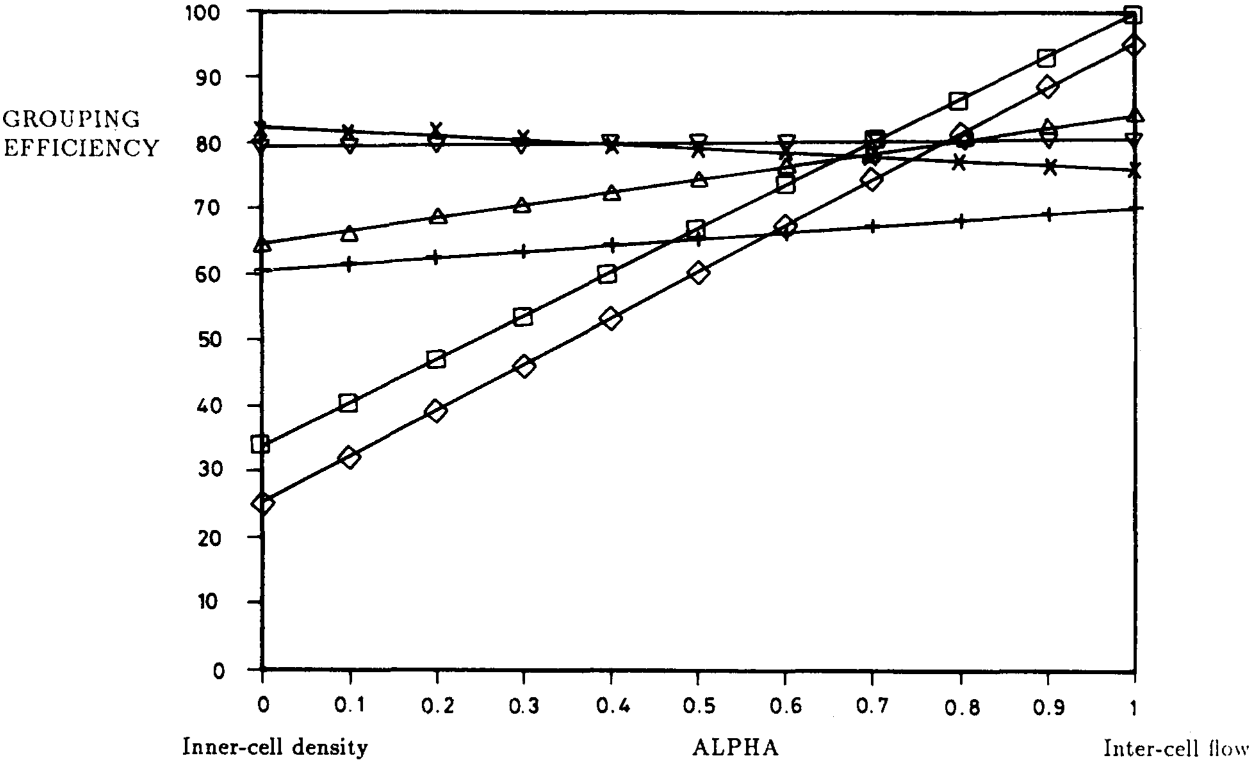


a) Clumpiness=1

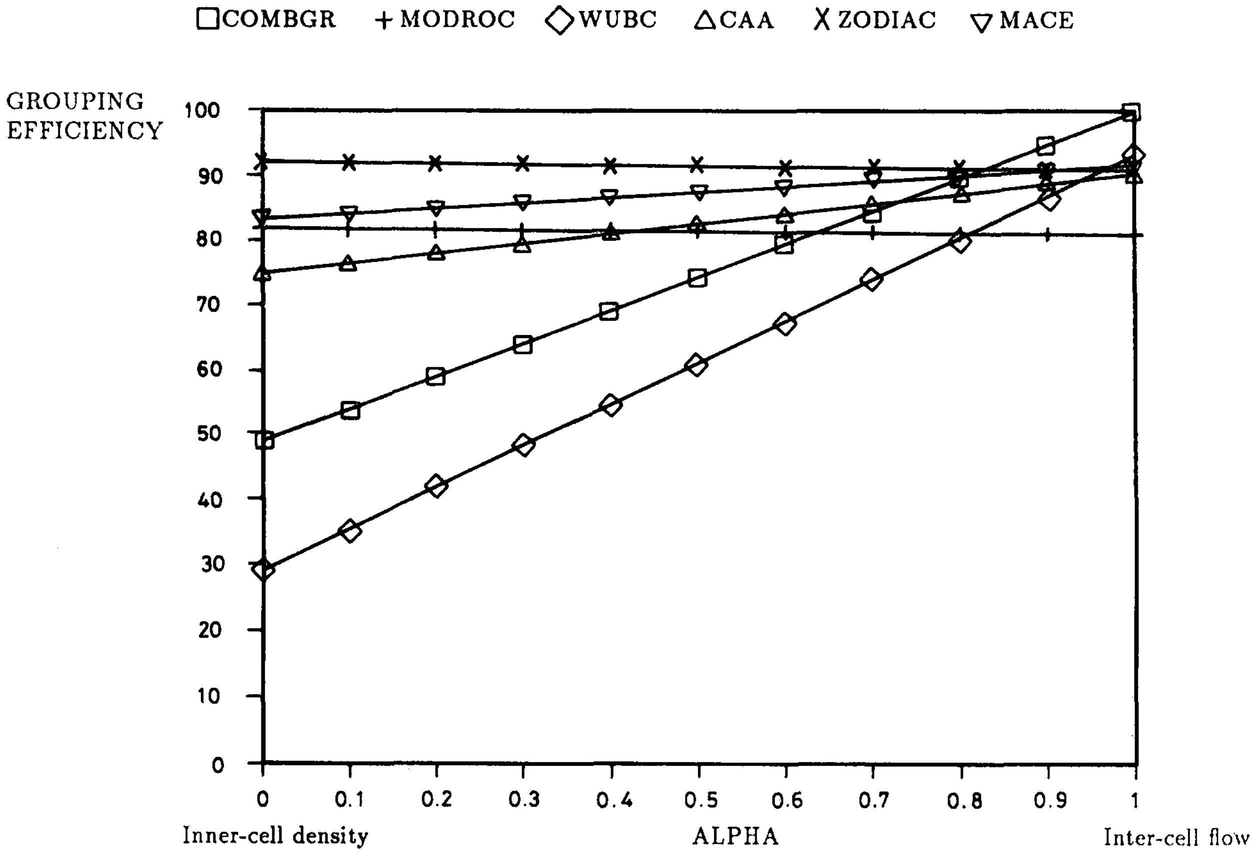
□COMBGR +MODROC ◇WUBC △CAA XZODIAC ▽MACE



b) Clumpiness=2

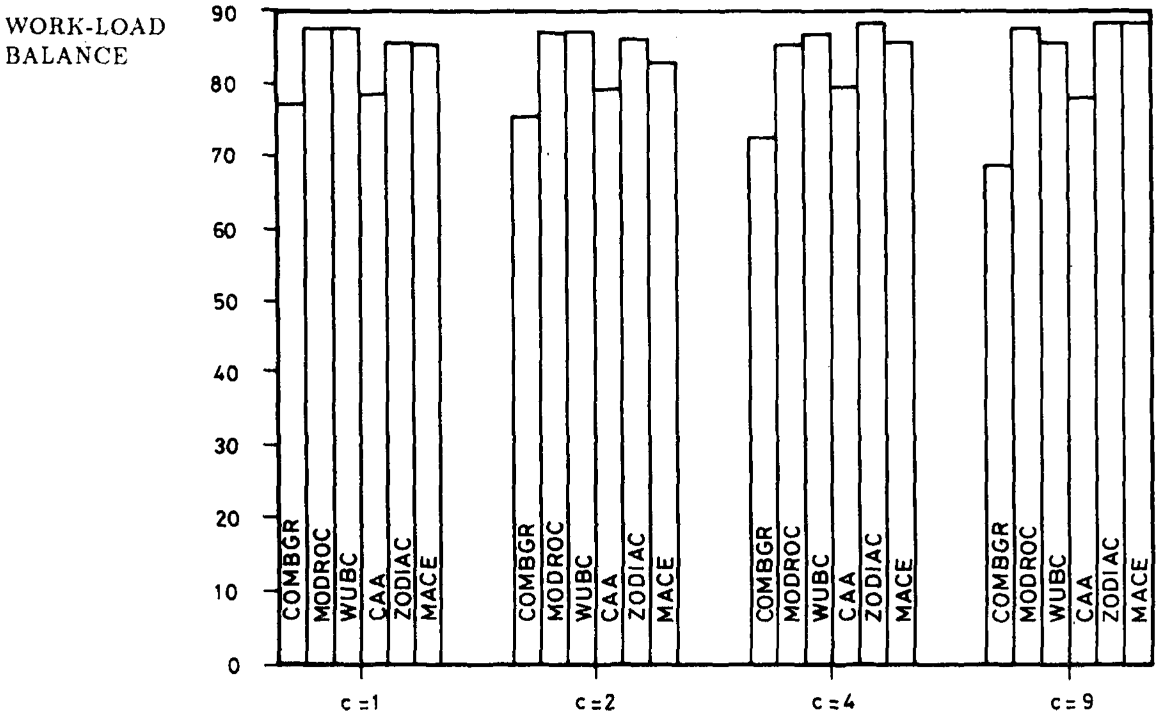


c) Clumpiness=4

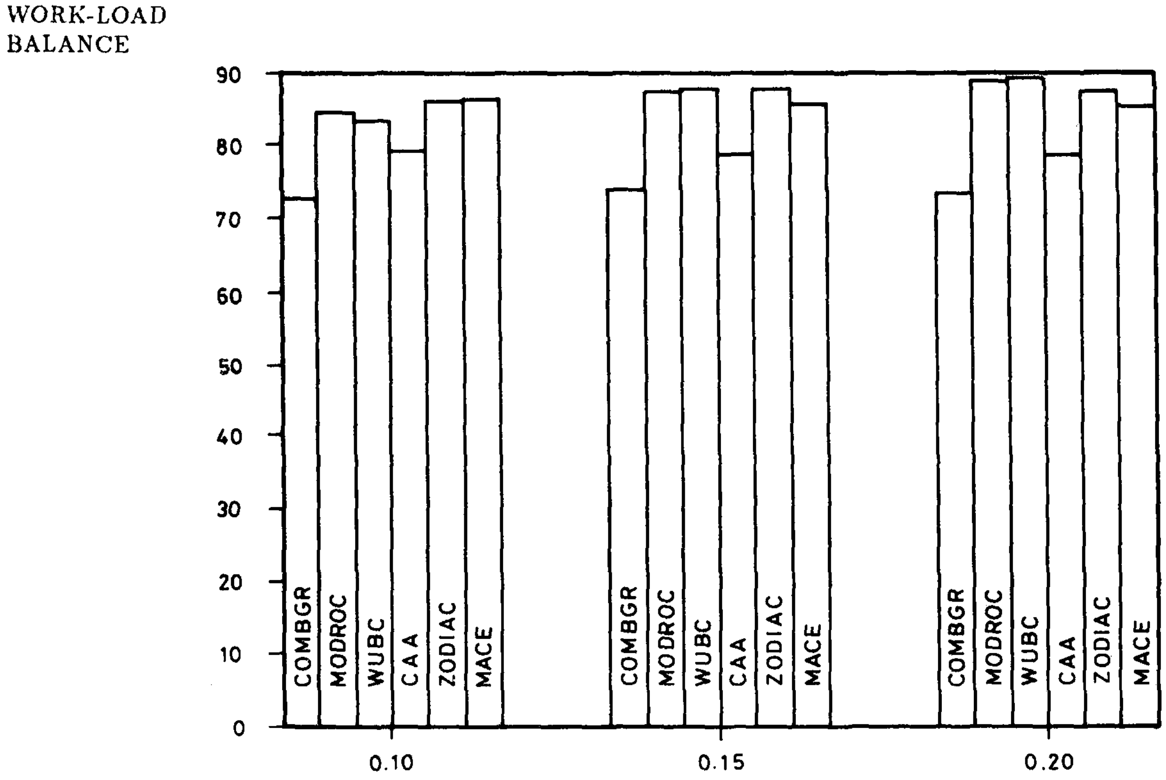


d) Clumpiness=9

Figure 5.2: Main clumpiness effects on grouping efficiencies.

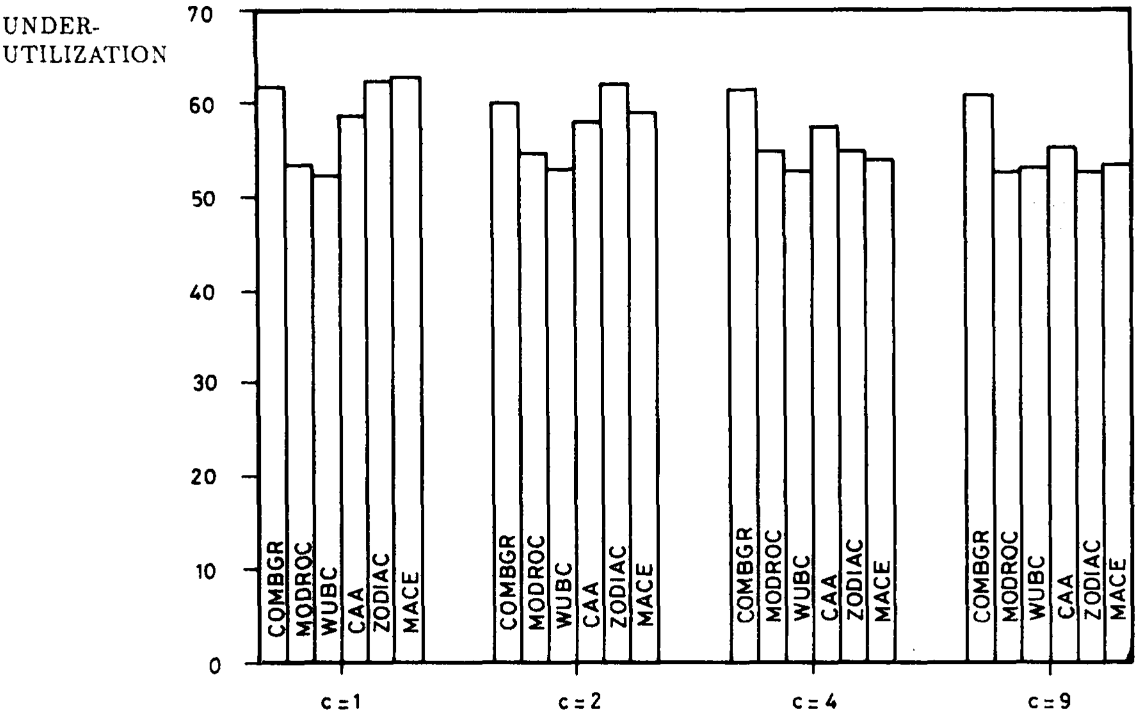


a) Main clumpiness effect

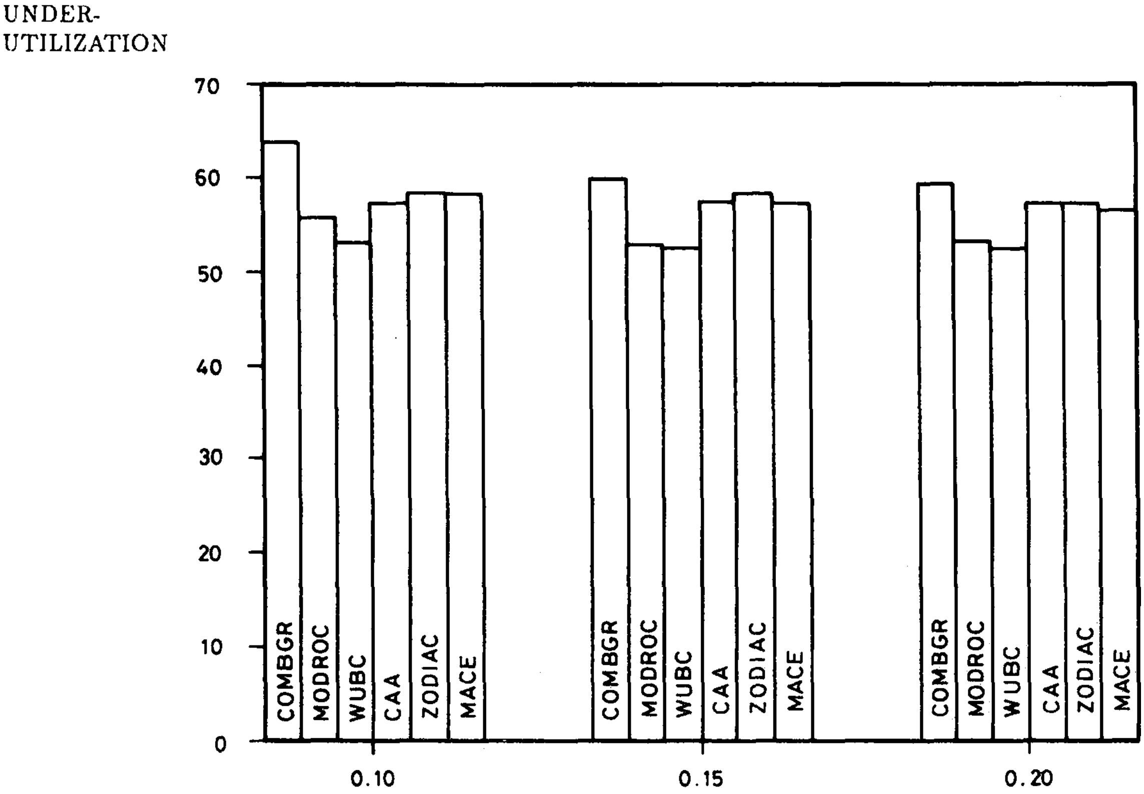


b) Main density effect

Figure 5.3: Main effects of the shape parameters on work-load balances.



a) Main clumpiness effect



b) Main density effect

Figure 5.4: Main effects of the shape parameters on under-utilizations.

The work-load balance values of the COMBGR's solutions are insensitive to different levels of the density factor. However, the work-load balance values decrease with an increase in the clumpiness value. The under-utilization measures of the PF/MG-F solutions generated by COMBGR seem to be independent of the clumpiness values. COMBGR generates better solutions in terms of under-utilization for problems with higher densities rather than the ones with lower densities. Nevertheless, as compared to other techniques COMBGR usually generates inferior PF/MG-F solutions relative to the under-utilization measure. The reason behind this is that COMBGR duplicates machines in order to generate solutions with no exceptional elements. This reason is also valid for the overall low work-load balance values.

Results show that with MODROC the PF/MG-F solutions improve with respect to inter-cell flow efficiencies as both the clumpiness and density values are increased. However, no general conclusion can be made about the effect of the density factor on the inner-cell densities. If the clumpiness value is set to its lower levels, the corresponding inner-cell densities are first decreased slightly by a change in the density factor from 0.10 to 0.15, and increased again slightly as a density value of 0.20 is used. When clumpiness is set to 4, the inner-cell efficiencies remain constant for the first two density levels, then they decrease by a factor of 10 for the highest density value. If clumpiness is set to 9, the inner-cell efficiencies first increase, then they remain constant with respect to increases in the density levels. But the changes are not significant. It can be stated that for a fixed clumpiness value the shape of the grouping efficiency graph does not change significantly with respect to different levels of density. The dominant factor that affects the shape of the grouping efficiency curves for MODROC is clumpiness. Inner-cell density measures increase considerably as near ideal CM systems on the manufacturing spectrum are considered. Slopes of the grouping efficiency curves decrease monotonically with respect to successive increases in the clumpiness values.

The PF/MG-F solutions generated by MODROC show only a slight increase in the work-load balance values as the density factor is increased. The work-load balance measures of MODROC's solutions are not significantly affected by the different levels of the clumpiness factor. The under-utilization values decrease first, then they start to increase as the density factor is increased from 0.10 to 0.20. The situation is reversed in the case of the clumpiness factor. The under-utilization values increase for the first two levels of the clumpiness factor. Thereafter, the under-utilizations decrease at a clumpiness of 9. These changes in the under-utilization values are not very significant.

The final PF/MG-F solutions of MODROC depend highly on the first two ROC iterations performed consecutively. These ROC iterations in turn, are affected by the existence of possible exceptional elements causing a decrease in the size of the block of 'ones' obtained at the top-left corner of the incidence matrix. Hence, MODROC creates inferior PF/MG-F solutions for job shop like environments. The quality of PF/MG-F solutions in terms of the grouping efficiencies improves in near ideal CM systems indicated by higher clumpiness values. Another important observation is the effect of the aspiration level indicator in selecting the final PF/MG-F solution among the hierarchical alternatives generated by MODROC. MODROC provides multiple PF/MG-F solutions for different values of the convex combination parameter. Since the grouping efficiency curves turned out to be linear, the overall effect of the convex combination parameter is insignificant. This insensitivity shows that either there is a dominating PF/MG-F solution in terms of the grouping efficiency no matter what value is set for the convex combination parameter, or the solutions generated with different convex combination values do not deviate much from each other with respect to the grouping efficiencies. However, the effect of the aspiration level indicator may be the reason of the nonhomogenity in some of the efficiency measures like in the inner-cell densities.

WUBC behaves like COMBGR in terms of grouping efficiencies. However, WUBC allows the existence of exceptional elements. The PF/MG-F solutions generated by COMBGR and WUBC have almost the same grouping efficiency values at different levels of the density factor for lower clumpiness values. As the clumpiness value is increased, the WUBC solutions with respect to the grouping efficiency measure are significantly inferior to the COMBGR solutions irrespective of the density factor. In all cases, the COMBGR's results are superior to that of WUBC's in terms of the grouping efficiency. Moreover, the gap between COMBGR and WUBC increases considerably as near ideal CM shops are considered. The WUBC technique generates solutions with high work-load balance values. The work-load balance values increase as density levels increase, whereas the effect of the clumpiness factor is negligible. The under-utilization values of the solutions generated by the WUBC technique are independent to both of the shape parameters.

The main focus of WUBC is on the under-utilization measure. WUBC allows exceptional elements in order to decrease the number of machine duplications. This increases machine utilizations and decreases the inter-cell flow efficiencies as compared to COMBGR. On the other hand, WUBC creates large-sized cells to prevent the existence of exceptional elements. Thus, the sizes of the manufacturing cells suggested by WUBC are usually larger than what COMBGR proposes. This causes the associated inner-cell densities to decrease. Since WUBC groups sufficiently loaded machines together to form a manufacturing cell, the corresponding work-load deviations between the machines clustered in a cell are small. This is the main reason why relatively higher work-load balance measures are achieved by WUBC.

CAA generated PF/MG-F solutions are insensitive to the density factor. However, as the value of the clumpiness factor is increased, the grouping efficiency values of the CAA's solutions improve. The grouping efficiency graph as a function of the convex combination parameter has a positive small slope which is almost invariant to changes in the clumpiness factor. The upward shifts of the grouping efficiency graphs occur as near ideal CM shops on the manufacturing spectrum are considered. Both the work-load balance and the under-utilization values from the CAA's PF/MG-F solutions show no change with respect to both of the shape parameters.

The grouping efficiency values of the PF/MG-F solutions generated by ZODIAC are quite sensitive to the density factor for small clumpiness values. The solutions to the test problems generated with a clumpiness of 1 and a density of 0.10 have a mean grouping efficiency behavior similar to that of COMBGR and WUBC. In this case, the grouping efficiency curve is linearly increasing with a positive slope of 70. For the same clumpiness value and the density factor set at a level of 0.15, the mean grouping efficiency curve decreases with a small slope. A further increase in the density to 0.20 makes the associated graph steeper with a negative slope of 65. A similar phenomenon is also observed for a clumpiness value of 2. The mean grouping efficiency graph linearly increases with a small slope when the density is 0.10. Density values of 0.15 and 0.20 result in decreasing grouping efficiency curves. The slope in these cases is steeper, having an approximate value of 48. For a clumpiness of 4, the graph is decreasing with decreasing slopes of 15, 4, and 3 with respect to the density values of 0.10, 0.15 and 0.20. The density factor becomes insensitive to the grouping efficiency results when the clumpiness value is 9. The associated graph is decreasing with a very small slope. This also indicates that the grouping efficiency values are insensitive to aspiration levels. The work-load balance values increase first, then remain constant when the density factor is increased from 0.10 to 0.15, then to 0.20, respectively. The mean work-load balance values increase as the clumpiness factor is increased. In general the changes in the work-load balance measure is insignificant. The under-utilization measure decreases slightly with an increase in the density. However, the amount of decrease is more significant for variations in the clumpiness factor.

The quality of the PF/MG-F solutions generated by ZODIAC depends totally on the choice of the final ideal seeds. The main purpose of the ZODIAC technique as a seed clustering algorithm is to find out the best ideal seeds. For job shop like environments, the possibility of a perfect block-diagonal structure after a series of column and row permutations is small. Since ZODIAC is designed to obtain a perfect block-diagonal structure through ideal seeds, the technique implicitly assumes the existence of such a structure. This is the main reason why inferior efficiency measures are obtained at low clumpiness values. At higher clumpiness values, the ZODIAC procedure gives satisfactorily good solutions because of the existence of near perfect block-diagonal structured problems.

The mean grouping efficiency curve of MACE decreases steeply for a clumpiness factor of 1. When the clumpiness parameter is set to 2, the grouping efficiency plot still decreases but with a smaller slope. The mean grouping efficiency values for small clumpiness values are independent of the changes in the density factor. The grouping efficiency graph increases with small slopes as the near ideal CM environment is approached. As the clumpiness parameter is increased the inner-cell densities first decrease, then increase. On the other hand, the inter-cell flow efficiencies increase monotonically as the clumpiness factor is increased. For a clumpiness value of 4, the inter-cell flow efficiency values increase as density is increased from 0.10 to 0.15, and then decrease with a further increase in density to 0.20. For a clumpiness value of 9, the inner-cell efficiency values decrease as density is increased from 0.10 to 0.15, and then remain unchanged as density is further increased to 0.20. The work-load balance measures initially decrease slightly, then increase as the clumpiness factor is increased. The under-utilization values decrease monotonically with an increase in the clumpiness factor. The effect of the density factor on both of the work-load balance and under-utilization measures is insignificant.

MACE creates a large number of small sized cells with high inner densities and large number of exceptional elements in job shop like environments. In job shop like environments, characterized by small clumpiness values, there are usually a small number of machines that are close to a specific machine in terms of similarity coefficients. MACE groups these small number of machines in a candidate cell. Another reason for the creation of a high number of small sized candidate cells is the use of threshold values. Each candidate cell consists of machine types that have similarity coefficients higher than the threshold value. In the second stage, only a few number of candidate cells are merged into a new larger cell because the similarity coefficients between the candidate cells are usually smaller than the threshold value. In job shop like environments, this results in a high number of exceptional elements and leads to low inter-cell flow efficiencies and high under-utilization values. As near ideal CM systems are approached, the number of cells generated decreases but the sizes of the candidate cells increase. This leads to decreases in the number of exceptional elements, inner-cell densities, work-load balances, and under-utilization values, and the grouping efficiency curve becomes flatter. Another reason for having better efficiencies for high clumpiness values is the reduced number of blocking machines demanding reallocations.

5.4 Comparison

The best PF/MG-F technique(s) suggested for each combination of the shape parameters in terms of the grouping efficiency measure are presented in Table 5.5. Similarly, the best PF/MG-F technique(s) according to the work-load

CLUMP. (c)	DENS. (d)	GROUPING EFFICIENCY = f(α)										
		0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
1	0.10	F	F	F	F	F	F	F	A,C	A,C	A,C	A,C
	0.15	F	F	F	F	F	F	A,C	A,C	A,C	A,C	A,C
	0.20	F	F	F	F	F	F	A,C	A,C	A,C	A,C	A,C
2	0.10	F	F	F	F	F	F	A	A	A	A	A
	0.15	E	E	E	E,D	F,D	A,D	A,C	A,C	A,C	A,C	A,C
	0.20	E	E	E	E	E,F,D	A	A	A	A	A	A
4	0.10	E	E	E,F	F,E	F,E	F	F	A,F	A	A	A
	0.15	E,F	F,E	F,E	F	F	F	F	F	F	A,F	A
	0.20	E	E	E	E	E	E	E	A,E,D	A	A	A
9	0.10	E	E	E	E,F	E,F	E,F	F,E	F,E	F,E	A,F	A
	0.15	E	E	E	E	E	E	E	E	E,F,A	A	A
	0.20	E	E	E	E	E	E	E	E	E,A	A	A

A: COMBGR B: MODROC C: WUBC D: CAA E: ZODIAC F: MACE

Table 5.5: Superior PF/MG-F technique(s) among the selected ones in terms of grouping efficiency.

balance measure and the under-utilization values are given in Table 5.6.

Related to the grouping efficiency, MACE seems to be the best PF/MG-F technique for near job shop like manufacturing environments if the inner-cell densities are considered to be more critical than the inter-cell flow efficiencies. COMBGR and WUBC are the best in similar environments when inter-cell flows gain importance. ZODIAC and MACE perform best as clumpiness is increased to 2 and inner-cell densities have preference over inter-cell flows. On the other hand, if the effect of exceptional elements gain importance, COMBGR and WUBC are the suggested techniques. In this case, COMBGR generates better PF/MG-F solutions in terms of the grouping efficiencies. As the clumpiness value is increased further, ZODIAC starts to dominate the other selected PF/MG-F techniques, especially at high densities. Nevertheless, COMBGR is still the best alternative in cases where the the existence of exceptional elements are extremely undesirable. If the two extremes are in balance, or equivalently the inner-cell densities and the inter-cell flows are equally important, ZODIAC and MACE should be preferred. Although the value of the convex combination parameter indicates the aspiration level of clumpiness factor is increased. For a clumpiness value of 4, the inter-cell flow efficiency values increase as density is increased from 0.10 to 0.15, and then decrease with a further increase in density to 0.20. For a clumpiness value of 9, the inner-cell efficiency values decrease as density is increased from 0.10 to 0.15, and then remain unchanged as density is further increased to 0.20. The work-load balance measures initially decrease slightly, then increase as the clumpiness factor is increased. The under-utilization values decrease monotonically with an increase in the clumpiness factor. The effect of the density factor on both of the work-load balance and under-utilization measures is insignificant.

MACE creates a large number of small sized cells with high inner densities and large number of exceptional elements in job shop like environments. In job shop like environments, characterized by small clumpiness values, there are usually a small number of machines that are close to a specific machine in terms of similarity coefficients. MACE groups these small number of machines in a candidate cell. Another reason for the creation of a high number of small sized candidate cells is the use of threshold values. Each candidate cell consists of machine types that have similarity coefficients higher than the threshold value. In the second stage, only a few number of candidate cells are merged into a new larger cell because the similarity coefficients between the candidate cells are usually smaller than the threshold value. In job shop like environments, this results in a high number of exceptional elements and leads to low inter-cell flow efficiencies and high under-utilization values. As near ideal CM systems are approached, the number of cells generated decreases but the sizes of the candidate cells increase. This leads to decreases in the number of exceptional elements, inner-cell densities, work-load balances, and under-utilization values, and the grouping efficiency curve becomes flatter. Another reason for having better efficiencies for high clumpiness values is the reduced number of blocking machines demanding reallocations.

5.4 Comparison

The best PF/MG-F technique(s) suggested for each combination of the shape parameters in terms of the grouping efficiency measure are presented in Table 5.5. Similarly, the best PF/MG-F technique(s) according to the work-load

CLUMP. (c)	DENS. (d)	WORK-LOAD BALANCE	UNDER UTILIZATION
1	0.10	F	C
	0.15	F,B,C	C
	0.20	C,B	C,B
2	0.10	E	C
	0.15	C,B	B,C
	0.20	C,B	C
4	0.10	E	C
	0.15	E,C	C,E,B
	0.20	B,C	C,E
9	0.10	E,F,B	B
	0.15	F,B,E,C	E,C,B
	0.20	E,F	C,E,B

A: COMBGR B: MODROC C: WUBC
D: CAA E: ZODIAC F: MACE

Table 5.6: Superior PF/MG-F technique(s) among the selected ones in terms of work-load balance and under-utilization measures.

the decision maker, usually moderate values are suggested. Hence, MACE, COMBGR, and WUBC are the best PF/MG-F techniques for near job shop systems whereas ZODIAC and MACE are the best ones for CM type systems at the other end of the manufacturing spectrum.

The PF/MG-F solutions generated by COMBGR and CAA are inferior in terms of the work-load balances. In job shop like environments, MACE, WUBC, and MODROC give better PF/MG-F solutions with respect to the work-load balance measure. As near ideal CM systems are considered, ZODIAC is added to the list of best PF/MG-F techniques.

The absolutely best PF/MG-F technique in terms of the under-utilization results is WUBC. In all cases considered, the technique that generate the best PF/MG-F solutions with respect to the equipment utilization criterion is unquestionably WUBC. In addition to WUBC, MODROC and ZODIAC techniques generate satisfactorily good PF/MG-F solutions with respect to the under-utilization measure in some cases.

6. SUMMARY, CONCLUSIONS, SUGGESTIONS

6.1 Summary

The first and the most important phase in the design of Cellular Manufacturing systems is the Part Family Machine Group Formation (PF/MG-F) problem. The PF/MG-F problem is concerned with the placement of functionally dissimilar machine groups together and the subsequent or simultaneous dedication of these machine groups to the manufacture of a specific range of parts. The PF/MG-F problem belongs to the $\mathcal{NP}$-Complete class. Hence, a large number of heuristics has been designed to handle the PF/MG-F problem. A taxonomy of the PF/MG-F techniques is presented and these techniques are reviewed in this framework.

The PF/MG-F problem is described by means of graph theoretical terms. It can be defined as permuting columns and rows of the machine-part incidence matrix so that a block-diagonal structure that identifies the cells is obtained. Equivalently, the PF/MG-F problem involves the reordering of the incidence matrix so that a minimum number of exceptional elements are obtained, provided that a block-diagonal structure exists. The PF/MG-F problem is extended by the utilization of work-load matrices. The improvement due to the use of the work-load matrix instead of the incidence matrix is explained.

Three efficiency indices are suggested for the evaluating the PF/MG-F results. For a specific PF/MG-F solution, all of the efficiency indices are computed from the final form of the corresponding work-load matrix. The first measure is the grouping efficiency which measures the extent of exceptional elements and inner-cell densities. The second measure is related to inner-cell work-load balances. The last index measures under-utilizations of individual machines.

Six analytical techniques designed to solve the PF/MG-F problem are selected for mutual comparison. These techniques are lattice theoretic combinatorial grouping (COMBGR), modified rank order clustering (MODROC), machine-component cell formation (MACE), within-cell utilization based clustering (WUBC), cost analysis algorithm (CAA), and zero-one data – ideal-seed clustering (ZODIAC). These techniques are analyzed in detail. Various modifications and extensions that improve the performance of each technique are suggested.

A PF/MG-F problem generator is developed to evaluate and compare the designed techniques. The selected techniques are evaluated by means of randomly generated test problems under different scenarios. The mean effects of the number of parts, shop densities, demand and depreciation cost variations, and manufacturing environments on the PF/MG-F problems for each selected technique are investigated. The selected PF/MG-F techniques are compared in terms of the mean values of the proposed efficiency measures.

6.2 Conclusions

Based on the results obtained from the experiment, the performance evaluations of the six selected PF/MG-F techniques lead to the following conclusions:

1. According to the mean results obtained from the test problems, all of the efficiency values of the solutions generated by all of the selected techniques are found to be insensitive to annual demand and depreciation cost variations. Moreover, the effect of changing the number of parts is also negligible. Based on the limited number of test problems, the shop density and the manufacturing environment indicator seem to be the most sensitive factors. However, the effect of the manufacturing environment indicator usually dominates the effect of the shop density.

2. COMBGR yields the best PF/MG-F solutions with respect to grouping efficiencies when the number of exceptional elements in the final work-load matrix is extremely important. COMBGR is a quite efficient PF/MG-F technique especially in job shop like manufacturing environments. COMBGR duplicates machines to eliminate exceptional elements and this leads to relatively inferior load balance and under-utilization measures.

3. The PF/MG-F solutions generated by MODROC achieve good work-load balance and under-utilization values. However, the performance of MODROC with respect to the grouping efficiency measure is relatively poor.

4. MACE generates the best PF/MG-F solutions when the inner-cell densities gain importance. This technique behaves much better in job shop like manufacturing environments than the other selected techniques. PF/MG-F solutions generated by MACE also lead to satisfactory work-load balance values. On the other hand, the under-utilization measures are relatively inferior in job shop like manufacturing environments.

5. Although WUBC proposes PF/MG-F solutions with relatively low under-utilization values, its performance with respect to the grouping efficiency measure is inferior. The work-load balance performance of the solutions generated by WUBC are considerably good.

6. The solutions created by CAA are relatively inferior in terms of all of the suggested efficiency indices with respect to the best solutions generated by the other techniques.

7. ZODIAC yields good PF/MG-F solutions related to all efficiency measures. The PF/MG-F proposals generated by the ZODIAC technique are among the best ones when near ideal CM environments are considered.

