Development of Manufacturing Cells Using an Artificial Ant-Based Algorithm with Different Similarity Coefficients

Mohammed Taboun
University of Windsor

Development of Manufacturing Cells Using an Artificial Ant-Based with Different Similarity Coefficients

by

Mohammed Salem Taboun

A Thesis
Submitted to the Faculty of Graduate Studies
through Industrial and Manufacturing Systems Engineering
in Partial Fulfillment of the Requirements for
the Degree of Master of Applied Science at the
University of Windsor

Windsor, Ontario, Canada

2010

Development of Manufacturing Cells Using an Artificial Ant-Based Algorithm with Different Similarity Coefficients

by

Mohammed Salem Taboun

APPROVED BY:

____
Dr. Kevin Li
Odette School of Business

____
Dr. Guoqing Zhang
Department of Industrial and Manufacturing Systems Engineering

____
Dr. Michael Wang, Advisor
Department of Industrial and Manufacturing Systems Engineering

____
Dr. Ahmed Azab, Chair of Defense
Department of Industrial and Manufacturing Systems Engineering

November 10, 2010

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

Although there exists several ways of solving the cellular manufacturing problem, including several ant-based algorithms, many of these algorithms focus on obtaining the best possible answer instead of efficiency and ease of programming. These existing ant-based algorithms which use similarity coefficients do not compare the efficiency of using different similarity coefficients within the algorithm either. An existing artificial-ant based algorithm was modified so that it is easier to manipulate. This modification was necessary to apply the algorithm to cellular manufacturing. The original algorithm, AntClass uses Euclidean vectors to measure the similarity between parts, because similarity is used to group parts together instead of distances, the modified version uses similarity coefficients. The concept of heaping clusters was also introduced to ant algorithms for cellular manufacturing. Instead of using Euclidean vectors to measure the distance to the center of a heap, as is such in the AntClass algorithm, an average similarity was introduced to measure the similarity between a part and a heap. The algorithm was tested on five common similarity coefficients to determine the similarity coefficient which gives the better quality solution as well as the most efficient process.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

CHAPTER I

INTRODUCTION

General Overview

Globalization and the development of global markets and technology are progressively accelerated forcing companies in today's competitive environment to change dramatically to satisfy the urgency and variability of consumer tastes and demands. To cope with this trend, companies must develop/adopt novel approaches and practical strategies to deal with various production parameters such quantity (smaller batches), variety (larger diversity) and so on to optimize their production systems. Batch production is one of the strategies that can be used to meet customer demand of lower volume and small batches; however, this strategy cannot be easily adopted in the efficient serial production lines.

On the other hand, low volume/high-variety manufacturing parts can be produced in batches using the flexibility of functionally arranged machines with the additional expense of some inefficiency. Batch production accounts for 50 – 75 % of world manufacturing systems (Zhao and Wu 2000). The inefficiency stemming from the inherent functionally arranged production systems include high set-up/operation time ratios, excessive non-value added material handling activities, greater work in process, long lead times, waiting periods and throughput, which leads to lower manufacturing productivity. To overcome productivity and inefficiency concerns, the concept of manufacturing cells has been considered as an acceptable solution that compromises between the efficiency of production lines and the flexibility of batch production systems.

Cellular Manufacturing Systems (CMS) is considered as an application of Group Technology (GT) concept to factory reconfiguration and shop floor layout design (Irani et al. 1999). Although both terms CMS and GT are occasionally used interchangeably, GT is an area of study devoted to parts clustering and machine cells formation and considered as the starting point of cell design. Furthermore, Da Silveira (1999) puts the grouping process of parts and machines in a central position of CMS implementation techniques. The grouping process of classifying similar parts facilitates both design and manufacturing; where part shape similarity is helpful in design but process similarity is important in the arrangement of machines and manufacturing g facilities.

In most cases, parts with similar shapes share the same process requirements such as rotational parts and prismatic parts. However, some parts with dissimilar shape may share the same set of machining requirements to produce them and vice versa. Burbridge (1992) indicated that routing information alone is sufficient to design manufacturing cell. The relationship between parts and their process requirements in terms of machines are arranged in a 0/1 binary structured format of two-dimensional matrix, known as part-machine incidence matrix. Most of the grouping and clustering approaches use this data structure as a starting point to form part families and machine cells.

During the past few decades many approaches have been proposed for solving part families and machine cells formation that can be classified into several techniques; (1) classification and coding techniques; (2) array-based techniques; (3) similarity coefficient techniques; (4) graph theoretic techniques; (5) mathematical programming techniques; and (6) artificial intelligence techniques.

Classification and coding techniques ranges from basic visual classification of part geometry to sophisticated computer coding techniques. Array-based technique makes use of the binary information to form part families and machine cells simultaneously by sorting both the rows and columns of part-machine incident matrix alternatively to form cluster blocks around the matrix diagonal (King 1980). Similarity coefficient techniques in GT is pioneered by Mc Auley (1972), which makes use of a similarity index to determine the similarity coefficients either between parts or machines then use this information in a clustering algorithm to form part families and machine cells.

Graph theoretic technique makes use of the part-machine incident matrix to develop a graph whose vertices correspond to the machines and whose edges represents the relationship created machines and parts using them. Rajagopalan and Batra used this technique to partition the machine-machine graph into a number of sub-graphs by removing edges with weak relationships to form machine cells and allocating parts to part families.

In mathematical programming techniques a number of part families/machine cells formation models have been developed using integer programming, mixed integer programming and goal programming. Kusiak (1987) formulated the machine cell formation problem into a 0/1 integer programming model with the objective of maximizing the sum of similarities while considering different system constraints. Although, different design objectives and system constraints can be incorporated into a mathematical program, grouping is a NP-complete problem, heuristic methods and artificial intelligence (AI) techniques are widely used to solve this problem in a reasonable time.

Finally, a number of artificial intelligence techniques have been used to solve part families and machine cells formation during the past two decades. Some of these techniques include artificial neural networks, genetic algorithms and knowledge-base systems. Chow and Hawaleshka (1993) used knowledge-based systems to form machine cells. Moon (1990), Chu (1993), Kaparthi et al. (1993), and Venugopal and Narendran (1994) employed artificial neural net works to form part families and machine cells. Venugopal and Narendran (1992) and Islier (1998) used genetic algorithms to form part families and machine cells. Recently, swarm intelligence techniques also known as Ant search algorithms have been used to form part families and machine cells. Islier (2005), Kao and Fu (2006), Kao and Li (2008) and Zhao et al. (2008) used this techniques to form part families and machine cells.

Since efficient and optimal grouping are the primary steps to a successful CMS implementation, research in this field will continue to develop novel grouping techniques. The proposed research topic is devoted to the development of an efficient algorithm ant based swarm intelligent technique. Various similarity measures used to determine the association between parts and machines will be integrated into the ant clustering model. Also, the impact of the similarity measures on optimal grouping will be compared and evaluated.

Objectives of the Research

The objectives of the research are as follows:

To select and manipulate a multistage ant-based swarm intelligent algorithm that can be used to solve part families and machine cells formation problem.

To evaluate the impact of different similarity measures on both the efficiency of the manipulated ant-based swarm intelligent algorithm and to evaluate the quality of the developed solutions

Organization of the Research

The research in this thesis proposal is organized as follows:

Chapter 1:	Introduction
Chapter 2:	Literature review on various aspects of part family and machine cell formation within the context of CMS and GT.
Chapter 3:	Development of the Ant-Based Swarm Intelligent Algorithmic Model.
Chapter 4:	Analysis of various similarity measures and the assessments of their impact on the model efficiency and optimal grouping solutions.
Chapter 5:	Numerical examples to test the model and its application.
Chapter 6:	Conclusions and recommendations for future research.

CHAPTER II

REVIEW OF LITERATURE

Group technology is the first step stone for the design of manufacturing cells. During the past few decades several approaches have been proposed for solving part families and machine cells formation. These approaches can be divided into the following:

- Classification and coding techniques
- Array-based techniques
- Similarity coefficient techniques
- Graph theoretic techniques
- Mathematical programming techniques
- Artificial intelligent techniques

Review of literature based on the above classification is describes in the following sections.

Classification and coding techniques

Classification and coding (CC) systems can be used as tool for GT by providing a structure for the classification of parts into groups based on selected part attributes and by assigning specific code to each part (Groover and Zimmers 1984, Hyer and Wemmerlov 1984, 1985). Some of the earliest coding systems include "Optiz Sytem", developed in the 1960's in Germany and perhaps is the most widely known and used coding system at that time in Europe (Optiz 1970, and Optiz and Wiendahl 1971). It has been used for both machined and non-machined parts.

Another CC system developed during the 1960's is known as the "Bisch Birn". Basically, it is a coding shell customized to a particular firm's needs (Gombinski 1969, Hyde 1981, Hyer et al. 1989). More recent commercial coding systems take advantage of advanced computing technology databases (Tatikonda and Wemmerlov 1992). Examples of these systems include Decision and Classification System (DCLASS), Computer Aided Process Planning (CAPP) systems, Manufacturing Information Classification System (MICLASS) and several other commercial systems that integrate both design and manufacturing information in various databases.

Array-Based Techniques

In array based clustering techniques a machine part index matrix is constructed. This matrix consists of 0, 1 entries where an entry 1 in the (i, j) position means that machine i is used to process part j, and an entry 0 means that machine i is not used to process part j. Algorithms are developed that transform the original matrix into a more structured form, and consequently, result in the formation of part families (Al-Sultan, 1997). Some examples of matrix formulation methods are similarity coefficient methods, the bond energy algorithm, the cluster identification algorithm and the extended cluster identification algorithm.

El-Essawy and Torrance (1972) proposed a method called component flow analysis (CFA). In some respects, the methodology of CFA differs from the of Burbridge's PFA procedure in the sense that CFA first partitions the problem, where PFA does not.

McCormick et al. (1972) developed a method called the Bond Energy Algorithm. This algorithm involves the evaluation of so called "bond energy" in the part machine matrix. A bond is said to exist between a pair of adjacent row elements or column elements if the pair of elements both have non-zero values. The value of the bond is equal to the product of the two adjacent elements. The total bond energy of the matrix is equal to summation of the product of any two adjacent elements. The algorithm manipulates the columns and rows of the part-machine matrix and tries to find a matrix containing the highest total bond energy. This algorithm can identify part families and machine cells simultaneously but still needs extensive manipulation of the final part-machine matrix to form cells of the required size.

King (1980) developed the Rank Order Clustering (ROC) algorithm which rearranges the rows and columns of the initial machine incidence matrix in decreasing binary values to obtain a block diagonal form. However, the applicability of the algorithm was restricted by the strong dependence of the results on the initial order of the machine-part matrix and existence of storage problems created by the usage of binary value used for reallocation.

Chan and Milner (1982) developed the Direct Clustering Algorithm (DCA) to solve the part family and machine grouping problems for cellular manufacturing systems. The Direct clustering Algorithm has four stages:

1. Count the number of positive entries in each row and column of the part-machine matrix
2. Starting from the first column, transfer the rows with positive entries in that column to the top portion of the matrix
3. Starting from the first column, transfer the rows with positive entries in that column to the top portion of the matrix
4. Iterate between steps (2) and (3) until no further transfer is required.