8. The suggested efficiency measures are quite effective in evaluating the selected PF/MG-F techniques.

6.3 Suggestions For Further Research

Research on the PF/MG-F problem is far from complete. This research has concentrated on identifying the basic guidelines necessary for evaluating and selecting among different PF/MG-F techniques under different situations. By this study, the first comparative work has been performed in this area. Modifications and extensions on the selected techniques have been made to eliminate some of the deficiencies of the PF/MG-F procedures. The selected techniques have been implemented on the computing environment. This specially designed computer support is necessary to handle the complexities of industrial data in generating solutions adhering to specific criteria. However, the definition of the PF/MG-F problem, the efficiency measures suggested to evaluate the PF/MG-F proposals, and the problem generator module require more in depth analysis and research than presented herein.

Some research areas for further investigation related to the PF/MG-F problem are itemized below:

1. *More powerful PF/MG-F techniques are required.* All of the reported techniques do not consider part routing alternatives. A technique taking routing alternatives into account can also serve the loading problem in Flexible Manufacturing Systems, or vice versa. Except WUBC, none of the PF/MG-F techniques employ work-load matrices that include the effect of annual part demands. Since almost all of the techniques focus on incidence matrices, they do not use the information on the total number of individual machines of each type. Moreover, majority of the existing techniques do not discriminate the machine types from each other. The drawback in utilizing the work-load matrices is the impossibility of representing the operation sequences. If a means of representing part operation sequences can be developed, inner-cell layouts and the resultant flow patterns could be analyzed quite easily. Hence new PF/MG-F techniques are needed to cope with alternative part routings for more flexibility. Also new techniques should consider the differences in parts and machine types using the work-load matrices together with depreciation cost information, and operation sequences

to analyze inner-cell layouts and flow patterns. Finally, these techniques should incorporate some performance criteria within themselves but not on an evaluation basis only. In this respect, Mathematical Programming based techniques seem to be suitable. The emphasis in the design of such techniques should be on both combinatorial effort and performance guarantees.

2. *Product lines can be used as a basis for PF/MG-F.* Parts and products belonging to the same product line and the machines related to these parts could be combined as a new approach. Clearly, this is not a pure Group Technology solution since the resultant system leads to the separation of similar parts in different product lines during manufacturing instead of clustering them into a cell. However, this approach generates 'focused factory' solutions that can facilitate the construction of closely linked production systems to realize the benefits of Just-In-Time philosophy. As indicated by Wemmerlöv and Hyer [44], it would be of great interest to study the relative advantages/disadvantages of such cell systems.

3. *More efficiency measures are needed.* In this research, three efficiency indices were introduced to evaluate the performance of the PF/MG-F solutions. Another index to measure the flexibility of the PF/MG-F solutions on a quantitative scale should be developed. The effect of the limitations imposed by an input on the efficiency indices should be removed. For instance, the minimum under-utilization is always attained in job shop environments. So, the limiting under-utilization value for each PF/MG-F problem can be obtained from the original (generated) shop. Thereafter, the relative under-utilization of each PF/MG-F solution should be computed by comparing to this value. It is also worthwhile to study the aspiration levels of the decision makers that are required to compute the grouping efficiencies. The decision makers' preferences between inner-cell densities and inter-cell flows and their corresponding utility curves can be identified using decision analysis tools which would be a valuable research in creating an opportunity to applying artificial intelligence in the context of PF/MG-F.

4. *A standard PF/MG-F problem generator is required.* A random generator to create statistically independent PF/MG-F test problems for

each scenario has been proposed in this study. This prototype can be modified and extended to obtain a more common base for testing and comparing different PF/MG-F approaches. A standard problem generator in evaluating different PF/MG-F techniques similar to the NETGEN system for the network optimization problems could be developed.

5. *Detailed analyses of PF/MG-F solutions are necessary.* This can be achieved either by field studies or by simulation. The former case implicitly requires the experiences of industry users. Specific organizations that are in the process of transforming their manufacturing system from job shop to CM and big job shop systems could be analyzed to identify the conditions for applicability of CM systems. Also the findings of other researchers or practitioners can be collected and documented. This would be extremely useful for researchers that are geographically far from such industrial zones. Furthermore, the effects of PF/MG-F solutions on the operating CM environments can be investigated from a modeling and experimentation point of view. Computer simulation seems to be the most appropriate tool for such analysis.

APPENDIX A

In this appendix, an example PF/MG-F problem is generated and the steps of the selected techniques are illustrated on this example. All of the selected PF/MG-F techniques and the problem generator module are coded on Data General Eclipse MV/20000 Model 1 Computer system under AOS/VS operating system. The complete list of Fortran-77 codes of the random generator, COMBGR, MODROC, MACE, WUBC, CAA and ZODIAC, and the results obtained from 5760 runs can be obtained from:

Levent Kandiller
Department of Industrial Engineering
Bilkent University
P. O. B. 8, Maltepe
TR-06572 ANKARA
E-Mail: KANDIL@TRBILUN.bitnet

Generator

S-1. Number of machine types: 50,
Number of parts: 100 ,
Density: 0.10 ,
Clumpiness: 4,
Number of parts in each imaginary cell: U[4;19],
Number of machines in each imaginary cell: U[2;8],
Processing rate distribution: U[2;9] ,
Demand distribution: U[50;4050] ,
Depreciation cost distribution: U[100;100100] ,
Annual machine capacities : 120000 min.

S-2. Inner-cell density: 0.775,
Off-diagonal density: 0.025,
Incidence Matrix :

```
1111100000000000000000000000000000100000000000000000000000000000000000000000000000000100000000000000
1010100000000000000000000000000000000000001000000000000000000000000000001000000000000000000000100000
1111110000000000000100000000000100000000000000000000000000000001000000000000000000000000000000000000
0000001111110000000000100000000000000000000000000000000000000000000000000000000000000000000000000100
0000010111110000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0010001111110000000000000000100000000000000000000000000000000000010000000000000000000000000000000000
0000001011110000000000000000000000000000000000010000000000000000000000000010000000100100000000000000
0000001100010000000000000100000000000000100000001000000000000000000000000000000000000000000000000000
0000001101100000000000000000000000000000000000000000000000000001000000000000000000010000000000000000
0010001011110000000001000000000000000000000000000000000001000000000000000001000000000000000000000000
0000001101110000000000000000000000000000001000000000000000000000000000000000000000000000000000000000
0000000000011101000000000000000000000000000000000000000000000001000000000010000000000000000000000000
0000000000001101000000000000000000000000000000000000100000000000000000000000000000000000000000000000
0010000000000110000000000000000000000000000000001100000000000000000001000000100000010000000000010000
0000000000000111000000000000000000000000010000000000000000000000000000000010000000001000000000000000
0000000100000011110000000000000000000000000010000000000000000000100000010000100000000000100000000000
0000000000000100011011011111101111100000000000000000000000000000000000000000001000000000000000000000
0000000000000000011111111110011111000000000000100010100000000000000000000000000000000000000000000000
0000000000001000100001111010111001000000010000000000000000000000000000010000000000000000000000000000
0000000000000000110111011100101011000000000000000000000000000000000000000000000000000000000000000000
0000000000000000011111011111101111010000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000010101111111111100000000000000000000000000000000000000000000000000
0000010000000000000000000000000000111111111111000111000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000011101110000000000000000000000000000000000000000010
0000000000000100000000000000000000000000000000000010110000000000000000000000000000000000000000000000
0000100000000000000000000000000100000000000000000010011111010000000000000000000000000001000000000000
0000000000000000000000000000000000000100001000000000000000011111000000000000000000000000000000000000
0000000000000000000000000000000000000100001000000000000000011111111000000000000000000000100000000000
0000000000000000000010000010010000000000000000000000001000011111111000000001000000000000000000000000
0000000000000000000000000000000000000000000000000000000000011111111000100000000000000000000000000000
0000000000000000000000000010000000000100000000000000000000011010111000000000000000000000000000000010
0000000000000000000000000000000000000000000000000000000000011111011000000000010000000000000000000000
0000000000000000010000000000000000000000000000000000000000000000000011111111011110001000000010000000000
0000000000000000000000000000000000000000000001000000000000000000000011111111100110000000000000000000000
0000000000000000000100000000010100000000000001000000000000000000000011111101101100000000000100000000000
0000000000000000000000000000000000000000000000000000000000000000000011101001111110000000001000001010000
0000000001000000000000000000000000000000000000000000000000000000000011111110111110000000000000000000000
0000000001000000000000000000000000000000000100000000000000000000000011111110111111000000010000000000000
1000000100000000000000000000000010000000000000000000000000000000000100000110000100111101111101101111111
0000000000000000000000000000000000100000000000000100000000000000010000000000000011101010010011111111111
0000000000000000000000001000000000000000000000000000000000000000000000000000000101111111111111000011101
0000000000000000000000000001010000000000000000000000000000000000000000000000000101011110111111101101011
0000000100000000000000000000000000000000000000000000000000000000000000000000000111111011011101111101010
0000000000010000000000000000000000000000000000000000000000000000000000000010000001100101111110011001111
0000000000000000000000000100000000000110000000000000000000000000000000000000000101011110111011111111110
0000000010000000000001000000000000000000000000000000000000000000000000000010000101110111111111111110011
0000000000000000000000000000000000100000000000000000000000000000000000000000000110111010111100001111011
0010000000000000000000000000000000000000000000000000000000000000000000000000000110110111011101011111111
0010000000000000000000000000000000000000000000000000000000000000000100000000000101111111111101110101111
0001000000000010000000000000000000000000000000000000000000000000000000000000000101101110111011001111011
```

S-3. Part No : Machine sequence (Standard time[min.]),

1 : 1(4)–39(6)– 2(4)– 3(16)
2 : 3(5)– 1(1)
3 : 48(2)– 1(8)–49(2)– 6(53)–14(8)– 3(8)–10(4)– 2(2)
4 : 50(4)– 1(2)– 3(16)
5 : 3(3)– 2(1)–26(2)– 1(4)
6 : 23(2)– 5(2)– 3(16)
7 : 11(5)–16(2)– 7(2)– 9(2)– 7(19)– 8(2)– 6(39)–10(16)– 4(5)
8 : 39(1)– 6(2)–11(5)– 5(3)– 8(1)– 4(9)–43(2)–11(3)–43(2)– 9(3)
9 : 5(2)– 7(3)– 4(6)–10(11)– 5(6)–46(2)– 5(2)– 6(30)
10 : 6(10)–11(4)–38(2)– 5(5)–38(3)–37(15)–10(5)– 7(3)– 4(19)– 9(1)
11 : 6(1)– 9(6)–11(22)–10(3)– 5(1)– 7(7)– 4(1)–10(8)
12 : 10(1)– 8(8)–11(1)– 4(2)–17(1)–12(3)– 6(4)–44(2)
13 : 12(2)–19(5)–12(14)–13(2)–16(1)
14 : 13(7)–16(10)–15(5)–14(16)–25(28)–12(12)
15 : 50(3)–15(1)–14(9)–16(10)
16 : 16(11)–15(2)–12(10)–17(5)–13(4)–15(1)
17 : 18(8)–19(4)–17(3)–20(1)–18(6)
18 : 33(1)– 3(6)–18(1)–21(3)–20(9)
19 : 17(5)–18(4)–17(4)–21(4)
20 : 18(1)–35(6)–21(8)–17(1)–20(2)–17(1)
21 : 29(3)–20(1)–18(1)–21(15)
22 : 10(1)–18(2)–17(5)–19(1)–10(2)–21(11)–10(4)–46(6)–20(7)
23 : 4(16)–18(3)–21(1)–17(5)–19(1)
24 : 20(3)–17(2)–18(9)–19(7)–17(1)
25 : 18(2)–21(1)–20(15)–41(22)–19(42)–20(9)–17(1)
26 : 31(1)–45(3)–20(3)–17(2)–21(15)
27 : 30(1)–21(9)–29(2)– 8(2)–19(4)– 8(3)
28 : 18(9)–42(10)– 6(4)–21(2)–17(1)
29 : 19(9)– 3(1)–20(6)–21(1)–19(16)–18(5)–17(3)
30 : 29(26)–35(2)–42(11)–17(1)–19(20)–18(9)
31 : 18(1)–19(5)–39(23)–20(3)–21(4)–17(4)
32 : 17(8)–26(3)–21(3)–35(9)–18(2)
33 : 17(4)–18(2)–21(7)–17(3)–20(11)
34 : 47(6)– 1(5)–20(1)– 1(3)–19(1)–17(4)–21(3)–18(1)–40(2)–19(2)
35 : 23(3)
36 : 31(2)–21(4)–23(3)–28(8)–22(11)
37 : 27(14)–45(13)–23(5)
38 : 23(13)–45(12)–22(6)–45(4)
39 : 23(4)
40 : 23(1)–22(12)–15(12)
41 : 8(12)–28(9)–23(5)–22(5)–27(1)–23(3)
42 : 23(3)–22(1)
43 : 23(2)– 2(2)–19(4)–11(18)–22(5)
44 : 38(2)–23(1)–22(5)
45 : 16(2)–22(10)
46 : 35(4)–22(10)
47 : 18(1)–34(3)–22(1)
48 : 22(3)–23(7)–22(1)– 7(11)–26(1)– 8(4)
49 : 14(24)–23(2)–22(11)–40(6)
50 : 14(7)–13(2)–23(3)–22(4)
51 : 18(2)–24(2)–25(1)–26(5)–24(2)
52 : 24(1)–26(4)
53 : 24(11)–25(12)–27(3)–18(1)–26(12)
54 : 29(14)–25(18)–29(17)–27(2)–26(1)

55 : 27(7)-24(1)
56 : 24(1)-26(4)-27(8)
57 : 24(1)
58 : 32(9)- 3(1)-31(8)-28(3)-30(4)
59 : 31(10)-32(29)-31(1)-29(2)-30(6)-28(11)
60 : 10(10)-28(4)-32(4)-29(32)-30(1)
61 : 32(13)-31(2)-29(15)-30(7)-29(6)-28(2)
62 : 12(8)-29(5)-28(6)-30(1)- 9(2)
63 : 30(3)-31(5)-29(19)-40(6)-28(2)-32(7)
64 : 31(3)-28(7)-32(1)-29(6)-30(6)
65 : 16(2)-29(2)-31(5)-28(2)-32(17)-49(6)-30(2)-39(3)-16(5)
66 : 35(1)-34(1)-38(1)- 6(5)-37(1)-33(17)-35(1)-34(8)-36(4)
67 : 33(19)-37(1)-35(4)-36(12)-38(2)-34(1)
68 : 37(14)-19(3)-38(6)-35(2)-33(7)-19(2)-34(10)-14(3)
69 : 2(28)-37(33)-36(2)-33(3)-34(4)-38(1)-36(4)-34(3)-30(2)-35(1)
70 : 33(2)-35(1)-38(1)-37(11)-34(1)
71 : 44(6)-37(10)-39(4)-33(4)-34(5)-38(2)-34(1)
72 : 36(6)-33(9)-46(4)-34(4)-16(12)-39(1)-35(6)-34(5)-36(2)
73 : 7(2)-35(1)-12(4)-36(1)-37(1)-15(1)-12(6)-38(1)
74 : 36(5)-29(30)-33(3)-37(11)-29(6)-38(1)
75 : 37(11)-35(1)-34(2)-37(16)-36(16)-38(4)-14(3)-33(1)
76 : 37(5)-34(2)-35(20)-36(3)-33(2)-38(2)-32(4)-37(1)-32(12)
77 : 16(3)-49(8)-17(7)-16(3)-39(10)-17(5)-43(2)-17(10)-44(2)-47(12)-50(3)-48(3)-46(5)
78 : 42(1)-45(3)-43(1)-44(18)-43(8)-47(4)-40(1)-48(2)-41(6)-10(3)
79 : 46(2)-43(1)-46(2)-50(2)-49(12)-40(4)
80 : 39(9)-33(8)-46(1)-41(7)-43(3)-49(20)-48(1)-50(5)-47(1)-46(2)-40(1)-48(2)-42(30)-49(1)-45(28)-42(7)
81 : 49(21)-41(16)-48(6)- 7(6)-47(16)-46(3)-39(12)-43(1)-44(6)-43(2)
82 : 41(4)-39(3)-42(15)-50(1)-47(6)-14(6)-39(5)-47(3)-41(13)-45(3)-40(1)-49(5)- 9(4)
83 : 41(11)-46(1)-49(21)-44(3)-39(1)-42(2)-15(4)-48(3)-43(5)-45(1)-39(3)-50(1)
84 : 46(1)-47(4)-48(1)-46(2)-40(6)-41(3)-49(2)-50(1)- 7(13)-42(3)-47(10)-44(5)-45(8)-43(3)
85 : 42(4)-46(7)-41(1)-44(1)-48(2)-49(6)-46(4)-41(1)-39(2)
86 : 44(3)-50(1)-41(4)-50(10)- 1(2)-39(6)-41(6)-49(3)-41(8)-45(3)-46(2)-43(1)-47(4)
87 : 39(6)-38(3)-50(3)-42(20)-46(4)-44(10)-45(3)-48(2)-47(9)-49(17)-40(2)-43(2)-41(7)
88 : 46(3)-50(8)-26(1)-42(17)-39(6)-43(1)-36(1)-44(3)-47(6)-45(3)-41(2)-42(2)-41(1)-49(3)-48(3)-46(3)-49(20)
89 : 42(6)-47(1)-39(5)-41(1)-48(4)-35(4)-42(22)-48(1)-28(1)-46(3)-16(1)
90 : 41(5)-42(17)-45(2)-40(6)-49(8)-42(17)-46(2)-43(1)-50(1)-46(1)
91 : 42(1)-44(17)-48(5)-45(6)-41(2)-43(1)-49(1)-40(2)-46(6)-39(4)-33(28)-50(2)
92 : 43(2)-40(3)-39(1)-49(6)-42(43)-44(5)-39(5)-45(5)-40(8)-46(1)
93 : 46(1)-48(2)-45(5)-43(12)-40(1)
94 : 36(1)-49(4)-47(3)-40(2)-50(2)-45(6)-46(5)-42(43)-39(18)-48(1)-14(13)-36(2)
95 : 45(12)-44(8)-43(2)-46(3)-42(11)-40(2)-50(3)-47(18)-39(1)- 2(4)-48(1)
96 : 46(1)-47(1)-50(2)-49(18)-48(2)-40(1)-44(3)-40(10)-39(2)-45(1)-48(7)-39(6)-36(2)-41(8)
97 : 50(1)-39(1)-45(2)-40(1)-43(3)-47(19)-49(10)-44(6)-41(12)-48(1)-42(8)
98 : 40(2)-45(1)-39(7)-44(6)-49(3)-41(10)-48(2)-41(9)- 4(8)
99 : 47(24)-50(1)-24(5)-48(1)-39(2)-49(12)-40(5)-49(3)-45(1)-44(1)-42(30)-31(3)-43(5)-46(3)-40(3)
100 : 46(2)-49(23)-41(11)-48(2)-50(2)-40(1)-39(4)-44(2)-47(9)-42(3)-48(10)-50(6)

S-3. Part No : \$ Total work-load cost[$TWLC_{I,J}$] : Demand,

01 : \$ 11621 : 933　02 : \$ 8625 : 3493　03 : \$ 67178 : 3439　04 : \$ 5534 : 725
05 : \$ 5468 : 1493　06 : \$ 11442 : 1314　07 : \$111115 : 3475　08 : \$ 1734 : 118
09 : \$ 14480 : 813　10 : \$ 77028 : 2854　11 : \$ 9969 : 357　12 : \$ 8026 : 972
13 : \$ 21017 : 3754　14 : \$ 18749 : 1533　15 : \$ 10406 : 1821　16 : \$ 848 : 82
17 : \$ 46243 : 3923　18 : \$ 5038 : 523　19 : \$ 18182 : 2349　20 : \$ 13294 : 1995
21 : \$ 14585 : 3682　22 : \$ 10199 : 747　23 : \$ 34792 : 3178　24 : \$ 39691 : 3010
25 : \$126588 : 2139　26 : \$ 4525 : 876　27 : \$ 2338 : 291　28 : \$ 29621 : 3172
29 : \$ 86997 : 3154　30 : \$ 58001 : 1332　31 : \$ 8226 : 422　32 : \$ 28370 : 2278
33 : \$ 43286 : 3335　34 : \$ 3718 : 287　35 : \$ 704 : 1056　36 : \$ 47830 : 3875
37 : \$ 36235 : 4043　38 : \$ 13409 : 1664　39 : \$ 2160 : 2428　40 : \$ 31455 : 3260
41 : \$ 54040 : 3387　42 : \$ 3485 : 3132　43 : \$ 71243 : 3521　44 : \$ 10123 : 3481
45 : \$ 1168 : 226　46 : \$ 766 : 116　47 : \$ 5891 : 2070　48 : \$ 42835 : 3333
49 : \$ 24706 : 2169　50 : \$ 8566 : 1786　51 : \$ 13608 : 2560　52 : \$ 3332 : 1364
53 : \$ 20577 : 1576　54 : \$ 71006 : 3065　55 : \$ 764 : 224　56 : \$ 22342 : 3871
57 : \$ 1784 : 3557　58 : \$ 10028 : 623　59 : \$ 38992 : 1048　60 : \$ 27421 : 900
61 : \$ 59601 : 1970　62 : \$ 8344 : 884　63 : \$ 45333 : 1765　64 : \$ 36102 : 2248
65 : \$ 77397 : 3425　66 : \$ 65158 : 4019　67 : \$ 24115 : 1860　68 : \$ 35009 : 1462
69 : \$124092 : 3281　70 : \$ 32443 : 3950　71 : \$ 62398 : 3568　72 : \$ 75755 : 3840
73 : \$ 5471 : 1648　74 : \$ 11578 : 349　75 : \$ 75958 : 3872　76 : \$ 62895 : 2428
77 : \$127668 : 3385　78 : \$ 62976 : 2637　79 : \$ 13184 : 2507　80 : \$ 80396 : 2022
81 : \$ 92753 : 2102　82 : \$ 57821 : 1987　83 : \$ 39143 : 1958　84 : \$ 97021 : 3170
85 : \$ 1691 : 158　86 : \$ 61542 : 3320　87 : \$126345 : 3458　88 : \$ 63478 : 2169
89 : \$ 14991 : 662　90 : \$ 26815 : 1309　91 : \$104053 : 2993　92 : \$ 87955 : 3065
93 : \$ 20930 : 3473　94 : \$ 13148 : 374　95 : \$ 99705 : 3296　96 : \$ 40110 : 1656
97 : \$ 38428 : 1202　98 : \$ 57820 : 2689　99 : \$ 82832 : 1852　100 : \$ 47101 : 1487

Work-load matrix[$WL_{i,j}$],

0.031 0.029 0.229 0.012 0.050 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 · · ·

0.031 0.000 0.057 0.000 0.012 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000

0.124 0.146 0.229 0.097 0.037 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.175 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000 0.145 0.009 0.041 0.452 0.003 0.016 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.022 0.000 0.003 0.068 0.119 0.003 0.000 0.000 0.000 0.000 0.000

0.000 0.000 1.519 0.000 0.000 0.000 1.129 0.002 0.203 0.238 0.003 0.032 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000 0.608 0.000 0.020 0.071 0.021 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000 0.058 0.001 0.000 0.000 0.000 0.065 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000 0.058 0.003 0.000 0.024 0.018 0.000 0.000 0.000 0.000 0.000

0.000 0.000 0.115 0.000 0.000 0.000 0.463 0.000 0.075 0.119 0.033 0.008 0.000 0.000 0.000 0.000

0.000 0.000 0.000 0.000 0.000 0.000 0.145 0.008 0.000 0.095 0.065 0.008 0.000 0.000 0.000 0.000

```
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.024 0.501 0.153 0.000 0.007
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.063 0.089 0.000 0.003
0.000 0.000 0.229 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.204 0.137 0.000
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.064 0.015 0.002
0.000 0.000 0.000 0.000 0.000 0.000 0.058 0.000 0.000 0.000 0.000 0.000 0.031 0.128 0.152 0.008
  ⋮                                                                                              ⋱
```

S-4. Machine No: Available number : Processing Rate $ Depreciation (%): Total work-load,

```
01: 1 : 5 $ 28446 (%): 042.579   02: 2 : 7 $ 48693 (%): 103.497
03: 1 : 9 $ 53576 (%): 086.603   04: 2 : 9 $ 42052 (%): 126.835
05: 1 : 4 $ 67250 (%): 021.449   06: 4 : 9 $ 19779 (%): 339.988
07: 2 : 6 $ 77818 (%): 150.213   08: 1 : 7 $ 56827 (%): 058.563
09: 1 : 5 $ 60425 (%): 018.347   10: 1 : 5 $ 27755 (%): 099.673
11: 1 : 6 $ 91391 (%): 084.949   12: 1 : 7 $ 03609 (%): 088.123
13: 1 : 8 $ 57579 (%): 018.449   14: 2 : 9 $ 23792 (%): 138.144
15: 1 : 4 $ 40779 (%): 048.610   16: 2 : 4 $ 42944 (%): 113.854
17: 2 : 4 $ 77072 (%): 169.724   18: 2 : 4 $ 52813 (%): 158.267
19: 3 : 6 $ 91217 (%): 233.595   20: 2 : 5 $ 79130 (%): 118.073
21: 2 : 4 $ 06014 (%): 140.978   22: 2 : 4 $ 53489 (%): 163.859
23: 2 : 3 $ 26689 (%): 126.902   24: 1 : 4 $ 60214 (%): 038.210
25: 1 : 9 $ 00263 (%): 099.638   26: 1 : 4 $ 58242 (%): 059.199
27: 1 : 4 $ 49929 (%): 086.152   28: 1 : 8 $ 77992 (%): 095.418
29: 3 : 9 $ 84429 (%): 236.998   30: 1 : 3 $ 93244 (%): 047.367
31: 1 : 4 $ 82423 (%): 056.107   32: 2 : 9 $ 68011 (%): 147.404
33: 3 : 9 $ 51871 (%): 242.292   34: 2 : 3 $ 78417 (%): 128.622
35: 2 : 5 $ 64404 (%): 117.501   36: 2 : 4 $ 09569 (%): 140.028
37: 4 : 9 $ 64699 (%): 317.633   38: 1 : 2 $ 27433 (%): 070.687
39: 3 : 5 $ 54856 (%): 237.860   40: 2 : 3 $ 25711 (%): 138.546
41: 4 : 9 $ 53772 (%): 312.163   42: 5 : 9 $ 47713 (%): 498.219
43: 2 : 3 $ 31515 (%): 124.271   44: 3 : 8 $ 83235 (%): 240.702
45: 3 : 5 $ 18700 (%): 258.434   46: 2 : 3 $ 40430 (%): 133.198
47: 3 : 7 $ 90864 (%): 293.050   48: 2 : 3 $ 92688 (%): 111.533
49: 5 : 8 $ 27842 (%): 411.164   50: 2 : 3 $ 00506 (%): 117.663
```

Efficiency measures, Inter-cell flow: 100.00 % ,
Inner-cell density: 18.67 %,
Work-load balance: 82.88 % ,
Under-utilization: 54.35 % .

COMBGR

S-1. Maximum machine-difference limit: 7.

S-2. Sorting by size and ordering by code significance.

```
Size is 13
PART NUMBER IS 87 WITH SIGNIFICANCE VALUE 1125762467889152.
PART NUMBER IS 88 WITH SIGNIFICANCE VALUE 1125109666414592.
PART NUMBER IS 99 WITH SIGNIFICANCE VALUE 1124526599438336.

Size is 12
PART NUMBER IS 84 WITH SIGNIFICANCE VALUE 1125350151028800.
PART NUMBER IS 80 WITH SIGNIFICANCE VALUE 1116833230880768.
PART NUMBER IS 91 WITH SIGNIFICANCE VALUE 1055260579725312.

Size is 11
PART NUMBER IS 96 WITH SIGNIFICANCE VALUE 1119062318907392.
PART NUMBER IS 94 WITH SIGNIFICANCE VALUE 1111365737521152.
PART NUMBER IS 97 WITH SIGNIFICANCE VALUE 1090440656846848.
PART NUMBER IS 83 WITH SIGNIFICANCE VALUE 1054706528960512.
PART NUMBER IS 95 WITH SIGNIFICANCE VALUE 843050540597250.

Size is 10
PART NUMBER IS 77 WITH SIGNIFICANCE VALUE 1104184552292352.
PART NUMBER IS 100 WITH SIGNIFICANCE VALUE 1103634796380160.
PART NUMBER IS 86 WITH SIGNIFICANCE VALUE 982138761510913.
PART NUMBER IS 82 WITH SIGNIFICANCE VALUE 936509028966656.

Size is 9
PART NUMBER IS 81 WITH SIGNIFICANCE VALUE 542334110400576.
PART NUMBER IS 89 WITH SIGNIFICANCE VALUE 249881331531776.
PART NUMBER IS 78 WITH SIGNIFICANCE VALUE 245740848808448.
PART NUMBER IS 10 WITH SIGNIFICANCE VALUE 206158432120.

Size is 8
PART NUMBER IS 90 WITH SIGNIFICANCE VALUE 905447825473536.
PART NUMBER IS 98 WITH SIGNIFICANCE VALUE 450524889481224.
PART NUMBER IS 3 WITH SIGNIFICANCE VALUE 422212465074727.
PART NUMBER IS 92 WITH SIGNIFICANCE VALUE 350469331353600.
PART NUMBER IS 65 WITH SIGNIFICANCE VALUE 281754015399936.
```

```
PART NUMBER IS 34 WITH SIGNIFICANCE VALUE 70918502023169.
PART NUMBER IS 12 WITH SIGNIFICANCE VALUE 8796093091496.
PART NUMBER IS 8 WITH SIGNIFICANCE VALUE 4672924419512.
PART NUMBER IS 69 WITH SIGNIFICANCE VALUE 271119810562.
PART NUMBER IS 7 WITH SIGNIFICANCE VALUE 34792.

Size is 7
PART NUMBER IS 85 WITH SIGNIFICANCE VALUE 469766342967296.
PART NUMBER IS 72 WITH SIGNIFICANCE VALUE 35523674537984.
PART NUMBER IS 22 WITH SIGNIFICANCE VALUE 35184374120960.
PART NUMBER IS 76 WITH SIGNIFICANCE VALUE 272730423296.
PART NUMBER IS 75 WITH SIGNIFICANCE VALUE 270582947840.
PART NUMBER IS 66 WITH SIGNIFICANCE VALUE 270582939680.
PART NUMBER IS 73 WITH SIGNIFICANCE VALUE 257698056256.
PART NUMBER IS 68 WITH SIGNIFICANCE VALUE 236223471616.
PART NUMBER IS 11 WITH SIGNIFICANCE VALUE 1912.

Size is 6
PART NUMBER IS 9 WITH SIGNIFICANCE VALUE 35184372089464.
PART NUMBER IS 71 WITH SIGNIFICANCE VALUE 9290014261248.
PART NUMBER IS 30 WITH SIGNIFICANCE VALUE 2216472018944.
PART NUMBER IS 25 WITH SIGNIFICANCE VALUE 1099513659392.
PART NUMBER IS 63 WITH SIGNIFICANCE VALUE 553916563456.
PART NUMBER IS 31 WITH SIGNIFICANCE VALUE 274879938560.
PART NUMBER IS 67 WITH SIGNIFICANCE VALUE 270582939648.
PART NUMBER IS 14 WITH SIGNIFICANCE VALUE 16840704.
PART NUMBER IS 29 WITH SIGNIFICANCE VALUE 2031620.

Size is 5
PART NUMBER IS 79 WITH SIGNIFICANCE VALUE 884557104545792.
PART NUMBER IS 93 WITH SIGNIFICANCE VALUE 198461848813568.
PART NUMBER IS 26 WITH SIGNIFICANCE VALUE 17593261424640.
PART NUMBER IS 28 WITH SIGNIFICANCE VALUE 2199024500768.
PART NUMBER IS 74 WITH SIGNIFICANCE VALUE 245081571328.
PART NUMBER IS 70 WITH SIGNIFICANCE VALUE 236223201280.
PART NUMBER IS 32 WITH SIGNIFICANCE VALUE 17214668800.
PART NUMBER IS 20 WITH SIGNIFICANCE VALUE 17181638656.
PART NUMBER IS 18 WITH SIGNIFICANCE VALUE 4296671236.
```

```
PART NUMBER IS 64 WITH SIGNIFICANCE VALUE 4160749568.
PART NUMBER IS 61 WITH SIGNIFICANCE VALUE 4160749568.
PART NUMBER IS 59 WITH SIGNIFICANCE VALUE 4160749568.
PART NUMBER IS 58 WITH SIGNIFICANCE VALUE 3892314116.
PART NUMBER IS 60 WITH SIGNIFICANCE VALUE 3087008256.
PART NUMBER IS 36 WITH SIGNIFICANCE VALUE 1215299584.
PART NUMBER IS 62 WITH SIGNIFICANCE VALUE 939526400.
PART NUMBER IS 27 WITH SIGNIFICANCE VALUE 806617216.
PART NUMBER IS 41 WITH SIGNIFICANCE VALUE 207618176.
PART NUMBER IS 53 WITH SIGNIFICANCE VALUE 125960192.
PART NUMBER IS 48 WITH SIGNIFICANCE VALUE 39846080.
PART NUMBER IS 43 WITH SIGNIFICANCE VALUE 6554626.
PART NUMBER IS 23 WITH SIGNIFICANCE VALUE 1507336.
PART NUMBER IS 16 WITH SIGNIFICANCE VALUE 120832.

Size is 4
PART NUMBER IS 15 WITH SIGNIFICANCE VALUE 562949953478656.
PART NUMBER IS 49 WITH SIGNIFICANCE VALUE 549762113536.
PART NUMBER IS 1 WITH SIGNIFICANCE VALUE 274877906951.
PART NUMBER IS 54 WITH SIGNIFICANCE VALUE 385875968.
PART NUMBER IS 21 WITH SIGNIFICANCE VALUE 270139392.
PART NUMBER IS 51 WITH SIGNIFICANCE VALUE 58851328.
PART NUMBER IS 5 WITH SIGNIFICANCE VALUE 33554439.
PART NUMBER IS 50 WITH SIGNIFICANCE VALUE 6303744.
PART NUMBER IS 33 WITH SIGNIFICANCE VALUE 1769472.
PART NUMBER IS 24 WITH SIGNIFICANCE VALUE 983040.
PART NUMBER IS 17 WITH SIGNIFICANCE VALUE 983040.
PART NUMBER IS 13 WITH SIGNIFICANCE VALUE 301056.

Size is 3
PART NUMBER IS 4 WITH SIGNIFICANCE VALUE 562949953421317.
PART NUMBER IS 37 WITH SIGNIFICANCE VALUE 17592257347584.
PART NUMBER IS 38 WITH SIGNIFICANCE VALUE 17592192335872.
PART NUMBER IS 44 WITH SIGNIFICANCE VALUE 137445244928.
PART NUMBER IS 47 WITH SIGNIFICANCE VALUE 8592162816.
PART NUMBER IS 56 WITH SIGNIFICANCE VALUE 109051904.
PART NUMBER IS 40 WITH SIGNIFICANCE VALUE 6307840.
PART NUMBER IS 6 WITH SIGNIFICANCE VALUE 4194324.
```

```
PART NUMBER IS 19 WITH SIGNIFICANCE VALUE 1245184.

Size is 2
PART NUMBER IS 46 WITH SIGNIFICANCE VALUE 17181966336.
PART NUMBER IS 55 WITH SIGNIFICANCE VALUE 75497472.
PART NUMBER IS 52 WITH SIGNIFICANCE VALUE 41943040.
PART NUMBER IS 42 WITH SIGNIFICANCE VALUE 6291456.
PART NUMBER IS 45 WITH SIGNIFICANCE VALUE 2129920.
PART NUMBER IS 2 WITH SIGNIFICANCE VALUE 5.

Size is 1
PART NUMBER IS 57 WITH SIGNIFICANCE VALUE 8388608.
PART NUMBER IS 39 WITH SIGNIFICANCE VALUE 4194304.
PART NUMBER IS 35 WITH SIGNIFICANCE VALUE 4194304.
```

S-3. Hosts and Guests,

HOSTS = { 87,88,99,84,80,91,96,94,83,95,77,86,82,81,89,78,10,98,03,65,34,12,08, 69,07,72,22,76,75,66,73,68,09,71,30,25,63,31,14,29,26,28,74,32,20,18,58,60,36,62, 27,41,53,48,43,23,16,15,49,01,54,21,05,50,13,04,37,38,44,47,40,06,46,45}