This procedure allows user interaction to deal with the problems of the bottlenecks and exceptional elements when they occur.

Chandrasekharan and Rajagopalan (1986) proposed an ideal seed non-hierarchical clustering algorithm which involves three primary stages. In the first stage, the problem is formulated as a bi-partite graph which consists of a machine sub-graph and a part sub-graph. The k-means algorithm is then used to construct k parts and k machines by grouping vectors which are close together. In the second stage, a performance measure called group efficiency is used to compare different grouping alternatives. In the third stage, parts and machines are rearranged to the closest 'imaginary groups' in an attempt to improve the initial assignment.

Chow and Hawaleshka (1993) developed an algorithm to solve the machine grouping problem that minimizes the intercellular movements with allocating a new machine. It is observed that the total number of exceptional parts generated by the (n+1) total number of machine cells is always greater than those generated by n total number of machine cells.

Abdule-Wahab et al. (2006) presented a new hybrid algorithm for data clustering, based off of the scatter search algorithm. Scatter search operates on a small set of solutions and makes only a limited use of randomization for diversification when searching for globally optimal solutions. The method proposed automatically discovers cluster number and cluster centers without prior knowledge of a possible number of classes, and without any initial partition. This algorithm was used by Rabbani et al. (2007) to solve the dynamic cell formation problem.

Similarity Coefficient Methods

Several researchers have developed techniques to form the part families and machine cells based on similarity coefficients. The similarity measures are generally based on sequence of operations, the processing requirements of parts, the tooling requirements of parts and availability of the tools on the machines etc. The first similarity coefficient was developed by McAuley (1972) was the first to apply the Jaccard similarity coefficient (Jaccard, 1908) to the machine cell formation problem and is the most widely used in the literature (Yin & Yasuda, 2006). Most of the similarity/dissimilarity coefficients based on binary data that can be found in literature (Baulieu, 1989). However, only a handful of these measures has been suggested and investigated within the context of GT/CMS for the purpose of cell formation and machine groupings.

De Witte (1980) proposed three similarity measures which can be used in production flow analysis. Since two of these coefficients showing the absolute relations and mutual interdependence, they were considered mainly for cell formation. Threshold values for these three similarity coefficients were arbitrary selected. In addition, the approach requires classification of machines as primary, secondary and tertiary. Similarly, Waghhodekar and Sahu (1984) proposed the use of one of three similarity coefficients for Machine-components CeEll formation (MACE). Similarity coefficient machine pairs can be either (i) additive type; (ii) product type or (iii) based on total flow of common components.

Selvam and Balasubramanian (1985) developed a dissimilarity measure based on operation sequence of manufacturing components. The dissimilarity matrix input considered the total number of components and processing sequence of each one as well as the production volume per period and handling cost per move between consecutive work centers. Other research work that used dissimilarity measures in GT include; Dutta et al. (1986) who suggested a dissimilarity coefficient using operation sequences and Kamrani and Parsaei (1993) who proposed a weighted dissimilarity index based on a disagreement measure of both design and manufacturing attributes between pairs of parts.

Choobineh (1988) developed a similarity measure, which based on the most relevant attributes of manufacturing parts. These attributes include manufacturing operations and their processing sequence that can be easily determined from their process plans. Subsequently, the measure is used to form part families and machine cells. Information obtained from manufacturing process plans were also utilized by Guiasingh and Lashkari (1986) to develop a similarity measure that expressed the capability between two machines in processing a set of parts requiring both machines. Machine capability is defined in terms of the tools available and tooling requirements to process the parts. Similarly, Tam (1990) suggested another similarity measure based on the operation sequence of manufacturing parts to form part families and machine cell groupings.

Gupta and Seifoddini (1990) proposed a new similarity index which took into consideration relevant production data that should be included in the early stages of the machine-component grouping process. The important production parameters incorporated in the computation of similarity coefficient were pair-wise routing sequence, part-wise average production volume, and unit operation time for each operation performed. It was indicated that by incorporating important production that the proposed measure has advantages and disadvantages. Some of the advantages included higher coefficient values that were indirectly assigned to pairs of machines which process parts with larger workload and responds to large differences in demand among parts. Gupta

(1993) suggested a new similarity coefficient which assigned pair-wise similarity among machines with usage factors of all alternative routings.

Kusiak and Cho (1992) proposed two similarity measures, the first one is based on binary information where a block diagonal structure is impeded into the machine-part matrix and took into consideration basic and alternative process plans. Basically, it is a binary measure that indicated weather one's part's process plan is a subset of another part's process plan. The second one is a modified version that generalizes the first similarity measure. The modified version can be used for parts or machines when the value of the first similarity measure would have been zero.

Moussa and Kamel (1996) proposed a new similarity measure based on the information provided in process plans. The information taken into consideration included manufacturing processing sequence of parts and their processing times during the assignment process. Jeon et al. (1998) extended the use of manufacturing attributes to include machine failure. Jeon and Leep (2006) proposed a new similarity measure, which took into consideration the number of available alternative process routes when available during machine failure. It was indicated that the measure draws on the number of alternative routes during machine failure when alternative routes are available instead of drawings on other production attributes including; operations, sequence, machine capabilities, production volume, processing requirements or operational times.

Islam and Sarker (2000) proposed a new similarity coefficient that is able to reflect the extent of true similarity of pairs of machines or parts in an incident matrix. The new measure of similarity is called relative matching coefficient. Unlike other similarity measures, the proposed similarity coefficient has the capability of conforming to commonly known similarity properties defined in literature such as (i) No mismatch, (ii) Minimum match, (iii) No match, (iv) Complete match, and (v) Maximum match. The new similarity coefficient is used as an intermediate tool to form cohesive manufacturing cells.

Comparative Studies of Similarity Coefficients

One of the earliest studies conducted to compare the effectives of various similarity measures or coefficients was reported by Mosier (1989). The study applied a mixture model experimental approach to compare seven similarity coefficients and four clustering algorithms. The similarity coefficients that were examined are given in Table 2.1.

The four well-known algorithms used in this study are, (i) Single Linkage (SLINK); (ii) Complete Linkage (CLINK); (iii) Centroid (CENT); and Ward's Method (WARD) by using Monte Carlo simulation to generate 30 problems with 100 parts and 100 machines. Four performance measures were used to evaluate the goodness of generated solutions including; 9i) Simple matching measure; (ii) Generalized matching measure; (iii) Product moment measure; and (iv) intercellular transfer measure.

Table 2.1 List of Similarity Coefficients Examined (Mosier, 1989)

Similarity Coefficient Name	Reference
McAuley's (Jaquard format)	McAuley (1972)
Multiple Weighted Similarity Coefficient	Mosier and Tube 1985
Additive Weighted Similarity Coefficient	Mosier and Tube 1985
Modified Multiplicative Weighted Similarity Coefficient	Moiser (1985)
Modified Yule Coefficient	Bishop et al. (1975)
Modified Humann Coefficient	Holly and Guilford (1964)
Modified Baroni-Urbani abd Buser Coefficient	Romesburg (1984)

The results of this study indicated that McAuley's similarity coefficient and the modified multiplicative weighted similarity coefficient are preferable compared with other similarity coefficients. However, Shafer and Rogers (1993) pointed out some of the limitations including that three of four performance measures are for measuring how closely the solution generated by the cell formation procedures matched the original machine-part matrix. In addition, the original machine-part matrix may not necessarily be the best or even a good configuration. Only the intercellular transfer measure of performance is considering specific objectives associated with machine cell formation problem. Further research recommendations to examine clustering efficacy and other measures were also sighted.

Shafer and Roger (1993) compared 16 similarity coefficients in conjunction with four clustering algorithms using 11 small example data based binary machine-part data sets mostly from the literature. Part family and machine cell grouping results were evaluated using four performance measures. The use of small well structured data set with some of the performance measures may not provide the discriminatory power needed to separate superior, from good and good from inferior techniques. In addition, the use of well structured small data set may provide results with a little general reliability due to strong dependency on the original input data. (Anderberg, 1973; Milligan and Cooper 1987; Vakharia and Wemmerlöv, 1995).

Seifoddinin and Hsu (1994) studied three different similarity coefficients (Jaccard's similarity coefficient, weighted similarity coefficient, and commonality score) 30 machine-component grouping problems. Several performance measures were used to evaluate the clustering results including grouping efficiency, grouping efficacy and the grouping capability index. Results showed that the weighted similarity coefficient generates better solutions based on the number of exceptional parts. On the hand, it was observed that grouping efficiency, grouping efficacy and the grouping capability index were not consistent performance measures.

Vakharia and Wemmerlöv (1995) conducted a study to evaluate the impact of dissimilarity measures and clustering algorithm techniques on the quality of solution with respect to part family formation and machine cell groupings. Eight dissimilarity measures were studied in conjunction with seven clustering algorithms using 24 binary data sets. Results of this study revealed that high internal cell cohesiveness and low levels of machine duplication were shown to be conflicting goals. The study also revealed that performance is sensitive to many factors, notably the underlying data and the stopping parameters. It was indicated that more research work is needed to link data structures to choice of clustering technique and dissimilarity measure. Also, more work is needed to find measures and methods under which cell system solutions can be compared at the aggregate level while considering individual cell properties.

Yin and Yasuda (2005 & 2006) conducted a study to evaluate the performance of 20 similarity coefficients shown in Table 2. In addition, a total of 94 data sets obtained literature and another 120 generated deliberately were used in this study in conjunction with three clustering algorithms (Single linkage clustering, SLC; complete linkage clustering, CLC; and average linkage clustering, ALC) were used in this study. Nine performance measures were used to evaluate the grouping solutions. The performance measures are the following:

- Number of exceptional elements (EE),

- Grouping efficiency,
- Group efficacy,
- Machine utilization index (grouping measure, GM),
- Clustering measure (CM),
- Grouping index (GI),
- Bond energy measure (BEM),
- Grouping capability index (GCI), and
- Alternative routing grouping efficiency (ARG efficiency)

Table 2.2 Similarity Coefficients Compared (Yin and Yasuda, 2006)

Similarity Coefficient		Range	Definition
1	Jaccard	0 to 1	a/(a+b+c)
2	Hamann	-1 to 1	[(a+d)-(b+c)]/[(a+d)+(b+c)]
3	Yule	-1 to 1	(ad-bc)/(ad+bc)
4	Simple Matching	0 to 1	(a+d)/(a+b+c+d)
5	Sonenson	0 to 1	2a/(2a+b+c)
6	Rogers and Tanimoto	0 to 1	(a+d)/[2(a+d)+b+c]
7	Sokal and Sneath	0 to 1	2(a+d)/[2(a+d)+b+c]
8	Russel and Rao	0 to 1	a/(a+b+c+d)
9	Baroni-Urbani and Buser	0 to 1	$[a+(ad)^{1/2}]/[a+b+c+(ad)^{1/2}]$
10	Phi	-1 to 1	$(ad-bc)/[(a+b)(a+c)(b+d)(c+d)^{1/2}]$
11	Ochiai	0 to 1	$a/[(a+b)(a+c)^{1/2}]$
12	PSC	0 to 1	$a^2/[(b+a)(c+a)]$
13	Dot-Product	0 to 1	a/(2a+b+c)
14	Kulezynski	0 to 1	1/2[a/(a+b) + a/(a+c)]
15	Sokal and Sneath 2	0 to 1	a/[a+2(b+c)]
16	Sokal and Sneath 4	0 to 1	1/4[a/(a+b) + a/(a+c) + d/(b+d) +d/(c+d)]
17	Relative Matching	0 to 1	$[a+ (ad)^{1/2}]/[a+b+c+d+(ad)^{1/2}]$
18	Chandrasekharan and Rajagopalan	0 to 1	a/Min[(a+b), (a+c)]
19	MaxSc	0 to 1	Max[a/(a+b), a/(a+c)]
20	Baker and Maropoulos	0 to 1	a/Max[(a+b),(a+c)

Where:

a is the number of machines which produce both components *i* and *j*

b is the number of machines which produce only component *i*

c is the number of machines which produce only component *j* *d* is the number of machines which produce neither components *i* or *j*

Study results revealed that three similarity coefficients are more efficient and four similarity coefficients are inefficient for solving the cell formation problem. In addition, it was found that Jaccard similarity coefficient is the most stable similarity coefficient. It was indicated that further research is needed to consider some production factors such as production volume, sequences of parts and so on.

Based on the above review of similarity measures, it can be revealed that most of these measures assume that the demand for each product during the planning period remains constant. The demand and processing times are assumed to be known with certainty. This may not be true in many production environments, hence there is a potential for discrepancy in the design solutions. In addition, none of these measures takes into consideration production lot size for each product and production scheduling constraints.

Graph Theoretic Techniques

Graph theoretic methods convert a machine part index matrix into a hypothetical graph where the vertices represent machines and/or parts and the edges stand for the similarity coefficients between machines. Matula (1969, 1970) was the first to demonstrate the applicability of high connectivity in similarity graphs to cluster analysis. Matula's approach is based on the cohesiveness function. This function is defined for every vertex and edge of a graph G to be the maximum edge-connectivity of any sub-graph containing that element. Hartuv and Shamir (2000) adopted the same technique to develop a clustering algorithm, where similarity data is used to form a similarity graph. Vertices are corresponding to elements with similarity values above the threshold and clusters are highly connected sub-graphs whose edge connectivity exceeds half the numbers of vertices.

Rajagopalan and Batra (1975) used graph partitioning approach to solve grouping problem of machine cells. Input data derived from the route cards of the components in analyzed and used to derive a graph whose vertices correspond to the machines and whose edges represents the relationship created between machines by the components using them. Once machine cells are formed by using the graph partitioning approach, the parts are allocated to the machine cells and the number of machines of a particular type in each cell is determined. One of the limitations of this technique is that machine cells and part families are not formed concurrently.

Kumar et al. (1986) used the graph theoretic technique and solved a graph partitioning problem to determine machine cells and part families for a fixed number of groups with machine cell size boundaries. Subsequently, Vannelli and Kumar (1986) extend the work and developed graph theoretic models to determine machines that need duplication in order to obtain a perfect block diagonal structure. In addition, Kumar and Vanelli (1987) used similar techniques for determining parts to be subcontracted to obtain a perfect block diagonal structure. Solutions obtained from these methods are found to depend on the choice of initial pivot elements.

Askin and Chiu (1990) developed a heuristic graph partitioning procedure to solve machine assignment and cell formation problem. First, a mathematical programming model is developed to incorporate costs of inventory, machine depreciation, machine setup, and material handling. The formulation is then divided into two phase/sub-problems; first sub-problem assigned components to specific machines, then the second sub-problem grouped machine into cells. Then the sub-problems are solved using a heuristic graph partitioning procedure. Finally, an approach to determine the economic batch size is also included.

Vohra *et al.* (1990) proposed a network-based algorithm to minimize the amount of machining times performed outside the part primary cells. A non-heuristic network approach is used to form manufacturing cells with minimum intercellular interactions. The machine-part matrix containing machining times is represented as a network which is subsequently partitioned by using a modified Gomory-Hu algorithm to find a minimum intercellular interaction.

Sinh and Mohanty (1991) developed a method for selecting an efficient path in fuzzy multi-objective networks to solve the routing problem in the manufacturing cell. An application of the methodology was also illustrated as the process plan selection problem. Askin *et al.* (1991) proposed a formulation for machine and part grouping problem, so called Hamiltonian Path approach. The part-matrix incidence matrix was used to represent the problem. The jaccard's similarity measure was used to form a distance measure for each machine pair and part pair.

Wu and Salvendy (1993) developed a network (an undirected graph) model to partition the machine graph into cells by considering operation sequences. Two algorithms are used in this model. The first algorithm partitions the network by finding the minimum cut sets in the network so that the resultant interaction between cells is minimal. The second algorithm is a simplified version of the first algorithm by selecting seed nodes in partitioning the network to further reduce the amount of computation.

However, the solution from this method is not guaranteed optimal (minimum intercellular movements).

Kandiller (1998) presented a cell formation technique using the hyper graph representation of the manufacturing systems. The proposed method approximates the hypergraph model by graphs so that the cuts are less affected by the approximation. A Gomory-Hu cut tree of the graph approximation then can be obtained. The minimum cuts between all pairs of vertices are calculated easily by the means of means of this tree, and a partition tree is produced. An algorithm is also presented to cut the partition tree. This algorithm is subjected to an experimentation of randomly generated manufacturing situations.

Recently, Zhao et al. (2008) developed a mathematical model of part clustering of product family based on weighted directed graph technique. The model is extended to incorporate swarm intelligent algorithm to develop e-manufacturing model, which can be used to solve the part family formations and machine cell groupings in e-manufacturing environment for mass customization. It was indicated that the system can be used as a support technology for mass customization, which is very important to develop optimal formation of manufacturing cells and could be more efficient in e-manufacturing mode than in traditional manufacturing mode.

Mathematical Programming Techniques

In mathematical programming techniques a number of part families/machine cells formation models have been developed using integer programming, mixed integer programming and goal programming. The objective functions of such models include, maximizing specific similarity measures. Other functions may in the form of minimizing the intercellular movements of parts, or machine workload deviations. Most of these models incorporate some kind of system constraints such as available capacity and/or machine cell size

Kusiak (1987) developed two different models that are based on p-median clustering techniques. The problems are formulated as 0/1 integer program to form part families and machine cells with the objective of maximizing the sum of similarities while considering different system constraints. In some cases, the models have difficulties in assigning the initial p-value. Ben-Arieh and Chang (1994) modified the p-median model by introducing p, the number of machine cells into the objective function to overcome the difficulty of assigning an initial p value; thus improving the optimization process to form part families and machine cells. Won (2000), and Won and Lee (2004) modified the p-median models to include new measures of similarity between machine pairs to solve machine grouping problem and deal with disadvantages of previous models such as large number of binary variables and constraints.

Co and Araar (1988) proposed a three-stage procedure to from machine cells to process specific sets of jobs. A mathematical program is formulated in the first stage to assign operations to machines with the objective of minimizing the deviations between workload assigned to machines and the available capacity. System constraints were based the available machining times. A direct search algorithm is implemented to define the composition of manufacturing cells.

Askin and Chiu (1990) proposed a mathematical model and solution procedure for the group technology configuration problem. In this model, costs of inventory, machine depreciation, machine set up and material handling are first incorporated into a mathematical programming formulation. The formulation is then divided into two sub-problems in order to find a solution. A heuristic graph partitioning procedure is then proposed for each sub-problem. The first sub problem assigns components to specific machines. The second sub-problem groups machines into cells.

Rajamani et al. (1990) developed three mathematical programming models to simultaneously form part families and machine groupings to analyze the effects of alternative process plans on the utilization of resources. The first model assigns machines to parts while minimizing the total investment cost subject to machine capacity and available budget. The second model assumes that part families are known and selects a process plan for each part, required machine for each operation and the number of machines in different cells. The objective in this case is to minimize the total investment cost subject to the same system constraints described in the first model. The third model determines both part families and machine groupings simultaneously subject to the same set of limitations. Comparisons of cost functions for the three models are also provided.

Demodaran et al. (1992), Liang and Taboun (1992), Shafer et al. (1992) and Rajamani et al. (1992) developed mathematical programming models that simultaneously form part families and machine groupings which minimizes the intercellular movement of parts and their associated costs. System limitations such as machine capacities, exceptional elements and precedence relationships of parts are some of those constraints considered for different models.

Dahel and smith (1993) proposed two mathematical programming models to group parts and machines into predefined number of cells simultaneously. The first model takes into consideration available machine capacity and cell size as system constraint while minimizing intercellular movements of parts. The second model is formulated as a multi-objective mathematical program to from machine cells which are flexible and have minimum interactions. Bothe models are analyzed and examined under the inter-cell routing flexibility criteria.

Logendran (1993) proposed a 0/1 quadratic mathematical program to simultaneously form part-machine grouping and evaluate the effectiveness of this grouping techniques in CMS. The objective function considered in this model consists of maximizing unified measure of effectiveness evaluated as the weighted sum of total moves and cell utilization subject to certain operational constraints. The constraints in processing times, sequence of operations, available machining capacities and non-consecutive operations scheduled on the same machine. The model is extended to take into consideration multiple routings for each part.

Adil et al. (1993) proposed a mathematical model which would take into consideration investment and operational costs during the cellular manufacturing design process. The majority of the cell formation models in literature consider grouping of parts and machines, based on clustering techniques. The performance of manufacturing cells formed therefore indicates that the cellular systems perform more poorly in terms of work-in-process inventory, average job waiting time and job flow time than the improved job shops. These cells, on the other hand, have superior performance in terms of average move times and setup. The mixed integer model developed by Adil et al. (1993) illustrates the trade-off relationships between operational and investment costs.

Moon and Gen (1999) and Sofianopoulou (1999) formulated a 0-1 integer mathematical programming models which consider both machine duplication and alternative process plans to form machine cells. Several manufacturing parameters including production volume levels, machining capacities, processing times, and the size of machine cells are taken into account as system constraints. Different optimization techniques are used to solve each model including genetic and simulated annealing algorithms.

Baykasoğlu et al. (2001) proposed an integer multi-objective non-linear model to solve part family and machine grouping problem simultaneously. The model uses generic capability units which are termed as resource elements to define processing capabilities of machine tools. Also, it takes into consideration important objectives such as minimization of part dissimilarity associated with production requirements and processing sequence of parts, minimization of machine cell workload imbalance and minimization of extra capacity requirements for cell formation.

Slomp *et al.* (2005) considered a new type of virtual cellular manufacturing (CM) system is considered, and proposed a multi-objective design procedure for designing such cells in real time. Retaining the functional layout, virtual cells are addressed as temporary groupings of machines, jobs and workers to realize the benefits of CM. The virtual cells are created periodically, for instance every week or every month, depending on changes in demand volumes and mix, as new jobs accumulate during a planning period. The proposed procedure includes labor grouping considerations in addition to part-machine grouping and is based on interactive goal programming methods. Factors such as capacity constraints, cell size restrictions, minimization of load imbalances, minimization of inter-cell movements of parts and provision of flexibility are considered. In labor grouping, the functionally specialized labor pools are partitioned and regrouped into virtual cells. Factors such as ensuring balanced loads for workers, minimization of inter-cell movements of workers and providing adequate levels of labor flexibility are considered in a pragmatic manner.