GUESTS= { 97,100,90,92,85,11,67,79,93,70,64,61,59,51,33,24,17,56,19,55,52,42,02, 57,39,35 }

Hospitality,

```
Host: 01 ={ 1, 2, 3, 4, 5, 8, 9 }
Host: 02 ={ 5 }
Host: 03 ={ 4, 8, 9, 24 }
Host: 04 ={ 3, 8, 9 }
Host: 05 ={ 3, 8, 9 }
Host: 06 ={ 3, 4, 5, 8, 9 }
Host: 07 ={ }
Host: 08 ={ }
Host: 09 ={ 5 }
Host: 10 ={ 9 }
Host: 11 ={ }
Host: 12 ={ }
Host: 13 ={ }
Host: 14 ={ }
```

```
Host: 15 ={ }
Host: 16 ={ }
Host: 17 ={ 6 }
Host: 18 ={ }
Host: 19 ={ 23 }
Host: 20 ={ 11, 12, 13 }
Host: 21 ={ 15, 16, 17, 19 }
Host: 22 ={ }
Host: 23 ={ }
Host: 24 ={ 7, 10 }
Host: 25 ={ }
Host: 26 ={ }
Host: 27 ={ 15, 16, 17, 19 }
Host: 28 ={ 7, 10 }
Host: 29 ={ 7, 10 }
Host: 30 ={ 7, 10 }
Host: 31 ={ }
Host: 32 ={ 10 }
Host: 33 ={ }
Host: 34 ={ }
Host: 35 ={ }
Host: 36 ={ 15, 16, 17, 19 }
Host: 37 ={ 11, 12, 13 }
Host: 38 ={ 15, 16, 17, 19 }
Host: 39 ={ }
Host: 40 ={ 15, 16, 17, 19 }
Host: 41 ={ }
Host: 42 ={ 19 }
Host: 43 ={ }
Host: 44 ={ 19 }
Host: 45 ={ 15, 19 }
Host: 46 ={ }
Host: 47 ={ }
Host: 48 ={ }
Host: 49 ={ 22, 25, 26 }
Host: 50 ={ }
```

Host: 51 ={ }

Host: 52 ={ 22, 25, 26 }

Host: 53 ={ 14, 18, 20, 21, 24 }

Host: 54 ={ 22, 25, 26 }

Host: 55 ={ 22, 25, 26 }

Host: 56 ={ 19 }

Host: 57 ={ }

Host: 58 ={ }

Host: 59 ={ 22, 25, 26 }

Host: 60 ={ 23 }

Host: 61 ={ }

Host: 62 ={ }

Host: 63 ={ 23 }

Host: 64 ={ 22, 25, 26 }

Host: 65 ={ }

Host: 66 ={ 23 }

Host: 67 ={ 25, 26 }

Host: 68 ={ 22, 25, 26 }

Host: 69 ={ 22, 25, 26 }

Host: 70 ={ }

Host: 71 ={ 22, 25, 26 }

Host: 72 ={ 25, 26 }

Host: 73 ={ }

Host: 74 ={ }

Flexibility,

Guest: 01 ={ 1 }

Guest: 02 ={ 1 }

Guest: 03 ={ 1, 4, 5, 6 }

Guest: 04 ={ 1, 3, 6 }

Guest: 05 ={ 1, 2, 6, 9 }

Guest: 06 ={ 17 }

Guest: 07 ={ 24, 28, 29, 30 }

Guest: 08 ={ 1, 3, 4, 5, 6 }

Guest: 09 ={ 1, 3, 4, 5, 6, 10 }

Guest: 10 ={ 24, 28, 29, 30, 32 }

Guest: 11 ={ 20, 37 }

Guest: 12 ={ 20, 37 }

Guest: 13 ={ 20, 37 }

Guest: 14 ={ 53 }

Guest: 15 ={ 21, 27, 36, 38, 40, 45 }

Guest: 16 ={ 21, 27, 36, 38, 40 }

Guest: 17 ={ 21, 27, 36, 38, 40 }

Guest: 18 ={ 53 }

Guest: 19 ={ 21, 27, 36, 38, 40, 42, 44, 45, 56 }

Guest: 20 ={ 53 }

Guest: 21 ={ 53 }

Guest: 22 ={ 49, 52, 54, 55, 59, 64, 68, 69, 71 }

Guest: 23 ={ 19, 60, 63, 66 }

Guest: 24 ={ 3, 53 }

Guest: 25 ={ 49, 52, 54, 55, 59, 64, 67, 68, 69, 71, 72 }

Guest: 26 ={ 49, 52, 54, 55, 59, 64, 67, 68, 69, 71, 72 }

S-4. Priorities(PR), Forward Relationships(FR), Inverse Relations(IR)

Host: 01 PR: 3 FR={ 4, 5, 6, 7, 9, 10, 12, 14, 16, 18 } IR={ }

Host: 02 PR: 3 FR={ 7, 9, 12, 14 } IR={ }

Host: 03 PR: 3 FR={ 10, 17 } IR={ }

Host: 04 PR: 4 FR={ 14, 16 } IR={ 1 }

Host: 05 PR: 4 FR={ 6 } IR={ 1 }

Host: 06 PR: 4 FR={ 9, 18 } IR={ 1, 5 }

Host: 07 PR: 4 FR={ 18 } IR={ 1, 2 }

Host: 08 PR: 3 FR={ 13, 58, 59, 66, 67, 68, 73, 74 } IR={ }

Host: 09 PR: 4 FR={ 12, 14, 18, 58, 66, 67, 68, 71, 73, 74 } IR={ 1, 2, 6 }

Host: 10 PR: 4 FR={ 16, 60, 66, 67, 68, 73, 74 } IR={ 1, 3 }

Host: 11 PR: 3 FR={ 74 } IR={ }

Host: 12 PR: 4 FR={ 66 } IR={ 1, 2, 9 }

Host: 13 PR: 4 FR={ 58, 59, 66, 67, 68, 73, 74 } IR={ 8 }

Host: 14 PR: 4 FR={ 73, 74 } IR={ 1, 2, 4, 9 }

Host: 15 PR: 3 FR={ 73, 74 } IR={ }

Host: 16 PR: 4 FR={ 67, 68, 73, 74 } IR={ 1, 4, 10 }

Host: 17 PR: 3 FR={ 33 } IR={ }

Host: 18 PR: 4 FR={ 67, 68, 73, 74 } IR={ 1, 6, 7, 9 }

```
Host: 19 PR: 3 FR={ 60, 63, 66 } IR={ }
Host: 20 PR: 3 FR={ 37, 47, 48, 74 } IR={ }
Host: 21 PR: 3 FR={ 36, 38, 40, 45, 56, 62 } IR={ }
Host: 22 PR: 3 FR={ 73, 74 } IR={ }
Host: 23 PR: 3 FR={ 72, 73, 74 } IR={ }
Host: 24 PR: 3 FR={ 28, 29, 30, 43, 73 } IR={ }
Host: 25 PR: 3 FR={ 74 } IR={ }
Host: 26 PR: 3 FR={ 73, 74 } IR={ }
Host: 27 PR: 3 FR={ 36, 38, 40, 45, 56, 62 } IR={ }
Host: 28 PR: 4 FR={ 29, 30, 43, 73 } IR={ 24 }
Host: 29 PR: 4 FR={ 30, 32, 43, 73 } IR={ 24, 28 }
Host: 30 PR: 4 FR={ 43, 73 } IR={ 24, 28, 29 }
Host: 31 PR: 3 FR={ 73 } IR={ }
Host: 32 PR: 4 FR={ 73 } IR={ 29 }
Host: 33 PR: 4 FR={ 72, 73 } IR={ 17 }
Host: 34 PR: 3 FR={ 43, 69, 70, 74 } IR={ }
Host: 35 PR: 3 FR={ 73 } IR={ }
Host: 36 PR: 4 FR={ 38, 40, 45 } IR={ 21, 27 }
Host: 37 PR: 4 FR={ 47, 48 } IR={ 20 }
Host: 38 PR: 4 FR={ 40, 45, 56, 62 } IR={ 21, 27, 36 }
Host: 39 PR: 3 FR={ 57, 58, 65, 74 } IR={ }
Host: 40 PR: 4 FR={ 45, 46, 56, 62 } IR={ 21, 27, 36, 38 }
Host: 41 PR: 3 FR={ 45, 62, 67, 68, 73, 74 } IR={ }
Host: 42 PR: 3 FR={ 44, 45, 56, 63, 70, 73, 74 } IR={ }
Host: 43 PR: 4 FR={ 69, 73, 74 } IR={ 24, 28, 29, 30, 34 }
Host: 44 PR: 4 FR={ 45, 73 } IR={ 42 }
Host: 45 PR: 4 FR={ 62, 73 } IR={ 21, 27, 36, 38, 40, 41, 42, 44 }
Host: 46 PR: 4 FR={ 62 } IR={ 40 }
Host: 47 PR: 4 FR={ 48, 66, 72, 73, 74 } IR={ 20, 37 }
Host: 48 PR: 4 FR={ 50, 73, 74 } IR={ 20, 37, 47 }
Host: 49 PR: 3 FR={ 68, 69, 71, 73, 74 } IR={ }
Host: 50 PR: 4 FR={ 73, 74 } IR={ 48 }
Host: 51 PR: 3 FR={ 62, 73, 74 } IR={ }
Host: 52 PR: 3 FR={ 67, 68, 69, 71, 73, 74 } IR={ }
Host: 53 PR: 3 FR={ 61 } IR={ }
Host: 54 PR: 3 FR={ 68, 69, 71, 73, 74 } IR={ }
```

Host: 55 PR: 3 FR={ 68, 69, 71, 73, 74 } IR={ }

Host: 56 PR: 4 FR={ 62, 70, 73, 74, } IR={ 21, 27, 36, 38, 40, 42 }

Host: 57 PR: 4 FR={ 65, 74 } IR={ 39 }

Host: 58 PR: 4 FR={ 74 } IR={ 8, 9, 13, 39 }

Host: 59 PR: 4 FR={ 64, 68, 69, 71, 73, 74 } IR={ 8, 13 }

Host: 60 PR: 4 FR={ 63, 66 } IR={ 10, 19 }

Host: 61 PR: 4 FR={ 67, 73, 74 } IR={ 53 }

Host: 62 PR: 4 FR={ 70, 73, 74 } IR={ 21, 27, 36, 38, 40, 41, 42, 45, 46, 51, 56 }

Host: 63 PR: 4 FR={ 66 } IR={ 19, 60 }

Host: 64 PR: 4 FR={ 68, 69, 71, 73, 74 } IR={ 59 }

Host: 65 PR: 4 FR={ 74 } IR={ 39, 57 }

Host: 66 PR: 4 FR={ 72, 73, 74 } IR={ 8, 9, 10, 12, 13, 19, 47, 60, 63 }

Host: 67 PR: 4 FR={ 68 } IR={ 8, 9, 10, 13, 16, 18, 41, 52, 61 }

Host: 68 PR: 4 FR={ 69, 71, 73, 74 } IR={ 8, 9, 10, 13, 16, 18, 41, 49, 52, 54, 55, 59, 64, 67 }

Host: 69 PR: 4 FR={ 71, 73, 74 } IR={ 34, 43, 49, 52, 54, 55, 59, 64, 68 }

Host: 70 PR: 4 FR={ 73, 74 } IR={ 34, 42, 56, 62 }

Host: 71 PR: 4 FR={ 73, 74 } IR={ 9, 49, 52, 54, 55, 59, 64, 68, 69 }

Host: 72 PR: 4 FR={ 73, 74 } IR={ 23, 33, 47, 66 }

Host: 73 PR: 4 FR={ 74 } IR={ 8, 9, 10, 13, 14, 15, 16, 18, 22, 23, 24, 26, 28, 29, 30, 31, 32, 33, 34, 35, 41, 42, 43, 44, 45, 47, 48, 49, 50, 51, 52, 54, 55, 56, 59, 61, 62, 64, 66, 68, 69, 70, 71, 72 }

Host: 74 PR: 2 FR={ } IR={ 8, 9, 10, 11, 13, 14, 15, 16, 18, 20, 22, 23, 25, 26, 33, 34, 39, 41, 42, 43, 47, 48, 49, 50, 51, 52, 54, 55, 56, 57, 58, 59, 61, 62, 64, 65, 66, 68, 69, 70, 71, 72, 73 }

S-5. Super-hosts,

Super-host 1 is = { 83, 38, 40 }

Merging Machinery ={ 22, 23 }

Machine Composition = [00000000000000100000011000000000000000101111110111]

Super-host 2 is = { 95, 1, 4 }

Merging Machinery ={ 1, 3 }

Machine Composition = [11100000000000000000000000000000000000110111111101]

Super-host 3 is = { 87, 84, 81 }

Merging Machinery ={ 7 }

```
Machine Composition = [ 000000100000000000000000000000000000000000001111111111111]

Super-host 4 is = { 89, 46, 45 }
Merging Machinery ={ 22 }
Machine Composition = [ 000000000000000010000010000010000010000001000101100011100]

Super-host 5 is = { 80, 91 }
Merging Machinery ={ 44 }
Machine Composition = [ 000000000000000000000000000000000000000100000111111111111]

Super-host 6 is = { 96, 98 }
Merging Machinery ={ 4 }
Machine Composition = [ 000100000000000000000000000000000000000000100111001111111]

Super-host 7 is = { 82, 15 }
Merging Machinery ={ 15, 16 }
Machine Composition = [ 000000001000011100000000000000000000000000000111100101011]

Super-host 8 is = { 78, 37 }
Merging Machinery ={ 23, 27 }
Machine Composition = [ 000000000100000000000000000010001000000000000001111101100]

Super-host 9 is = { 76, 75 }
Merging Machinery ={ 14 }
Machine Composition = [ 000000000000010000000000000000000000001111110000000000000]

Super-host 10 is = { 66, 74 }
Merging Machinery ={ 29 }
Machine Composition = [ 000001000000000000000000000000000001000111111000000000000]

Super-host 11 is = { 9, 6 }
Merging Machinery ={ 3, 23 }
Machine Composition = [ 001111100100000000000001000000000000000000000000000010000]

Super-host 12 is = { 25, 31 }
Merging Machinery ={ 39 }
Machine Composition = [ 000000000000000011111000000000000000000000001010000000000]

Super-host 13 is = { 63, 58 }
Merging Machinery ={ 3 }
Machine Composition = [ 001000000000000000000000000000000111110000000010000000000]

Super-host 14 is = { 29, 20 }
Merging Machinery ={ 35 }
```

```
Machine Composition = [ 001000000000000000111110000000000010000000000000000]

Super-host 15 is = { 18, 21 }
Merging Machinery ={ 29 }
Machine Composition = [ 001000000000000001011000000010001000000000000000]

Super-host 16 is = { 60, 62 }
Merging Machinery ={ 9, 12 }
Machine Composition = [ 000000001101000000000000001110100000000000000000]

Super-host 17 is = { 23, 47 }
Merging Machinery ={ 22, 34 }
Machine Composition = [ 000100000000000011011000000000001000000000000000]

Super-host 18 is = { 16, 13 }
Merging Machinery ={ 19 }
Machine Composition = [ 000000000001101110100000000000000000000000000000]

Super-host 19 is = { 49, 50 }
Merging Machinery ={ 13 }
Machine Composition = [ 000000000000110000000110000000000000010000000000]

Super-host 20 is = { 88, 86 }
Merging Machinery ={ 1 }
Machine Composition = [ 100000000000000000000000100000000010010111111111]

Super-host 21 is = { 3, 5 }
Merging Machinery ={ 26 }
Machine Composition = [ 111001000100010000000000100000000000000000000110]

Super-host 22 is = { 71, 44 }
Merging Machinery ={ 22, 23 }
Machine Composition = [ 000000000000000000001100000000110011100001000000]

Super-host 23 is = { 28, 32 }
Merging Machinery ={ 26, 35 }
Machine Composition = [ 000001000000000011001000010000001000000100000000]

Super-host 24 is = { 53, 54 }
Merging Machinery ={ 29 }
Machine Composition = [ 000000000000000001000011110100000000000000000000]

Super-host 25 is = { 99 }
Machine Composition = [ 000000000000000000000010000001000000110111111111]
```

```
Super-host 26 is = { 94 }

Machine Composition = [ 00000000000001000000000000000000000100110100111111]

Super-host 27 is = { 77 }

Machine Composition = [ 00000000000000011000000000000000000000100011011111]

Super-host 28 is = { 10 }

Machine Composition = [ 00011110111000000000000000000000000011000000000000]

Super-host 29 is = { 65 }

Machine Composition = [ 00000000000000010000000000011111000000100000000010]

Super-host 30 is = { 34 }

Machine Composition = [ 10000000000000001111100000000000000000010000001000]
Super-host 31 is = { 12 }

Machine Composition = [ 00010101011100001000000000000000000000000001000000]

Super-host 32 is = { 8 }

Machine Composition = [ 00011101101000000000000000000000000000100010000000]

Super-host 33 is = { 69 }

Machine Composition = [ 01000000000000000000000000000100111111000000000000]

Super-host 34 is = { 7 }

Machine Composition = [ 00010111111000010000000000000000000000000000000000]

Super-host 35 is = { 72 }

Machine Composition = [ 00000000000000010000000000000000111100100000010000]

Super-host 36 is = { 22 }

Machine Composition = [ 00000000010000001111100000000000000000000000010000]

Super-host 37 is = { 73 }

Machine Composition = [ 00000010000100100000000000000000001111000000000000]

Super-host 38 is = { 68 }

Machine Composition = [ 00000000000001000010000000000000111011000000000000]

Super-host 39 is = { 30 }

Machine Composition = [ 00000000000000001110000000001000001000000100000000]

Super-host 40 is = { 14 }

Machine Composition = [ 00000000000111110000000010000000000000000000000000]

Super-host 41 is = { 26 }

Machine Composition = [ 00000000000000001001100000000010000000000000100000]
```

```
Super-host 42 is = { 36 }
Machine Composition = [ 00000000000000000000111000010010000000000000000000]
Super-host 43 is = { 27 }
Machine Composition = [ 00000001000000000010100000001100000000000000000000]
Super-host 44 is = { 41 }
Machine Composition = [ 00000001000000000000011000110000000000000000000000]
Super-host 45 is = { 48 }
Machine Composition = [ 00000011000000000000011001000000000000000000000000]
Super-host 46 is = { 43 }
Machine Composition = [ 01000000001000000010011000000000000000000000000000]
```

Shop Configuration

```
CELL# 1
MACHINES( 13)={ 15, 22, 23, 39, 41, 42, 43, 44, 45, 46, 48, 49, 50 }
P A R TS( 5)={ 83, 85, 38, 42, 40 }
CELL# 2
MACHINES( 13)={ 1, 2, 3, 39, 40, 42, 43, 44, 45, 46, 47, 48, 50 }
P A R TS( 5)={ 95, 93, 1, 4, 2 }
CELL# 3
MACHINES( 14)={ 7, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50 }
P A R TS( 7)={ 87, 97,100, 84, 90, 79, 81 }
CELL# 4
MACHINES( 10)={ 16, 22, 28, 35, 39, 41, 42, 46, 47, 48, 49, 50 }
P A R TS( 3)={ 89, 46, 45 }
CELL# 5
MACHINES( 13)={ 33, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50 }
P A R TS( 3)={ 80, 91, 92 }
CELL# 6
MACHINES( 12)={ 4, 36, 39, 40, 41, 44, 45, 46, 47, 48, 49, 50 }
P A R TS( 2)={ 96, 98 }
CELL# 7
MACHINES( 12)={ 9, 14, 15, 16, 39, 40, 41, 42, 45, 47, 49, 50 }
P A R TS( 2)={ 82, 15 }
```

```
CELL# 8

MACHINES( 11)={ 10, 23, 27, 40, 41, 42, 43, 44, 45, 47, 48 }

P A R TS( 4)={ 78, 37, 39, 35 }

CELL# 9

MACHINES( 8)={ 14, 32, 33, 34, 35, 36, 37, 38, 48, 49, 50 }

P A R TS( 4)={ 76, 67, 70, 75 }

CELL# 10

MACHINES( 8)={ 6, 29, 33, 34, 35, 36, 37, 38, 47, 48, 50 }

P A R TS( 2)={ 66, 74 }

CELL# 11

MACHINES( 8)={ 3, 4, 5, 6, 7, 10, 23, 46, 49, 50 }

P A R TS( 2)={ 9, 6 }

CELL# 12

MACHINES( 7)={ 17, 18, 19, 20, 21, 39, 41, 47, 49, 50 }

P A R TS( 4)={ 25, 24, 17, 31 }

CELL# 13

MACHINES( 7)={ 3, 28, 29, 30, 31, 32, 40, 47, 49, 50 }

P A R TS( 5)={ 63, 64, 61, 59, 58 }

CELL# 14

MACHINES( 7)={ 3, 17, 18, 19, 20, 21, 35, 48, 49 }

P A R TS( 3)={ 29, 20, 33 }

CELL# 15

MACHINES( 6)={ 3, 18, 20, 21, 29, 33, 46, 47, 48 }

P A R TS( 2)={ 18, 21 }

CELL# 16

MACHINES( 7)={ 9, 10, 12, 28, 29, 30, 32, 47, 48 }

P A R TS( 2)={ 60, 62 }

CELL# 17

MACHINES( 7)={ 4, 17, 18, 19, 21, 22, 34, 37, 38 }

P A R TS( 2)={ 23, 47 }

CELL# 18

MACHINES( 6)={ 12, 13, 15, 16, 17, 19, 48, 49 }

P A R TS( 2)={ 16, 13 }
```

```
CELL# 19
MACHINES( 5)={ 13, 14, 22, 23, 40, 14, 48, 49 }
P A R TS( 2)={ 49, 50 }

CELL# 20
MACHINES( 14)={ 1, 26, 36, 39, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50 }
P A R TS( 2)={ 88, 86 }

CELL# 21
MACHINES( 9)={ 1, 2, 3, 6, 10, 14, 26, 48, 49 }
P A R TS( 2)={ 3, 5 }

CELL# 22
MACHINES( 8)={ 22, 23, 33, 34, 37, 38, 39, 44 }
P A R TS( 2)={ 71, 44 }

CELL# 23
MACHINES( 7)={ 6, 17, 18, 21, 26, 35, 42, 43 }
P A R TS( 3)={ 28, 19, 32 }

CELL# 24
MACHINES( 6)={ 18, 24, 25, 26, 27, 29, 37, 38 }
P A R TS( 7)={ 53, 51, 56, 55, 52, 57, 54 }

CELL# 25
MACHINES( 13)={ 24, 31, 39, 40, 42, 43, 44, 45, 46, 47, 48, 49, 50 }
P A R TS( 1)={ 99 }

CELL# 26
MACHINES( 11)={ 14, 36, 39, 40, 42, 45, 46, 47, 48, 49, 50 }
P A R TS( 1)={ 94 }

CELL# 27
MACHINES( 10)={ 16, 17, 39, 43, 44, 46, 47, 48, 49, 50 }
P A R TS( 1)={ 77 }

CELL# 28
MACHINES( 9)={ 4, 5, 6, 7, 9, 10, 11, 37, 38 }
P A R TS( 2)={ 10, 11 }

CELL# 29
MACHINES( 8)={ 16, 28, 29, 30, 31, 32, 39, 49 }
P A R TS( 1)={ 65 }
```

```
CELL# 30
MACHINES( 8)={ 1, 17, 18, 19, 20, 21, 40, 47 }
PARTS( 1)={ 34 }
CELL# 31
MACHINES( 8)={ 4, 6, 8, 10, 11, 12, 17, 44 }
PARTS( 1)={ 12 }
CELL# 32
MACHINES( 8)={ 4, 5, 6, 8, 9, 11, 39, 43 }
PARTS( 1)={ 8 }
CELL# 33
MACHINES( 8)={ 2, 30, 33, 34, } 35, 36, 37, 38 }
PARTS( 1)={ 69 }
CELL# 34
MACHINES( 8)={ 4, 6, 7, 8, 9, 10, 11, 16 }
PARTS( 1)={ 7 }
CELL# 35
MACHINES( 7)={ 16, 33, 34, 35, 36, 39, 46 }
PARTS( 1)={ 72 }
CELL# 36
MACHINES( 7)={ 10, 17, 18, 19, 20, 21, 46 }
PARTS( 1)={ 22 }
CELL# 37
MACHINES( 7)={ 7, 12, 15, 35, 36, 37, 38 }
PARTS( 1)={ 73 }
CELL# 38
MACHINES( 7)={ 14, 19, 33, 34, 35, 37, 38 }
PARTS( 1)={ 68 }
CELL# 39
MACHINES( 6)={ 17, 18, 19, 29, 35, 42 }
PARTS( 1)={ 30 }
CELL# 40
MACHINES( 6)={ 12, 13, 14, 15, 16, 25 }
PARTS( 1)={ 14 }
```

CELL# 41
MACHINES(5)={ 17, 20, 21, 31, 45 }
PARTS(1)={ 26 }

CELL# 42
MACHINES(5)={ 21, 22, 23, 28, 31 }
PARTS(1)={ 36 }

CELL# 43
MACHINES(5)={ 8, 19, 21, 29, 30 }
PARTS(1)={ 27 }

CELL# 44
MACHINES(5)={ 8, 22, 23, 27, 28 }
PARTS(1)={ 41 }

CELL# 45
MACHINES(5)={ 7, 8, 22, 23, 26 }
PARTS(1)={ 48 }

CELL# 46
MACHINES(5)={ 2, 11, 19, 22, 23 }
PARTS(1)={ 43 }

⋮

S-5. ··· Shop configuration (final iteration);

CELL# 1
MACHINES(48)={ 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50 }
PARTS(81)={ 89, 46, 45, 65, 63, 64, 61, 59, 58, 60, 62, 27, 41, 49, 50, 36, 34, 23, 47, 83, 85, 38, 42, 40, 43, 22, 25, 24, 17, 31, 95, 93, 1, 4, 2, 78, 37, 39, 35, 3, 5, 53, 51, 56, 55, 52, 57, 54, 14, 16, 13, 29, 20, 33, 30, 28, 19, 32, 18, 21, 26, 87, 97,100, 84, 90, 79, 81, 88, 86, 96, 98, 94, 82, 15, 80, 91, 92, 99, 77, 72 }

CELL# 2
MACHINES(21)={ 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 16, 17, 22, 23, 26, 37, 38, 39, 43, 44, 46 }
PARTS(8)={ 10, 11, 9, 6, 7, 48, 12 }

CELL# 3
MACHINES(20)={ 2, 6, 7, 12, 14, 15, 19, 22, 23, 29, 30, 32, 33, 34, 35, 36, 37, 38, 39, 44 }
PARTS(11)={ 71, 44, 76, 67, 70, 75, 66, 74, 68, 69, 73 }

S-6. Efficiency measures, Inter-cell flow: 100.00 % ,
Inner-cell density: 23.67 %,
Work-load balance: 74.70 % ,
Under-utilization: 65.05 % .
Extra Investment : $ 1677126.

MODROC

S-1. Lower limit on similarity coefficient: 0.75,
Upper Limit on number of independent pairs: 5.

S-2. Two ROC iterations:

MACHINE-PART INCIDENCE MATRIX (iteration 1.1)

```
PART:0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000001
PART:0000000001111111111222222222233333333334444444444555555555566666666667777777777888888888899999999990
PART:1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
LOC: 1234567890000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
LOC: 1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
M/C LOC
03 01 1111110000000000100000000000100000000000000000000000000001000000000000000000000000000000000000000000
01 02 1111100000000000000000000000000001000000000000000000000000000000000000000000000000010000000000000000
02 03 1010100000000000000000000000000000000000001000000000000000000000000000100000000000000000000000100000
39 04 1000000100000000000000000000001000000000000000000000000000000000000010000011000010011101111110110111111
06 05 0010001111100000000000001000000000000000000000000000000000000000010000000000000000000000000000000000
10 06 0010001011110000000010000000000000000000000000000000000000001000000000000000000010000000000000000000
14 07 0010000000000110000000000000000000000000000000011000000000000000000001000000000100000000000001000000
49 08 0010000000000000000000000000000000000000000000000000000000000000100000000000101111111111011101011111
48 09 0010000000000000000000000000000000000000000000000000000000000000000000000000011011011101110101111111
50 10 0001000000000010000000000000000000000000000000000000000000000000000000000000101101110111011001111011
26 11 0000100000000000000000000000000100000000000000001001111010000000000000000000000000000010000000000000
05 12 0000010111100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
23 13 0000010000000000000000000000000000111111111100011100000000000000000000000000000000000000000000000000
04 14 0000001111100000000010000000000000000000000000000000000000000000000000000000000000000000000000000100
11 15 0000001101110000000000000000000000000000001000000000000000000000000000000000000000000000000000000000
09 16 0000001101100000000000000000000000000000000000000000000000000001000000000000000000100000000000000000
08 17 0000001100010000000000000100000000000000010000100000000000000000000000000000000000000000000000000000
07 18 0000001011100000000000000000000000000000000001000000000000000000000000000010000000001001000000000000000
16 19 0000001000001111000000000000000000000000001000000000000000000000100000000100001000000000010000000000
43 20 0000000100000000000000000000000000000000000000000000000000000000000000000000111110110111011110101010
46 21 0000000010000000000010000000000000000000000000000000000000000000000000001000010111011111111111110011
38 22 0000000001000000000000000000000000000000010000000000000000000000011111101111000000000000100000000000
37 23 0000000001000000000000000000000000000000000000000000000000000000011111101111000000000000000000000000
12 24 0000000000111010000000000000000000000000000000000000000000001000000000001000000000000000000000000000
17 25 0000000000100011011011111011111110000000000000000000000000000000000000000000010000000000000000000000
44 26 0000000000010000000000000000000000000000000000000000000000000000000001000001100101111111001100111111
13 27 0000000000011010000000000000000000000000000000010000000000000000000000000000000000000000000000000000
19 28 0000000000001000100001111010111001000000001000000000000000000000000100000000000000000000000000000000
15 29 0000000000000111000000000000000000000010000000000000000000000000000000001000000000100000000000000000
25 30 0000000000000100000000000000000000000000000000000101100000000000000000000000000000000000000000000000
18 31 0000000000000001111111110011111110000000000000001000101000000000000000000000000000000000000000000000
20 32 0000000000000001101110111001010110000000000000000000000000000000000000000000000000000000000000000000
21 33 0000000000000001111101111101111010000000000000000000000000000000000000000000000000000000000000000000
33 34 0000000000000000100000000000000000000000000000000000000000000000011111110111000100000000001000000000
35 35 0000000000000000010000000001010000000000000100000000000000000000011111011011000000000000010000000000
29 36 0000000000000000001000001001000000000000000000000000100000111111111000000001000000000000000000000000
41 37 0000000000000000000000010000000000000000000000000000000000000000000000000000101111111111111000011101
31 38 0000000000000000000000001000000000100000000000000000000001101011100000000000000000000000000000000010
45 39 0000000000000000000000001000000000011000000000000000000000000000000000000000010101110111011111111110
30 40 0000000000000000000000000100000000000000000000000000000001111111100010000000000000000000000000000000
42 41 0000000000000000000000000101000000000000000000000000000000000000000000000000010101111011111101101011
40 42 0000000000000000000000000000000001000000000000001000000000000010000000000000001110101001001111111111
47 43 0000000000000000000000000000000001000000000000000000000000000000000000000000110111010111100001111011
22 44 0000000000000000000000000000000000101011111111111100000000000000000000000000000000000000000000000000
28 45 0000000000000000000000000000000000100001000000000000000000111111100000000000000000000001000000000000
27 46 0000000000000000000000000000000000001000100000000000000011110000000000000000000000000000000000000000
34 47 0000000000000000000000000000000000000000000000100000000000000000011111111001100000000000000000000000
24 48 0000000000000000000000000000000000000000000000000011101110000000000000000000000000000000000000000010
32 49 0000000000000000000000000000000000000000000000000000000001111011100000000001000000000000000000000000
36 50 0000000000000000000000000000000000000000000000000000000000000001101001111100000000000001000001010000
```

MACHINE-PART INCIDENCE MATRIX (iteration 1.2)

```
PART:0000000000000000000000000010000000000000000000000000000000000000000000000000000000000000000000000000
PART:0000002158394609887889989909886987731100162726154167897943555554443334332627114767721232312244656655
PART:1354269886453984287731907608155292110197268820509485409382314621406872593273635670457400396176391457
LOC: 1234567890000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
LOC: 1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
M/C LOC
03 01 1111111110000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
01 02 1111100001100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
02 03 1110000000011100000000000000000000000000000000000000000000000000000000000000000000000000000000000000
39 04 1000000001010011111111111111111100000000000000000000000000000000000000000000000000000000000000000000
06 05 0100000000000010000000000000000000111111100000000000000000000000000000000000000000000000000000000000
10 06 0100000000000000000000000000000000111110011100000000000000000000000000000000000000000000000000000000
14 07 0100000000000001100000000000000000000000000000111111000000000000000000000000000000000000000000000000
49 08 0100000001000001111111111111111000000000000000000000111000000000000000000000000000000000000000000000
48 09 0100000000010001011111111111110010000000000100000000100100000000000000000000000000000000000000000000
50 10 0001000000101000111111111111100000000000000000100000111000000000000000000000000000000000000000000000
26 11 0010000000000000001000000000000000000000000000000000000011111100000000000000000000000000000000000000
05 12 0000010000000010000000000000000000111000000000000000000000000000000000000000000000000000000000000000
23 13 0000010000001000000000000000000000000000000000011000000001000000111111110000000000000000000000000000
04 14 0000000000000010000000000001000000001111100000000000000000000000000000001000000000000000000000000000
11 15 0000000000001010000000000000000000110110000000000000000000000000000000000000000000000000000000000000
09 16 0000000000000101000000000000000000011010000000000000000000000000000000000100000000000000000000000000
08 17 0000000000000010000000000000000000011000000000000000000000000001000000010000000000000100000000000000
07 18 0000000000000000000000000000100000001111000000000000010001000000000000000000100000000000000000000000
16 19 0000000000000000001000000000001011000001000000100100000000000000000000000000111000000000000000000000
43 20 0000000001010010011111111000100100000000000010000000111100000000000000000000000000000000000000000000
46 21 0000000001010001011111110110110111000010000010000000111100000000000000000000000000000000000000000000
38 22 0000000000001000001000000000000000101000010000000011000000000000100000000001000111000000000000000000
37 23 0000000000001000000000000000000000101000010000000011000000000000000000000001000111000000000000000000
12 24 0000000000000000000000000000000000000000100000000100000000000000000000000001011100000000000000000000
17 25 0000010001000000010000000000000000001000010101000000000000010000000000000000100010000011111111000000
44 26 0000000001010000011111101111111010010000010010000000100000000000000000000000000000000000000000000000
13 27 0000000000000000000000000000000000000000000000010100000000000000000000000001100000000000000000000000
19 28 0000001000101000000000000000000000100000000010000010000000000000000000000010100100001111000000000000
15 29 0000000000000000001000000000000000000000000000100100000000000000010000000001100000000000000000000000
25 30 0000000000000000000000000000000000000000000000000100000000111000000000000000000000000000000000000000
18 31 0000011001000000000000000000000001000000101000000000000000111000000000000010000000000111111011000000
20 32 0000011001000000000000000000000001000000001000000000000000000000000000000000000000111011011000000000
21 33 0000011001000000000000000000000001000000101000000000000001000000010000010100000000001000111110000000
33 34 0000000100000100000001010000000001100000001000000011000000000000000000000000000011110000000000000000
35 35 0000000000001000000000000000000001100000001000000011000000100000000000000001000111000011000001000000
29 36 0000000000000000000000000000010000000000000001000000000000000100000000000110000000100010000100111100
41 37 0000000001000000110111011111110010000000000001000000011000000000000000000000000000100000000000000000
31 38 0000000010000000000000010000000100000000000000000000000000000000100000000000000000000000001000111100
45 39 0000000001010001110111111101000100000000000010000000110100000000000001100000000000000000001000000000
30 40 0000000010000100000000000000001000000000000001000000000000000000000000000110000000000000000000111100
42 41 0000000000100011101111110100101100000000011000000000110000000000000000000000000000000010000000000000
40 42 0000000001100011001011111110001000000000000100001000111100000000000000000000000000000000000000100000
47 43 0000000011100011111001111101000100000000000100000000100000000000000000000000000000000000000000000000
22 44 0000000000001000000000000000000000000000000000011000000001000000111110100000000010000000000011000000
28 45 0000000010000000000000000000001010000000000000010000000000000000001001000000010000000000000000111100
27 46 0000000000000000000000000000000000000000000000000000000000001011010000100000000000000000000000000010
34 47 0000000000001000000000000000000001100000010000000001100000000000000000000000000111000000000010000000
24 48 0000000000000000000000100000000000000000000000000000000000001101100000000000000000000000000000000011
32 49 0000000010000000000000000000000100000000000001000000000000000000000000000000000100000000000000111100
36 50 0000000000000101010000000010000000010000000010000001000000000000000000000001000110100000000000000000
```