Dafersha and Chen (2006) proposed a comprehensive mathematical model for the design of CMS based on tooling requirements of the parts and tooling available on the machines. The model incorporates dynamic cell configuration, alternative routings, lot splitting, and sequence of operations, multiple units of identical machines, machine capacity, and workload balancing among cells, operation cost, and cost of subcontracting part processing, tool consumption cost, setup cost, cell size limits, and machine adjacency constraints. Computational experience on small problems showed that a significant amount of cost savings can be achieved by considering system reconfigurations, lot splitting and system flexibility; and that there are significant differences on workload distribution among the cells, if workload balancing is not attempted.

Satoglu and Suresh (2009) proposed a goal-programming model for the design of hybrid cellular manufacturing (HCM) systems, in a dual resource constrained environment, considering many real-world application issues. The procedure consists of three phases. The initial phase involves a Pareto analysis of demand volumes and volatility. In the second phase, a machine-grouping phase is conducted to form manufacturing cells, and a residual functional layout. In this phase, over-assignment of parts to the cells, machine purchasing cost, and loss of functional synergies are attempted to be minimized. Following the formation of cells and the functional layout, a labor allocation phase (the third phase) is carried out by considering worker capabilities and capacities. The total costs of cross-training, hiring, firing and over-assignment of workers to more than one cell are sought to be minimized.

Arikan and Güngör (2009) proposed a new multi-objective fuzzy mathematical model for the cellular manufacturing system (CMS) design and its solution methodology. The goal of their m model is to handle two important problems of CMS design called cell formation and exceptional elements simultaneously in fuzzy environment. The objective functions of the model are minimization of the cost of exceptional element elimination, minimization of the number of outer cell operations and maximization of the utilized machine capacity. The fuzziness stems from model parameters which are part demand, machine capacity and the exceptional elements' elimination costs. To illustrate the model, an example problem with fuzzy extension is adopted from literature and computational results are obtained by using the two-phased solution procedure proposed in their study. The approach is performed to reach simultaneous optimal solutions for all objective functions. The model solutions are investigated by using well-known performance measures and also three problem-specific performance measures are proposed. The model is capable of expressing vagueness of all the system parameters and gives the decision-maker (DM) alternative decision plans for different grades of precision.

Artificial Intelligence Techniques

An artificial neural network is a mathematical model or computational model that tries to simulate the structure and/or functional aspects of the brain. It consists of an interconnected group of artificial neurons and it processes information using a connectionist approach. Artificial neural networks at adaptive forms of artificial intelligence and are capable of learning. Du (2010) has outlined several neural network clustering algorithms, including C-means clustering, competitive learning, and mountain and subtractive clustering..

The most well-known data clustering technique is the statistical C-means, also known as the k-means (Du, 2010). The C-means algorithm approximates the maximum likelihood (ML) solution for determining the location of the means of a mixture density of component densities (Moody & Darken, 1989). The C-means can be implemented in either the batch mode (Linde, Buzo, & Gray, 1980; Moody & Darken, 1989) or the incremental mode (MacQueen, 1967). The batch C means (Linde et al. 1980), is applied when the whole training set is available. The incremental C-means is suitable for a training set that is obtained on-line. In the batch C-means, the initial partition is arbitrarily defined by placing each input pattern into a randomly selected cluster, and the prototypes are defined to be the average of the patterns in the individual clusters (Du, 2010). When the C-means is performed, at each step the patterns keep changing from one cluster to the closest cluster ck according to the nearest-neighbor rule and the prototypes are then recalculated as the mean of the samples in the clusters (Du, 2010).

Competitive learning can be implemented using a two-layer neural network. The input and output layers are fully connected. The output layer is called the competition layer, wherein lateral connections are used to perform lateral inhibition. Based on the mathematical statistics problem called cluster analysis, competitive learning is usually derived by minimizing the mean squared error function (Tsypkin, 1973).

The mountain clustering is a simple and effective method for estimating the number of clusters and the initial locations of the cluster centers (Yager & Filev, 1994).

The method grids the data space and computes a potential value for each grid point based on its distance to the actual data points. Each grid point is a potential cluster center. The potential for each grid is calculated based on the density of the surrounding data points. The grid with the highest potential is selected as the first cluster center and then the potential values of all the other grids are reduced according to their distances to the first cluster center. The next cluster center is located at the grid point with the highest remaining potential. This process is repeated until the remaining potential values of all the grids fall below a threshold. However, the grid structure causes the complexity to grow exponentially with the dimension of the problem. The subtractive clustering (Chiu, 1994a), as a modified mountain clustering, uses all the data points to replace all the grid points as potential cluster centers. This effectively reduces the number of grid points to N (Chiu, 1994a).

Genetic Algorithms

Genetic Algorithms (GA) have been developed by Holland (1975) at the University of Michigan. Holland's research had two primary goals. The first was to abstract and rigorously explain the adaptive processes of natural systems. The second was to design artificial system software that retains the important mechanisms of natural systems. This approach has led to important discoveries in both natural and artificial systems science. Genetic algorithms start with an initial set of random solutions called the population. Each individual in the population is called a chromosome, representing a solution to the problem at hand. A chromosome is a string of symbols, and is usually a binary string. There are two kinds of operations encountered in genetic algorithms. The first is Genetic operations (crossover and mutations) and the second is evolution operation (selection or reproduction). Some of the GA applications include optimization, group technology and manufacturing cell formation. The following literature describes some recent developments in GA within the context of GT and machine cell formation.

Gonçalves and Resende (2004) proposed a new approach for obtaining machine cells and product families. This approach combines a local search heuristic with a genetic algorithm. The genetic algorithm uses a random keys alphabet, an elitist selection strategy, and a parameterized uniform crossover. Computational experiences performed on 34 different group technology problems, show that the algorithm performs remarkably well when compared other algoritms.

Jeon and Leep (2006) developed a methodology which can be used to form manufacturing cells using both a new similarity coefficient based on t he number of alternative routes during machine failure and demand changes for multiple periods. The methodology is divided into two phases. The first phase suggests a new similarity coefficient, and the second phase uses a genetic algorithm for cell formation. This GA considers the scheduling and operational aspects in cell design under demand changes. Finally, machines are assigned to part families using mixed integer programming.

Tariq *et al.* (2008) developed a hybrid genetic algorithm for machine-part grouping. This algorithm is an approach that combines a local search heuristic (LSH) with genetic algorithms (GA). The GA uses integer type representation, multipoint crossover and roulette wheel selection procedure. The computational experience done show that the algorithm converges to the best solution in the initial generations but also produces solutions that are as accurate as any result reported in literature. They also observed that the proposed algorithm was more consistent in terms of accuracy with respect to the problem size, when compared to other algorithms.

Venugopal and Narendran (1992) proposed a genetic algorithm approach to the machine-component grouping problem with multiple objectives. The algorithm is bi-criteria mathematical model with a solution procedure based on the genetic algorithm. This study is a first of its kind in group technology literature, and a successful demonstration of the application of genetic algorithm to the machine-component grouping problem. The algorithm is found to be effective in offering a collection of satisfactory solutions, which is essential in a multi-objective environment, to enable the decision maker to choose the best alternative. It is independent of the nature of the objective functions. It is inherently parallel and is capable of super linear speed-up in multi-processor systems. With the availability of parallel computers, this algorithm will be particularly useful in solving part-family problems in complex, large scale FMS environments.

Hsu and Su (1998) proposed a genetic algorithm based procedure to solve the cellular manufacturing grouping problem. More specifically, they aimed to minimize (i) total cost, which includes inter-cell and intra-cell part transportation costs and machines investment cots; (ii) intra-cell machine loading imbalance; and (iii) inter-cell machine loading imbalance under many realistic considerations. The procedure they proposed is extremely adaptive, flexible, and efficient; and can be used to solve real manufacturing grouping problem problems in factories by providing robust manufacturing cell formation in a short execution time.

Zhao and Wu (2000) proposed a genetic algorithm approach to the machine-component grouping problem with multiple objectives. These objectives were minimizing the costs due to intercell and intracell part movements; minimizing the total within cell load variation and minimizing exceptional elements. They developed specific genetic operators in order to make problem solving easier. During the cell formation process, the routing sequence of parts, production volume, workload balance and the constraints of cell number and cell-size are carefully considered. They argue that although taking alternative routes does increase the time consumption of the genetic algorithm, the calculation time is still very limited. The method developed by them is feasible for medium sized tasks.

Caux *et al.* (2000) addressed the problem of manufacturing cell formation with alternative process plans and machine capacity constraints. Given routings, capacities of machines and quantities of parts to produce, the problem consisted of grouping machines into manufacturing cells and in selecting one process plan for each part. The objective of their research was to minimize the inter-cell traffic, respecting machine capacity constraints. A new approach combining the simulated annealing method for the cell formation and a branch-and-bound method for the routing selection was proposed. This method permits the simultaneously solving of the cell formation problem and the part-routing assignment problem whereas other methods are based on two heuristics or algorithms: one of the two problems is then solved from the solutions of the second one. Although exact methods, like the branch and branch and bound method often lead to large computational times, the method they proposed provides solutions very quickly. This feature makes the method more robust to variations of production. Although acceleration processes have been introduced in the branch-and-bound, the method can be limited with large-sized problems or unconstrained problems due to calculation time.

Onwubolu and Mutingi (2001) proposed a genetic algorithm (GA) meta-heuristic-based cell formation procedure to solve the cell formation problem. The cell formation problem solved by them is to simultaneously group machines and part-families into cells, so that intercellular movements are minimized. Also included is an option for considering the minimization of cell load variation is included and another, which combines minimization of intercellular movements and cell load-variation, exists. The algorithm solves this problem through improving a cell configuration using the GA meta-heuristic. The number of cells required and lower and upper bounds on cell size are allowed to be specified. This makes the GA scheme flexible for solving the cell formation problems. The solution procedure was found to perform well on tested large-scale problems and published data sets.

Uddin and Shanker (2002) addressed generalized grouping problem where each part has more than one process routes. The problem of simultaneously assigning machines and process routes (parts) to cells was formulated as an integer-programming problem. The objective of minimization of inter-cell movements is achieved by minimizing the number of visits to various cells required by a process route for processing the corresponding part. The proposed a procedure based on genetic algorithm which was quite effective in finding the global optimal solution to the grouping problem within a reasonable time, since the GAs are inherently parallel and is capable of super linear speed-up in multiprocessor systems. With the availability of parallel computers, two sub-problems can be solved simultaneously and this algorithm is particularly useful in solving large size grouping problem.

Simulated Annealing

The simulated annealing methodology draws its analogy from the ‘annealing’ process used in the metallurgical industry. The process of annealing is a way metals are slowly cooled to produce low energy-state crystals. Simulated annealing is a heuristic search procedure for combinatorial optimization (Metropolis et al, 1953).

Sofianopoulou (1997) addressed the cell formation problem by modeling it as a linear integer programming problem with the objective of minimizing the number of intercellular moves subject to cell-size constraints and taking into account the machine operation sequence of each part. An interesting feature of the proposed formulation is that there is no need of specifying (before hand) the number of cells to be used, which is automatically adjusted within the solution procedure. A very efficient random search heuristic algorithm, based on the simulated annealing method, was adopted for its solution. The heuristic is tested on a number of problems and its performance was evaluated.

Saha and Bandyopadhyay (2009) proposed a multi-objective clustering technique which optimizes simultaneously two objectives, one reflecting the total “quality” present in the data set in terms of total compactness of the clusters, and the other reflecting the total symmetry present in the clusters of the data set. The algorithm uses a simulated annealing based multi-objective optimization method as the underlying optimization criterion and center based encoding is used. The multi-objective clustering technique is able to suitably evolve Assignment of points to different clusters is done based on the newly developed point symmetry based these cluster centers in such a way so that the two objectives are optimized ‘simultaneously’ distance rather than the Euclidean distance. Results on eight artificial and six real-life data sets show that the proposed technique was well suited to detect true partitioning from data sets with clusters having either the hyper-spherical shape or point symmetric structure. Results were compared with those obtained by five existing clustering techniques, one multi-objective clustering technique, MOCK, average linkage clustering algorithm, expectation maximization clustering algorithm, well-known genetic algorithm based K-means clustering technique (GAK-means) and a newly developed genetic algorithm with point symmetry based clustering technique (GAPS).

Lin *et al.* (2010) proposed a simulated annealing based meta-heuristic for solving the part-machine cell formation problem. The effectiveness of the proposed approach was compared to conventional algorithms across a set of part-machine cell formation problem s available in literature. The experimental results obtained indicate that the proposed approach is a state-of-the-art algorithm for part-machine cell formation problems, as seen through a comparison of the obtained results with the best-known solutions of the 13 conventional algorithms with respect to four types of performance measures. Given the difficulty in solving part-machine cell formation problems, the results obtained by the proposed simulated annealing based meta-heuristic may encourage practitioners to apply it to real world problems.

Swarm Intelligence

Swarms consist of many simple agents that have local interactions, including interacting with the environment. The emergence of complex, or macroscopic, behaviors and the ability to achieve optimal solutions as a team result from combining simple, or microscopic, behaviors (Hinchey *et al.*, 2007). Beni and Wang (1989) introduced the term swarm intelligence. Swarm intelligence techniques are population based stochastic methods used in combinatorial optimization problems in which the collective behavior of relatively simple individuals arises from their local interactions with their environment to produce functional global patterns. Swarm intelligence represents a meta-heuristic approach to solve a variety of problems.

Ant algorithms were first proposed by Dorigo et al. (1991) as a multi-agent approach to difficult combinatorial optimization problems such as the travelling salesman problem and the quadratic assignment problem (Dorigo, 1999). Ant algorithms were inspired by the observation of real ant colonies. Ants are social insects, they live in colonies and their behaviour is directed more to the survival of the whole colony as opposed to the survival of a single ant. An important behaviour of the ant colonies is their foraging behaviour, specifically, how ants can find the shortest paths between food sources and their nest (Dorigo, 1999). This behaviour has been a core foundation of recent research work and development of optimal cell formation.

Labroche et al. (2003) proposed an ant clustering system called AntClust. This algorithm is inspired from the chemical recognition system of ants. In the system proposed by Labroche et al (2003), the continuous interactions between the nest mates generate a "Gestalt" colonial odour. The Gestalt effect refers to the form-forming capability of our senses, particularly with respect to the visual recognition of figures and whole forms instead of just a collection of simple lines and curves (Hothersall, 2004). Similarly, this clustering algorithm associates an object of the data set to the odour of an ant and then simulates meetings between ants. Artificial ants that share a similar odour are grouped in the same nest, which provides the expected partition.

Runkler (2005), simplified the original ant system to create a generalized ant colony optimization system, which could be used to solve a wide variety of discrete optimization problems. This literature shows how objective function based clustering models such as hard and fuzzy c-means can be optimized using particular extensions of this simplified ant optimization algorithm. Experiments with artificial and real datasets show that ant clustering produces better results than alternating optimization because it is less sensitive to local extrema.

Islier (2005) proposed a method for solving the cellular manufacturing problem, by using an ant system algorithm in the group technology formulation. The method presented a technique where the grouping problem was first represented as an artificial ant system. The ants rearrange constantly obtaining a better grouping every cycle. These ants are semi-blind and use a communication-supported random search process. The data structure used by this ant system is the pheromone matrix. This matrix starts out empty, and is gradually formed by the experiences of the individual ants. The ant system uses this matrix to determine if the new grouping is better than the previous state.

Kao and Fu (2006) proposed a part clustering algorithm that used the concept of ant-based clustering in order to resolve machine cell formation problems. This three-phase algorithm mainly utilizes distributed agents which mimic the way real ants collect similar objects to form meaningful piles. In the first phase of this algorithm, an ant-based clustering model is adopted to form the initial part families. Kao and Fu part modified a part similarity coefficient and used it in the similarity density function of the model for the purpose of clustering. In the second phase, the K-means method is employed in order to achieve a better grouping result. In the third phase, artificial ants are used again to merge the small, refined part families into larger part families in a hierarchical manner. Kao and Fu (2006) argued that that this algorithm would increase the flexibility of determining the number of final part families for the factory layout designer.

Peterson et al (2008) introduced two improvements that can be incorporated into any ant clustering algorithm. These improvements kernel function similarity weights and a similarity memory model replacement scheme. A kernel function assigns a weight to each object within an ant's neighborhood according to the object distance and provides an alternate interpretation of the similarity of objects in an ant's neighborhood. In this ant clustering system, ants can hill-climb the kernel gradients as they look for a suitable place to drop a carried object. The similarity memory model equips ants with a small memory consisting of a sampling of the current clustering space. These improvements were compared to a basic ant clustering algorithm, and it was shown that kernel functions and the similarity memory model increase clustering speed and cluster quality, especially for datasets with an unbalanced class distribution, such as network intrusion.

Kao and Li (2008) proposed an ant colony recognition system for part clustering problems. This algorithm mimics the random meetings of real ants to build up the ability of object recognition and then to form many initial part clusters with high similarities. These initial part clusters are further merged into larger and larger clusters in a collective way until the designated number of part families is reached. The characteristics of artificial ants (such as randomization and collective behaviour) allow the algorithm to re-cluster wrongly grouped parts into the proper clusters. It is argued that this system can eliminate the chaining effects resulting from the interference of abnormal parts during the clustering process.

Motivation of The Research

In the literature, we have seen that there are several ways in which to solve the cellular manufacturing problem. Many of these existing methods however, are not as flexible as the swarm intelligence methods. Even though there are existing ant algorithm models, there has been limited comparison of the processes within the ant algorithm, with other replaceable processes. For example, the efficiency ant algorithms which use similarity coefficients have only been measured using that one similarity coefficient. Therefore there exists a need to investigate the effects that different similarity coefficients have on ant-based algorithm optimization techniques.

Most of the ant algorithm models in literature focus on developing part families to be as optimal as possible, rather than focusing on the efficiency of the algorithm itself. Many of the ant algorithm optimization techniques in literature are also developed into software, which are used by practitioners in the industry. The operational requirements of these ant algorithm software are very demanding. Thus, there exists a need for an efficient, easy to program ant algorithm that would create the optimal cellular manufacturing problem solution, and in doing so would use minimum resources. Therefore, it would be beneficial to this study to develop an ant algorithm that is efficient as well as easy to program, in order to compare the effects of different similarity coefficients on it.

CHAPTER III

DESIGN AND METHODOLOGY

There are several ways in which to solve the cellular manufacturing problem, however recent research has shown that optimization with artificial intelligence methods are more efficient for optimization. One of the advantages of swarm intelligence methods is that they are very flexible and efficient. In order to compare the effect of similarity coefficients on an ant algorithm, a modified version of Monmarché's AntClust has been created. The algorithm starts by creating an artificial environment for the ants to operate in.

The Environment: A Two Dimensional Chessboard

In order to have the artificial ants interact with the environment, an artificial environment must be created. The simplest way to do this is to create a grid on which the ants will move, pick up parts and drop parts. As in Monmarché *et al.*'s (1999) AntClass Algorithm, this is a two dimensional matrix *C* with a size of *m* x *m*.

The number of cells on this chessboard has to be greater than the number of ants added with the number of parts, in order for the artificial ants to move the parts and create families. If the chessboard is too large, there will be a lot of time wasted when the ants travel and relocate parts. Monmarché et al (1999) have determined that the size of the two dimensional chessboard has to be a function of the number of objects in order to scale automatically to the problem size, and have developed a formula to calculate the size of the two dimensional chessboard:

Equation 1 Calculating the size of the 2 dimensional chessboard

$$m^2 = n_p \times 4 \textbf{ OR } m = \sqrt{n_p \times 4}$$

Where m is the length and width of the two dimensional chessboard, and n_p is the number of parts. For example, a problem with four parts would need a chessboard of size 3 x 3 (Kao and Fu, 2006).

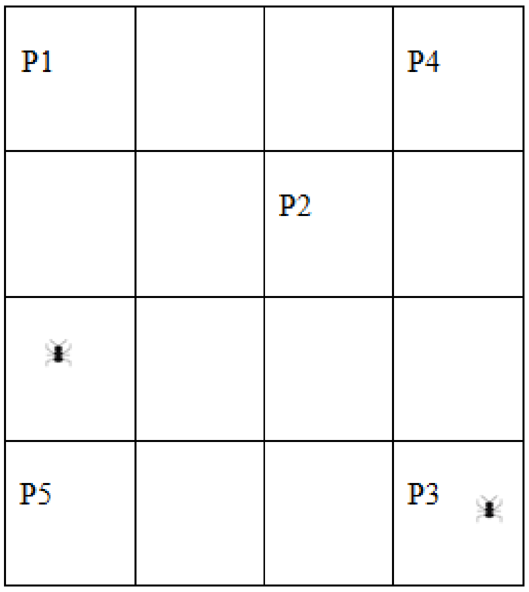


Figure 3. 1 Two Dimensional Chessboard

Along with a number of cells on the two-dimensional chessboard, there are a number of ants to be randomly spread with the parts. The number of parts has to be calculated so that there are not too many parts, along with enough parts to ensure that the algorithm will be completed quickly. The number of ants can be calculated as:

Equation 2 Calculating the number of artificial ants

$$n_{ants} = \frac{n_p}{10}$$

<u>Collection of Parts into Heaps</u>

Unlike the algorithm proposed by Kao and Fu in 2006, the artificial ant in this algorithm will be able to collect parts into heaps, and also build or destroy these heaps. A heap of parts (*H*) is a collection of two or more parts, and is located on a single cell on the two dimensional chessboard. A major advantage of having the ants collect parts into heaps is that a heap type cluster can easily be identified, while non-heaped special patterns of parts, such as that used by Kao and Fu (2006), may touch each other on the two dimensional chessboard. When spatial clusters touch each other, identification of clusters becomes difficult. Another advantage of using heaps is that more accurate heuristics for dropping or removing parts from these heaps are able to be defined (Monmarché et al., 1999). Ants will be able to remove the least similar part from a heap, or add a part to a heap if it is similar to the parts in the heap.

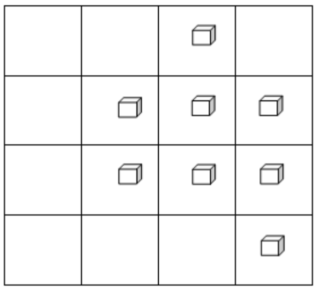


Figure 3. 2 Non-Heaped Cluster(s)

From the above image, it is difficult to tell whether this is one large cluster of five parts, or if it is a small cluster of three parts touching a small cluster of two parts.