MACHINE-PART INCIDENCE MATRIX (after two iterations)

```
PART:0000000000000000000000010000000000000000000000000000000000000000000000000000000000000000000000000000
PART:0000002158394698789989908898869803778977901110261456712643555553433444332232211234221176776566644755
PART:1354269886453948701977603185252981214098390127865908542082316246187402595031397407673667403914265357
LOC: 1234567890000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
LOC: 1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
M/C LOC
03 01 1111111110000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
01 02 1111100001100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
02 03 1110000000011100000000000000000000000000000000000000000000000000000000000000000000000000000000000000
39 04 1000000001010011111111111111111111111111110000000000000000000000000000000000000000000000000000000000
49 05 0100000000100001111111111111111111000001110000000000000000000000000000000000000000000000000000000000
48 06 0100000000010011111111111111001000010011000000000000000000000000000000000000000000000000000000000000
06 07 0100000000000000000000000000000000010000000011111100000000000000000000000000000000000000000000000000
14 08 0100000000000001000000000000001000000000000000001111110000000000000000000000000000000000000000000000
10 09 0100000000000000000000000000000000000001011111100000000110000000000000000000000000000000000000000000
26 10 0010000000000000100000000000000000000000000000000000000011111110000000000000000000000000000000000000
50 11 0001000000101001111111111110001000000111000000000100000000000000000000000000000000000000000000000000
23 12 0000010000001000000000000000000000000000000000001100000100000011111111100000000000000000000000000000
05 13 0000010000000000000000000000000010000000011100000000000000000000000000000000000000000000000000000000
18 14 0000001100100000000000000000000100000000000001000000001001110000000000000111111111000000000000000000
21 15 0000001100100000000000000000000100000000000001000000001001000001000000000111111000110000000000000000
20 16 0000001100100000000000000000000100000000000000000000001000000000000000000111100110010000000000000000
19 17 0000001000101000000000000000000100000000000000000001001000000000000000001000101110011000000000000000
17 18 0000001000100000100000000000000100000000000101000000001001000000000000001110111110100100000000000000
33 19 0000000100000100011000000000000110000000000000100011000000000000000000000000000000000011110000000000
30 20 0000000010000100000000000000000100000000000000000000000100000000000000000000000000010000000111110000
31 21 0000000010000000000100000000010000000000000000000000000000000000100000000000000000010000000111100000
28 22 0000000010000000000000000000101000000000000000000000000010000000001100000000000000000000000111110000
32 23 0000000010000000000000000000010000000000000000000000000010000000000000000000000000000000100011100000
47 24 0000000001110011110111111010010010000100100000000000000000000000000000000000000000000000000000000000
43 25 0000000001010001111111001100001010001111100000000000000000000000000000000000000000000000000000000000
45 26 0000000001010011011111101010101000001101100000000000000000000000000000001100000000000010000000000000000
46 27 0000000001010011111101111010011001011101100000000000000100000000000000000000000000000000000000000000
44 28 0000000001010001101111111111001000011001000100000000000000000000000000000000000000000000000000000000
41 29 0000000001000001011011111111100100001101000000000000000000000000000000000010000000000000000000000000
40 30 0000000000110010011111110010101000001111100000000010000000000000000000000000000000000000010000000000
42 31 0000000000010011011111011001101100001101000000010000000000000000000000000000000010000000000000000000
11 32 0000000000001000000000000000000000100000000011110000000000000000000000000000000000000000000000000000
22 33 0000000000001000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
36 34 0000000000000111000000100000000000000100000000000100001000000000000000000000000000000001110000000100
38 35 0000000000000100000001000000000000001000000010000100011000000000000000010000000000000000111100000000100
35 36 0000000000000100000000000000000100100000000000000100011000010000000000000001000000100000110100000010100
34 37 0000000000000100000000000000000110000000000000000100011000000000000000000000000000010000110100000000000
37 38 0000000000000100000000000000000000001000000010000100011000000000000000000000000000000000111100000000100
09 39 0000000000000000000000000000100010000000001101000000000000000000000000000000000000000000000000100000
04 40 0000000000000000000000000010000010000000011111000000000000000000000000000000010000000000000000000000
08 41 0000000000000000000000000000000010000000000011000000000000001000000010000000000000010000000000000000
16 42 0000000000000000000001000000000000001010010001001000010000000000000000000000000000001100000000001000
15 43 0000000000000000000000000100000000000000000000001000010000000000000000010000000000000100000000000100
24 44 0000000000000000000100000000000000000000000000000000000000011110000000000000000000000000000000000011
07 45 0000000000000000000000000100000000001000011101000000000000010000000000000000000000000000000000000100
29 46 0000000000000000000000000000010000000000000000000000000000100000010000000000010000100100001011111100000
12 47 0000000000000000000000000000000000000000000010000000001000000000000000000000000000001100000000100100
13 48 0000000000000000000000000000000000000000000000000100100000000000000000000000000000011000000000000000
25 49 0000000000000000000000000000000000000000000000000000000010000110010000000000000000000000000000000000
27 50 0000000000000000000000000000000000000000000000000000000000010101010100000000000000000000000000000010
```

S-3. First candidate cells

```
CELL # 1 IS :
MACHINES = { 3, 1, 2 }.
P A R TS = { 1, 3, 5 }.
Other candidate cells:

CELL # 2 IS :
MACHINES = { 3, 1 }.
P A R TS = { 4, 2 }.

CELL # 3 IS :
MACHINES = { 3 }.
P A R TS = { 6, 29, 18, 58 }.

CELL # 4 IS :
MACHINES = { 3, 1, 46, 43, 44, 49, 45, 41 }.
P A R TS = { 86 }.

CELL # 5 IS :
MACHINES = { 3, 1 }.
P A R TS = { 34 }.

CELL # 6 IS :
MACHINES = { 3 }.
P A R TS = { 77 }.

CELL # 7 IS :
MACHINES = { 3 }.
P A R TS = { 80 }.

CELL # 8 IS :
MACHINES = { 3 }.
P A R TS = { 91 }.

CELL # 9 IS :
MACHINES = { 3 }.
P A R TS = { 87 }.

CELL # 10 IS :
MACHINES = { 3 }.
P A R TS = { 99 }.
```

```
CELL # 11 IS :
MACHINES = { 3 }.
P A R TS = { 78 }.
CELL # 12 IS :
MACHINES = { 3 }.
P A R TS = { 96 }.
CELL # 13 IS :
MACHINES = { 3 }.
P A R TS = { 92 }.
CELL # 14 IS :
MACHINES = { 3 }.
P A R TS = { 22 }.
CELL # 15 IS :
MACHINES = { 3 }.
P A R TS = { 19 }.
CELL # 16 IS :
MACHINES = { 3 }.
P A R TS = { 85 }.
CELL # 17 IS :
MACHINES = { 3 }.
P A R TS = { 61 }.
CELL # 18 IS :
MACHINES = { 3 }.
P A R TS = { 50 }.
CELL # 19 IS :
MACHINES = { 3 }.
P A R TS = { 25 }.
CELL # 20 IS :
MACHINES = { 3 }.
P A R TS = { 98 }.
CELL # 21 IS :
MACHINES = { 3 }.
P A R TS = { 93 }.
```

```
CELL # 22 IS :
MACHINES = { 3 }.
PARTS = { 40 }.

CELL # 23 IS :
MACHINES = { 3 }.
PARTS = { 42 }.

CELL # 24 IS :
MACHINES = { 3 }.
PARTS = { 39 }.

CELL # 25 IS :
MACHINES = { 3 }.
PARTS = { 75 }.

CELL # 26 IS :
MACHINES = { 3 }.
PARTS = { 32 }.

CELL # 27 IS :
MACHINES = { 3 }.
PARTS = { 66 }.

CELL # 28 IS :
MACHINES = { 3 }.
PARTS = { 56 }.

CELL # 29 IS :
MACHINES = { 3 }.
PARTS = { 9 }.

CELL # 30 IS :
MACHINES = { 3 }.
PARTS = { 49 }.

CELL # 31 IS :
MACHINES = { 3 }.
PARTS = { 38 }.

CELL # 32 IS :
MACHINES = { 3 }.
PARTS = { 59 }.
```

```
CELL # 33 IS :
MACHINES = { 3 }.
P A R TS = { 100 }.

CELL # 34 IS :
MACHINES = { 3 }.
P A R TS = { 53 }.

CELL # 35 IS :
MACHINES = { 3 }.
P A R TS = { 28 }.

CELL # 36 IS :
MACHINES = { 3 }.
P A R TS = { 90 }.

CELL # 37 IS :
MACHINES = { 3 }.
P A R TS = { 76 }.

CELL # 38 IS :
MACHINES = { 3, 1 }.
P A R TS = { 84, 74, 81, 62 }.

CELL # 39 IS :
MACHINES = { 3 }.
P A R TS = { 65, 89, 48, 88 }.

CELL # 40 IS :
MACHINES = { 39, 18 }.
P A R TS = { 10, 41, 30, 57 }.

CELL # 41 IS :
MACHINES = { 34, 21, 6 }.
P A R TS = { 43, 51, 69, 73, 7 }.

CELL # 42 IS :
MACHINES = { 34 }.
P A R TS = { 35, 21 }.

CELL # 43 IS :
MACHINES = { 21 }.
P A R TS = { 71, 27 }.
```

CELL # 44 IS :
MACHINES = { 40, 44, 49 }.
PARTS = { 15, 54, 23, 33 }.

CELL # 45 IS :
MACHINES = { 7, 2, 13 }.
PARTS = { 79, 11, 67 }.

CELL # 46 IS :
MACHINES = { 7, 2, 5, 19, 33, 36 }.
PARTS = { 46, 20, 64, 37, 97, 72, 24 }.

CELL # 47 IS :
MACHINES = { 7, 29, 2, 50 }.
PARTS = { 47, 63, 16, 82, 8, 94 }.

CELL # 48 IS :
MACHINES = { 7, 19 }.
PARTS = { 26, 12, 44 }.

CELL # 49 IS :
MACHINES = { 30, 50, 36, 33 }.
PARTS = { 68, 31, 45 }.

CELL # 50 IS :
MACHINES = { 30, 22, 35, 15, 5 }.
PARTS = { 83, 14 }.

CELL # 51 IS :
MACHINES = { 2 }.
PARTS = { 52, 17 }.

CELL # 52 IS :
MACHINES = { 23, 20, 25, 28, 12 }.
PARTS = { 60, 13, 36 }.
CELL # 53 IS :
MACHINES = { 24, 42, 16, 37, 14 }.
PARTS = { 70 }.

CELL # 54 IS :
MACHINES = { 38, 13, 32 }.
PARTS = { 95, 55 }.

S-4. Similarity coefficient matrix:

$$\begin{array}{ccccccccccc}
: & 1.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & \cdots \\
 & : & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & \cdots \\
 & & : & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & \cdots \\
 & & & : & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & \cdots \\
 & & & & : & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & \cdots \\
 & & & & & : & 0.500 & 0.500 & 0.500 & 0.000 & \cdots \\
 & & & & & & : & 0.750 & 0.667 & 0.750 & \cdots \\
 & & & & & & & : & 0.667 & 0.000 & \cdots \\
 & & & & & & & & : & 0.333 & \cdots \\
 & & & & & & & & & : & \cdots \\
 & & & & & & & & & & \ddots
\end{array}$$

S-5. Merges:

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 2.
MACHINES ={ 3, 1, 2 }
PARTS ={ 1, 3, 5, 4, 2 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 3.
PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 5.
PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 6.
PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 7.
PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 8.
PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 9.
PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 10.
PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 11.
PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 12.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 13.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 14.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 15.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 16.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 17.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 18.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 19.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 20.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 21.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 22.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 23.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 24.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 25.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39, 75 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 26.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39, 75, 32 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 27.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 28.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 29.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 30.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 31.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 32.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38, 59 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 33.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38, 59,100 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 34.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38, 59,100, 53 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 35.

PARTS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85, 61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38, 59,100, 53, 28 }

```
PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 36.
P A R TS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85,
61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38, 59,100, 53, 28, 90 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 37.
P A R TS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85,
61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38, 59,100, 53, 28, 90, 76 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 38.
P A R TS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85,
61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38, 59,100, 53, 28, 90, 76, 84, 74,
81, 62 }

PRIMARY CELL 4 IS MERGED WITH PRIMARY CELL 39.
P A R TS ={ 86, 65, 89, 48, 88 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 51.
P A R TS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85,
61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38, 59,100, 53, 28, 90, 76, 84, 74,
81, 62, 52, 17 }

PRIMARY CELL 41 IS MERGED WITH PRIMARY CELL 42.
P A R TS ={ 43, 51, 69, 73, 7, 35, 21 }

PRIMARY CELL 46 IS MERGED WITH PRIMARY CELL 48.
P A R TS ={ 46, 20, 64, 37, 97, 72, 24, 26, 12, 44 }

PRIMARY CELL 41 IS MERGED WITH PRIMARY CELL 43.
P A R TS ={ 43, 51, 69, 73, 7, 35, 21, 71, 27 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 4.
P A R TS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85,
61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38, 59,100, 53, 28, 90, 76, 84, 74,
81, 62, 52, 17, 86, 65, 89, 48, 88 }

PRIMARY CELL 45 IS MERGED WITH PRIMARY CELL 46.
P A R TS ={ 79, 11, 67, 46, 20, 64, 37, 97, 72, 24, 26, 12, 44 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 44.
P A R TS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85,
61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38, 59,100, 53, 28, 90, 76, 84, 74,
81, 62, 52, 17, 86, 65, 89, 48, 88, 15, 54, 23, 33 }

PRIMARY CELL 45 IS MERGED WITH PRIMARY CELL 47.
```

```
P A R TS ={ 79, 11, 67, 46, 20, 64, 37, 97, 72, 24, 26, 12, 44, 47, 63, 16, 82, 8, 94 }

PRIMARY CELL 45 IS MERGED WITH PRIMARY CELL 49.
P A R TS ={ 79, 11, 67, 46, 20, 64, 37, 97, 72, 24, 26, 12, 44, 47, 63, 16, 82, 8, 94,
68, 31, 45 }

PRIMARY CELL 45 IS MERGED WITH PRIMARY CELL 50.
P A R TS ={ 79, 11, 67, 46, 20, 64, 37, 97, 72, 24, 26, 12, 44, 47, 63, 16, 82, 8, 94,
68, 31, 45, 83, 14 }

PRIMARY CELL 45 IS MERGED WITH PRIMARY CELL 54.
P A R TS ={ 79, 11, 67, 46, 20, 64, 37, 97, 72, 24, 26, 12, 44, 47, 63, 16, 82, 8, 94,
68, 31, 45, 83, 14, 95, 55 }

PRIMARY CELL 1 IS MERGED WITH PRIMARY CELL 45.
P A R TS ={ 1, 3, 5, 4, 2, 6, 29, 18, 58, 34, 77, 80, 91, 87, 99, 78, 96, 92, 22, 19, 85,
61, 50, 25, 98, 93, 40, 42, 39, 75, 32, 66, 56, 9, 49, 38, 59,100, 53, 28, 90, 76, 84, 74,
81, 62, 52, 17, 86, 65, 89, 48, 88, 15, 54, 23, 33, 79, 11, 67, 46, 20, 64, 37, 97, 72, 24,
26, 12, 44, 47, 63, 16, 82, 8, 94, 68, 31, 45, 83, 14, 95, 55 }
```

S-6. Based on α, choose the solution with the highest grouping efficiency value among all PF/MG-F proposals,

(a) α : 0. 0
# of cells: 6
Inter-cell flow efficiency: .4297
Inner-cell density efficiency: .5184
Work-load balance efficiency: .8138
Under-utilization efficiency: .7226
Extra investment: $ 0.

(b) α : 0. 1
# of cells: 6
Inter-cell flow efficiency: .4297
Inner-cell density efficiency: .5184
Work-load balance efficiency: .8138
Under-utilization efficiency: .7226
Extra investment: $ 0.

(c) α : 0. 2
# of cells: 6
Inter-cell flow efficiency: .4297
Inner-cell density efficiency: .5184
Work-load balance efficiency: .8138
Under-utilization efficiency: .7226
Extra investment: $ 0.

(d) α : 0. 3
# of cells: 6
Inter-cell flow efficiency: .4297
Inner-cell density efficiency: .5184
Work-load balance efficiency: .8138
Under-utilization efficiency: .7226
Extra investment: $ 0.

(e) α : 0. 4
# of cells: 5
Inter-cell flow efficiency: .4304
Inner-cell flow efficiency: .4220
Work-load balance efficiency: .8091
Under-utilization efficiency: .6746
Extra investment: $ 0.

(f) α : 0. 5
# of cells: 5
Inter-cell flow efficiency: .4304
Inner-cell flow efficiency: .4220
Work-load balance efficiency: .8091
Under-utilization efficiency: .6746
Extra investment: $ 0.

(g) α : 0. 6
# of cells: 5
Inter-cell flow efficiency: .4304
Inner-cell flow efficiency: .4220
Work-load balance efficiency: .8091
Under-utilization efficiency: .6746
Extra investment: $ 0.

(h) α : 0. 7

# of cells: 5
Inter-cell flow efficiency: .4304
Inner-cell flow efficiency: .4220
Work-load balance efficiency: .8091
Under-utilization efficiency: .6746
Extra investment: $ 0.

(i) α : 0. 8

# of cells: 5
Inter-cell flow efficiency: .4304
Inner-cell flow efficiency: .4220
Work-load balance efficiency: .8091
Under-utilization efficiency: .6746
Extra investment: $ 0.

(j) α : 0. 9

# of cells: 5
Inter-cell flow efficiency: .4304
Inner-cell flow efficiency: .4220
Work-load balance efficiency: .8091
Under-utilization efficiency: .6746
Extra investment: $ 0.

MACE

S-1. Similarity coefficients: PSC,
Threshold values: 0.10.

S-2. Number of common parts (NCC), total number of parts (TNC), total flow of common parts processed (TFC)

Machine # : 1
NCC = (0, 3, 5, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 1, 1, 0, 1, 1, 1, 1, 2, 1, 2, 2)
TNC = 7 TFC = 32

Machine # : 2
NCC = (3, 0, 3, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1,
0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 2, 1, 0, 1, 1, 1, 1, 1, 1, 2, 1, 1)
TNC = 6 TFC = 34

Machine # : 3
NCC = (5, 3, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 2, 1, 2, 2, 0, 1, 0, 0, 1,
0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1)
TNC = 9 TFC = 31

Machine # : 4
NCC = (0, 0, 0, 0, 4, 6, 4, 3, 4, 5, 5, 1, 0, 0, 0, 1, 2, 1, 1, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 2, 1, 1, 0, 1, 2, 1, 1, 0, 1, 1, 0)
TNC = 8 TFC = 51

Machine # : 5
NNC = (0, 0, 1, 4, 0, 4, 3, 1, 3, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0)
TNC = 5 TFC = 28

Machine # : 6
NCC = (1, 1, 1, 6, 4, 0, 4, 3, 4, 6, 5, 1, 0, 1, 0, 1, 2, 1, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0)
TNC = 9 TFC = 57

Machine # : 7
NCC = (0, 0, 0, 4, 3, 4, 0, 2, 3, 4, 3, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 2, 2, 1, 1, 2, 1, 2, 2, 1, 3, 2, 2, 2, 1)
TNC = 8 TFC = 55

Machine # : 8
NCC = (0, 0, 0, 3, 1, 3, 2, 0, 2, 2, 3, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 2, 2, 0, 0, 1,
1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0)
TNC = 6 TFC = 33

Machine # : 9
NCC = (0, 0, 0, 4, 3, 4, 3, 2, 0, 3, 4, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 2, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1)
TNC = 6 TFC = 41

Machine # : 10
NCC = (1, 1, 1, 5, 3, 6, 4, 2, 3, 0, 4, 1, 0, 1, 0, 1, 2, 1, 1, 1, 1, 0, 0, 0, 0, 0,
0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 2, 1, 2, 1, 2, 1, 0)
TNC = 9 TFC = 58

Machine # : 11
NCC = (0, 1, 0, 5, 3, 5, 3, 3, 4, 4, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0)
TNC = 6 TFC = 39

Machine # : 12
NCC = (0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 3, 1, 3, 3, 2, 0, 1, 0, 0, 0, 0, 0, 1, 0,
0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0)
TNC = 6 TFC = 29

Machine # : 13
NCC = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 2, 2, 3, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
TNC = 4 TFC = 15

Machine # : 14
NCC = (1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 2, 0, 2, 2, 0, 0, 1, 0, 0, 2, 2, 0, 1, 0,
0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 3, 1, 2, 0, 0, 2, 1, 2, 2, 3, 3)
TNC = 9 TFC = 52

Machine # : 15
NCC = (0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 3, 2, 2, 0, 3, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 2)
TNC = 6 TFC = 30

Machine # : 16
NCC = (0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 3, 3, 2, 3, 0, 2, 0, 1, 0, 0, 1, 0, 0, 1, 0,
0, 2, 1, 1, 1, 1, 1, 1, 2, 1, 0, 0, 4, 0, 1, 1, 1, 1, 0, 3, 2, 2, 2, 2)
TNC = 10 TFC = 53

Machine # : 17
NCC = (1, 0, 1, 2, 0, 2, 0, 1, 0, 2, 1, 2, 1, 0, 1, 2, 0,14, 9,10,12, 0, 0, 0, 0, 1,
0, 0, 1, 0, 1, 0, 0, 0, 3, 0, 0, 0, 2, 1, 1, 2, 1, 2, 1, 2, 2, 1, 1, 1)
TNC = 18 TFC = 84

Machine # : 18
NCC = (1, 0, 2, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,14, 0, 9,11,13, 1, 0, 2, 2, 3,
1, 0, 2, 0, 0, 0, 1, 1, 3, 0, 0, 0, 1, 1, 1, 2, 0, 0, 0, 1, 1, 0, 0, 0)
TNC = 19 TFC = 76

Machine # : 19
NCC = (1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 9, 9, 0, 7, 7, 1, 1, 0, 0, 0, 0, 0, 2, 1, 0, 0, 1, 1, 2, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0)
TNC = 13 TFC = 60

Machine # : 20
NCC = (1, 0, 2, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,10,11, 7, 0,10, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0)
TNC = 12 TFC = 52

Machine # : 21
NCC = (1, 0, 2, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,12,13, 7,10, 0, 1, 1, 0, 0, 1, 0, 1, 2, 1, 2, 0, 1, 0, 2, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0)
TNC = 16 TFC = 68

Machine # : 22
NCC = (0, 1, 0, 0, 0, 0, 1, 2, 0, 0, 1, 0, 1, 2, 1, 1, 0, 1, 1, 0, 1, 0,10, 0, 0, 1, 1, 2, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0)
TNC = 13 TFC = 33

Machine # : 23
NCC = (0, 1, 1, 0, 1, 0, 1, 2, 0, 0, 1, 0, 1, 2, 1, 0, 0, 0, 1, 0, 1,10, 0, 0, 0, 1, 2, 2, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0)
TNC = 14 TFC = 33

Machine # : 24
NCC = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 2, 4, 3, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
TNC = 7 TFC = 23

Machine # : 25
NCC = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 2, 0, 0, 0, 0, 0, 2, 0, 3, 2, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
TNC = 4 TFC = 15

Machine # : 26
NCC = (1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 3, 0, 0, 1, 1, 1, 4, 3, 0, 3, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
TNC = 9 TFC = 36

Machine # : 27
NCC = (0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 2, 3, 2, 3, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0)
TNC = 6 TFC = 16

Machine # : 28
NCC = (0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 2, 0, 0, 0, 0, 1, 2, 2, 0, 0, 0, 1, 0, 7, 8, 7, 7, 0, 0, 1, 0, 0, 0, 2, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0)
TNC = 11 TFC = 52

Machine # : 29
NCC = (0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 2, 2, 1, 2, 0, 0, 0, 1, 1,
1, 7, 0, 8, 5, 6, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0)
TNC = 12 TFC = 51

Machine # : 30
NCC = (0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0,
0, 8, 8, 0, 6, 7, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)
TNC = 10 TFC = 47

Machine # : 31
NCC = (0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 2, 1, 1, 1, 0, 0,
0, 7, 5, 6, 0, 6, 0, 0, 0, 0, 0, 0, 2, 2, 0, 1, 1, 1, 2, 1, 1, 1, 2, 1)
TNC = 9 TFC = 48

Machine # : 32
NCC = (0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 7, 6, 7, 6, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)
TNC = 8 TFC = 38

Machine # : 33
NCC = (0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 2, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0,
0, 0, 1, 1, 0, 1, 0, 9, 8, 7, 9, 9, 4, 2, 2, 2, 2, 2, 2, 3, 1, 2, 2, 2)
TNC = 13 TFC = 81

Machine # : 34
NCC = (0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 2, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 1, 0, 1, 9, 0, 8, 6, 8, 8, 2, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0)
TNC = 10 TFC = 53

Machine # : 35
NCC = (0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 2, 1, 2, 3, 3, 2, 1, 2, 1, 0, 0, 0, 1,
0, 1, 1, 1, 0, 1, 8, 8, 0, 7, 8, 8, 2, 0, 1, 2, 0, 0, 0, 2, 1, 1, 0, 0)
TNC = 14 TFC = 74

Machine # : 36
NCC = (0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 2, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 1, 1, 0, 1, 7, 6, 7, 0, 7, 7, 4, 2, 2, 2, 1, 2, 3, 4, 3, 3, 3, 3)
TNC = 11 TFC = 78

Machine # : 37
NCC = (0, 1, 0, 1, 1, 2, 2, 0, 1, 1, 1, 1, 0, 2, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 1, 0, 1, 9, 8, 8, 7, 0,11, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0)
TNC = 11 TFC = 63

Machine # : 38
NCC = (0, 1, 0, 1, 1, 2, 2, 0, 1, 1, 1, 1, 0, 2, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0,
0, 0, 1, 1, 0, 1, 9, 8, 8, 7,11, 0, 2, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1)
TNC = 13 TFC = 77

```
Machine # : 39
NCC = ( 2, 2, 1, 2, 1, 1, 1, 1, 2, 0, 1, 0, 0, 2, 1, 4, 2, 1, 1, 1, 1, 0, 0, 1, 0, 1,
0, 2, 1, 1, 2, 1, 4, 2, 2, 4, 1, 2, 0,12,14,14,13,16,14,17,14,16,18,14)
TNC = 25 TFC =213

Machine # : 40
NCC = ( 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 3, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
0, 1, 1, 1, 2, 1, 2, 0, 0, 2, 0, 1,12, 0,11,13,12,11,15,13,12,13,14,13)
TNC = 20 TFC =167

Machine # : 41
NCC = ( 1, 0, 0, 1, 0, 0, 2, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1,
0, 1, 0, 0, 0, 0, 2, 0, 1, 2, 0, 1,14,11, 0,13,11,13,13,13,12,14,15,12)
TNC = 18 TFC =163

Machine # : 42
NCC = ( 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 2, 1, 1, 2, 2, 1, 0, 1, 0, 0, 1, 0, 1,
0, 1, 1, 0, 1, 0, 2, 0, 2, 2, 0, 1,14,13,13, 0,12,12,14,14,12,14,14,13)
TNC = 19 TFC =172

Machine # : 43
NCC = ( 1, 1, 0, 1, 1, 1, 2, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1,
0, 0, 0, 0, 1, 0, 2, 0, 0, 1, 0, 1,13,12,11,12, 0,13,14,15,11,13,14,13)
TNC = 18 TFC =162

Machine # : 44
NCC = ( 1, 1, 0, 2, 0, 1, 2, 1, 0, 2, 1, 1, 0, 0, 1, 1, 2, 0, 0, 0, 0, 0, 0, 1, 0, 1,
0, 0, 0, 0, 1, 0, 2, 1, 0, 2, 1, 2,16,11,13,12,13, 0,13,14,12,15,15,12)
TNC = 19 TFC =173

Machine # : 45
NCC = ( 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 2, 1, 0, 1, 0, 0, 1, 1, 1, 2, 1, 0, 1,
1, 0, 0, 0, 2, 0, 2, 0, 0, 3, 0, 1,14,15,13,14,14,13, 0,14,12,14,15,14)
TNC = 21 TFC =178

Machine # : 46
NCC = ( 1, 1, 0, 1, 1, 1, 3, 0, 0, 2, 0, 0, 0, 1, 1, 3, 2, 1, 1, 1, 1, 0, 0, 1, 0, 1,
0, 1, 0, 0, 1, 0, 3, 1, 2, 4, 0, 1,17,13,13,14,15,14,14, 0,13,16,17,15)
TNC = 23 TFC =197

Machine # : 47
NCC = ( 2, 1, 0, 0, 0, 0, 2, 0, 1, 1, 0, 0, 0, 2, 0, 2, 2, 1, 1, 1, 1, 0, 0, 1, 0, 1,
0, 1, 0, 0, 1, 0, 1, 0, 1, 3, 0, 1,14,12,12,12,11,12,12,13, 0,14,13,13)
TNC = 17 TFC =165

Machine # : 48
NCC = ( 1, 2, 1, 1, 0, 1, 2, 0, 0, 2, 0, 0, 0, 2, 1, 2, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1,
0, 1, 0, 0, 1, 0, 2, 0, 1, 3, 0, 1,16,13,14,14,13,15,14,16,14, 0,16,13)
TNC = 20 TFC =185
```

Machine # : 49
NCC = (2, 1, 1, 1, 0, 1, 2, 0, 1, 1, 0, 0, 0, 3, 1, 2, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1,
0, 1, 1, 1, 2, 1, 2, 0, 0, 3, 0, 1,18,14,15,14,14,15,15,17,13,16,0,16)
TNC = 22 TFC =198

Machine # : 50
NCC = (2, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 3, 2, 2, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1,
0, 0, 0, 0, 1, 0, 2, 0, 0, 3, 0, 1,14,13,12,13,13,12,14,15,13,13,16,0)
TNC = 19 TFC =171

Similarity coefficient matrix:

:	.300	.455	.000	.000	.067	.000	.000	.000	.067	.000	.000	.000	.000	···
	:	.250	.000	.000	.071	.000	.000	.000	.071	.091	.000	.000	.071	···
		:	.000	.077	.059	.000	.000	.000	.059	.000	.000	.000	.059	···
			:	.444	.545	.333	.273	.400	.417	.556	.077	.000	.000	···
				:	.400	.300	.100	.375	.273	.375	.000	.000	.000	···
					:	.308	.250	.364	.500	.500	.071	.000	.059	···
						:	.308	.000	.167	.273	.308	.273	.077	···
							:	.200	.154	.333	.091	.000	.000	···
								:	.250	.500	.091	.000	.071	···
									:	.364	.071	.000	.059	···
										:	.000	.000	.000	···
											:	.429	.071	···
												:	.182	···
													:	⋱

S-3. The first candidate cell,

Cell-# : 01 Machines={ 37, 38, 36 }

S-4. Other candidate cells,

Cell-# : 02 Machines={ 33, 34, 35 }
Cell-# : 03 Machines={ 49, 50, 39, 41, 46, 43 }
Cell-# : 04 Machines={ 30, 32, 28, 29, 31 }
Cell-# : 05 Machines={ 44, 48, 42, 47 }
Cell-# : 06 Machines={ 17, 18, 19, 21, 20 }
Cell-# : 07 Machines={ 22, 23 }
Cell-# : 08 Machines={ 40, 45 }
Cell-# : 09 Machines={ 4, 11, 5, 6, 7, 8, 9, 10 }
Cell-# : 10 Machines={ 1, 3, 2 }
Cell-# : 11 Machines={ 12, 13, 15, 16 }

Cell-# : 12 Machines={ 24, 26, 25, 27 }

Cell-# : 13 Machines={ 14 }

S-5. Inter-cell flows,
Number of common parts (SNCC), total number of parts (STNC), total flow of common parts processed (STFC),

Cell-#01 SNCC = [0, 11, 6, 3, 5, 1, 1, 4, 3, 1, 2, 1, 3] STNC = 17 STFC = 41
Cell-#02 SNCC = [11, 0, 5, 5, 5, 6, 2, 2, 2, 2, 3, 1, 2] STNC = 20 STFC = 46
Cell-#03 SNCC = [6, 5, 0, 3, 25, 4, 0, 20, 9, 5, 6, 2, 4] STNC = 36 STFC = 89
Cell-#04 SNCC = [3, 5, 3, 0, 3, 5, 2, 3, 4, 2, 3, 3, 0] STNC = 20 STFC = 36
Cell-#05 SNCC = [5, 5, 25, 3, 0, 5, 0, 20, 8, 4, 4, 2, 3] STNC = 29 STFC = 84
Cell-#06 SNCC = [1, 6, 4, 5, 5, 0, 3, 2, 6, 4, 4, 3, 1] STNC = 28 STFC = 44
Cell-#07 SNCC = [1, 2, 0, 2, 0, 3, 0, 3, 4, 2, 3, 3, 2] STNC = 17 STFC = 25
Cell-#08 SNCC = [4, 2, 20, 3, 20, 2, 3, 0, 4, 3, 1, 3, 3] STNC = 26 STFC = 68
Cell-#09 SNCC = [3, 2, 9, 4, 8, 6, 4, 4, 0, 3, 4, 2, 2] STNC = 24 STFC = 51
Cell-#10 SNCC = [1, 2, 5, 2, 4, 4, 2, 3, 3, 0, 0, 1, 1] STNC = 14 STFC = 28
Cell-#11 SNCC = [2, 3, 6, 3, 4, 4, 3, 1, 4, 0, 0, 1, 3] STNC = 16 STFC = 34
Cell-#12 SNCC = [1, 1, 2, 3, 2, 3, 3, 3, 2, 1, 1, 0, 1] STNC = 15 STFC = 23
Cell-#13 SNCC = [3, 2, 4, 0, 3, 1, 2, 3, 2, 1, 3, 1, 0] STNC = 9 STFC = 25

Similarity coefficient matrix,

:	.064	.010	.006	.007	.001	.001	.006	.004	.001	.003	.001	.009
	:	.006	.015	.006	.018	.003	.001	.002	.003	.006	.001	.003
		:	.003	.084	.004	.000	.066	.018	.010	.012	.002	.007
			:	.003	.016	.004	.004	.009	.004	.007	.011	.000
				:	.007	.000	.070	.015	.007	.006	.002	.004
					:	.008	.001	.016	.013	.011	.009	.001
						:	.005	.013	.006	.011	.016	.006
							:	.005	.005	.000	.006	.005
								:	.006	.009	.003	.003
									:	.000	.002	.000
										:	.001	.011
											:	.002
												:

Part assignments,

Part-# 01 is assigned to cell-# 10.
Part-# 02 is assigned to cell-# 10.
Part-# 03 is assigned to cell-# 9.
Part-# 04 is assigned to cell-# 10.
Part-# 05 is assigned to cell-# 10.
Part-# 06 is assigned to cell-# 10.
Part-# 07 is assigned to cell-# 9.
Part-# 08 is assigned to cell-# 9.
Part-# 09 is assigned to cell-# 9.
Part-# 10 is assigned to cell-# 9.
Part-# 11 is assigned to cell-# 9.
Part-# 12 is assigned to cell-# 9.
Part-# 13 is assigned to cell-# 6.
Part-# 14 is assigned to cell-# 11.
Part-# 15 is assigned to cell-# 11.
Part-# 16 is assigned to cell-# 11.
Part-# 17 is assigned to cell-# 6.
Part-# 18 is assigned to cell-# 6.
Part-# 19 is assigned to cell-# 6.
Part-# 20 is assigned to cell-# 6.
Part-# 21 is assigned to cell-# 4.
Part-# 22 is assigned to cell-# 6.
Part-# 23 is assigned to cell-# 9.
Part-# 24 is assigned to cell-# 6.
Part-# 25 is assigned to cell-# 6.
Part-# 26 is assigned to cell-# 6.
Part-# 27 is assigned to cell-# 6.
Part-# 28 is assigned to cell-# 6.
Part-# 29 is assigned to cell-# 6.
Part-# 30 is assigned to cell-# 6.
Part-# 31 is assigned to cell-# 3.
Part-# 32 is assigned to cell-# 6.
Part-# 33 is assigned to cell-# 6.
Part-# 34 is assigned to cell-# 6.
Part-# 35 is assigned to cell-# 7.
Part-# 36 is assigned to cell-# 4.
Part-# 37 is assigned to cell-# 12.
Part-# 38 is assigned to cell-# 7.
Part-# 39 is assigned to cell-# 7.
Part-# 40 is assigned to cell-# 7.
Part-# 41 is assigned to cell-# 4.
Part-# 42 is assigned to cell-# 7.
Part-# 43 is assigned to cell-# 9.
Part-# 44 is assigned to cell-# 7.
Part-# 45 is assigned to cell-# 7.
Part-# 46 is assigned to cell-# 7.
Part-# 47 is assigned to cell-# 2.
Part-# 48 is assigned to cell-# 9.
Part-# 49 is assigned to cell-# 7.
Part-# 50 is assigned to cell-# 7.
Part-# 51 is assigned to cell-# 12.
Part-# 52 is assigned to cell-# 12.
Part-# 53 is assigned to cell-# 12.
Part-# 54 is assigned to cell-# 4.
Part-# 55 is assigned to cell-# 12.
Part-# 56 is assigned to cell-# 12.
Part-# 57 is assigned to cell-# 12.
Part-# 58 is assigned to cell-# 4.
Part-# 59 is assigned to cell-# 4.
Part-# 60 is assigned to cell-# 4.
Part-# 61 is assigned to cell-# 4.
Part-# 62 is assigned to cell-# 4.
Part-# 63 is assigned to cell-# 4.
Part-# 64 is assigned to cell-# 4.
Part-# 65 is assigned to cell-# 4.
Part-# 66 is assigned to cell-# 2.
Part-# 67 is assigned to cell-# 2.
Part-# 68 is assigned to cell-# 2.
Part-# 69 is assigned to cell-# 1.
Part-# 70 is assigned to cell-# 1.
Part-# 71 is assigned to cell-# 1.
Part-# 72 is assigned to cell-# 2.
Part-# 73 is assigned to cell-# 9.
Part-# 74 is assigned to cell-# 4.
Part-# 75 is assigned to cell-# 1.
Part-# 76 is assigned to cell-# 2.
Part-# 77 is assigned to cell-# 6.
Part-# 78 is assigned to cell-# 5.
Part-# 79 is assigned to cell-# 3.
Part-# 80 is assigned to cell-# 5.
Part-# 81 is assigned to cell-# 5.
Part-# 82 is assigned to cell-# 5.
Part-# 83 is assigned to cell-# 3.
Part-# 84 is assigned to cell-# 5.