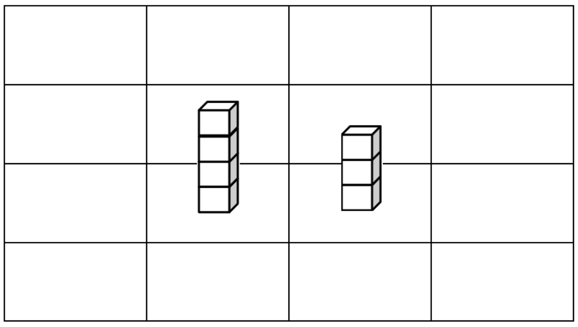


Figure 3. 3 Two clusters or distinct heaps

The image above shows a heap of parts. It is clearly identified as one large cluster of five parts.

Phase 1: Creating Part Families

The colony of artificial ants, or the artificial ants across the two dimensional chessboard, consists of p amount of ants. The artificial ants are labeled ant$_1$ ant$_2$...ant$_p$ and each artificial ant, ant$_i$, is located on one cell of the board. Initially, the artificial ants are spread out randomly. Each ant then moves according to the process outlined in Figure 3.4. The motion of each artificial ant is not completely random. Initially each artificial ant moves, and could possibly pick up or drop a part depending on its status. Each ant has a probability $P_{direction}$ to further continue in its direction when moving next. Each ant also has a speed parameter which tells it how many steps it will move in the selected direction before stopping. Once an ant has moved, it may pick up or drop a part, and this is repeated for a predefined number of steps. The ants will perform this algorithm until they reached the predefined number of cycles which is the number of parts multiplied by 500. After the ants have finished all of these steps, the algorithm then moves to Phase 2.

Ant is unloaded: Picking up a Part

When an ant is unloaded, it looks for a possible part to pick up by considering the eight (or six if it is on an edge) cells around its current position. As soon as one part or a heap of parts is discovered then the artificial ant will react based on whether there is one part, a heap of two parts, or a heap larger than two parts. If there is one part on the cell, the artificial ant will calculate the similarity density function $f(P_k)$ and the pickup probability $P_{pick}(P_k)$. After this, the artificial ant will compare the pickup probability to a randomly produced probability P_r. If $P_{pick}(P_k) > P_r$, the artificial ant will pick up the encountered part and its status will become loaded. If there is a heap consisting of two parts in the cell, then the artificial ant will pick up a random part. In this case, three random probabilities will be generated, P_{r1}, P_{r2} and $P_{destroy}$. P_{r1} will be associated with one part and P_{r2} to the other. Then the artificial ant will compare P_{r1} and P_{r2} to the random probability $P_{destroy}$. The part which has a random probability closer to $P_{destroy}$ will be treated as if it was the only part in the cell and possibly pick it up. Finally, if there are more than two parts in the heap then the ant will choose the least similar part out of the heap (provided that the part is beyond a predefined threshold) to pick up. It will then choose the least similar part from the heap and generate and random probablilty P_r. If P*r* is greater than the threshold value $P_{destroy}$, the artificial ant will then treat the least similar part as if it is the only part in the cell.

Equation 3: Similarity density function to measure the similarity of a part *P*$_k$ with its surroundings

$$f(P_k) = \frac{\sum s\,(P_k, P_l)}{nP_l}, P_l \in\ n^2$$

Equation 4: Probability transfer function for an artificial ant to pick up a part

$$\boldsymbol{P_{pick}(P_k)} = \left(\frac{\boldsymbol{k_p}}{\boldsymbol{k_p + f(P_k)}}\right)^{\boldsymbol{2}}$$

Where:

$f(P_k)$ similarity density function to measure the similarity of a part P_k with its surroundings

P_k is the part held or encountered by an artificial ant

P_l is the part located in one of the 3-8 surrounding cells on the 2-dimensional chessboard

$S(P_k, P_l)$ is the similarity between parts P_k and P_l

*n*2 is the surrounding area that is recognizable to an artificial ant (3-8 cells)

$P_{pick}(P_k)$ is the probability transfer function for an artificial ant to pick up the part P_k

k_p constant value with their range between $0 < k_p<1$

<u>Ant is Loaded: Dropping a Part</u>

When an ant is carrying a part, it will look at the eight surrounding cells. Then will act based on three conditions. The fist condition is encountered when the cell is empty. If this is the case, the artificial ant will compute $f(P_k)$ and $P_{drop}(P_k)$. The artificial and will then compare $P_{drop}(P_k)$ to a randomly generated probability P_r. If $P_{drop}(P_k) > P_r$, then the artificial ant will drop the part. Otherwise, the artificial ant will keep its status as loaded and continue. If there is a part in the cell already, the artificial ant will check to see if the similarity coefficient of the two parts is beyond the similarity threshold. If it is, the ant will drop the part and create a heap of parts. In the third case, the ant will compare the parts similarity to the heap. If the part is more similar to the heap than the least similar ant in the heap, then the ant will add the part to the heap.

Equation 5: Probability transfer function for an artificial ant to drop a part

$$P_{drop}(P_k) = \begin{cases} 2f(P_k) & if\ f(P_k) < k_d \\ 1 & otherwise \end{cases}$$

Where:

$f(P_k)$ similarity density function to measure the similarity of a part P_k with its surroundings

$P_{drop}(P_k)$ probability transfer function for an artificial ant to lay aside the Part P_k

k_d is a constant value with a range of $0 < k_d < 1$

Phase 2: Refining Part Families

Because the method in phase 1 uses random distributions, it tends to create many small homogenous part families. In order to improve the quality of the clustering, we use the K-means algorithm as done by Kao and Fu (2006) and by Monmarché *et al.* (1999). This is an important phase because the random clustering performed in phase 1 may have parts "inappropriately distributed to wrong part families" (Kao and Fu, 2006).

To perform the K-means algorithm the following steps must be taken. First, make the number of initial part families obtained in the first phase act as the number of K-means group, and then calculate the center vector (or average similarity) of each part family. If there are single parts (parts not in heaps), compute the average similarity to each of the closest heaps and add it to the most similar heap. Once there are no single parts left, assign each of the heaps a number. Start with the first heap, and find the least similar part in the heap (as was done in phase 1). Compute its average similarity with each of the other heaps. Select the heap with the highest similarity. If the part is more similar to the heap than the least similar part in the heap, move the part to this heap. Calculate the new center vectors for each heap, and then repeat for the second heap. Do this until no parts can be moved.

Equation 6: Average similarity between a part and a heap

$$\bar{s}(P_k, H_j) = \sum_{i=1}^{h} \frac{s(P_k, P_i)}{nP_h + 1}$$

Where:

H_j is the current heap

nP_h is the number of parts in H_j

h is the number of parts in the H_j

$\bar{s}(P_k, H_j)$ is the average similarity between a part and a heap

$s(P_k, P_i)$ is the similarity between part *k* and part *i*

Phase 3: Combining Part Families

In the first phase, many small homogenous part families are formed because of the random nature of the clustering. This randomness generally creates more part families than sought out. For the third phase, all of the refined part families are treated as single objects and scattered randomly across the two dimensional chessboard. Then, new artificial ants with part family merging "thought processes" are randomly dropped on the two dimensional chessboard. These ants re-cluster the families until a predefined number of part groups are reached.

When the objects (families) and ants are randomly scattered over the two dimensional chessboard, the family merging process begins. When an artificial ant comes across a family on the chessboard, it will generate a random probability (P_r) and compare it to a predefined probability called the Family Pick up probability (P_{fpu}). If the random probability is less than the family pick up probability, then the artificial ant will pick up the object and become loaded. If the randomly generated probability is not less than the family pick up probability, than the ant will move randomly past the part.

If a loaded ant comes across a part family, it will determine the distance between the two object centers (the total average similarity) and the maximum distance between the parts (the two least similar parts). The maximum distance is then divided by the average similarity and compared to a predefined similarity threshold. If the ratio is smaller than the threshold value then the two part families will be merged into one, and the ant will not be loaded with the new family.

These steps are repeated until the number of part families which were predefined by the facility designer are reached, or a predefined number of maximum steps have been reached. The reason that there are a maximum number of steps is included in this phase is to avoid infinite looping.

Performing the ant algorithm

Performing the ant algorithm by hand can prove to be a long and tedious process, as it greatly depends on numbers stored in charts along with keeping track of many random numbers simultaneously. Therefore, this algorithm was performed using a small program made in Borland C++. An example of the source code for this program, using on similarity coefficient can be found in Appendix D.