Part-# 85 is assigned to cell-# 3.
Part-# 86 is assigned to cell-# 3.
Part-# 87 is assigned to cell-# 5.
Part-# 88 is assigned to cell-# 5.
Part-# 89 is assigned to cell-# 5.
Part-# 90 is assigned to cell-# 5.
Part-# 91 is assigned to cell-# 5.
Part-# 92 is assigned to cell-# 5.
Part-# 93 is assigned to cell-# 3.
Part-# 94 is assigned to cell-# 5.
Part-# 95 is assigned to cell-# 5.
Part-# 96 is assigned to cell-# 3.
Part-# 97 is assigned to cell-# 5.
Part-# 98 is assigned to cell-# 3.
Part-# 99 is assigned to cell-# 5.
Part-#100 is assigned to cell-# 5.

S-6. Final cells,

CELL-#: 1
Machines={ 37; 38; 36 }
Parts={ 69; 70; 71; 75 }

CELL-#: 2
Machines={ 33; 34; 35 }
Parts={ 47; 66; 67; 68; 72; 76 }

CELL-#: 3
Machines={ 49; 50; 39; 41; 46; 43 }
Parts={ 31; 79; 83; 85; 86; 93; 96; 98 }

CELL-#: 4
Machines={ 30; 32; 28; 29; 31 }
Parts={ 21; 36; 41; 54; 58; 59; 60; 61; 62; 63; 64; 65; 74 }

CELL-#: 5
Machines={ 44; 48; 42; 47 }
Parts={ 78; 80; 81; 82; 84; 87; 88; 89; 90; 91; 92; 94; 95; 97; 99;100 }

CELL-#: 6
Machines={ 17; 18; 19; 21; 20 }
Parts={ 13; 17; 18; 19; 20; 22; 24; 25; 26; 27; 28; 29; 30; 32; 33; 34; 77 }

CELL-#: 7
Machines={ 22; 23 }
Parts={ 35; 38; 39; 40; 42; 44; 45; 46; 49; 50 }

CELL-#: 8
Machines={ 40; 45 }

CELL-#: 9

Machines={ 4; 11; 5; 6; 7; 8; 9; 10 }

Parts={ 3; 7; 8; 9; 10; 11; 12; 23; 43; 48; 73 }

CELL-#: 10

Machines={ 1; 3; 2 }

Parts={ 1; 2; 4; 5; 6 }

CELL-#: 11

Machines={ 12; 13; 15; 16 }

Parts={ 14; 15; 16 }

CELL-#: 12

Machines={ 24; 26; 25; 27 }

Parts={ 37; 51; 52; 53; 55; 56; 57 }

CELL-#: 13

Machines={ 14 }

Blocking machines and reassignments,

Cell # 8 has no parts assigned.

Blocking machine # 40 is assigned to cell # 5.

Blocking machine # 40 is assigned to cell # 3.

Blocking machine # 45 is assigned to cell # 5.

Blocking machine # 45 is assigned to cell # 3.

Blocking machine # 45 is assigned to cell # 10.

Cell # 13 has no parts assigned.

Blocking machine # 14 is assigned to cell # 10.

S-7. Efficiency measures, Inter-cell flow: 69.95 %,
Inner-cell density: 75.74 %,
Work-load balance: 82.39 %,
Under-utilization: 62.27 %.

WUBC

S-1. Cell admission factor (CAF) : 0.60,
Cell size upper limit: 50,
Key machine type selection rule: A4,
Part assignment rule: B2.

S-2. Key machine selection,

Key machine type is 42.

Machine type 42 is inserted into the FCFS queue.

S-3. Examine all parts routed through the key machine type,

Part 28 is examined, and it is admitted to the cell.

Part 30 is examined, and it is admitted to the cell.

Part 78 is examined, and it is admitted to the cell.

Part 80 is examined, and it is admitted to the cell.

Part 82 is examined, and it is admitted to the cell.

Part 83 is examined, and it is admitted to the cell.

Part 84 is examined, and it is admitted to the cell.

Part 85 is examined, and it is admitted to the cell.

Part 87 is examined, and it is admitted to the cell.

Part 88 is examined, and it is admitted to the cell.

Part 89 is examined, and it is admitted to the cell.

Part 90 is examined, and it is admitted to the cell.

Part 91 is examined, and it is admitted to the cell.

Part 92 is examined, and it is admitted to the cell.

Part 94 is examined, and it is admitted to the cell.

Part 95 is examined, and it is admitted to the cell.

Part 97 is examined, and it is admitted to the cell.

Part 99 is examined, and it is admitted to the cell.

Part 100 is examined, and it is admitted to the cell.

S-4. Evaluate non-key machines,

Machine type 09 is rejected from the cell because of WLF= .36101.

Machine type 33 is rejected from the cell because of WLF= .34387.

Machine type 07 is rejected from the cell because of WLF= .22862.

Machine type 24 is rejected from the cell because of WLF= .20195.

Machine type 15 is rejected from the cell because of WLF= .13427.

Machine type 38 is rejected from the cell because of WLF= .12230.

Machine type 02 is rejected from the cell because of WLF= .10615.

Machine type 39 is rejected from the cell because of WLF= .09059.

Machine type 10 is rejected from the cell because of WLF= .06614.

Machine type 40 is rejected from the cell because of WLF= .04724.

Machine type 06 is rejected from the cell because of WLF= .03110.

Machine type 26 is rejected from the cell because of WLF= .03053.

Machine type 14 is rejected from the cell because of WLF= .02933.

Machine type 17 is rejected from the cell because of WLF= .00654.

Machine type 16 is rejected from the cell because of WLF= .00485.

S-5. FCFS queue update,

Machine type 42 is deleted from the FCFS queue.

The FCFS queue is empty.

S-6. Release cell,

Cell 1 is discarded since it contains only the key machine which is prevented from being a key in the next iterations.

Go to S-2.

S-2. Key machine selection,

New key machine type is 49.

Machine type 49 is inserted into the FCFS queue.

S-3. Examine all parts routed through the key machine type,

Part 03 is examined, and it is admitted to the cell.

Part 65 is examined, and it is admitted to the cell.

Part 77 is examined, and it is admitted to the cell.

Part 79 is examined, and it is admitted to the cell.

Part 80 is examined, and it is admitted to the cell.

Part 81 is examined, and it is admitted to the cell.

Part 82 is examined, and it is admitted to the cell.

Part 83 is examined, and it is admitted to the cell.

Part 84 is examined, and it is admitted to the cell.

Part 85 is examined, and it is admitted to the cell.

Part 86 is examined, and it is admitted to the cell.

Part 87 is examined, and it is admitted to the cell.

Part 88 is examined, and it is admitted to the cell.

Part 90 is examined, and it is admitted to the cell.

Part 91 is examined, and it is admitted to the cell.

Part 92 is examined, and it is admitted to the cell.

Part 94 is examined, and it is admitted to the cell.

Part 96 is examined, and it is admitted to the cell.

Part 97 is examined, and it is admitted to the cell.

Part 98 is examined, and it is admitted to the cell.

Part 99 is examined, and it is admitted to the cell.

Part 100 is examined, and it is admitted to the cell.

S-4. Evaluate non-key machines,

Machine type 01 is admitted to the cell with WLF .66840.

Machine type 09 is rejected from the cell because of WLF= .36101.

Machine type 33 is rejected from the cell because of WLF= .34387.

Machine type 16 is rejected from the cell because of WLF= .32414.

Machine type 07 is rejected from the cell because of WLF= .29859.

Machine type 24 is rejected from the cell because of WLF= .20195.

Machine type 04 is rejected from the cell because of WLF= .14134.

Machine type 15 is rejected from the cell because of WLF= .13427.

Machine type 38 is rejected from the cell because of WLF= .12230.

Machine type 40 is rejected from the cell because of WLF= .10756.

Machine type 39 is rejected from the cell because of WLF= .09059.

Machine type 26 is rejected from the cell because of WLF= .03053.

Machine type 14 is rejected from the cell because of WLF= .02933.

Machine type 36 is rejected from the cell because of WLF= .01971.

S-5. FCFS queue update,

Machine type 01 is inserted into the FCFS queue.

Machine type 49 is deleted from the FCFS queue.

Go to S-3.

S-3. Examine all parts routed through the key machine type,

Part 01 is examined, and it is admitted to the cell.

Part 02 is examined, and it is admitted to the cell.

Part 03 is examined, and it remained in its previously assigned cell.

Part 04 is examined, and it is admitted to the cell.

Part 05 is examined, and it is admitted to the cell.

Part 34 is examined, and it is admitted to the cell.

Part 86 is examined, and it remained in its previously assigned cell.

S-5. FCFS queue update,

Machine type 1 is deleted from the FCFS queue.

The FCFS queue is empty.

S-6. *Go to* S-3.

S-7. Form the cell,

The key machine type 49 is assigned to the cell with WCU of 4.11164 .

Machine type 1 is assigned to the cell with WCU of .42579 .

S-8. Part assignments,

Part # 03 is assigned to cell # 1 .

Part # 65 is assigned to cell # 1 .

Part # 77 is assigned to cell # 1 .

Part # 79 is assigned to cell # 1 .

Part # 80 is assigned to cell # 1 .

Part # 81 is assigned to cell # 1 .

Part # 82 is assigned to cell # 1 .

Part # 83 is assigned to cell # 1 .

Part # 84 is assigned to cell # 1 .

Part # 85 is assigned to cell # 1 .

Part # 86 is assigned to cell # 1 .

Part # 87 is assigned to cell # 1 .

Part # 88 is assigned to cell # 1 .

Part # 90 is assigned to cell # 1 .

Part # 91 is assigned to cell # 1 .

Part # 92 is assigned to cell # 1 .

Part # 94 is assigned to cell # 1 .

Part # 96 is assigned to cell # 1 .

Part # 97 is assigned to cell # 1 .

Part # 98 is assigned to cell # 1 .

Part # 99 is assigned to cell # 1 .

Part # 100 is assigned to cell # 1 .

Part # 01 is assigned to cell # 1 .

Part # 02 is assigned to cell # 1 .

Part # 04 is assigned to cell # 1 .

Part # 05 is assigned to cell # 1 .

Part # 34 is assigned to cell # 1 .

S-* Iteration,

New key machine type is 6.

Machine type 6 is inserted into the FCFS queue.

Part 03 is examined, and it is admitted to the cell.

Part 07 is examined, and it is admitted to the cell.

Part 08 is examined, and it is admitted to the cell.

Part 09 is examined, and it is admitted to the cell.

Part 10 is examined, and it is admitted to the cell.

Part 11 is examined, and it is admitted to the cell.

Part 12 is examined, and it is admitted to the cell.

Part 28 is examined, and it is admitted to the cell.

Part 66 is examined, and it is admitted to the cell.

Machine type 4 is rejected from the cell because of WLF= .52458.

Machine type 6 is deleted from the FCFS queue.

The FCFS queue is empty.

Cell 2 is discarded since it contains only the key machine which is prevented from being a key in the next iterations.

⋮

S-* Iteration,

New key machine type is 9.

Machine type 9 is inserted into the FCFS queue.

Part 07 is examined, and it remained in its previously assigned cell.

Part 08 is examined, and it is admitted to the cell.

Part 10 is examined, and it is admitted to the cell.

Part 11 is examined, and it is admitted to the cell.

Part 62 is examined, and it remained in its previously assigned cell.

Part 82 is examined, and it is admitted to the cell.

Machine type 4 is rejected from the cell because of WLF= .36560.

Machine type 9 is deleted from the FCFS queue.

The FCFS queue is empty.

Cell 3 is discarded since it contains only the key machine which is prevented from being a key in the next iterations.

Go to S-2.

S-2. Key machine selection,

No more key machines

Go to S-10.

S-10. Add all left-over machines into the remainder cell,

Machine type 03 is assigned to the remainder cell.

Machine type 04 is assigned to the remainder cell.

Machine type 05 is assigned to the remainder cell.

Machine type 06 is assigned to the remainder cell.

Machine type 07 is assigned to the remainder cell.

Machine type 08 is assigned to the remainder cell.

Machine type 09 is assigned to the remainder cell.

Machine type 10 is assigned to the remainder cell.

Machine type 11 is assigned to the remainder cell.

Machine type 13 is assigned to the remainder cell.

Machine type 14 is assigned to the remainder cell.

Machine type 15 is assigned to the remainder cell.

Machine type 16 is assigned to the remainder cell.

Machine type 17 is assigned to the remainder cell.

Machine type 18 is assigned to the remainder cell.

Machine type 19 is assigned to the remainder cell.

Machine type 20 is assigned to the remainder cell.

Machine type 21 is assigned to the remainder cell.

Machine type 22 is assigned to the remainder cell.

Machine type 23 is assigned to the remainder cell.

Machine type 24 is assigned to the remainder cell.

Machine type 25 is assigned to the remainder cell.

Machine type 26 is assigned to the remainder cell.

Machine type 27 is assigned to the remainder cell.

Machine type 28 is assigned to the remainder cell.

Machine type 29 is assigned to the remainder cell.

Machine type 30 is assigned to the remainder cell.

Machine type 31 is assigned to the remainder cell.

Machine type 32 is assigned to the remainder cell.

Machine type 33 is assigned to the remainder cell.

Machine type 34 is assigned to the remainder cell.

Machine type 35 is assigned to the remainder cell.

Machine type 36 is assigned to the remainder cell.

Machine type 37 is assigned to the remainder cell.

Machine type 38 is assigned to the remainder cell.

Machine type 39 is assigned to the remainder cell.

Machine type 40 is assigned to the remainder cell.

Machine type 41 is assigned to the remainder cell.

Machine type 42 is assigned to the remainder cell.

Machine type 43 is assigned to the remainder cell.

Machine type 44 is assigned to the remainder cell.

Machine type 45 is assigned to the remainder cell.

Machine type 46 is assigned to the remainder cell.

Machine type 47 is assigned to the remainder cell.

Machine type 48 is assigned to the remainder cell.

Machine type 49 is assigned to the remainder cell.

Machine type 50 is assigned to the remainder cell.

Examine parts for possible reassignment to the remainder cell;

Part # 01 is assigned to the remainder cell.

Part # 02 is assigned to the remainder cell.

Part # 03 is assigned to the remainder cell.

Part # 04 is assigned to the remainder cell.

Part # 06 is assigned to the remainder cell.

Part # 07 is assigned to the remainder cell.

Part # 08 is assigned to the remainder cell.

Part # 09 is assigned to the remainder cell.

Part # 10 is assigned to the remainder cell.

Part # 11 is assigned to the remainder cell.

Part # 12 is assigned to the remainder cell.

Part # 14 is assigned to the remainder cell.

Part # 15 is assigned to the remainder cell.

Part # 16 is assigned to the remainder cell.

Part # 17 is assigned to the remainder cell.

Part # 18 is assigned to the remainder cell.

Part # 19 is assigned to the remainder cell.

Part # 20 is assigned to the remainder cell.

Part # 21 is assigned to the remainder cell.

Part # 22 is assigned to the remainder cell.

Part # 23 is assigned to the remainder cell.

Part # 24 is assigned to the remainder cell.

Part # 25 is assigned to the remainder cell.

Part # 26 is assigned to the remainder cell.

Part # 27 is assigned to the remainder cell.

Part # 28 is assigned to the remainder cell.

Part # 29 is assigned to the remainder cell.

Part # 30 is assigned to the remainder cell.

Part # 31 is assigned to the remainder cell.

Part # 32 is assigned to the remainder cell.

Part # 33 is assigned to the remainder cell.

Part # 34 is assigned to the remainder cell.

Part # 35 is assigned to the remainder cell.

Part # 36 is assigned to the remainder cell.

Part # 37 is assigned to the remainder cell.

Part # 38 is assigned to the remainder cell.

Part # 39 is assigned to the remainder cell.

Part # 40 is assigned to the remainder cell.

Part # 41 is assigned to the remainder cell.

Part # 42 is assigned to the remainder cell.

Part # 43 is assigned to the remainder cell.

Part # 44 is assigned to the remainder cell.

Part # 45 is assigned to the remainder cell.

Part # 46 is assigned to the remainder cell.

```
Part # 47 is assigned to the remainder cell.
Part # 48 is assigned to the remainder cell.
Part # 49 is assigned to the remainder cell.
Part # 50 is assigned to the remainder cell.
Part # 51 is assigned to the remainder cell.
Part # 52 is assigned to the remainder cell.
Part # 53 is assigned to the remainder cell.
Part # 54 is assigned to the remainder cell.
Part # 55 is assigned to the remainder cell.
Part # 56 is assigned to the remainder cell.
Part # 57 is assigned to the remainder cell.
Part # 58 is assigned to the remainder cell.
Part # 59 is assigned to the remainder cell.
Part # 60 is assigned to the remainder cell.
Part # 61 is assigned to the remainder cell.
Part # 62 is assigned to the remainder cell.
Part # 63 is assigned to the remainder cell.
Part # 64 is assigned to the remainder cell.
Part # 65 is assigned to the remainder cell.
Part # 66 is assigned to the remainder cell.
Part # 67 is assigned to the remainder cell.
Part # 68 is assigned to the remainder cell.
Part # 69 is assigned to the remainder cell.
Part # 70 is assigned to the remainder cell.
Part # 71 is assigned to the remainder cell.
Part # 72 is assigned to the remainder cell.
Part # 74 is assigned to the remainder cell.
Part # 75 is assigned to the remainder cell.
Part # 76 is assigned to the remainder cell.
Part # 77 is assigned to the remainder cell.
Part # 78 is assigned to the remainder cell.
Part # 79 is assigned to the remainder cell.
Part # 80 is assigned to the remainder cell.
Part # 81 is assigned to the remainder cell.
Part # 82 is assigned to the remainder cell.
Part # 83 is assigned to the remainder cell.
```

Part # 84 is assigned to the remainder cell.

Part # 85 is assigned to the remainder cell.

Part # 86 is assigned to the remainder cell.

Part # 87 is assigned to the remainder cell.

Part # 88 is assigned to the remainder cell.

Part # 89 is assigned to the remainder cell.

Part # 90 is assigned to the remainder cell.

Part # 91 is assigned to the remainder cell.

Part # 92 is assigned to the remainder cell.

Part # 93 is assigned to the remainder cell.

Part # 94 is assigned to the remainder cell.

Part # 95 is assigned to the remainder cell.

Part # 96 is assigned to the remainder cell.

Part # 97 is assigned to the remainder cell.

Part # 98 is assigned to the remainder cell.

Part # 99 is assigned to the remainder cell.

Part #100 is assigned to the remainder cell.

S-11. Update other cells,

No machine release

S-12. Solution,

CELL # : 1
MACHINES = { 16, 12 }
PARTS = { 13, 73 }

CELL # : 2
MACHINES = { 2, 1 }
PARTS = { 5 }

CELL # : 3 (remainder cell)

MACHINES = { 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50 }

PARTS = { 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100 }

Efficiency measures, Inter-cell flow: 97.67 % ,
Inner-cell density: 19.03 %,
Work-load balance: 83.28 % ,
Under-utilization: 57.16 % .

CAA

S-1. Cell admission factor: 0.20,
Extra factor on cell size limit: 0,
Cell size upper limit: 14.

S-2. Key part is 77.

Machine type 16 is added into the cell.
Machine type 17 is added into the cell.
Machine type 39 is added into the cell.
Machine type 43 is added into the cell.
Machine type 44 is added into the cell.
Machine type 46 is added into the cell.
Machine type 47 is added into the cell.
Machine type 48 is added into the cell.
Machine type 49 is added into the cell.
Machine type 50 is added into the cell.

S-3. Search for candidate parts,

Part # 01 is examined and not rejected but it cannot be a candidate because of its value (3.000).
Part # 03 is examined and rejected since it leads an increase in the cell size of 2 machine types.
Part # 04 is examined and not rejected but it cannot be a candidate because of its value (2.000).
Part # 07 is examined and rejected since it leads an increase in the cell size of 3 machine types.
Part # 08 is examined and rejected since it leads an increase in the cell size of 2 machine types.
Part # 09 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 12 is examined and rejected since it leads an increase in the cell size of 2 machine types.
Part # 13 is examined and not rejected but it cannot be a candidate because of its value (3.000).
Part # 14 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 15 is examined and not rejected but it cannot be a candidate because of its value (1.000).
Part # 16 is examined and not rejected but it cannot be a candidate because of its value (1.500).
Part # 17 is examined and not rejected but it cannot be a candidate because of its value (3.000).
Part # 19 is examined and not rejected but it cannot be a candidate because of its value (2.000).
Part # 20 is examined and not rejected but it cannot be a candidate because of its value (4.000).
Part # 22 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 23 is examined and not rejected but it cannot be a candidate because of its value (4.000).
Part # 24 is examined and not rejected but it cannot be a candidate because of its value (3.000).

```
Part # 25 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 26 is examined and not rejected but it cannot be a candidate because of its value ( 4.000).
Part # 28 is examined and not rejected but it cannot be a candidate because of its value ( 4.000).
Part # 29 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 30 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 31 is examined and not rejected but it cannot be a candidate because of its value ( 2.000).
Part # 32 is examined and not rejected but it cannot be a candidate because of its value ( 4.000).
Part # 33 is examined and not rejected but it cannot be a candidate because of its value ( 3.000).
Part # 34 is examined and rejected since it leads an increase in the cell size of 2 machine types.
Part # 45 is examined and not rejected but it cannot be a candidate because of its value ( 1.000).
Part # 65 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 71 is examined and not rejected but it cannot be a candidate because of its value ( 2.000).
Part # 72 is examined and not rejected but it cannot be a candidate because of its value ( 1.333).
Part # 78 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 79 is examined and not rejected but it cannot be a candidate because of its value ( .250).
Part # 80 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 81 is examined and not rejected but it cannot be a candidate because of its value ( .286).
Part # 82 is examined and rejected since it leads an increase in the cell size of 2 machine types.
Part # 83 is examined and not rejected but it cannot be a candidate because of its value ( .571).
Part # 84 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 85 is examined and not rejected but it cannot be a candidate because of its value ( .400).
Part # 86 is examined and not rejected but it cannot be a candidate because of its value ( .429).
Part # 87 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 88 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 89 is examined and not rejected but it cannot be a candidate because of its value ( .800).
Part # 90 is examined and not rejected but it cannot be a candidate because of its value ( 1.000).
Part # 91 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 92 is examined and not rejected but it cannot be a candidate because of its value ( .600).
Part # 93 is examined and not rejected but it cannot be a candidate because of its value ( .667).
Part # 94 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 95 is examined and not rejected but it cannot be a candidate because of its value ( .571).
Part # 96 is examined and not rejected but it cannot be a candidate because of its value ( .571).
Part # 97 is examined and not rejected but it cannot be a candidate because of its value ( .571).
Part # 98 is examined and not rejected but it cannot be a candidate because of its value ( 1.000).
Part # 99 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part #100 is examined and not rejected but it cannot be a candidate because of its value ( .429).
```

Go to S-5.

S-5. *Go to* S-2.

S-2. Key part is 25.

Machine type 17 is added into the cell.
Machine type 18 is added into the cell.
Machine type 19 is added into the cell.
Machine type 20 is added into the cell.
Machine type 21 is added into the cell.
Machine type 41 is added into the cell.

Search for candidate parts,

Part # 17 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 19 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 24 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 29 is examined and it becomes a candidate part with value .200.

Part # 31 is examined and it becomes a candidate part with value .200.

Part # 33 is examined and automatically assigned to the cell since it leads no increase in the cell size.

S-4. Cell expansion,

Part # 29 is assigned to the cell.
Machine # 3 is added into the cell.

Go to S-3.

S-3. Search for further candidate parts,

Part # 31 is examined and it becomes a candidate part with value .200.

S-4. Cell expansion,

Part # 31 is assigned to the cell.
Machine # 39 is added into the cell.

Go to S-3.

S-3. Search for further candidate parts,

Part # 01 is examined and not rejected but it cannot be a candidate because of its value (1.000).

Part # 02 is examined and not rejected but it cannot be a candidate because of its value (1.000).

Part # 03 is examined and rejected since it leads an increase in the cell size of 1 machine types.

Part # 04 is examined and not rejected but it cannot be a candidate because of its value (2.000).

Part # 05 is examined and not rejected but it cannot be a candidate because of its value (3.000).

Part # 06 is examined and not rejected but it cannot be a candidate because of its value (2.000).

Part # 08 is examined and rejected since it leads an increase in the cell size of 1 machine types.

Part # 12 is examined and rejected since it leads an increase in the cell size of 1 machine types.

Part # 13 is examined and not rejected but it cannot be a candidate because of its value (3.000).

Part # 16 is examined and not rejected but it cannot be a candidate because of its value (4.000).

Part # 18 is examined and not rejected but it cannot be a candidate because of its value (.250).

Part # 20 is examined and not rejected but it cannot be a candidate because of its value (.250).

Part # 21 is examined and not rejected but it cannot be a candidate because of its value (.333).

Part # 22 is examined and not rejected but it cannot be a candidate because of its value (.400).

Part # 23 is examined and not rejected but it cannot be a candidate because of its value (.250).

Part # 26 is examined and not rejected but it cannot be a candidate because of its value (.667).

Part # 27 is examined and not rejected but it cannot be a candidate because of its value (1.500).

Part # 28 is examined and not rejected but it cannot be a candidate because of its value (.667).

Part # 30 is examined and not rejected but it cannot be a candidate because of its value (1.000).

Part # 32 is examined and not rejected but it cannot be a candidate because of its value (.667).

Part # 34 is examined and not rejected but it cannot be a candidate because of its value (.600).

Part # 36 is examined and not rejected but it cannot be a candidate because of its value (4.000).

Part # 43 is examined and not rejected but it cannot be a candidate because of its value (4.000).

Part # 47 is examined and not rejected but it cannot be a candidate because of its value (2.000).

Part # 51 is examined and not rejected but it cannot be a candidate because of its value (3.000).

Part # 53 is examined and not rejected but it cannot be a candidate because of its value (4.000).

Part # 58 is examined and not rejected but it cannot be a candidate because of its value (4.000).

Part # 65 is examined and rejected since it leads an increase in the cell size of 1 machine types.

Part # 68 is examined and not rejected but it cannot be a candidate because of its value (6.000).

Part # 71 is examined and not rejected but it cannot be a candidate because of its value (5.000).

Part # 72 is examined and not rejected but it cannot be a candidate because of its value (6.000).

Part # 78 is examined and rejected since it leads an increase in the cell size of 2 machine types.

Part # 80 is examined and rejected since it leads an increase in the cell size of 4 machine types.

Part # 81 is examined and rejected since it leads an increase in the cell size of 1 machine types.

Part # 82 is examined and rejected since it leads an increase in the cell size of 2 machine types.

```
Part # 83 is examined and rejected since it leads an increase in the cell size of 3 machine types.
Part # 84 is examined and rejected since it leads an increase in the cell size of 5 machine types.
Part # 85 is examined and not rejected but it cannot be a candidate because of its value ( 2.500).
Part # 86 is examined and rejected since it leads an increase in the cell size of 2 machine types.
Part # 87 is examined and rejected since it leads an increase in the cell size of 5 machine types.
Part # 88 is examined and rejected since it leads an increase in the cell size of 5 machine types.
Part # 89 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 90 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 91 is examined and rejected since it leads an increase in the cell size of 4 machine types.
Part # 92 is examined and rejected since it leads an increase in the cell size of 1 machine types.
Part # 94 is examined and rejected since it leads an increase in the cell size of 4 machine types.
Part # 95 is examined and rejected since it leads an increase in the cell size of 4 machine types.
Part # 96 is examined and rejected since it leads an increase in the cell size of 3 machine types.
Part # 97 is examined and rejected since it leads an increase in the cell size of 3 machine types.
Part # 98 is examined and not rejected but it cannot be a candidate because of its value ( 3.000).
Part # 99 is examined and rejected since it leads an increase in the cell size of 6 machine types.
Part #100 is examined and rejected since it leads an increase in the cell size of 2 machine types.
```

Go to S-5.

S-5. *Go to* S-2.

S-2. Key part is 87.

Machine type 38 is added into the cell.
Machine type 39 is added into the cell.
Machine type 40 is added into the cell.
Machine type 41 is added into the cell.
Machine type 42 is added into the cell.
Machine type 43 is added into the cell.
Machine type 44 is added into the cell.
Machine type 45 is added into the cell.
Machine type 46 is added into the cell.
Machine type 47 is added into the cell.
Machine type 48 is added into the cell.
Machine type 49 is added into the cell.
Machine type 50 is added into the cell.

Search for candidate parts,

Part # 78 is examined and it becomes a candidate part with value .125.

Part # 79 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 80 is examined and it becomes a candidate part with value .091.

Part # 81 is examined and it becomes a candidate part with value .125.

Part # 83 is examined and it becomes a candidate part with value .100.

Part # 84 is examined and it becomes a candidate part with value .091.

Part # 85 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 86 is examined and it becomes a candidate part with value .111.

Part # 90 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 91 is examined and it becomes a candidate part with value .091.

Part # 92 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 93 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 95 is examined and it becomes a candidate part with value .100.

Part # 96 is examined and it becomes a candidate part with value .100.

Part # 97 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 98 is examined and it becomes a candidate part with value .143.

Part #100 is examined and automatically assigned to the cell since it leads no increase in the cell size.

S-4. Cell expansion,

Part # 81 is assigned to the cell.

Part # 84 is automatically assigned to the cell without increasing the cell size.

Machine # 7 is added into the cell.

Go to S-5.

S-5. *Go to* S-2.

S-* Key part is 7.

Machine type 4 is added into the cell.

Machine type 6 is added into the cell.

Machine type 7 is added into the cell.

Machine type 8 is added into the cell.

Machine type 9 is added into the cell.

Machine type 10 is added into the cell.

Machine type 11 is added into the cell.

Machine type 16 is added into the cell.

Part # 11 is examined and it becomes a candidate part with value .167.

Part # 11 is assigned to the cell.

Machine # 5 is added into the cell.

S-* Key part is 54.

Machine type 25 is added into the cell.

Machine type 26 is added into the cell.

Machine type 27 is added into the cell.

Machine type 29 is added into the cell.

S-* Key part is 61.

Machine type 28 is added into the cell.

Machine type 29 is added into the cell.

Machine type 30 is added into the cell.

Machine type 31 is added into the cell.

Machine type 32 is added into the cell.

Part # 59 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 63 is examined and it becomes a candidate part with value .200.

Part # 64 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 63 is assigned to the cell.

Machine # 40 is added into the cell.

S-* Key part is 40.

Machine type 15 is added into the cell.

Machine type 22 is added into the cell.

Machine type 23 is added into the cell.

Part # 35 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 39 is examined and automatically assigned to the cell since it leads no increase in the cell size.

Part # 42 is examined and automatically assigned to the cell since it leads no increase in the cell size.

S-* Key part is 38.

Machine type 22 is added into the cell.
Machine type 23 is added into the cell.
Machine type 45 is added into the cell.

S-* Key part is 57.

Machine type 24 is added into the cell.

S-6. Rejected parts and assignments,

Part 01 is subcontracted, or assigned to cell# 2.
Part 02 is subcontracted, or assigned to cell# 2.
Part 03 is subcontracted, or assigned to cell# 4.
Part 04 is subcontracted, or assigned to cell# 2.
Part 05 is subcontracted, or assigned to cell# 2.
Part 06 is subcontracted, or assigned to cell# 2.
Part 08 is subcontracted, or assigned to cell# 4.
Part 09 is subcontracted, or assigned to cell# 4.
Part 10 is subcontracted, or assigned to cell# 4.
Part 12 is subcontracted, or assigned to cell# 4.
Part 13 is subcontracted, or assigned to cell# 2.
Part 14 is subcontracted, or assigned to cell# 5.
Part 15 is subcontracted, or assigned to cell# 1.
Part 16 is subcontracted, or assigned to cell# 1.
Part 18 is subcontracted, or assigned to cell# 2.
Part 20 is subcontracted, or assigned to cell# 2.
Part 21 is subcontracted, or assigned to cell# 2.
Part 22 is subcontracted, or assigned to cell# 2.
Part 23 is subcontracted, or assigned to cell# 4.
Part 26 is subcontracted, or assigned to cell# 2.
Part 27 is subcontracted, or assigned to cell# 2.

Part 28 is subcontracted, or assigned to cell# 2.

Part 30 is subcontracted, or assigned to cell# 2.

Part 32 is subcontracted, or assigned to cell# 2.

Part 34 is subcontracted, or assigned to cell# 2.

Part 36 is subcontracted, or assigned to cell# 7.

Part 37 is subcontracted, or assigned to cell# 8.

Part 41 is subcontracted, or assigned to cell# 7.

Part 43 is subcontracted, or assigned to cell# 4.

Part 44 is subcontracted, or assigned to cell# 7.

Part 45 is subcontracted, or assigned to cell# 7.

Part 46 is subcontracted, or assigned to cell# 7.

Part 47 is subcontracted, or assigned to cell# 2.

Part 48 is subcontracted, or assigned to cell# 4.

Part 49 is subcontracted, or assigned to cell# 7.

Part 50 is subcontracted, or assigned to cell# 7.

Part 51 is subcontracted, or assigned to cell# 5.

Part 52 is subcontracted, or assigned to cell# 5.

Part 53 is subcontracted, or assigned to cell# 5.

Part 55 is subcontracted, or assigned to cell# 5.

Part 56 is subcontracted, or assigned to cell# 5.

Part 58 is subcontracted, or assigned to cell# 6.

Part 60 is subcontracted, or assigned to cell# 6.

Part 62 is subcontracted, or assigned to cell# 6.

Part 65 is subcontracted, or assigned to cell# 6.

Part 66 is subcontracted, or assigned to cell# 4.

Part 67 is subcontracted, or assigned to cell# 3.

Part 68 is subcontracted, or assigned to cell# 3.

Part 69 is subcontracted, or assigned to cell# 6.

Part 70 is subcontracted, or assigned to cell# 3.

Part 71 is subcontracted, or assigned to cell# 3.

Part 72 is subcontracted, or assigned to cell# 1.

Part 73 is subcontracted, or assigned to cell# 3.

Part 74 is subcontracted, or assigned to cell# 5.

Part 75 is subcontracted, or assigned to cell# 3.

Part 76 is subcontracted, or assigned to cell# 6.

Part 78 is subcontracted, or assigned to cell# 3.

Part 80 is subcontracted, or assigned to cell# 3.

Part 82 is subcontracted, or assigned to cell# 3.

Part 83 is subcontracted, or assigned to cell# 3.

Part 86 is subcontracted, or assigned to cell# 3.

Part 88 is subcontracted, or assigned to cell# 3.

Part 89 is subcontracted, or assigned to cell# 3.

Part 91 is subcontracted, or assigned to cell# 3.

Part 94 is subcontracted, or assigned to cell# 3.

Part 95 is subcontracted, or assigned to cell# 3.

Part 96 is subcontracted, or assigned to cell# 3.

Part 98 is subcontracted, or assigned to cell# 3.

Part 99 is subcontracted, or assigned to cell# 3.

S-7. Solution,

CELL # : 1
MACHINES = { 16, 39, 43, 44, 46, 47, 48, 49, 50 }
PARTS = { 77, 16 }

CELL # : 2
MACHINES = { 17, 18, 19, 20, 21, 41, 3, 39 }
PARTS = { 25, 17, 19, 24, 33, 29, 31, 1, 2, 4, 5, 6, 13, 18, 20, 21, 22, 26, 27, 28, 30, 32, 34, 36, 47 }

CELL # : 3
MACHINES = { 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 7 }
PARTS = { 87, 79, 85, 90, 92, 93, 97, 100, 81, 84, 67, 68, 70, 71, 73, 75, 78, 80, 82, 83, 86, 88, 89, 91, 94, 95, 96, 98, 99 }

CELL # : 4
MACHINES = { 4, 6, 7, 8, 9, 10, 11, 16, 5 }
PARTS = { 7, 11, 3, 8, 9, 10, 12, 23, 43, 48, 66 }

CELL # : 5
MACHINES = { 25, 26, 27, 29 }
PARTS = { 54, 51, 52, 53, 55, 56, 74 }

CELL # : 6
MACHINES = { 28, 29, 30, 31, 32, 40 }
PARTS = { 61, 59, 64, 63, 58, 60, 62, 65, 69, 76 }

CELL # : 7

MACHINES = { 15, 22, 23 }

PARTS = { 40, 35, 39, 42, 36, 41, 44, 45, 46, 49, 50 }

CELL # : 8

MACHINES = { 22, 23, 45 }

PARTS = { 38, 37 }

CELL # : 9

MACHINES = { 24 }

PARTS = { 57 }

Efficiency measures, Inter-cell flow: 74.80 % ,
Inner-cell density: 63.13 %,
Work-load balance: 78.61 % ,
Under-utilization: 60.62 % .

ZODIAC

S-1. Weighting factor: 0.50,
Threshold value for representative seeds: 7,
Maximum allowable number of cells: 41.

S-2. Artificial seeds for columns,

Seed(01)=(11000000000000000000000000000000000000000000000000)

Seed(02)=(00110000000000000000000000000000000000000000000000)

Seed(03)=(00001100000000000000000000000000000000000000000000)

Seed(04)=(00000011000000000000000000000000000000000000000000)

Seed(05)=(00000000110000000000000000000000000000000000000000)

Seed(06)=(00000000001100000000000000000000000000000000000000)

Seed(07)=(00000000000011000000000000000000000000000000000000)

Seed(08)=(00000000000000110000000000000000000000000000000000)

Seed(09)=(00000000000000001100000000000000000000000000000000)

Seed(10)=(00000000000000000010000000000000000000000000000000)

Seed(11)=(00000000000000000001000000000000000000000000000000)

Seed(12)=(00000000000000000000100000000000000000000000000000)

Seed(13)=(00000000000000000000010000000000000000000000000000)

Seed(14)=(00000000000000000000001000000000000000000000000000)

Seed(15)=(00000000000000000000000100000000000000000000000000)

Seed(16)=(00000000000000000000000010000000000000000000000000)

Seed(17)=(00000000000000000000000001000000000000000000000000)

Seed(18)=(00000000000000000000000000100000000000000000000000)

Seed(19)=(00000000000000000000000000010000000000000000000000)

Seed(20)=(00000000000000000000000000001000000000000000000000)

Seed(21)=(00000000000000000000000000000100000000000000000000)

Seed(22)=(00000000000000000000000000000010000000000000000000)

Seed(23)=(00000000000000000000000000000001000000000000000000)

Seed(24)=(00000000000000000000000000000000100000000000000000)

Seed(25)=(00000000000000000000000000000000010000000000000000)

Seed(26)=(00000000000000000000000000000000001000000000000000)

Seed(27)=(00000000000000000000000000000000000100000000000000)

Seed(28)=(00000000000000000000000000000000000010000000000000)

Seed(29)=(00000000000000000000000000000000000001000000000000)

```
Seed(30)=(000000000000000000000000000000000000000100000000000)
Seed(31)=(000000000000000000000000000000000000000010000000000)
Seed(32)=(000000000000000000000000000000000000000001000000000)
Seed(33)=(000000000000000000000000000000000000000000100000000)
Seed(34)=(000000000000000000000000000000000000000000010000000)
Seed(35)=(000000000000000000000000000000000000000000001000000)
Seed(36)=(000000000000000000000000000000000000000000000100000)
Seed(37)=(000000000000000000000000000000000000000000000010000)
Seed(38)=(000000000000000000000000000000000000000000000001000)
Seed(39)=(000000000000000000000000000000000000000000000000100)
Seed(40)=(000000000000000000000000000000000000000000000000010)
Seed(41)=(000000000000000000000000000000000000000000000000001)
```

Part clustering:

Part # 01 is assigned to cluster # 01.

Part # 02 is assigned to cluster # 01.

Part # 03 is assigned to cluster # 01.

Part # 04 is assigned to cluster # 01.

Part # 05 is assigned to cluster # 01.

Part # 06 is assigned to cluster # 02.

Part # 07 is assigned to cluster # 04.

Part # 08 is assigned to cluster # 03.

Part # 09 is assigned to cluster # 03.

Part # 10 is assigned to cluster # 03.

⋮

Part # 100 is assigned to cluster # 20.

Representative seeds for columns,

```
Seed(01)=(111001000100010000000000000000000000000000000000110)-Prt#: 03
Seed(02)=(000111101110000000000000000000000000011000000000000)-Prt#: 10
Seed(03)=(000101111110000100000000000000000000000000000000000)-Prt#: 07
Seed(04)=(000000000001111100000000010000000000000000000000000)-Prt#: 14
Seed(05)=(000000000001101110000000000000000000000000000000000)-Prt#: 16
Seed(06)=(100000000000000011111000000000000000000010000001000)-Prt#: 34
Seed(07)=(000000000000001000010000000000000111011000000000000)-Prt#: 68
Seed(08)=(001000000000000000010110000000000000100000000000000000)-Prt#: 18
Seed(09)=(000000010000000000000000110001100000000000000000000)-Prt#: 41
```

```
Seed(10)=(001010000000000000000000010000000000000000000000000000000)-Prt#: 06
Seed(11)=(000000000000000000000000100000010000000110111111111)-Prt#: 99
Seed(12)=(000000000000000100000000000010000001000101100011100)-Prt#: 89
Seed(13)=(000000000000000000000000000000000100000111110111111)-Prt#: 80
Seed(14)=(000000000000010000000000000000000000100110100111111)-Prt#: 94
Seed(15)=(000000000000010000000000000000000000101111110111)-Prt#: 83
Seed(16)=(0000001000000000000000000000000000000001111111111)-Prt#: 84
Seed(17)=(000000000000000011110000000000000000000000000000000)-Prt#: 17
Seed(18)=(000000000000000010011000000000010000000000000100000)-Prt#: 26
Seed(19)=(000000010000000000101000000011000000000000000000000)-Prt#: 27
Seed(20)=(000001000000000011001000000000000000000000100000000)-Prt#: 28
Seed(21)=(00000000000000000000000000000000000000101101010110)-Prt#: 85
Seed(22)=(10000000000000000000000000000000000000101011111011)-Prt#: 86
Seed(23)=(000000000000000000000000010000000001001011111111111)-Prt#: 88
Seed(24)=(00000000000000000000000000000000000000011110110011)-Prt#: 90
Seed(25)=(00000000000000000000000000000000000000110111110010)-Prt#: 92
Seed(26)=(00000000000000000000000000000000000000100101101100)-Prt#: 93
Seed(27)=(000000000000000000001100000000000000000000000100000)-Prt#: 38
Seed(28)=(010000000001000000010011000000000000000000000000000)-Prt#: 43
Seed(29)=(000000000100000000000000000111010000000000000000000)-Prt#: 60
Seed(30)=(000000001001000000000000000111000000000000000000000)-Prt#: 62
Seed(31)=(000000000000000100000000000111110000001000000000010)-Prt#: 65
Seed(32)=(000001000000000000000000000000111111000000000000000)-Prt#: 66
Seed(33)=(000000000000000100000000000000001111001000000010000)-Prt#: 72
Seed(34)=(000000100001001000000000000000000011110000000000000)-Prt#: 73
Seed(35)=(000000000000000000000000000010001001110000000000000)-Prt#: 74
Seed(36)=(000000000000000110000000000000000000010001101111)-Prt#: 77
Seed(37)=(000000000100000000000000000000000000001111101100)-Prt#: 78
Seed(38)=(000000000000000001000001110000000000000000000000000)-Prt#: 51
Seed(39)=(000000000000000000000000111010000000000000000000000)-Prt#: 54
Seed(40)=(000000000000000000000000100100000000000000000000000)-Prt#: 55
Seed(41)=(01000000000000000000000000000000000000110111111101)-Prt#: 95
```

Part clustering.

S-3. Number of non-null column clusters: 41,
Artificial seeds for rows,

```
Seed(01)=(1110000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(02)=(0001110000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(03)=(0000001110000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(04)=(0000000001110000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(05)=(0000000000001110000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(06)=(0000000000000001110000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(07)=(0000000000000000001110000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(08)=(0000000000000000000001110000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(09)=(0000000000000000000000001110000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(10)=(0000000000000000000000000001110000000000000000000000000000000000000000000000000000000000000000000000)
Seed(11)=(0000000000000000000000000000001110000000000000000000000000000000000000000000000000000000000000000000)
Seed(12)=(0000000000000000000000000000000001110000000000000000000000000000000000000000000000000000000000000000)
Seed(13)=(0000000000000000000000000000000000001110000000000000000000000000000000000000000000000000000000000000)
Seed(14)=(0000000000000000000000000000000000000001110000000000000000000000000000000000000000000000000000000000)
Seed(15)=(0000000000000000000000000000000000000000001110000000000000000000000000000000000000000000000000000000)
Seed(16)=(0000000000000000000000000000000000000000000001110000000000000000000000000000000000000000000000000000)
Seed(17)=(0000000000000000000000000000000000000000000000001110000000000000000000000000000000000000000000000000)
Seed(18)=(0000000000000000000000000000000000000000000000000001110000000000000000000000000000000000000000000000)
Seed(19)=(0000000000000000000000000000000000000000000000000000001100000000000000000000000000000000000000000000)
Seed(20)=(0000000000000000000000000000000000000000000000000000000011000000000000000000000000000000000000000000)
Seed(21)=(0000000000000000000000000000000000000000000000000000000000110000000000000000000000000000000000000000)
Seed(22)=(0000000000000000000000000000000000000000000000000000000000001100000000000000000000000000000000000000)
Seed(23)=(0000000000000000000000000000000000000000000000000000000000000011000000000000000000000000000000000000)
Seed(24)=(0000000000000000000000000000000000000000000000000000000000000000110000000000000000000000000000000000)
Seed(25)=(0000000000000000000000000000000000000000000000000000000000000000001100000000000000000000000000000000)
Seed(26)=(0000000000000000000000000000000000000000000000000000000000000000000011000000000000000000000000000000)
Seed(27)=(0000000000000000000000000000000000000000000000000000000000000000000000110000000000000000000000000000)
Seed(28)=(0000000000000000000000000000000000000000000000000000000000000000000000001100000000000000000000000000)
Seed(29)=(0000000000000000000000000000000000000000000000000000000000000000000000000011000000000000000000000000)
Seed(30)=(0000000000000000000000000000000000000000000000000000000000000000000000000000110000000000000000000000)
Seed(31)=(0000000000000000000000000000000000000000000000000000000000000000000000000000001100000000000000000000)
Seed(32)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000011000000000000000000)
Seed(33)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000110000000000000000)
Seed(34)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000001100000000000000)
Seed(35)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000011000000000000)
Seed(36)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000110000000000)
Seed(37)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000001100000000)
Seed(38)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000011000000)
Seed(39)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000110000)
Seed(40)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000001100)
Seed(41)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000011)
```

Machine clustering,

Representative seeds for rows,

```
Seed(01)=(111110000000000001000000000010000000000000000000000000000010000000000000000000000000000000000000000000000)-
-Machine#: 03
Seed(02)=(0010001111110000000000000001000000000000000000000000000000000001000000000000000000000000000000000000000)-
-Machine#: 06
Seed(03)=(001000101111000000000010000000000000000000000000000000000000010000000000000000001000000000000000000000000)-
-Machine#: 10
Seed(04)=(000000100000111000000000000000000000000000001000000000000000000010000001000010000000000001000000000000)-
-Machine#: 16
Seed(05)=(000000000000000011111111001111111000000000000100010100000000000000000000000000000000000000000000000000000)-
-Machine#: 18
Seed(06)=(000000000001000110110111110111111100000000000000000000000000000000000000000000100000000000000000000000000)-
-Machine#: 17
Seed(07)=(000010000000000000000000000000001000000000000000100111101000000000000000000000000000000001000000000000)-
-Machine#: 26
Seed(08)=(000000000000000000000000000000000000000000000000000111011100000000000000000000000000000000000000000000010)-
-Machine#: 24
Seed(09)=(000000000000000000001000001001000000000000000000000000001000011111110000000010000000000000000000000000000)-
-Machine#: 29
Seed(10)=(000000000000000000010000000001010000000000000100000000000000000000111110110110000000000000100000000000)-
-Machine#: 35
Seed(11)=(001000000000000000000000000000000000000000000000000000000000000000000000000011011011101110101111111)-
-Machine#: 48
Seed(12)=(000000001000000000000100000000000000000000000000000000000000000000000001000010111011111111111111110011)-
-Machine#: 46
Seed(13)=(000000000000000000000000001000000000011000000000000000000000000000000000000000101011101110111111111110)-
-Machine#: 45
Seed(14)=(000000111111000000000010000000000000000000000000000000000000000000000000000000000000000000000000000000100)-
-Machine#: 04
Seed(15)=(000001011110000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)-
-Machine#: 05
Seed(16)=(000000101110000000000000000000000000000000000000100000000000000000000000000010000000100100000000000000000)-
-Machine#: 07
Seed(17)=(000000110001000000000000000100000000000000010000001000000000000000000000000000000000000000000000000000000)-
-Machine#: 08
Seed(18)=(000000110110000000000000000000000000000000000000000000000000001000000000000000000001000000000000000000000)-
-Machine#: 09
Seed(19)=(000000000001110100000000000000000000000000000000000000000000100000000001000000000000000000000000000000000)-
-Machine#: 12
Seed(20)=(000000000000110100000000000000000000000000000000010000000000000000000000000000000000000000000000000000000)-
-Machine#: 13
Seed(21)=(001000000000011000000000000000000000000000000000110000000000000000010000001000000100000000000001000000)-
-Machine#: 14
Seed(22)=(000000000000111000000000000000000000000001000000000000000000000000000000000100000000100000000000000000)-
-Machine#: 15
Seed(23)=(000000000000000010000000000000000000000000000000000000000000000011111110111000100000000001000000000)-
-Machine#: 33
Seed(24)=(000000000000000000000000000000000000000000000001000000000000000000011111110011000000000000000000000000)-
-Machine#: 34
Seed(25)=(000000000000000000000000000000000000000000000000000000000000000000000110100111110000000000100000101000000)-
-Machine#: 36
Seed(26)=(000000001000000000000000000000000000000000000000000000000000000000000011111101111000000000000000000000000)-
-Machine#: 37
Seed(27)=(000000000000000000000000000000000000100001000000000000000001111111100000000000000000000000001000000000)-
-Machine#: 28
Seed(28)=(000000000000000000000000001000000000000000000000000000000000011111111000100000000000000000000000000000)-
-Machine#: 30
Seed(29)=(000000000000000000000000010000000001000000000000000000000000110101110000000000000000000000000000000000010)-
-Machine#: 31
Seed(30)=(000000000000000000000000000000000010000000000000001000000000000010000000000000111010100100111111111)-
-Machine#: 40
Seed(31)=(000000000000000000000010000000000000000000000000000000000000000000000000000000101111111111111000011101)-
-Machine#: 41
Seed(32)=(000000000000000000000000000101000000000000000000000000000000000000000000000000101011110111111101101011)-
-Machine#: 42
Seed(33)=(000000100000000000000000000000000000000000000000000000000000000000000000000111110110111011110101010)-
-Machine#: 43
Seed(34)=(000000000010000000000000000000000000000000000000000000000000000000000001000001100101111111001100111111)-
-Machine#: 44
Seed(35)=(000000000000000000000000000000000010000000000000000000000000000000000000000110111010111100001111011)-
-Machine#: 47
Seed(36)=(111110000000000000000000000000001000000000000000000000000000000000000000000000100000000000000)-
-Machine#: 01
Seed(37)=(101010000000000000000000000000000000000010000000000000000000000000000010000000000000000000000000100000)-
-Machine#: 02
Seed(38)=(000000110111000000000000000000000000000010000000000000000000000000000000000000000000000000000000000000000)-
-Machine#: 11
Seed(39)=(000000000000100010000111101011100100000000100000000000000000000000000100000000000000000000000000000000000)-
-Machine#: 19
Seed(40)=(000000000000000110111011100101011000000000000000000000000000000000000000000000000000000000000000000000000)-
-Machine#: 20
Seed(41)=(000000000000000011111101111110111101000000000000000000000000000000000000000000000000000000000000000000000)-
-Machine#: 21
```

Machine clustering,

Number of non-null row clusters: 41.

S-4. Number of clusters: 41,

Reordered incidence matrix,

MACHINE-PART INCIDENCE MATRIX AT THE END OF STAGE 1

```
Part: 0000000000000000000000000000000000000000000000000000000001000000000000000000000000000000000000000000
Part: 0011001111136777123440000033988899899881222222333221238890887999334444444545566666677777775555559
Part: 3901782453648015816181245659990717246347023459013679821580689023780245679038901342567962347812345675
Loc : 0000000000000000000200000000030000000004000000000500000000060000000007000000000800000000090000000000
Loc : 1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
M/C LOC
03 01 1000000000000000010000111110000000000000000000000100000000000000000000000000000100000000000000000000000
06 02 1111110000000000000000000000000000000000000000000000000010000000000000000000000000001000000000000000
10 03 1111101000000000000000000000000000000000010000000000000000000000000000000000001000000000000000100000000
16 04 0000100111100000000000000000001000000000000000000000000000000000000000000001000000000010000100100000000
18 05 0000000000100001100000000000000000000000011111111101110000000000000000000001000000000000000000010100000
17 06 0000001000110000000000000000000000000000111111111110111000000000000000000000000000000000000001000000000
26 07 0000000000000000000100010000000000000000000000000000000100000100000000000000000000000000000011110100
24 08 0000000000000000000000000000100000000000000000000000000000000000000000000000000000000000000011101110
27 09 0000000000000000001000000000000000000000000000000000000000000000010000000000000000000000000000111100
29 10 0000000000000000010000000000000000000000000000100010000000000000000000000000111111100000010000010000
35 11 0000000000001101000000000000001000000000100000100000010000000000000000010000000000000111111000000000000
48 12 1000000000000000000000000000001111101111000000000000000111101000100000000000000000000000001100000001
49 13 1000000000000000000000000000101111111111000000000000000111111111000000000000000000100000001000000000
46 14 0100000000000000000000000000111110011110010000000000000011011111100000000000000000000001001000000001
45 15 0000000000000000000000000000010111111111000000000010000001011011111000000000000000000000000100000001
04 16 0111111000000000000000000000000000000000001000000000000000010000000000000000000000000000000000000000
05 17 0111010000000000000000000000010000000000000000000000000000000000000000000000000000000000000000000000
07 18 0111100000000000000100000000000000000001000000000000000010000000000000000000000000000000100000000000
08 19 0000111000000000000110000000000000000000000000000010000000000000000000000000000000000000000000000000
22 20 0000000000000000001110000000000000000000000000000000000000000000001111111111000000000000000000000000
23 21 0000000000000000001110000111000000000000000000000000000000000000001111100011100000000000000000000000
09 22 0011110000000000000000000000000000001000000000000000000000000000000000000000000001000000000000000000
12 23 0000001101100000000000000000000000000000000000000000000000000000000000000000000001000000100000000000
13 24 0000000101100000000000000000000000000000000000000000000000000000000000000001000000000000000000000000
25 25 0000000100000000000000000000000000000000000000000000000000000000000000000000000000000000000010110000
14 26 1000000110001001000000000000000000000110000000000000000000000000000000000011000000000000000000000000
15 27 0000000110100000000000000000000000000010000000000000000000000000001000000000000000000000100000000000
33 28 0000000000001111100000000000000101000000000000000000000000000000000000000000000000011111010000000000
34 29 0000000000001111000000000000000000000000000000000000000000000000000000000010000000000111110000000000
36 30 0000000000000000100000000000000000000011000000000000000000000001000000000000000000000011111110000000000
37 31 0010000000001111000000000000000000000000000000000000000000000000000000000000000000011110110000000000
38 32 0010000000001111000000000000000010000000000000000000000000000000000000100000000000011110110000000000
28 33 0000000000000000001100000000000100000000000000000000000000000000000000000000111111111000000000000000000
30 34 0000000000000000000000000000000000000000000000000100000000000000000000000000111111111100100000000000000
32 35 0000000000000000000000000000000000000000000000000000000000000000000000000000111111010001000000000000
31 36 0000000000000000001000000000010000000000000000000100000000000000000000000000110111101000000000000000
40 37 0000000000010000000000000000010111111110100000000000000000110011110000000001000000100000000000100000001
41 38 0000000000000000000000000000000011111110111000010000000000000111111010000000000000000000000000100000000
42 39 0000000000000000000000000000011111111011000000010000000100101010110000000000000000000000000100000001
43 40 0000010000000000000000000000010111100011000000000000000000010001111110000000000000000000000011000000001
39 41 0000010000000010000001000000011111111111000000010000000000111111001000000000000000000010000100100000001
44 42 0000001000000010000000000000010011100111000000000000000000011111100100000000000000000000000011000000001
47 43 0000000000010000000000000000011110111101000000000000000000010011100000000000000000000000000011000000001
50 44 0000000010000000000000001000010111111111100000000000000000011111000000000000000000000000000001000000001
01 45 1000000000010000000000111100000000000000000000000000000000100000000000000000000000000000000000000000
02 46 1000000000000000000001001000000000000000000000000000000000000000000000000000010000000000100000000000001
11 47 0011111000000000000000000000000000000000000000000000000000000000000000000000010000000000000000000000000
19 48 0000000001011000000000000000000000000000010111111100100000000000000000000000010000000000000000000000000
20 49 0000000000010000110000000000000000000000001110111011100000000000000000000000000000000000000000000000
21 50 0000000000010000111000000000000000000000001110110111111100000000000000000000000000000000000000000000
```

S-5. E_i^j information,

1 : 1 0 0 0 0 0 0 1 0 5 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
2 : 1 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
3 : 1 3 2 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0
4 : 0 0 1 2 2 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0 1 0 0 0 0 0
5 : 0 0 0 0 0 1 0 2 0 0 0 0 0 0 0 010 0 0 3 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 2 0 0 0
⋮

E_i information,

9 9 9 10 19 18 9 13 12 14 42 23 21 8 5 8 33 6 6 8 9 6 13 10 11 24 11 18 9 20 18 19 18 44 36 7 6 6 13 12 16

Similarity coefficients and part cluster allocations,

$\xi_{1,01}$= .95960 $\xi_{1,2}$= .45361 $\xi_{1,03}$= .45361 $\xi_{1,04}$= .45408
$\xi_{1,05}$= .45408 $\xi_{1,06}$= .45455 $\xi_{1,07}$= .45313 $\xi_{1,08}$= .70918
$\xi_{1,09}$= .45361 $\xi_{1,10}$= .83564 $\xi_{1,11}$= .45455 $\xi_{1,12}$= .45455
$\xi_{1,13}$= .45313 $\xi_{1,14}$= .45361 $\xi_{1,15}$= .45455 $\xi_{1,16}$= .45455
$\xi_{1,17}$= .50556 $\xi_{1,18}$= .45455 $\xi_{1,19}$= .45455 $\xi_{1,20}$= .45361
$\xi_{1,21}$= .45313 $\xi_{1,22}$= .45455 $\xi_{1,23}$= .45455 $\xi_{1,24}$= .45408
$\xi_{1,25}$= .45455 $\xi_{1,26}$= .45455 $\xi_{1,27}$= .45000 $\xi_{1,28}$= .45455
$\xi_{1,29}$= .54078 $\xi_{1,30}$= .45455 $\xi_{1,31}$= .45455 $\xi_{1,32}$= .45313
$\xi_{1,33}$= .45455 $\xi_{1,34}$= .45455 $\xi_{1,35}$= .45455 $\xi_{1,36}$= .45455
$\xi_{1,37}$= .45455 $\xi_{1,38}$= .45361 $\xi_{1,39}$= .45455 $\xi_{1,40}$= .45361
$\xi_{1,41}$= .45455
PART CLUSTER 01 IS ALLOTED TO MACHINE CLUSTER 01.

PART CLUSTER 02 IS ALLOTED TO MACHINE CLUSTER 02.
PART CLUSTER 37 IS ALLOTED TO MACHINE CLUSTER 03.
PART CLUSTER 04 IS ALLOTED TO MACHINE CLUSTER 04.
PART CLUSTER 17 IS ALLOTED TO MACHINE CLUSTER 05.
PART CLUSTER 20 IS ALLOTED TO MACHINE CLUSTER 06.
PART CLUSTER 38 IS ALLOTED TO MACHINE CLUSTER 07.
PART CLUSTER 40 IS ALLOTED TO MACHINE CLUSTER 08.
PART CLUSTER 19 IS ALLOTED TO MACHINE CLUSTER 09.
PART CLUSTER 32 IS ALLOTED TO MACHINE CLUSTER 10.
PART CLUSTER 13 IS ALLOTED TO MACHINE CLUSTER 11.

```
PART CLUSTER 24 IS ALLOTED TO MACHINE CLUSTER 12.
PART CLUSTER 14 IS ALLOTED TO MACHINE CLUSTER 13.
PART CLUSTER 03 IS ALLOTED TO MACHINE CLUSTER 14.
PART CLUSTER 10 IS ALLOTED TO MACHINE CLUSTER 15.
PART CLUSTER 16 IS ALLOTED TO MACHINE CLUSTER 16.
PART CLUSTER 09 IS ALLOTED TO MACHINE CLUSTER 17.
PART CLUSTER 30 IS ALLOTED TO MACHINE CLUSTER 18.
PART CLUSTER 05 IS ALLOTED TO MACHINE CLUSTER 19.
PART CLUSTER 39 IS ALLOTED TO MACHINE CLUSTER 20.
PART CLUSTER 07 IS ALLOTED TO MACHINE CLUSTER 21.
PART CLUSTER 15 IS ALLOTED TO MACHINE CLUSTER 22.
PART CLUSTER 33 IS ALLOTED TO MACHINE CLUSTER 23.
PART CLUSTER 27 IS ALLOTED TO MACHINE CLUSTER 24.
PART CLUSTER 23 IS ALLOTED TO MACHINE CLUSTER 25.
PART CLUSTER 34 IS ALLOTED TO MACHINE CLUSTER 26.
PART CLUSTER 29 IS ALLOTED TO MACHINE CLUSTER 27.
PART CLUSTER 31 IS ALLOTED TO MACHINE CLUSTER 28.
PART CLUSTER 11 IS ALLOTED TO MACHINE CLUSTER 29.
PART CLUSTER 06 IS ALLOTED TO MACHINE CLUSTER 30.
PART CLUSTER 21 IS ALLOTED TO MACHINE CLUSTER 31.
PART CLUSTER 12 IS ALLOTED TO MACHINE CLUSTER 32.
PART CLUSTER 22 IS ALLOTED TO MACHINE CLUSTER 33.
PART CLUSTER 25 IS ALLOTED TO MACHINE CLUSTER 34.
PART CLUSTER 36 IS ALLOTED TO MACHINE CLUSTER 35.
PART CLUSTER 18 IS ALLOTED TO MACHINE CLUSTER 36.
PART CLUSTER 28 IS ALLOTED TO MACHINE CLUSTER 37.
PART CLUSTER 26 IS ALLOTED TO MACHINE CLUSTER 38.
PART CLUSTER 35 IS ALLOTED TO MACHINE CLUSTER 39.
PART CLUSTER 08 IS ALLOTED TO MACHINE CLUSTER 40.
PART CLUSTER 41 IS ALLOTED TO MACHINE CLUSTER 41.
```

Reordered incidence matrix,

```
                MACHINE-PART INCIDENCE MATRIX AT THE END OF STAGE 2

Part: 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000010000000000
Part: 001171112222223331235555552666788997989900100000338344611567778733444444587556666693889088972497129
Part: 3901845702345901398212356776796071790246782124565946182364801532780245679083890134594158096276334815
Loc : 0000000000000000000200000000030000000004000000000500000000060000000007000000000800000000090000000000
Loc : 1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
M/C LOC
03 01 1000000000000001000000000000000000000000111110000000000000000000000000000000000000000000000000100
06 02 1111000000000000000100000000010000000000000111000000000000000000000000000000000000000000000000000000
10 03 1111100001000000000000000000000000000000001010000000000000000000000000000000010000000000000000000000
16 04 0000011000000000000000000000000000000000100000000000001100000000100000100000000000000000100000010010000000
18 05 0000000111111111111111010000000000000000000000000000000000000000000000000010000000000000100000000000110
17 06 0000000111111111111111000000000000000000000000000010000000000000100000000000000000000000000000001000000011000000
26 07 0000000000000000000111101000000000000000000000000000100000010001000000000000000000010000000000000000000000000000
24 08 0000000000000000000111110000000000000000000000000000000000000000000000000000000000001000000000000000
27 09 0000000000000000000001110000000000000000000000000001000010000000100000000000000000000000000000000000
29 10 0000000000000010000000000000100000000000000000000010010000000000000000000011111100000000000000001010
35 11 0000000010000010000100000001110000000000000000000000000001101010000000100001000000000000000100000000000
48 12 1000100000000000000000000000011110001100000000001000000000000010000000000100000000010111111001001000100
49 13 1000000000000000000000000000011111111100000000001000000000000010000000000001000000000110111101110000000
46 14 0100000001000000000000000000011101101100000000001000000000000011000000000100000000010110111111100100011
45 15 0000100000000000000000000000011101111100000000001000000000001011000000000100000000010001001101010001
04 16 0111000001000000000000000000000000000000111000000000000000000000000000000000000000000000100000000000
05 17 0111000000000000000000000000000000000000000100000100000000000000000000000000000000000000000000000000
07 18 0111000000000000000000000000000000000000001000000000100100000000000000000000100000000010000000000000
08 19 0000000000000000000000000100000000000000111000000000000001100000000000000000000000000000000000000000
22 20 0000000000000000000000000000000000000000000000011100000000000001111111110000000000000000000000100000
23 21 0000000000000000000000000000000000000000000001110111000000000000111110001100000000000000000000100000
09 22 0011000000000000000000000000000000000001001100000000000000010000000000000000000000000000000000000000
12 23 0000010000000000000000000000000000000000010000000000111000000000000000000100000000000000000000000000
13 24 0000010000000000000000000000000000000000000000000001100000000000000000010000000000000000000000000000
25 25 0000010000000000000001010000000000000000000000000000000000000100000000000000000000000000000000000000
14 26 1000011000000000000000000000000001100000000000000000100100000000000001100000000000000000000000000000
15 27 0000011000000000000000000000000000000000000000000000100000100010000000001000000000000000000000000000
33 28 0000000000000000000000011110100000000000000000000000000000111101000000000000000000000000000000001100
34 29 0000000000000000000000011110000000000000000000000000000000111101000000001000000000000000000000000000
36 30 0000000000000000000000011110000000110000000000000000000000001010000000000011000000000000000000001000
37 31 0010000000000000000000011110000000000000000000000000000000111100000000000000100000000000000000001000
38 32 0010000000000000000000011110100000000000000000000000000000111100000001000000100000000000000000001000
28 33 0000000000000000000000000000000000000000000000000000110100000000000000000001111111000001000000000000
30 34 0000000000000000000000001001000000000000000000000000000100000000000000000000011111100000000000000000
32 35 0000000000000000000000000100000000000000000000000000000000000000000000000000011111100000000000000000
31 36 0000000000000000000000000000000000000000000000000010000000000000000000000000011011111000000100000000000
40 37 0000100000000000000000000000011111111100000000000001000000000000000000001000000000100110011001000100001
41 38 0000100000000010000000000000011101101000000000000001000000000000100000000001000000000011111100000000000
42 39 0000100000000100010000000000011101111000000000000001000000000000100000000001000000000100101101000000001
43 40 0000100000000000000000000000011111100001000000000001000000000000100000000001000000000101000011100100010001
39 41 0000000000000010000000000000011100111010100000000000000000000010110000000000100000000110111111110000001
44 42 0000100000000000000000000000011100001001000000000001000000000010100000000001000000000101111011100000001
47 43 0000100000000000000000000000011010011100000000000001000000000000000000000001000000000111001110100000001
50 44 0000010000000000000000000000011111111100000010000100000000000001000000000001000000000100001010100000001
01 45 1000000000000000000000000000000000000000011110000000000000000000000000000000000000010000010000000000
02 46 1000000000000000000000000100000000000000100100000000000000000000000000000000000000000000000000100001
11 47 0011000000000000000000000000000000000000001110000000000000000000000000000000000000000000000000100000
19 48 0000000101111111100000000001000000000000000000000001001000000000000000000000000000001000000000100000
20 49 0000000111011101100000000000000000000000000000000000000000000000000000000000000000001000000000100011 0
21 50 0000000011101101111100000010000000000000000000000000100000000000000000000000000000001000000000100011 0
```

S-6. Inter-cell flow : 0.8947,

Inner-cell density: 0.7966,

Clustering efficiency: 0.8457,

Limiting efficiency: 0.9536,

Relative efficiency: 0.8868.

S-7. Ideal seeds for machines,

```
Seed(01)=(0010000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(02)=(0000000011100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(03)=(0000001100010000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(04)=(0000000000000110000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(05)=(0000000000001001000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(06)=(0000000000000000000000000000000001000000000000000000000000000000000000000000000000000000000000000000)
Seed(07)=(0000000000000000000000000000000000000000000000000000000000000000000101100010000000000000000000000000)
Seed(08)=(0000000000000000010010000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(09)=(0000000000000000000000000000000000010000100000010000000000000000000000000000000000000000000000000000)
Seed(10)=(1101110000000000000000000000000000100010000000000000000000000000000000000000000000000000000000000000)
Seed(11)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000010)
Seed(12)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000100000000000)
Seed(13)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000100000010001000001000)
Seed(14)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000001000000000001010000)
Seed(15)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000100000000000000000)
Seed(16)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000010000000000000000)
Seed(17)=(0000000000000000100101111000111010000000000000000000000000000000000000000000000000000000000000000000)
Seed(18)=(0000000000000000000000000100000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(19)=(0000000000000000000000000010000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(20)=(0000000000000000001000000001000100000000000000000000000000000000000000000000000000000000000000000000)
Seed(21)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000010001000000000000101)
Seed(22)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000100000000000000)
Seed(23)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000001000000000000)
Seed(24)=(0000000000000000000000000000000000000000000000000000000000000000000000000000001000000000010000000000)
Seed(25)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000100000000)
Seed(26)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000010000000)
Seed(27)=(0000000000000000000000000000000000001101010111101100000000000000000000000000000000000000000000000000)
Seed(28)=(0000000000000000000000000000000000000000001000000000000000000000000000000000000000000000000000000000)
Seed(29)=(0000000000000000000000000000000000000000000000000000000111101100000000000000000000000000000000000000)
Seed(30)=(0000000000000000000000000000000000000000000000000000000000000100000000000000000000000000000000000000)
Seed(31)=(0000000000000000000000000000000000000000000000000000000000000000100000000000000000000000000000000000)
Seed(32)=(0000000000000000000000000000000000000000000000000000000000000000011010000001000000000000000000000000)
Seed(33)=(0000000000000000000000000000000000000000000000000000000000000000000000010000000000000000000000000000)
Seed(34)=(0000000000000000000000000000000000000000000000000000000000000000000000001000000000000000000000000000)
Seed(35)=(0000000000000000000000000000000000000000000000000000000000000000000000000100000000000000000000000000)
Seed(36)=(0000000000000000000000000000000000000000000000000000000000000000000000000000100000000000000000000000)
Seed(37)=(0000000000000000000000000000000000000000000000000000000000000000000000000000010000000000000000000000)
Seed(38)=(0000000000000000000000000000000000000000000000000011100000000000000000000000000000000000000000000000)
Seed(39)=(0000000000000000000000000000000000000000000000000000010000000000000000000000000000000000000000000000)
Seed(40)=(0000000000000000000000000000000000000000000000000000001110000000000000000000000000000000000000000000)
Seed(41)=(0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000010000000)
```

Row clustering,

Number of non-null clusters: 16,

Ideal seeds for parts,

```
Seed(01)=(11000000000000000000000000000000000000000000000000)
Seed(02)=(00011110111000000000000000000000000000000000000000)
Seed(03)=(00000001001000000000000000000000000000000000000000)
Seed(04)=(00000000000001110000000000000000000000000000000000)
Seed(05)=(00000000000110000000000000000000000000000000000000)
Seed(06)=(00000000000000000000000000000000110011000000000000)
Seed(07)=(00100000000000000000000000000000000000000000000000)
Seed(08)=(00000000000000000000000000000000000000111110110111)
Seed(09)=(00000000000000000000000000000000000000000000001000)
Seed(10)=(00000000000000001111100000000000000000000000000000)
Seed(11)=(00000000000000000000000000000000000000000001000000)
Seed(12)=(00000000000000000000011000000000000000000000000000)
Seed(13)=(00000000000000000000000000011111000000000000000000)
Seed(14)=(00000000000000000000000000000000001100000000000000)
Seed(15)=(00000000000000000000000111000000000000000000000000)
Seed(16)=(00000000000000000000000000100000000000000000000000)
```

Column clustering,

Number of non-null clusters: 15,

Representative seeds for parts,

```
Seed(01)=(11100100010001000000000000000000000000000000000110)-Rrt#: 03
Seed(02)=(00011110111000000000000000000000000000110000000000)-Rrt#: 10
Seed(03)=(00010101011100001000000000000000000000000001000000)-Rrt#: 12
Seed(04)=(00000000000111110000000001000000000000000000000000)-Rrt#: 14
Seed(05)=(00000000000110111000000000000000000000000000000000)-Rrt#: 16
Seed(06)=(01000000000000000000000000001001111111000000000000)-Rrt#: 69
Seed(07)=(00100000000000000101100000000000001000000000000000)-Rrt#: 18
Seed(08)=(00000000000000000000000000000000000001111111111111)-Rrt#: 87
Seed(09)=(10000000000000000111110000000000000000010000001000)-Rrt#: 34
Seed(10)=(00000000000000000001110000100100000000000000000000)-Rrt#: 36
Seed(11)=(00000000000000010000000000011111000000100000000010)-Rrt#: 65
Seed(12)=(00000000000000010000000000000000011110100000010000)-Rrt#: 72
Seed(13)=(00000000000000000100000111100000000000000000000000)-Rrt#: 53
Seed(14)=(00000000000000000000001000100000000000000000100000)-Rrt#: 37
Seed(15)=(00000000000000000111000000100000100000001000000000)-Rrt#: 30
```

Column clustering,

Number of non-null clusters: 15,

Ideal seeds for machines,

```
Seed(01)=(1010100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(02)=(0000001111100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(03)=(0000000000010000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(04)=(0000000000000110000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(05)=(0000000000001001000000000000000000000000000010000000000000000000000000000000000000000000000000000000)
Seed(06)=(0000000000000000000000000000000000000000000000000000000000000000011111101111000000000000000000000000)
Seed(07)=(0101000000000000011110000100100010000000000000100000000000000000000000000000000000000000000000000000)
Seed(08)=(0000000000000000000000000000000000000000000000000000000000000000000000000000111111111111011101111111)
Seed(09)=(0000000000000000100001111000001001000000000000000000000000000000000000000000000000000000000000000000)
Seed(10)=(0000000000000000000000000000000000010001111101011100000000000000000000000000000000000000000000000000)
Seed(11)=(0000000000000000000000000000000000000000000000000000000111111110000000000000000000000000000000000000)
Seed(12)=(0000000000000000000000000000000000000000000000000000000000000000000000010000000000000000100000000000)
Seed(13)=(0000000000000000000000000000000000000000000000000011111110000000000000000000000000000000000000000000)
Seed(14)=(0000010000000000000000000000000000101110000000000000000000000000000000000000000000000000000010000000)
Seed(15)=(0000000000000000000000000011010100000000000000000000000000000000000000000000000000000000000000000000)
```

Row clustering.