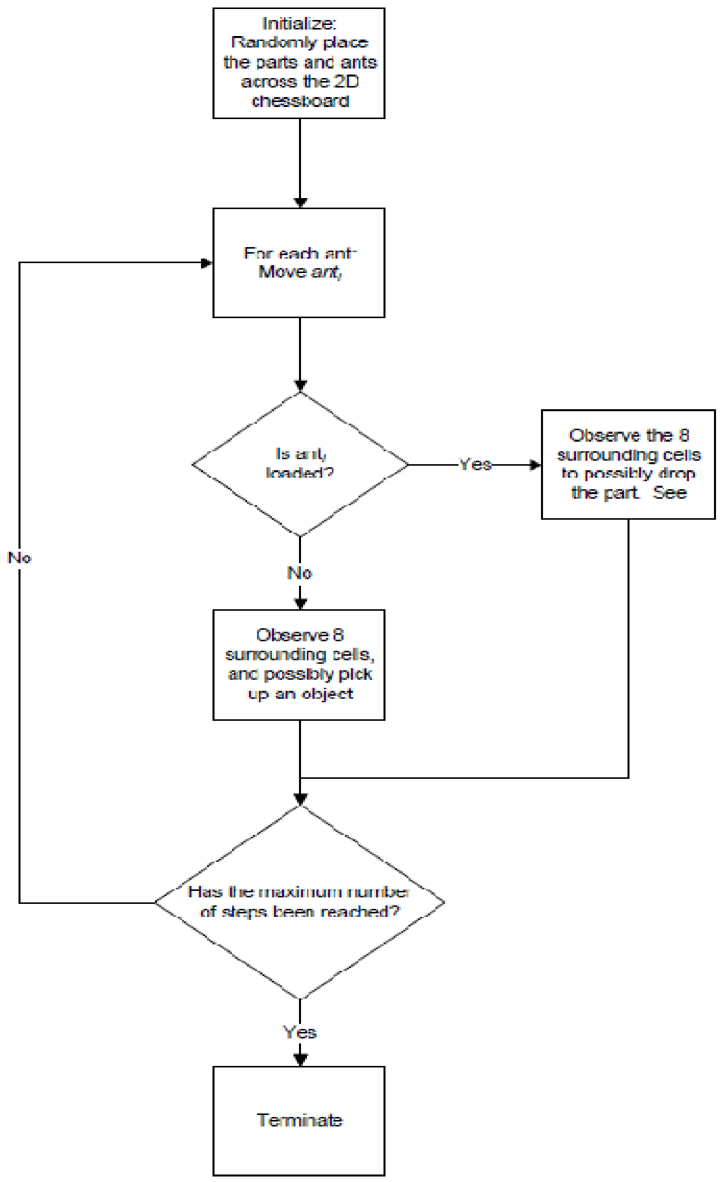
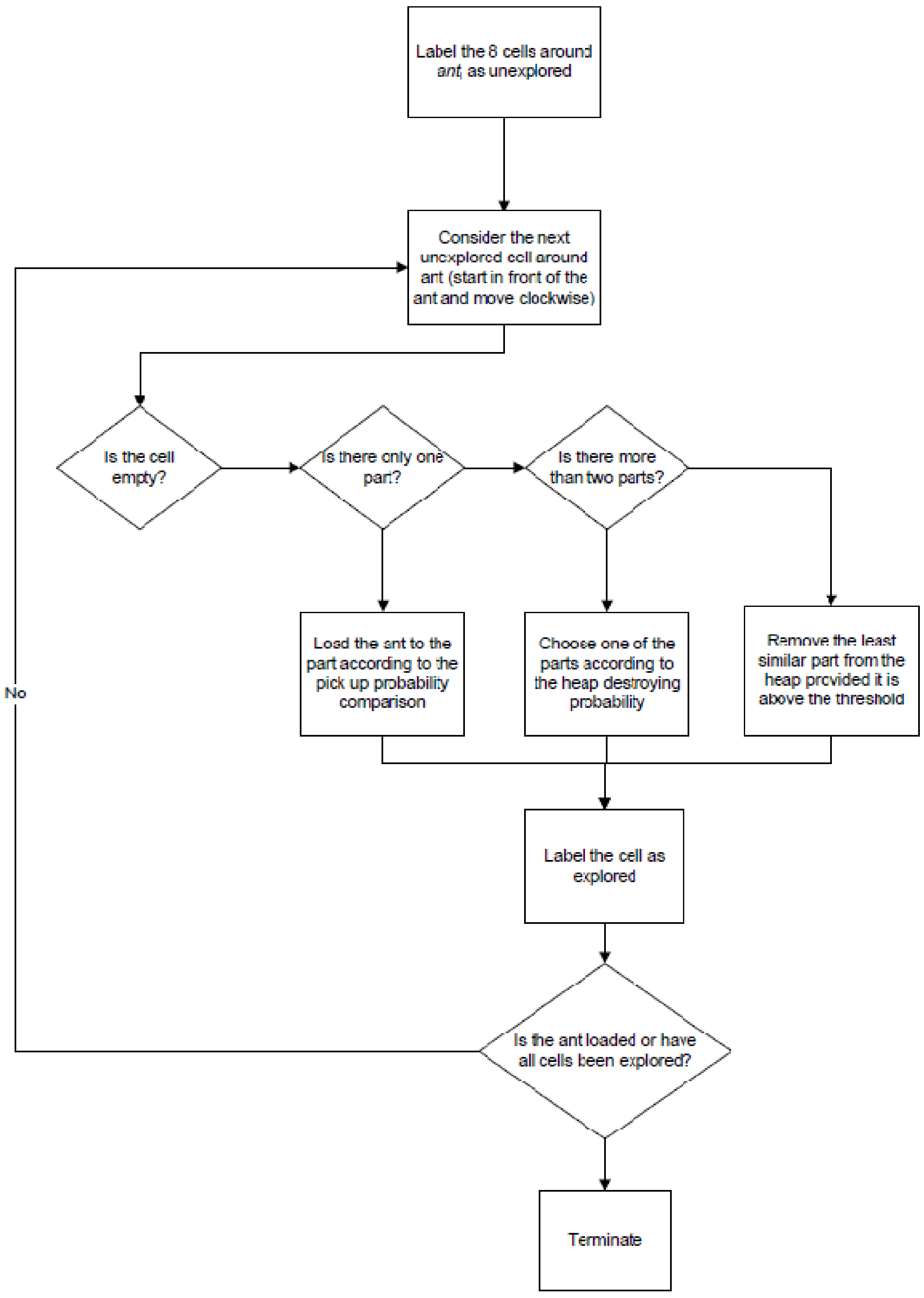

Initialize: Randomly place the parts and ants across the 2D chessboard
For each ant: Move ant_i
Is ant_i loaded?
Yes
Observe the 8 surrounding cells to possibly drop the part. See
No
Observe 8 surrounding cells, and possibly pick up an object
Has the maximum number of steps been reached?
No
Yes
Terminate

Figure 3. 4 Core Artificial Ant-Based Algorithm



Label the 8 cells around ant, as unexplored
Consider the next unexplored cell around ant (start in front of the ant and move clockwise)
Is the cell empty?
Is there only one part?
Is there more than two parts?
Load the ant to the part according to the pick up probability comparison
Choose one of the parts according to the heap destroying probability
Remove the least similar part from the heap provided it is above the threshold
No
Label the cell as explored
Is the ant loaded or have all cells been explored?
Terminate

Figure 3. 5 Artificial Ant Thought Process Picking up a Part

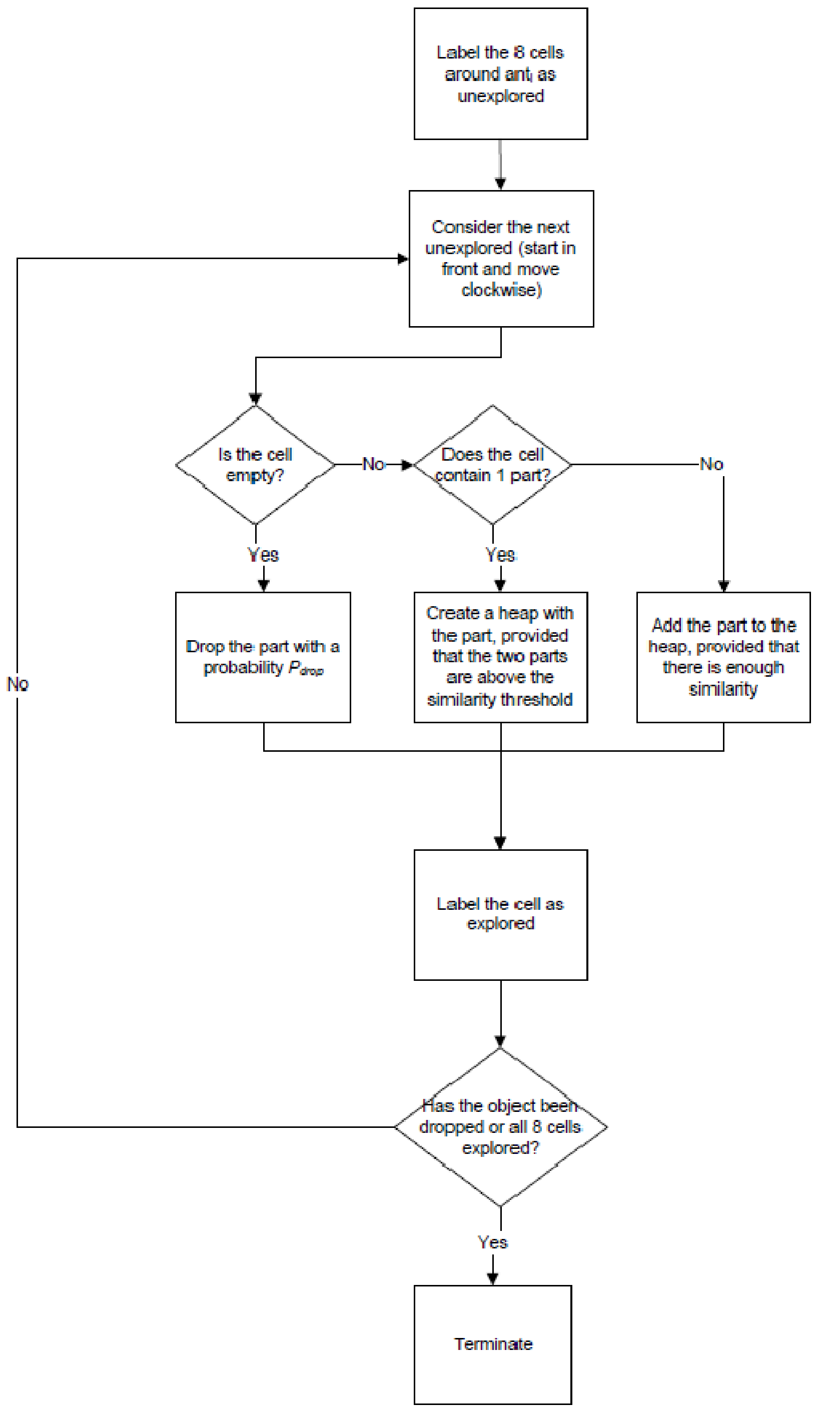

Label the 8 cells around ant_i as unexplored
Consider the next unexplored (start in front and move clockwise)
Is the cell empty?
No
Does the cell contain 1 part?
No
Yes
Yes
Drop the part with a probability P_{drop}
Create a heap with the part, provided that the two parts are above the similarity threshold
Add the part to the heap, provided that there is enough similarity
No
Label the cell as explored
Has the object been dropped or all 8 cells explored?
Yes
Terminate

Figure 3. 6 Artificial Ant Logic: Dropping a Part

Phase 4: Comparing Different Similarity Coefficients

In this phase, a number of similarity coefficients will be used in the algorithm to study the algorithm created, in order to determine which similarity coefficient works best with this method. In order to determine the best similarity coefficients, the best similarity coefficients from literature were selected to be compared. These similarity coefficients are the Jaccard coefficient, Russel and Rao's Similarity Coefficient, the Simple Matching Similarity Coefficient, the Relative Matching Similarity Coefficient and the Baroni-Urbani and Buser Similarity Coefficient. These similarity coefficients are defined below.

Equation 7: Jaccard Similarity Coefficient

$$\sigma = \frac{a}{a + b + c}$$

Equation 8: Russel and Rao's Similarity Coefficient

$$\sigma = \frac{a}{a + b + c + d}$$

Equation 9: Simple Matching Coefficient

$$\sigma = \frac{a + d}{a + b + c + d}$$

Equation 10: Relative Matching Coefficient

$$\sigma = \frac{a + \sqrt{ad}}{a + b + c + d + \sqrt{ad}}$$

Equation 11: Baroni-Urbani and Buser Similarity Coefficient

$$\sigma = \frac{a + \sqrt{ad}}{a + b + c + d + \sqrt{ad}}$$

Where:

a is the number of machines which produce both components *i* and *j*
b is the number of machines which produce only component *i*

c is the number of machines which produce only component *j*

d is the number of machines which produce neither components *i* or *j*

The similarity coefficients will be tested on three problems: One of small size, a medium sized problem and a large sized problem. The reason for testing problems of three different sizes is to determine how the similarity coefficient affects the behavior of the artificial ants in different sized environments. The first problem is cellular manufacturing problem with 11 parts manufactured on 5 machines. This small problem was presented by Chow and Howaleshka (1992). The second problem is a cellular manufacturing problem with 20 parts being produced on 8 machines. This medium sized problem was introduced by Chandrasekharan and Rajagopalan in 1986. The third and last problem is a cellular manufacturing problem involving 40 parts being manufactured on 24 machines. This problem was introduced by Chandrasekharan and Rajagopalan in 1989. These problems will be outlined in detail in the next chapter.