S-8. *Go to* S-9.

S-9. Ideal seeds for parts,

```
Seed(01)=(1110000000000000000000000000000000000000)
Seed(02)=(0001111011100000000000000000000000000000)
Seed(03)=(0000000100010000000000000000000000000000)
Seed(04)=(0000000000000110000000000100000000000000)
Seed(05)=(0000000000001001000000000000000000000000)
Seed(06)=(0000000000000000000000011111100000000000)
Seed(07)=(0000000000000000000010000000000000000000)
Seed(08)=(0000000000000000000000000000011111111111)
Seed(09)=(0000000000000000111100000000000000000000)
Seed(10)=(0000000000000000000001100000000000000000)
Seed(11)=(0000000000000000000000001111100000000000)
Seed(12)=(0000000000000000000000010110000000000000)
Seed(13)=(0000000000000000000000000000000000000000)
Seed(14)=(0000000000000000000000000000000000000000)
Seed(15)=(0000000000000000000000000000000000000000)
```

Column clustering,

Number of non-null clusters: 13,

Ideal seeds for machines,

```
Seed(01)=(1111100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(02)=(0000001111100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(03)=(0000000000010000000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(04)=(0000000000000110000000000000000000000000000000000000000000000000000000000000000000000000000000000000)
Seed(05)=(0000000000001001000000000000000000000000000010000000000000000000000000000000000000000000100000000000)
Seed(06)=(0000000000000000000000000000000000000000000000000000000000000000011111111111000000000000000000000000)
Seed(07)=(0000000000000000011010000111000100000000000000000000000000000000000000000000000000000000000000000000)
Seed(08)=(0000000000000000000000000000000000000000000000000000000000000000000000000000110111111111011101111111)
Seed(09)=(0000000000000000100101111000111011000000000000000000000000000000000000000000000000000000000000000000)
Seed(10)=(0000010000000000000000000000000000111111111101111100000000000000000000000000000000000000000000000000)
Seed(11)=(0000000000000000000000000000000000000000000000000000000001111111100000000000000000000000000000000000)
Seed(12)=(0000000000000000000000000000000000000000000000000011111100000000000000000000000000000000000000000000)
Seed(13)=(0000000000000000000000000000000000000000000000000000000010000000000000000000001000000000000010000000)
```

Row clustering,

Number of non-null clusters: 12,

Representative seeds for machines,

```
Seed(01)=(1111110000000000001000000000100000000000000000000000000000100000000000000000000000000000000000000000)-
-Machine#: 03
Seed(02)=(0010001111110000000000000001000000000000000000000000000000000000010000000000000000000000000000000000)-
-Machine#: 06
Seed(03)=(0000001100010000000000000010000000000000000100000010000000000000000000000000000000000000000000000000)-
-Machine#: 08
Seed(04)=(0010000000000110000000000000000000000000000000000110000000000000000100000010000001000000000001000000)-
-Machine#: 14
Seed(05)=(0000000000000000001000000000001010000000000000010000000000000000000001111101101100000000000010000000000)-
-Machine#: 35
Seed(06)=(1000000100000000000000000000000100000000000000000000000000000000100000110000100111101111101101111111)-
-Machine#: 39
Seed(07)=(0000000000000000111111111001111111100000000000010001010000000000000000000000000000000000000000000000)-
-Machine#: 18
Seed(08)=(0000010000000000000000000000000000111111111100011100000000000000000000000000000000000000000000000000)-
-Machine#: 23
Seed(09)=(0000000000000000000010000010010000000000000000000000000010000111111100000000010000000000000000000000000)-
-Machine#: 29
Seed(10)=(0000100000000000000000000000000010000000000000000010011110100000000000000000000000000001000000000000)-
-Machine#: 26
Seed(11)=(0000000000000100000000000000000000000000000000000000101100000000000000000000000000000000000000000000)-
-Machine#: 25
Seed(12)=(0000000000000000000000000000000000000010001000000000000111100000000000000000000000000000000000000000)-
-Machine#: 27
```

Row clustering,

Number of non-null clusters: 12,

Ideal seeds for parts,

```
Seed(01)=(111000000000000000000000000000000000000000000000000)
Seed(02)=(000111101110000000000000000000000000000000000000000)
Seed(03)=(000000010000000000000000000000000000000000000000000)
Seed(04)=(000000000000010000000000000000000000000000000000000)
Seed(05)=(000000000000000000000000000000001111110000000000000)
Seed(06)=(000000000000000000000000000000000000001111111111111)
Seed(07)=(000000000000000011111000000000000000000000000000000)
Seed(08)=(000000000000000000000011000000000000000000000000000)
Seed(09)=(000000000000000000000000000111110000000000000000000)
```

```
Seed(10)=(00000000000000000000000000010000000000000000000000)

Seed(11)=(00000000000011011000000011000000000000000000000000)

Seed(12)=(00000000000000000000000000001000000000000000000000)
```

Column clustering.

S-10. Reordered incidence matrix,

```
Part: 000000001112579166667777777777888888888889999999990111222222222233333033334444444444455566666665555511135
Part: 1234578901277935678901234567801234567890124567890789012345689012346568901234567890890123451234634675
Loc : 0000000000000000000200000000030000000004000000000500000000060000000007000000000800000000090000000000
Loc : 123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
M/C LOC
01 01 111110000000000000000000000000000100000000000000000000000000001000000000000000000000000000000
02 02 101010000000000000100000000000000000100000000000000000000000000000001000000000000000000000
03 03 111110000000000000000000000000000000000000010000000001000001000000000000000100000000000000000
04 04 000001111100000000000000000000000000000000001000000000100000000000000000000000000000000000
05 05 000000111000000000000000000000000000000000000000000000000100000000000000000000000000000000
06 06 001001111110000010000000000000000000000000000000000001000000000000000000000000000000000000
07 07 000001011100000000000001000000100100000000000000000000000000000000010000000000000000000000
08 09 000001101100000000000000000000100000000000000000000000000000000000000000000010000000000000
09 10 001001011110000000000000001000000000000000000000010000000000000000000000001000000000000000
10 11 000001101110000000000000000000000000000000000000000000000000000001000000000000000000000000
11 08 000001100011000000000000000000000000000000000000000000000000001000000100000000000000000000
12 14 001000000000000010010000001000001000000000001000000000000000000000000011000000000000001000
13 33 000000000000000011111101110010000000000010000000001000000000000000000000000000000000000000
14 34 000000000000000011111110011000000000000000000000000000000000000000010000000000000000000000
15 35 000000000000000011111011011000000000000010000000001010000000000000000100000000000000000000
16 36 000000000000000011010011111000000000000100001010000000000000000000000000000000000000000000
17 37 000000001000000011111101111000000000000000000000000000000000000000000000000000000000000000
18 38 000000001000000011111101111100000000010000000000000000000000000000000100000000000000000000
19 39 100000100000000000000001100001011110111110111111111100000000000001000000000000010000000000
20 40 000000000000011000000000000001101010010011111111110000000000000000001000001000001000000000
21 41 000000000000000000000000000011111111111110001110100000000010000000000000000000000000000000
22 42 000000000000000000000000000011011110111111111010110000000000001010000000000000000000000000
23 43 000000100000011000000000000011101101110111010010100000000000000000000000000000000000000000
24 44 000000000100000000000100000110101111110011011111100000000000000000000000000000000000000000
25 45 000000000000010000000000000110111011101111111110000000000010000000001000000000000000000010
26 46 000000010000011000000010000101101111111111111100110000010000000000000000000000000000000000
27 47 000000000000000000000000000011110101111000111101100000000000000000001000000000000000000000
28 48 001000000000010000000000000011101110111010111111100000000000000000000000000000000000000000
29 49 001000000000010000000000000010111111111101111011111000000000000000000000000001000000000000
30 50 000100000000010100000000000010101110111011011110110000000000000000000000000000000000000000
31 17 000000000100000000000000010000000000000000000001011011111111111100000000000000000000000100
32 18 000000000000000000000000000000000000000000000011111111101111111110000001000000000001010000000
33 19 000000000010000001000000000000000000000000000001000011110011100100000000001000000000000010000
34 20 000000000000000000000000000000000000000000000000110111011101010110000000000000000000000000
35 21 000000000010000000000000000000000000000000000000111111011110111100100000000000000000000000
36 22 000000000000000000000000000000000000000000000000000000000000110111111111100000000000000000
37 23 000000000000000000000000000000000000000000000000000000001111111111000111000000000000000010
38 28 000000000000000000000000000000001000000000000000000000000001000100000000011111110000000000
39 29 000000000010000000000000100000000000000000100000000010000000000000000000000011111110001000000
40 30 000000000010000000100000000000000000000000000000000000000000000000000000011111110000000000
41 31 000000000000000000000000000000000000000001000000001000000000100000000000001101011000000000000
42 32 000000000000000000000000010000000000000000000000000000000000000000000000001111011100000000000
43 26 000010000000000000000000000000000010000000000000000000000001000000000010000000001111100000
44 12 000000000100000000000000100000000000000000000000000000000000000000000000000001000000011100
45 13 000000000000000000000000000000000000000000000000000000000000000000000000100000000000011100
46 15 000000000000000100000001000000001000000000000000000000000000000100000000000000000000001100
47 16 000001000000000010000001000010000000000010000000000000000000000000100000000000010000011100
48 24 000000000001000000000000000000000000000000100000000000000000000000000000000000001110100001
49 25 000000000000000000000000000000000000000000000000000000000000000000000000000000001011001000
50 27 000000000000000000000000000000000000000000000000000000000000000010000000000000000011100011
```

Inter-cell flow : 0.9797,

Inner-cell density: 0.7944,

```
Clustering efficiency: 0.8880,
Limiting efficiency: 1.0000,
Relative efficiency: 0.8880,
Smallest block is liquidated.
⋮
Reordered incidence matrix,
```

MACHINE-PART INCIDENCE MATRIX AT THE END OF STAGE 3.2

```
Part: 0000000000000000000000000000000000000000000100000000000000000000000000000000000000000000000000000000
Part: 0000000011166667777777778888888888999999999011122222222233330333334444444444555666666555555111123579
Part: 1234578901267890123456780123456789012456789078901234568912346567890123456789089012345123456345670793
Loc :0000000000000000000200000000030000000004000000000500000000060000000007000000000800000000090000000000
Loc :1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
M/C LOC
01 01 1111100000000000000000000000010000000000000000000000000000001000000000000000000000000000000000000000
02 02 1010100000000010000000000000000000000100000000000000000000000000000000010000000000000000000000000000
03 03 1111100000000000000000000000000000000000000001000000000100001000000000000000010000000000000000000000
04 04 0000011111100000000000000000000000000000010000000010000000000000000000000000000000000000000000000000
05 05 0000001110000000000000000000000000000000000000000000000000010000000000000000000000000000000000000000
06 06 0010011111100000000000000000000000000000000000000000000100000000000000000000000000000000000000000000
07 07 0000010111000000010000010010000000000000000000000000000000000000000000000000100000000000000000000000
08 08 0000011000100000000000000000000000000000000000000000000000000000000001000000100000000000000000010000
09 09 0000011011000000000000000100000000000000000000000000000000000000000000000000000001000000000000000000
10 10 0010010111100000000000100000000000000000000000001000000000000000000000000000000100000000000000000000
11 11 0000011011100000000000000000000000000000000000000000000000000000000001000000000000000000000000000000
33 12 0000000000111111101110010000000000000100000000010000000000000000000000000000000000000000000000000000
34 13 0000000000111111100110000000000000000000000000000000000000000000000000000001000000000000000000000000
35 14 0000000000111110110110000000000000010000000000000000010000000001000000000000100000000000000000001000
36 15 0000000000110100111110000000000000100000101000000000000000000000000000000000000000000000000000000000
37 16 0000000010011111101111000000000000000000000000000000000000000000000000000000000000000000000000000000
38 17 0000000010011111101111000000010000000000000000000000000000000000000000000100000000000000000000000000
39 18 1000001000000000110000101111011111101111111111000000000000000000100000000000000000000000000001000000000000
40 19 0000000000000000000000011010100100111111111100000000000000000010000000000000010000000010000000000000011
41 20 0000000000000000000000011111111111111000111010000000000100000000000000000000000000000000000000000000
42 21 0000000000000000000000011011110111111110101100000000000001000000000000000000000000000000000000001000
43 22 0000001000000000000000011110110111011101010100000000000000000000000000000000000000000000000000000011
44 23 0000000001000001000001101011111100110111111100000000000000000000000000000000000000000000000000000000
45 24 0000000000000000000000011011101110111111111100000000001000000000011000000000000000000000000000000001
46 25 0000000100000000010000101101111111111111001100000010000000000000000000000000000000000000000000000011
47 26 0000000000000000000000011110101111000111101100000000000000000000100000000000000000000000000000000000
48 27 0010000000000000000000011101110111010111111000000000000000000000000000000000000000000000000000000001
49 28 0010000000000000000000010111111111101111011111100000000000000000000000000000000000001000000000000010
50 29 0001000000000000000000010101110111011011110110000000000000000000000000000000000000000000000001000010
17 30 0000000000100000000000000100000000000000000000001011011111111111000000000000000000000000000000101000
18 31 0000000000000000000000000000000000000000000001111111101111110000000000010000000000000101000000001000
20 32 0000000000000000000000000000000000000000000011011101110110110000000000000000000000000000000000000000
21 33 0000000000000000000000000000000000000000000011111101111111100100000000000000000000000000000000010000
22 34 0000000000000000000000000000000000000000000000000000000000010101111111111110000000000000000000000000
23 35 0000000000000000000000000000000000000000000000000000000001111111111100011100000000000000000000000000
28 36 0000000000000000000000000000000000001000000000000000000000000000000001000010000111111110000000000000
29 37 0000000000000000001000000000000000000000000000001000000000000000000000000000011111110001000000011000
30 38 0000000000000001000000000000000000000000000000000000000000000000000000000000111111100000000000010000
31 39 0000000000000000000000000000000000000000000000010000000000001000000000000000011010111000000000000000
32 40 0000000000000000000001000000000000000000000000000000000000000000000000000001111011100000000000000000
24 41 0000000000000000000000000000000000000000000000010000000000000000000000000000000000000111011000000100
25 42 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000101100010000000
26 43 0000100000000000000000000000000001000000000000000000000001000000000000000010000000000000011110100000000
27 44 0000000000000000000000000000000000000000000000000000000000001000100000000000000000000000111100000000000
12 45 0000000001000000010000000000000000000000000000000000000000000000000000000000000001000000000001101000000
13 46 0000000000000000000000000000000000000000000000000000000000000000000000000001000000000000001101000000
14 47 0010000000000100000010000010000000000010000000000000000000000000000000000011000000000000000110000000
15 48 0000000000000000001000000001000000000000000000000000000000000000010000000000000000000000000111000000
16 49 0000010000000000001000010000000000001000000000000000000000000000000000010000000000000010000000001111000000
19 50 0000000000000100000000000000000000000000000000010000111100110010000100000000000000000000000000100011000
```

Inter-cell flow : 0.9870,
Inner-cell density: 0.7922,
Clustering efficiency: 0.8896,
Limiting efficiency: 1.0000,
Relative efficiency: 0.8896,
Smallest block is liquidated.
⋮
Inter-cell flow : 0.9877,
Inner-cell density: 0.7887,
Clustering efficiency: 0.8882,
Limiting efficiency: 1.0000,
Relative efficiency: 0.8882,
Revert to earlier grouping.

S-11. Solution,

Cell # 1
Machines={ 1, 2, 3 }
Parts={ 1, 2, 3, 4, 5 }

Cell # 2
Machines={ 4, 5, 6, 7, 8, 9, 10, 11 }
Parts={ 7, 8, 9, 10, 11, 12 }

Cell # 3
Machines={ 33, 34, 35, 36, 37, 38 }
Parts={ 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76 }

Cell # 4
Machines={ 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50 }
Parts={ 77, 78, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 94, 95, 96, 97, 98, 99, 100 }

Cell # 5
Machines={ 17, 18, 20, 21 }
Parts={ 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 28, 29, 31, 32, 33, 34 }

Cell # 6
Machines={ 22, 23 }
Parts={ 6, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50 }

Cell # 7

Machines={ 28, 29, 30, 31, 32 }

Parts={ 58, 59, 60, 61, 62, 63, 64, 65 }

Cell # 8

Machines={ 24, 25, 26, 27 }

Parts={ 51, 52, 53, 54, 55, 56 }

Cell # 9

Machines={ 12, 13, 14, 15, 16, 19 }

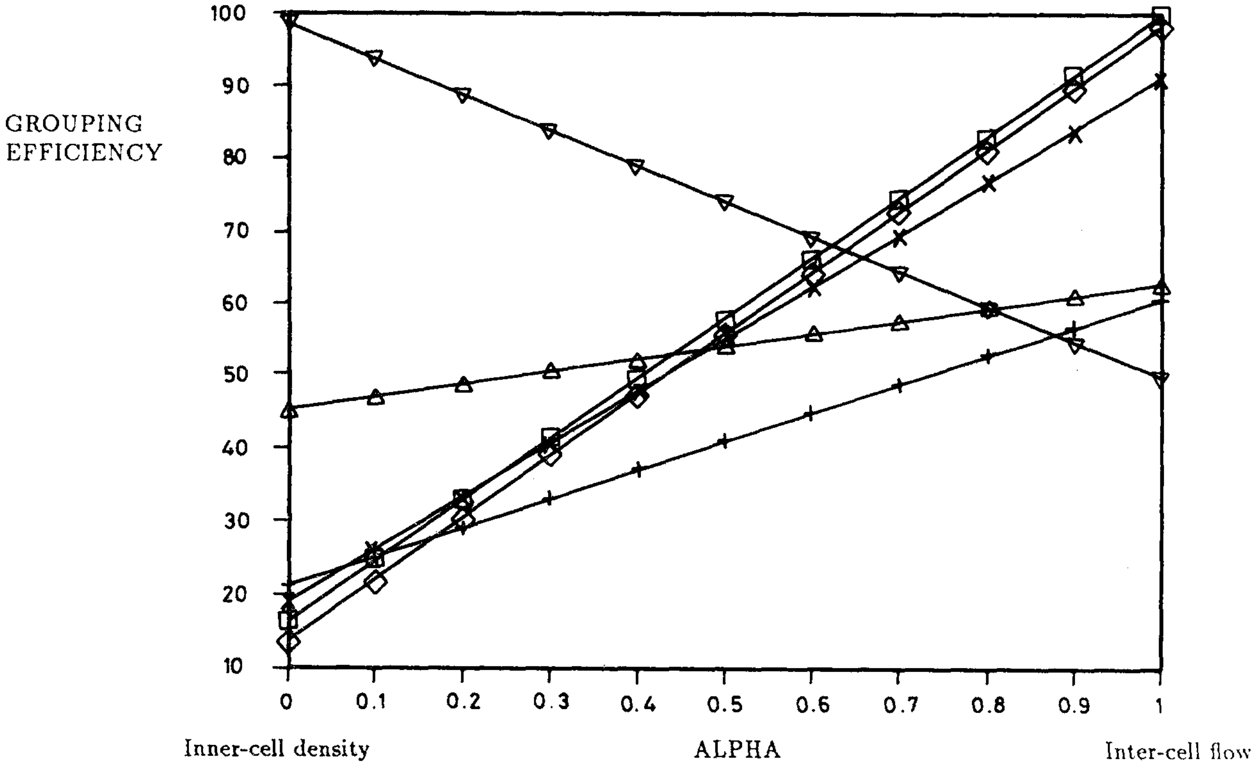
Parts={ 13, 14, 16, 16, 27, 30, 57, 79, 93 }

Efficiency measures, Inter-cell flow: 71.62 % ,
Inner-cell density: 83.32 %,
Work-load balance: 82.79 % ,
Under-utilization: 60.93 % .

APPENDIX B

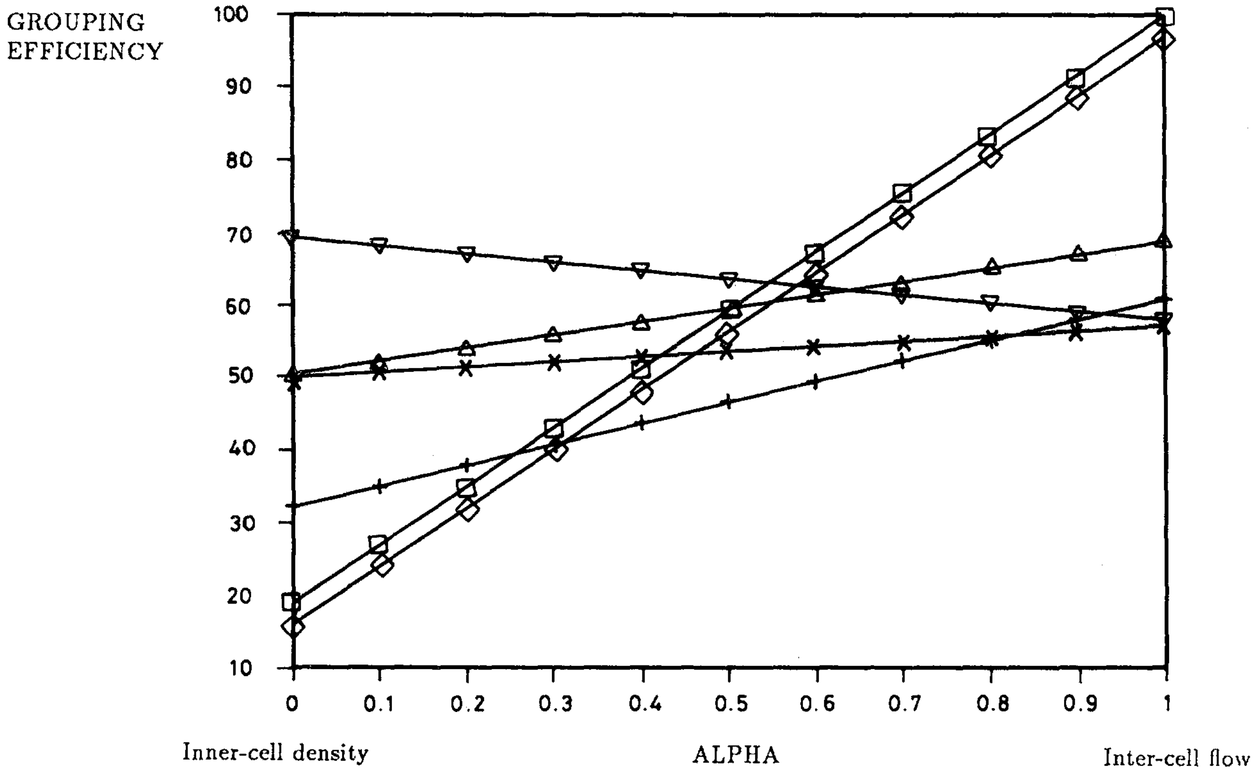
In this appendix, two way effects of the shape paremeters (clumpiness and density) on the selected PF/MG-F techniques are presented with respect to grouping efficiency measure, work-load balance measure, and under-utilization measure.

Two way effects of shape parameters in terms of grouping efficiency measure.

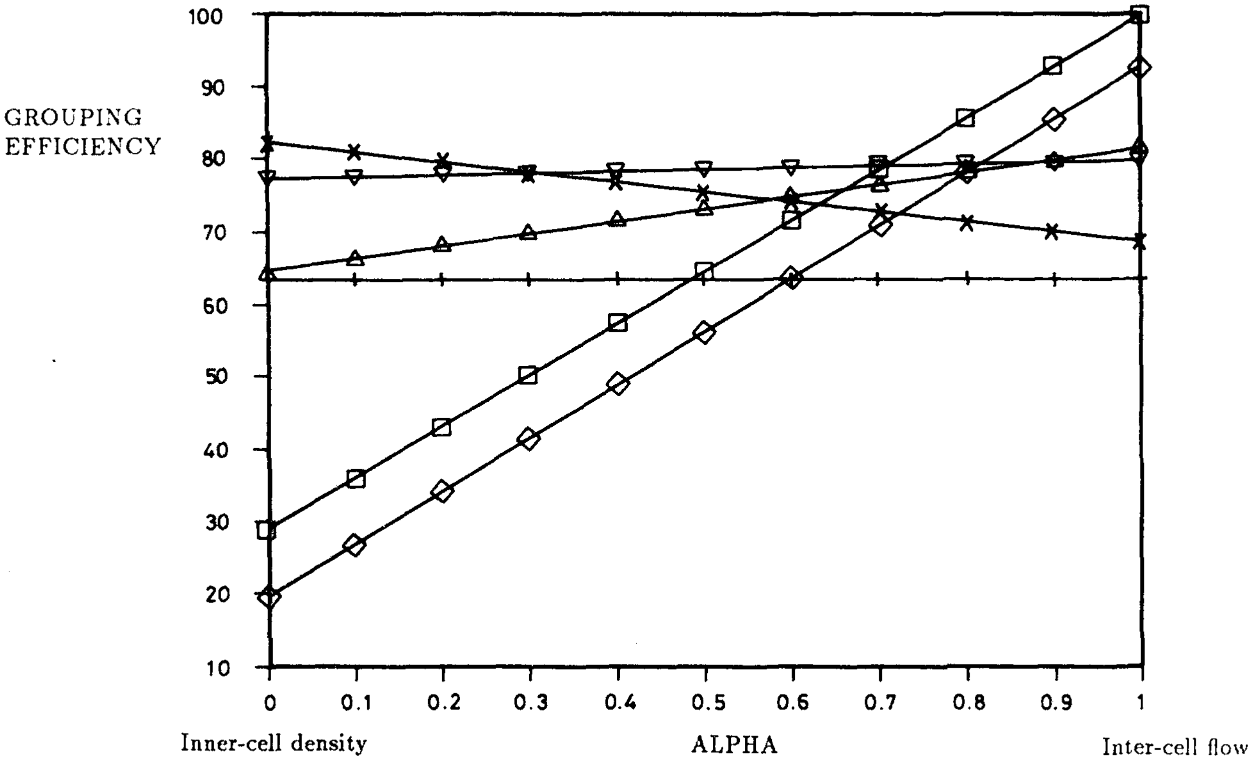


a) Clumpiness=1, Density=0.10

□ COMBGR + MODROC ◇WUBC △CAA X ZODIAC ▽MACE

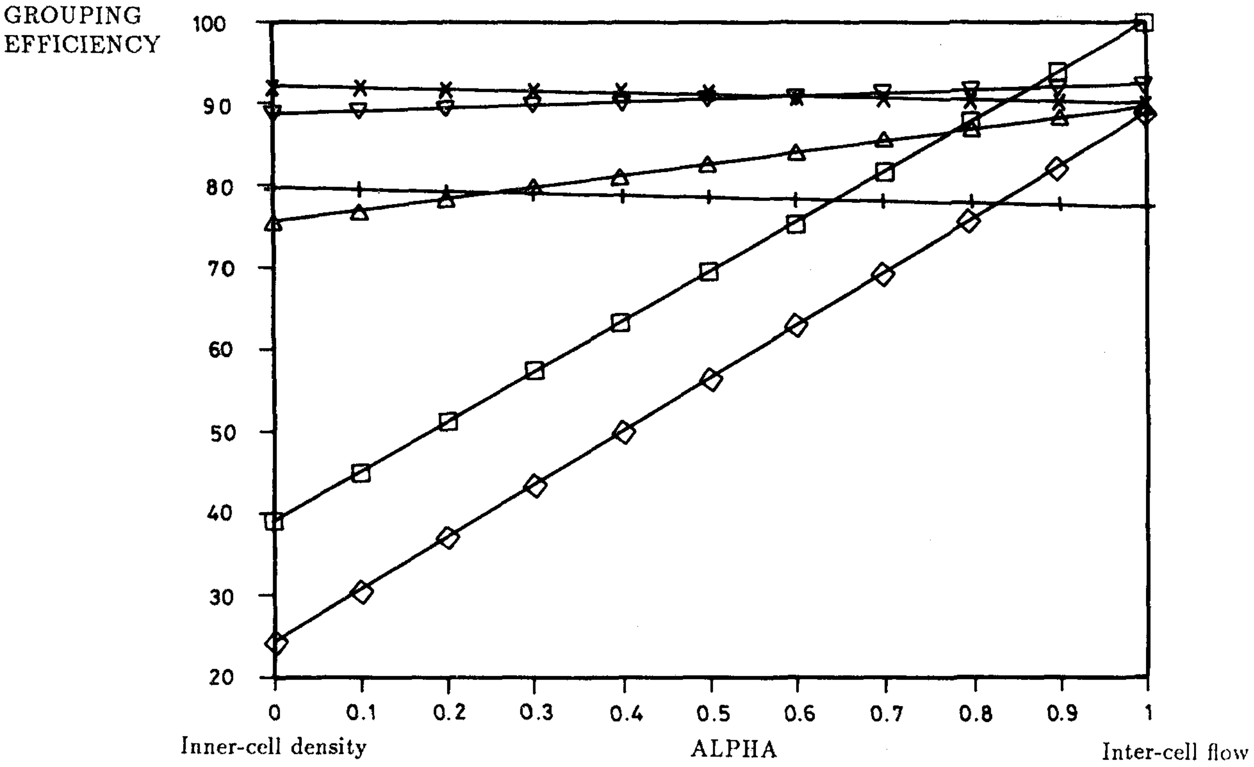


b) Clumpiness=2, Density=0.10

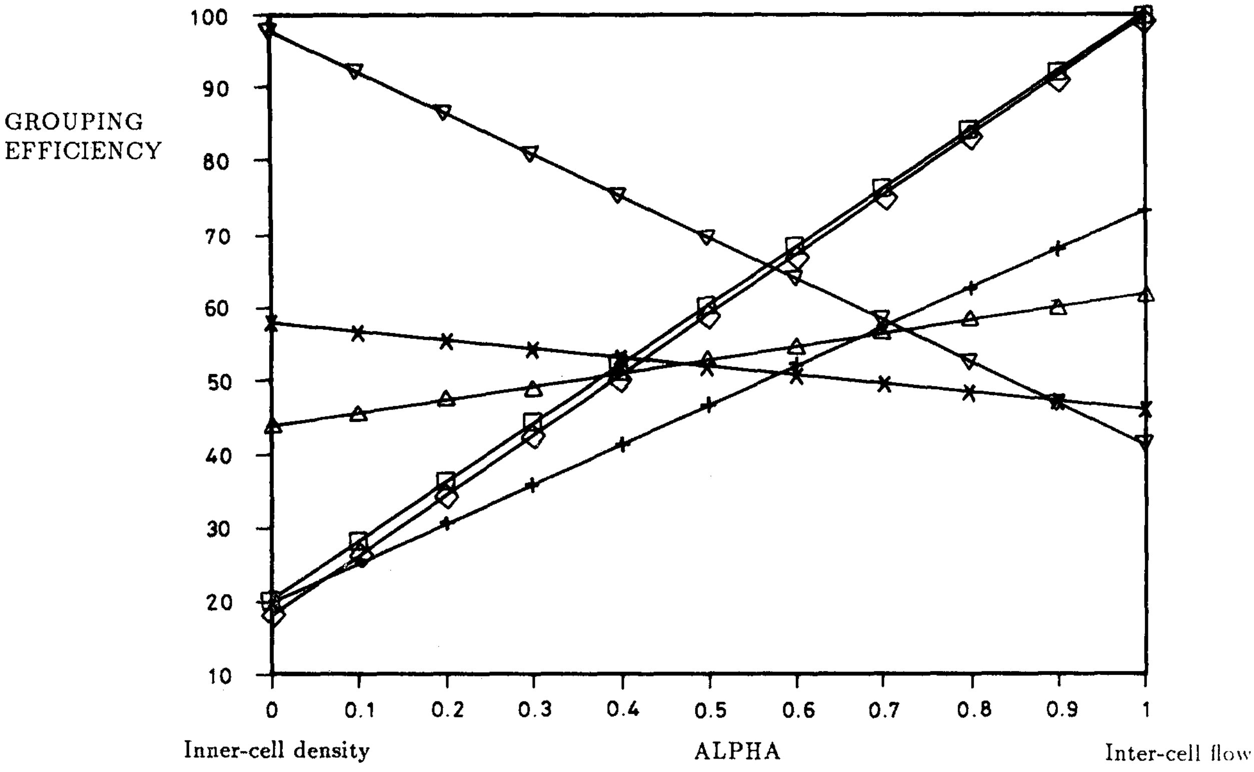


c) Clumpiness=4, Density=0.10

□COMBGR +MODROC ◇WUBC △CAA XZODIAC ▽MACE

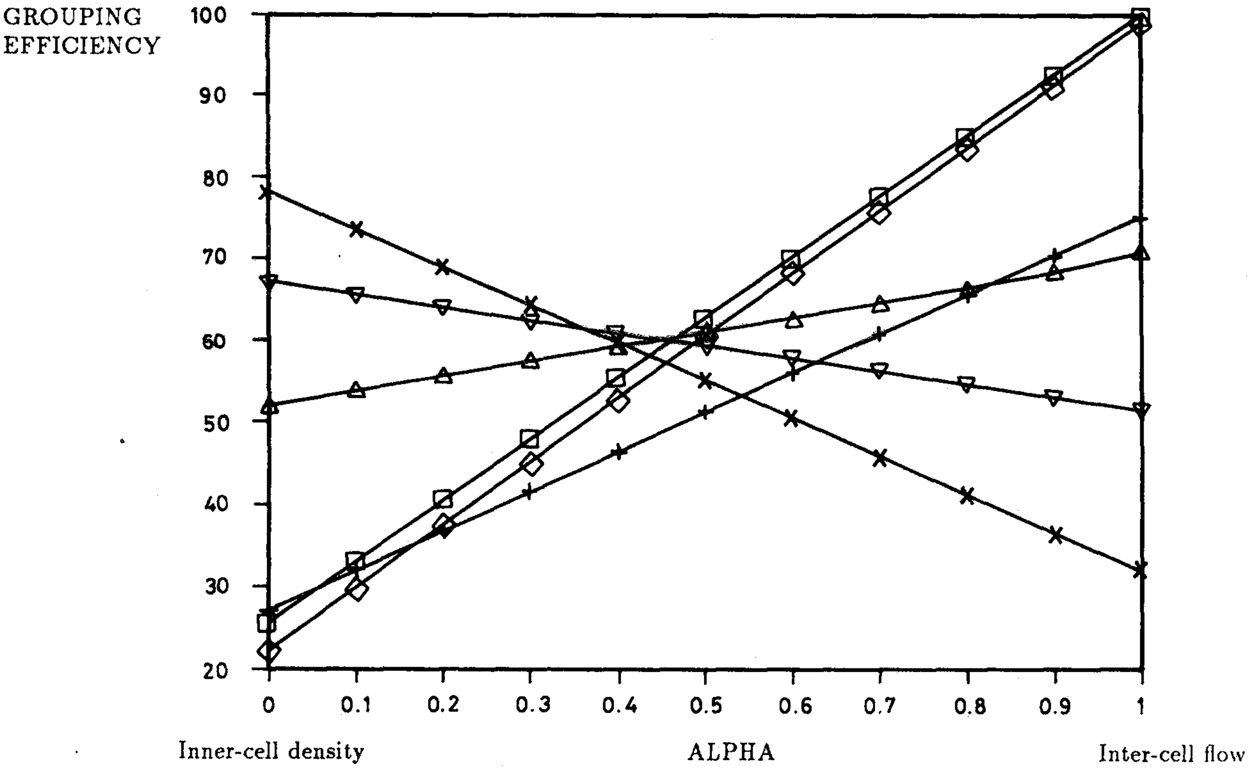


d) Clumpiness=9, Density=0.10

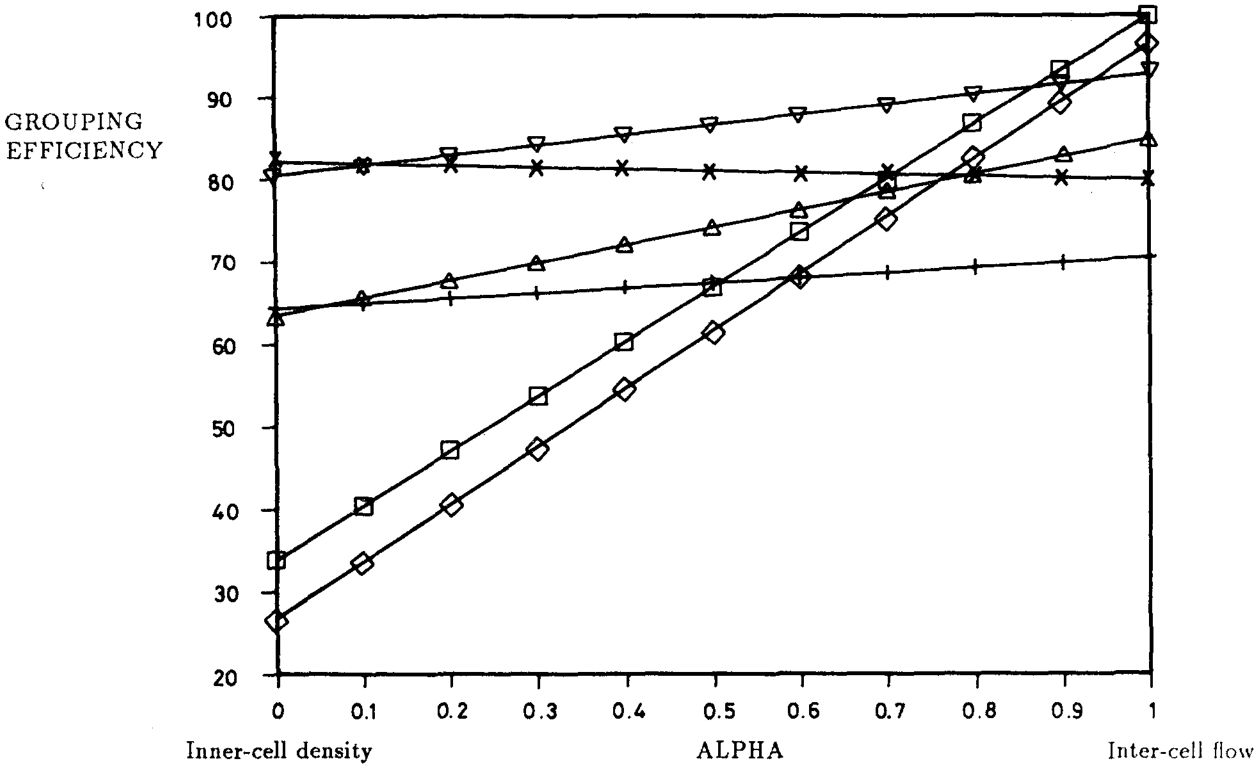


e) Clumpiness=1, Density=0.15

□ COMBGR + MODROC ◇ WUBC △ CAA X ZODIAC ▽ MACE

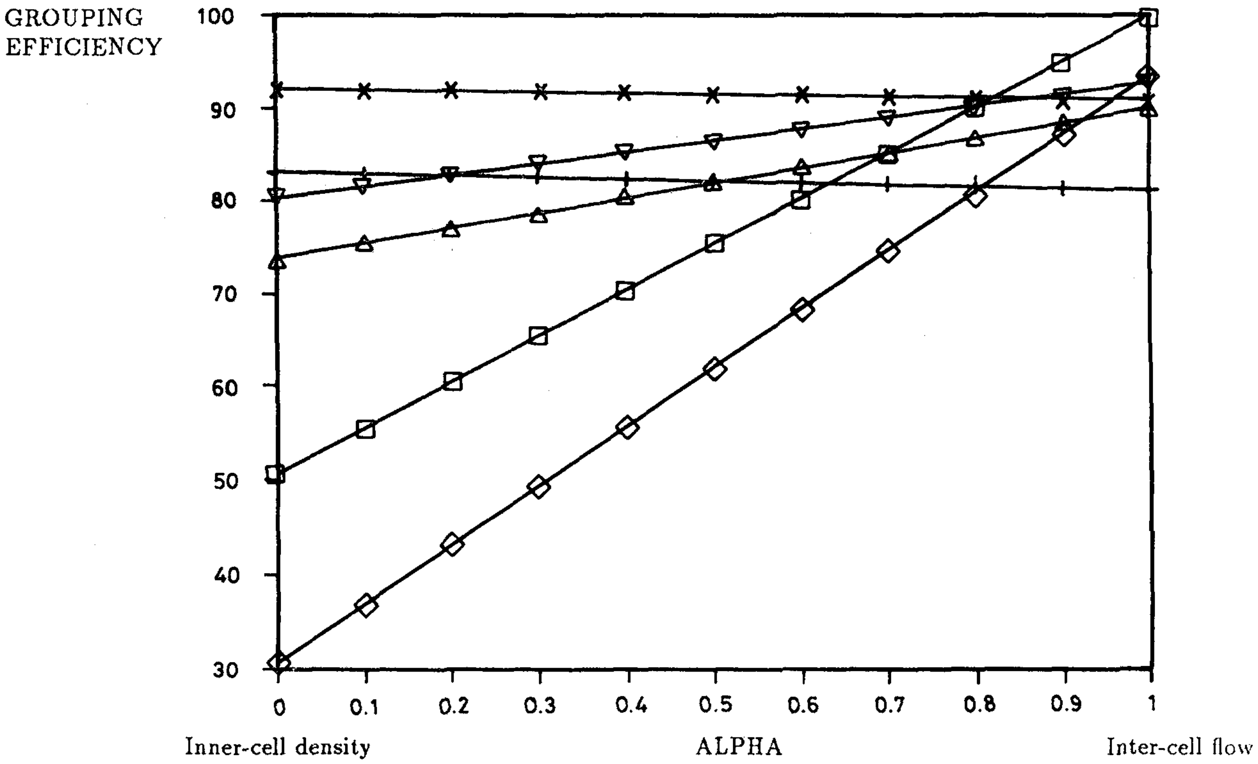


f) Clumpiness=2, Density=0.15

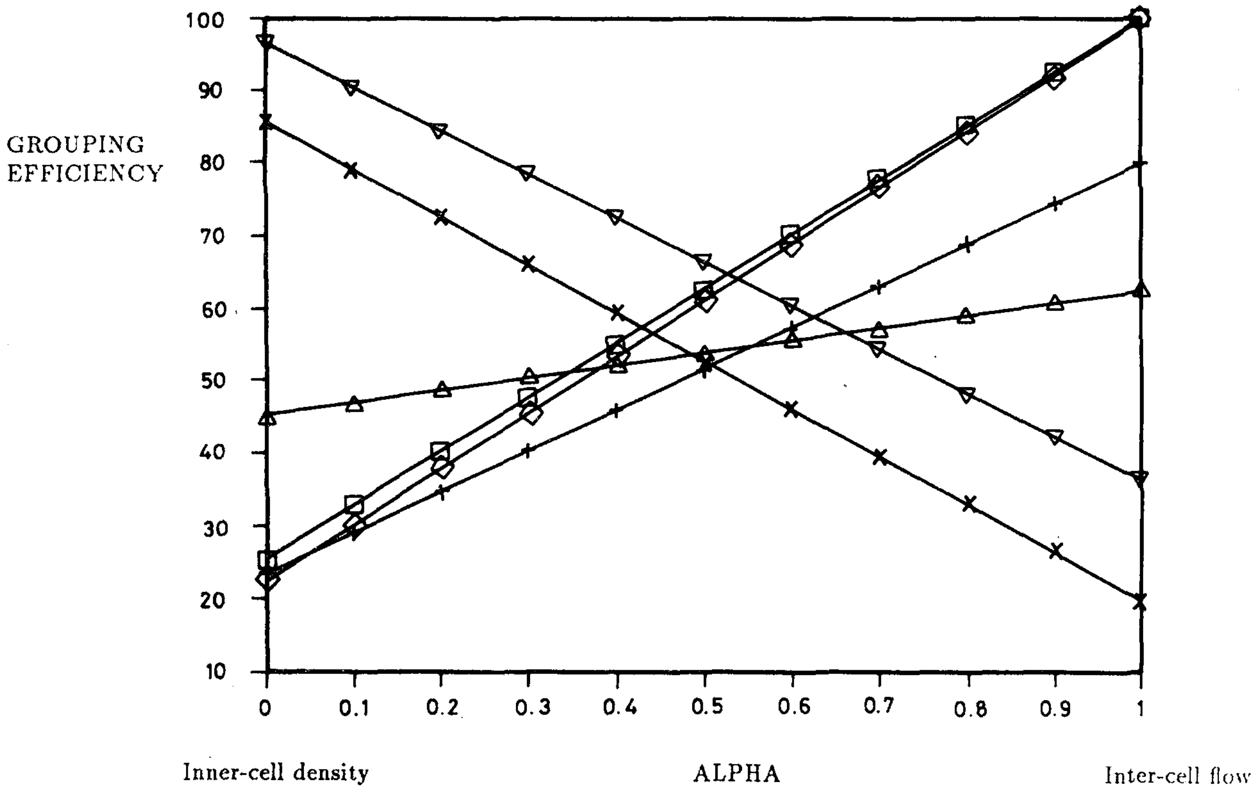


g) Clumpiness=4, Density=0.15

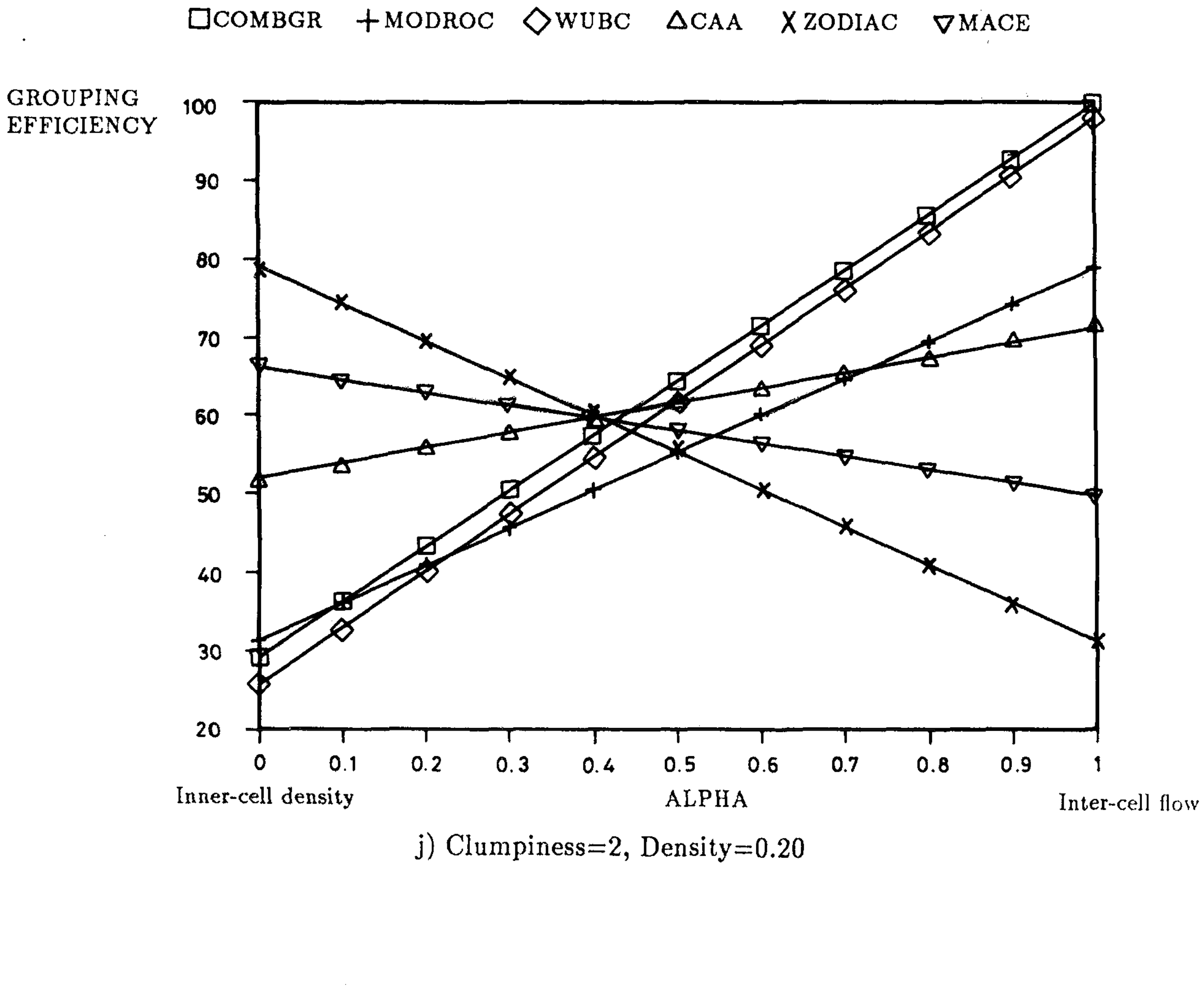
□ COMBGR +MODROC ◇WUBC △CAA X ZODIAC ▽MACE



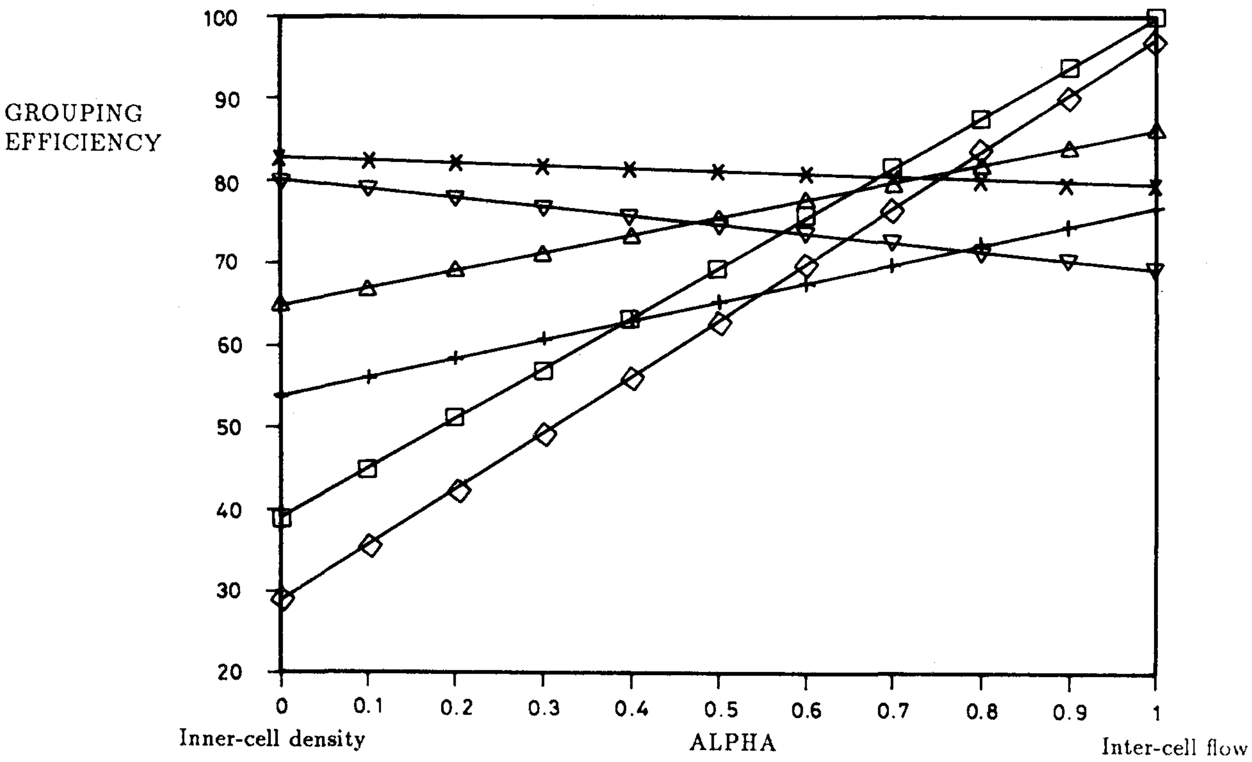
h) Clumpiness=9, Density=0.15



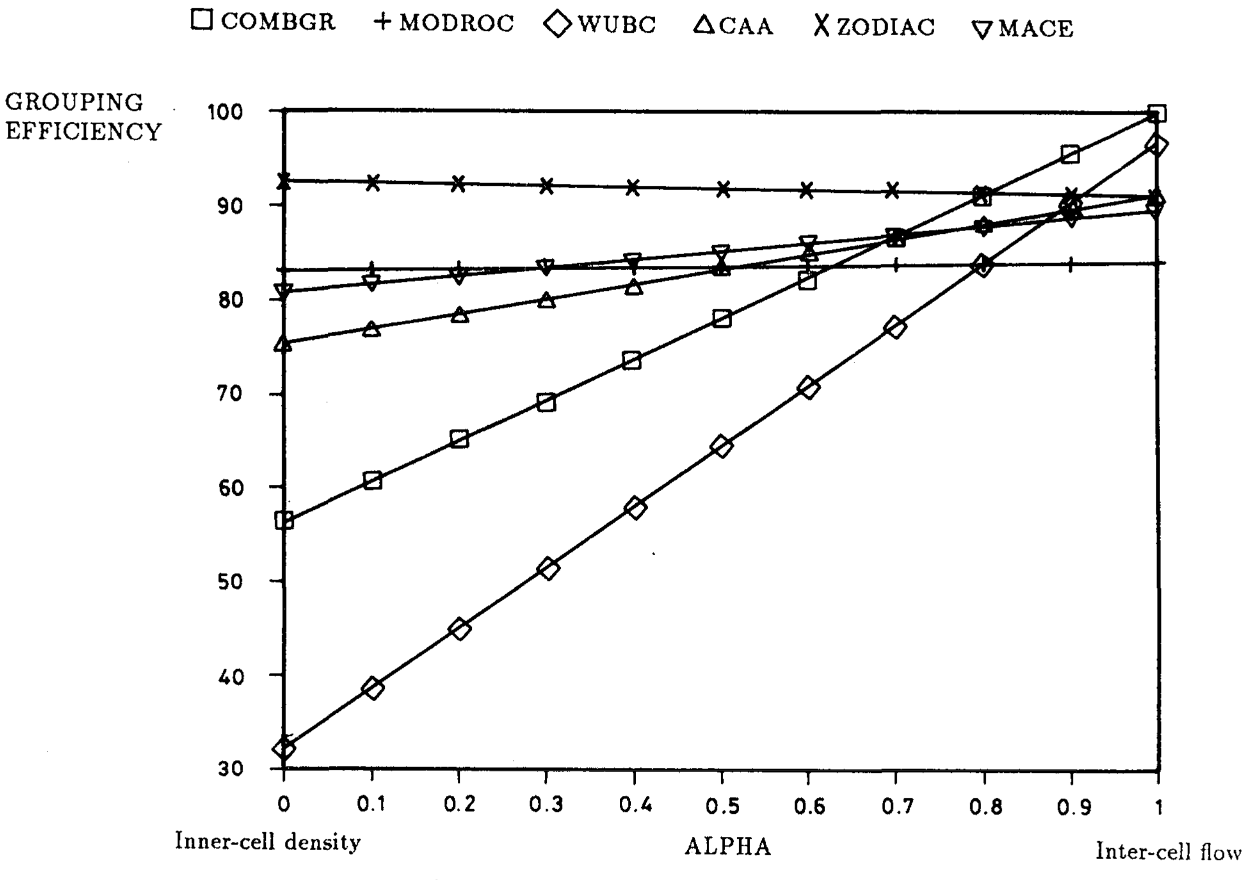
i) Clumpiness=1, Density=0.20



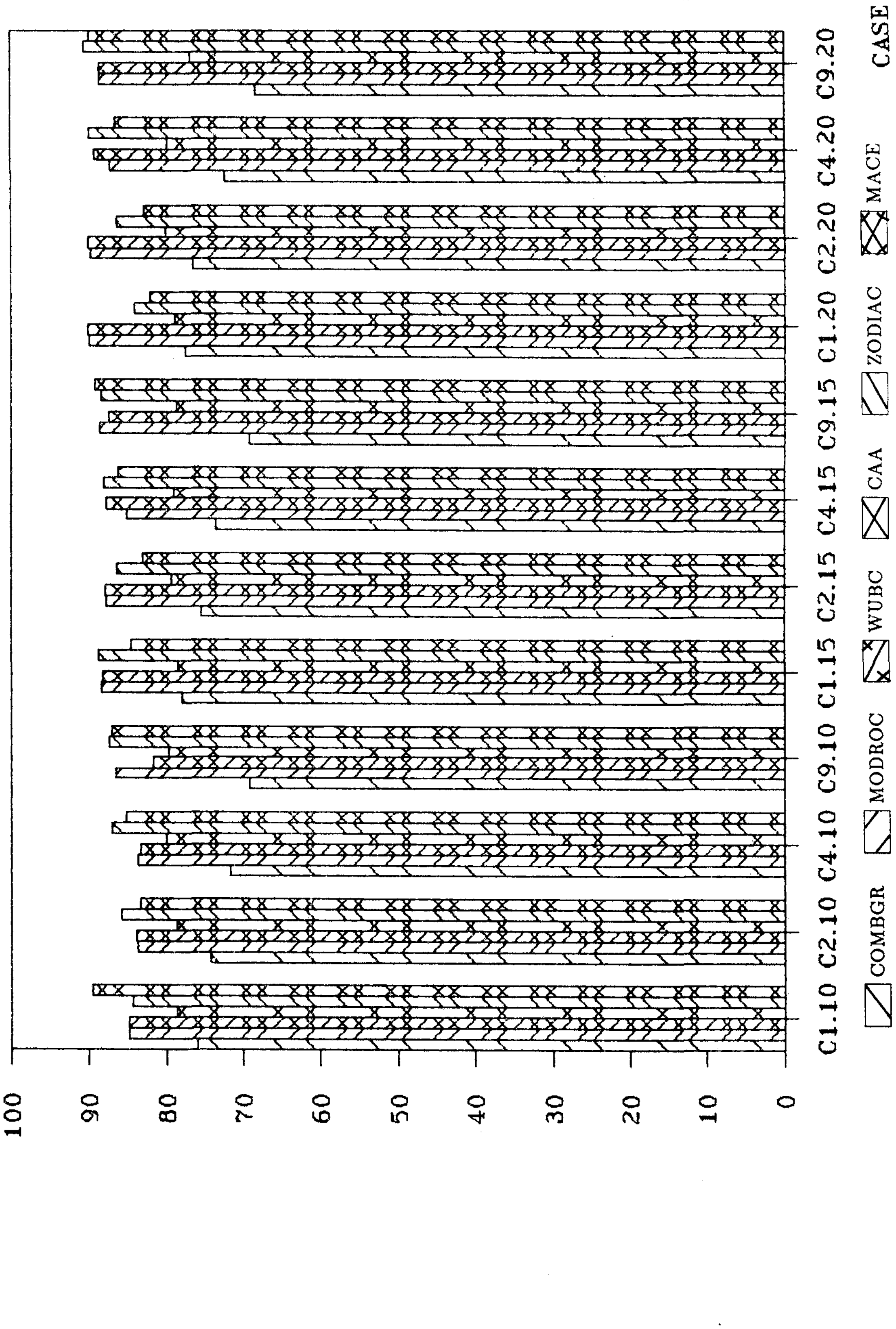
j) Clumpiness=2, Density=0.20



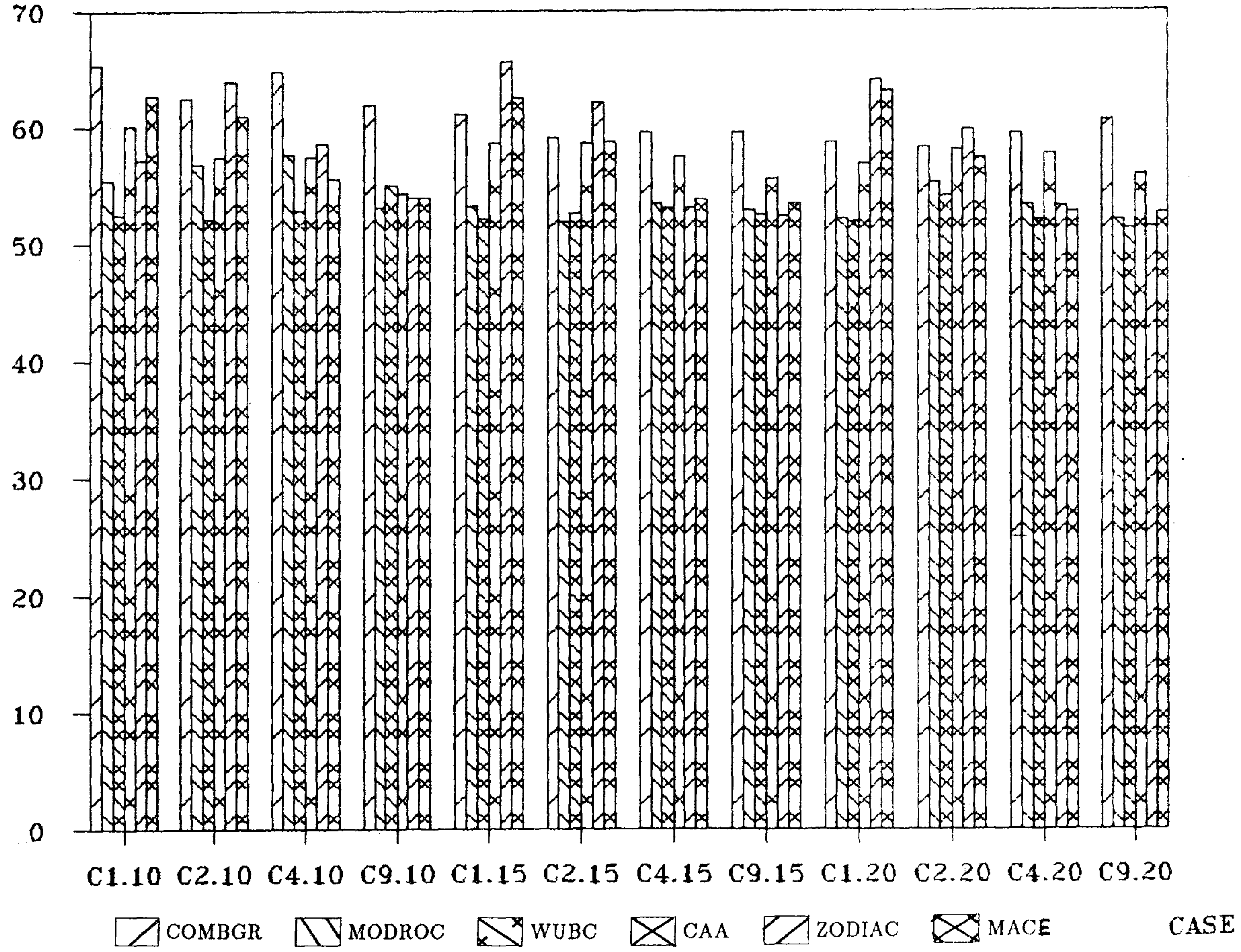
k) Clumpiness=4, Density=0.20



l) Clumpiness=9, Density=0.20



Two way effects of shape parameters in terms of work-load balance measure.



Two way effects of shape parameters in terms of under-utilization measure.

REFERENCES

[1] Askin, R. G. Subramanian, S. P. (1987), "A Cost–based Heuristic for Group Technolgy Configuration", *International Journal of Production Research*, Vol. 25, No. 1, pp. 101–113.

[2] Ballakur, A. (1985) *An Investigation of Part Family / Machine Group Formation in Designing a Cellular Manufacturing System*, Unpublished Ph. D. Thesis, University of Winconsin–Madison.

[3] Ballakur, A. Steudel, H. J. (1987), "A Within-cell Utilization Based Heuristics for Designing Cellular Manufacturing Systems", *International Journal of Production Research*, Vol. 25, No. 5, pp. 639–665.

[4] Black, J. T. (1983), "Cellular Manufacturing Systems Reduce Setup Time, Make Small Lot Production More Economical", *Industrial Engineering*, Vol. 15, No. 11, pp. 36–48.

[5] Burbidge, J. L. (1971), "Prodution Flow Analysis", *The Production Engineer*, Vol. 50, Nos. 4/5, pp. 139–152.

[6] Burbidge, J. L. (1975), *The Introduction of Group Technology*, Heinemann, London.

[7] Burbidge, J. L. (1982), "The Simplification of Material Flow Systems", *International Journal of Production Research*, Vol. 20, No. 1, pp. 117–133.

[8] Carrie, A. S. (1973), "Numerical Taxonomy Applied to Group Technology and Plant Layout", *International Journal of Production Research*, Vol. 11, No. 4, pp. 399–416.

[9] Chan, H. Milner, D. (1982), "Direct Clustering Algorithm for Group Formation in Cellular Manufacturing", *Journal of Manufacturing Systems*, Vol. 1, No. 1, pp. 65–75.

[10] Chandrasekharan, M. P. Rajagopalan, R. (1986a), “An Ideal Seed Non-hierarchical Clustering Algorithm for Cellular Manufacturing”, *International Journal of Production Research*, Vol. 24, No. 2, pp. 451–464.

[11] Chandrasekharan, M. P. Rajagopalan, R. (1986b), “MODROC: An Extension of Rank Order Clustering for Group Technology”, *International Journal of Production Research*, Vol. 24, No. 5, pp. 1221–1233.

[12] Chandrasekharan, M. P. Rajagopalan, R. (1987), “ZODIAC- An Algorithm for Concurrent Formation of Part–Families and Machine–Cells”, *International Journal of Production Research*, Vol. 25, No. 6, pp. 835–850.

[13] Choobineh, F. (1988), “A Framework for the Design of Cellular Manufacturing Systems”, *International Journal of Production Research*, Vol. 26, No. 7, pp. 1161–1172.

[14] Co, H. S. (1988), “Configuring Cellular Manufacturing Systems”, *International Journal of Production Research*, Vol. 26, No. 9, pp. 1523–1533.

[15] De Beer, C. De Witte, J. (1978), “Production Flow Synthesis”, *CIRP Annalen*, Vol. 27, pp. 389–441.

[16] De Witte, J. (1980), “The Use of Similarity Coefficients in Production Flow Analysis”, *International Journal of Production Research*, Vol. 18, No. 4, pp. 503–514.

[17] El-Essawy, I. Torrance, J. (1972), “Component Flow Analysis”, *The Production Engineer*, Vol. 51, pp. 165–170.

[18] Francis, R. L. White, J. A. (1974), *Facility Layout and Location*, Prentice–Hall, Inc. Englewood Cliffs, New Jersey.

[19] Greene, T. J. Sadowski, R. P. “Problems Associated with Cellular Manufacturing Control”, *Society of Manufacturing Engineers*, Technical Paper # MS82-941.

[20] Houtzeel, A. Brown, C. S. ”A Management Overview of Group Technology”, *Society of Manufacturing Engineers*, Technical Paper # MS82-167.

[21] Hyer, N. L. Wemmerlöv, U. (1982), "MRP/GT: A Framework for Production Planning and Control of Cellular Manufacturing", *Decision Sciences*, Vol. 13, pp. 681–701.

[22] Irastorza, J. C. Deane, R. H. (1974), "A Loading and Balancing Methodology for Job Shop Control", *AIIE Transactions*, Vol. 6, No. 4, pp. 302–307.

[23] King, J. R. (1980), "Machine-Component Grouping in Production Flow Analysis: An Approach Using a Rank Order Clustering Algorithm", *International Journal of Production Research*, Vol. 18, No. 2, pp. 213–232.

[24] King, J. R. Nakornchai V. (1982), "Machine–Component Group Formation in Group Technology: Review and Extension", *International Journal of Production Research*, Vol. 20, No. 2, pp. 117–133.

[25] Kumar, K. R. , Kusiak, A. ve Vannelli A. (1986), "Grouping of Parts and Components in Flexible Manufacturing Systems", *European Journal of Operational Research*, Vol. 24, pp. 387–397.

[26] Kusiak, A. (1987), "The Generalized Group Technology Concept", *International Journal of Production Research*, Vol. 25, No. 4, pp. 561–569.

[27] Kusiak, A. Chow W. S. (1987), "Efficient Solving of the Group Technology Problem", *Journal of Manufacturing Systems*, Vol. 6, No. 2, pp 117–124.

[28] Kusiak, A. İbrahim, W. M. (1988), " KBGT: A Knowledge-based System for Grouping Machines and Parts", Working Paper # 03/88, Department of Mechanical and Industrial Engineering, University of Manitoba, Winnipeg–Manitoba.

[29] Lawler, E. L. , Lenstra, J. K. , Rinnoy Kan, A. G. H. Shmoys D. B. (Editors) (1985), *The Travelling Salesman Problem: A Guided Tour of Combinatorial Optimization*, John Wiley, New York.

[30] MacQueen, J. (1967), "Some Methods for Classification and Analysis of Multivariate Observations", *Proceeding of the Fifth Symposium on Mathematical Statistics and Probability*, University of California, Berkeley, U. S. A. , Vol. 1, pp. 281–297.

[31] McAuley, J. (1972), "Machine Grouping for Efficient Production", *The Production Engineer*, Vol. 51, No. 2, pp. 53–57.

[32] McCormick, W. T. , Schweitzer, P. J. White, T. W. (1972), "Problem Decomposition and Data Reorganization by a Clustering Technique", *Operations Research*, Vol. 20, pp. 993–1009.

[33] Olivia-Lopez, E. Purcheck, G. F. (1979), "Load Balancing for Group Technology Planning and Control", *International Journal of MTDR*, Vol. 19, pp. 259–274.

[34] Purcheck, G. F. (1974), "Combinatorial Grouping – A Lattice-Theoretic Method for the Design of Manufacturing Systems", *Journal of Cybernetics*, Vol. 4, No. 3, pp. 27-60.

[35] Purcheck, G. F. (1985), "Machine–Component Group Formation: An Heuristic Method for Flexible Production Cells and Flexible Manufacturing Systems", *International Journal of Production Research*, Vol. 23, No. 5, pp. 911–943.

[36] Royden, H. L. (1968), *Real Analysis*, Second Edition, The Macmillan Company, Collier-macmillan Ltd./, London.

[37] Shunk, D. L. (1985), "Group Technology Provides Organized Approach to Realizing Benefits of CIMS", *Industrial Engineering*, Vol. 17, No. 41, pp 74–80.

[38] Sokal, R. R. Sneath, P. H. A. (1968), *Principles of Numerical Taxonomy*, Freeman.

[39] Rajagopalan, R. Batra, J. L. (1975), "Design of Cellular Production Systems: A Graph-Theoretic Approach", *International Journal of Production Research*, Vol. 13, No. 6, pp. 567–579.

[40] Vakharia, A. J. Wemmerlöv U. (1988), "Designing a Cellular Manufacturing System: A Materials Flow Approach Based on Operation Sequences", Working Paper, Graduate School of Business, University of Wisconsin-Madison.

[41] Vannelli, A. Kumar, K. R. (1986), "A Method for Finding Minimal Bottleneck Cells for Grouping Part–Machine Families", *International Journal of Production Research*, Vol. 24, No. 2, pp. 387–400.

[42] Waghodekar, P. H. Sahu, S. (1984), "Machine–Componenet Cell Formation in Group Technology: MACE", *International Journal of Production Research*, Vol. 22, No. 6, pp. 937–948.

[43] Wemmerlöv, U. Hyer, N. L. (1986a), "Procedures for the Part Family / Machine Group Identification Problem in Cellular Manufacturing", *Journal of Operations Management*, Vol. 6, No. 2, pp. 125–147.

[44] Wemmerlöv, U. Hyer, N. L. (1986b), "Research Issues in Cellular Manufacturing", *International Journal of Production Research*, Vol. 25, No. 3, pp 413–431.

[45] Wemmerlöv, U. (1988), *Production Planning and Control Procedures for Cellular Manufacturing Systems: Concepts and Practices*, APICS.