The similarity coefficients will be tested using several performance measures. The first performance measure will be the number of exceptional elements (EE or e_e). The number of exceptional elements is the source of inter-cellular movement between cells (Yin and Yasuda, 2006). Since one of the objectives of cellular manufacturing is to reduce the material handling costs, a reduction in the number of exceptional elements is directly related to the cellular manufacturing problem.

The second performance measure to be used to measure the quality of the solution is the grouping efficiency (η). Grouping efficiency was developed by Chandrasekharan and Rajagopalan (1986). Grouping efficiency (η) is defined as the weighted average of two efficiencies η_1 and η_2.

Equation 12: Grouping Efficiency for a Machine-Part Matrix

$$\eta = w\eta_1 + (1 - w)\eta_2$$

η_1 and η_2 can be defined as:

Equation 13: Left side partial grouping efficiency

$$\eta_1 = \frac{e_c}{\sum_{r=1}^{k} M_r N_r}$$

Equation 14: Right Side grouping efficiency

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right]$$

Where:

k is the number of diagonal blocks on the machine-part matrix
N_r is the number of components in the rth family
M_r is the number of machines in the r^{th} cell
n_m is the number of machines
n_p is the number of parts
o is the number of operations in the machine part matrix
v is the number of voids in the solution
e_c is the number of non-exceptional elements
e_e is the number of exceptional elements
w is a constant relating the importance of intercellular movement (equal to 0.5 in the study)

The similarity coefficients will be compared according to several aspects. The first comparison will be made to the number of steps taken to complete the algorithm. The second comparison will be made to as the number of part families made at the end of Phase one.

CHAPTER IV

ANALYSIS OF SAMPLE PROBLEMS

The differences in the similarity coefficients will be seen at the end of the first phase. The differences will be corrected by the second phase, or the K- means refining. Therefore, in this section, the first phase will be done for each of the similarity coefficients. The solution of the first phase will then be tested for grouping efficiency. After the first phase, the algorithm will be continued as it normally does and then tested for overall efficiency.

Small Problem (Chow and Howaleshka, 1992)

This example is a cellular manufacturing problem with 11 parts manufactured on 5 machines. For the small example, the artificial ants will perform initial clustering with 500(11)(5) cycles.

Table 4. 1 Initial Machine-Part Matrix for the Small Example

	Parts											
		1	2	3	4	5	6	7	8	9	10	11
Machines	1	1	1			1	1	1	1		1	
	2	1			1	1				1		1
	3			1			1		1		1	
	4		1		1			1		1		1
	5			1			1		1		1	

The number of cells on the two-dimensional chessboard can be defined as:

$$m = \sqrt{n \times 4} = \sqrt{11 \times 4} = 6.6 = 7$$

The number of artificial ants on the two dimensional chessboard can be defined as:

$$n_{ants} = \frac{n_{parts}}{10} = \frac{11}{10} \approx 1$$

Therefore there is 1 artificial ant and 11 parts scattered randomly on the two dimensional chessboard, as can be seen in figure 4.1.

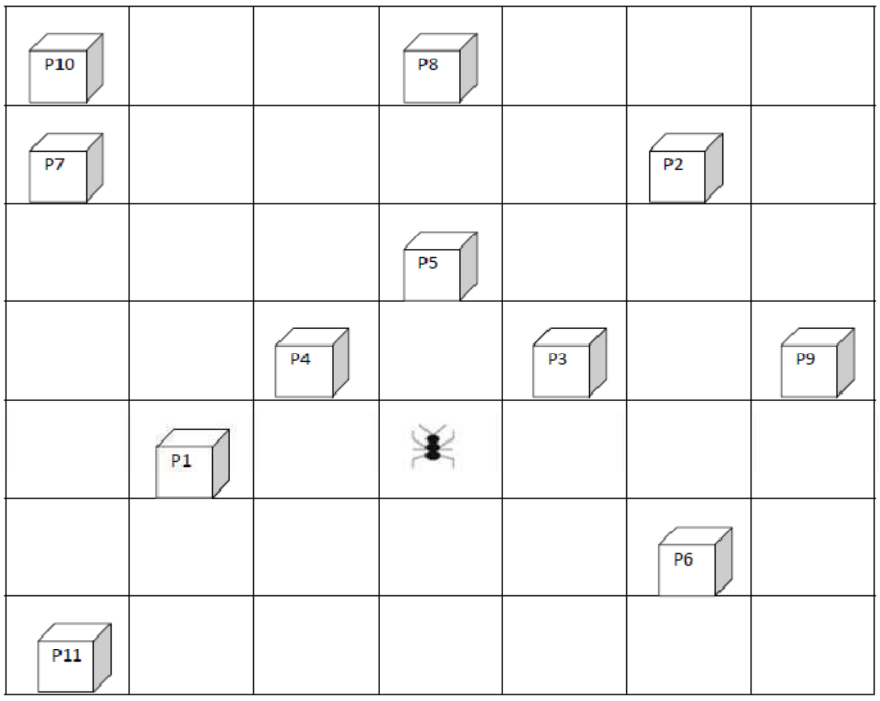


Figure 4. 1 Initial Machine-Part Matrix for Small Example

Small Problem: Jaccard Similarity Coefficient

Solving the small example's first phase with the Jaccard Similarity Coefficient the machine part matrix seen in Table 4.2, and the two-dimensional chessboards layout seen in Figure 4.2 are obtained.

Table 4. 2 Machine Part Matrix for the first Phase of the Small Example using the Jaccard Similarity Coefficient

	Parts											
		9	11	4	1	5	7	2	6	10	8	3
Machines	2	1	1	1	1	1						
	4	1	1	1			1	1				
	1				1	1	1	1	1	1	1	
	5								1	1	1	1
	3								1	1	1	1

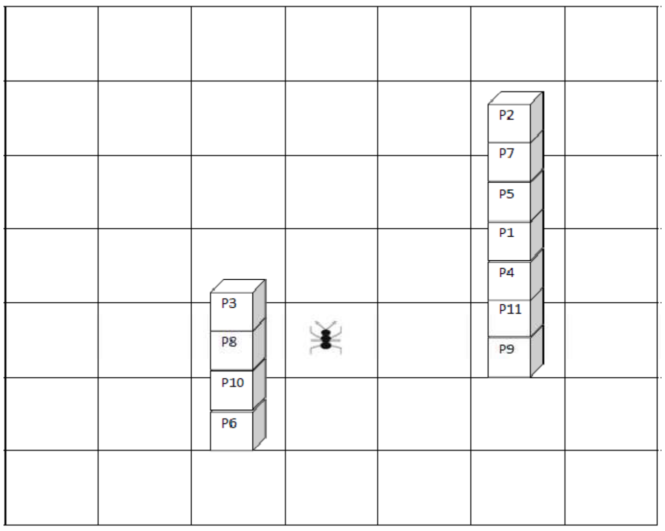


Figure 4. 2 Layout of the 2-Dimensional chessboard following the first phase for the small example, using the Jaccard Similarity Coefficient

From the machine part matrix, it can be seen that there are 2 part families with 3 exceptional elements. Therefore, the variables can be set as:

n_m=5

n_p=11

M_1=3

M_2=2

N_1=7

N_2=4

e_d=20

e_o=3

k=2

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{22}{29} = 0.759$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{3}{26} = 0.885$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1 - w)\eta_2 = 0.821$$

Therefore, the Jaccard Similarity Coefficient gives a grouping efficiency of 87.3% at the end of phase 1.

<u>Small Problem: Russel and Rao's Similarity Coefficient</u>

Solving the small example's first phase with the Russel and Rao's Similarity Coefficient, the machine part matrix seen in Table 4.3 is obtained.

Table 4. 3 Machine-Part Matrix for Phase 1 of a small example using Russel and Rao's Similarity Coefficient

		Parts										
		9	11	4	7	2	1	5	6	8	10	3
Machines	2	1	1	1			1	1				
	4	1	1	1	1	1						
	1				1	1	1	1	1	1	1	
	3								1	1	1	1
	5								1	1	1	1

The two-dimensional chessboard's layout for the small example can be seen in figure 4.3 is obtained from the first phase of clustering.

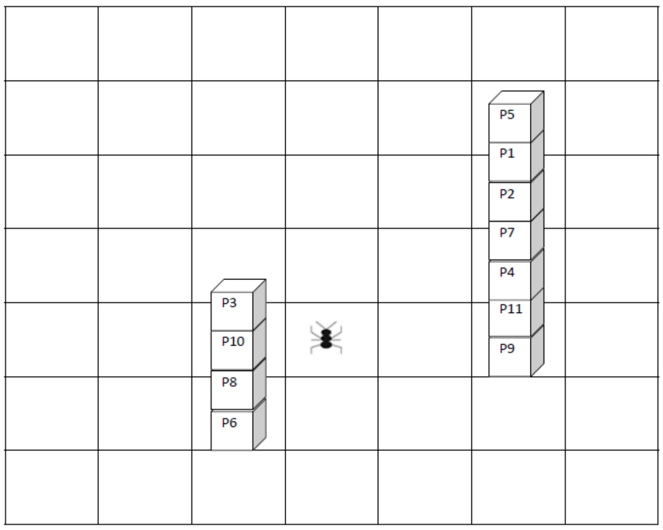


Figure 4. 3 2-D Chessboard's Layout for Phase 1 of the small example using R and R Similarity Coefficient

From the machine part matrix, it can be seen that there are 2 part families with 3 exceptional elements. Therefore, the variables can be set as:

n_m=5

n_p=11

M_1=3

M_2=2

N_1=7

N_2=4

e_d=20

e_o=3

k=2

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{22}{29} = 0.759$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{3}{26} = 0.885$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1-w)\eta_2 = 0.821$$

Therefore, Russel and Rao's Similarity Coefficient gives a grouping efficiency of 82.1% at the end of phase 1.

Small Problem: Simple Matching Similarity Coefficient

Solving the small example's first phase with the Simple Matching Similarity Coefficient, the machine part matrix seen in Table 4.4 is obtained.

Table 4. 4 Machine-Part matrix for Phase one of a Small problem solved using Simple Matching Similarity Coefficient

		Parts										
Machines		9	11	4	1	5	7	2	6	8	10	3
	4	1	1	1			1	1				
	2	1	1	1	1	1						
	1				1	1	1	1	1	1	1	
	3								1	1	1	1
	5								1	1	1	1

The two-dimensional chessboard's layout for the small example can be seen in figure 4.5 is obtained from the first phase of clustering with the Simple Matching Similarity Coefficient.

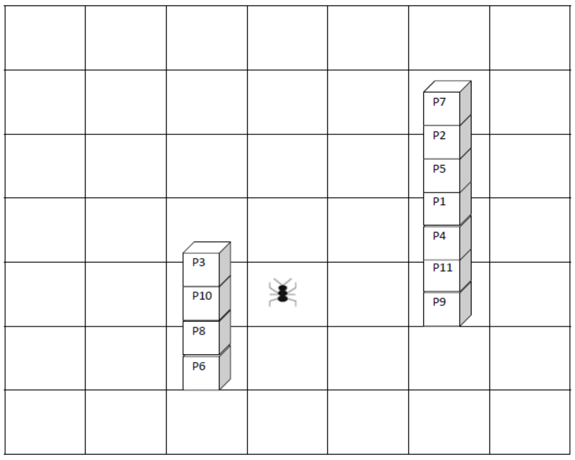


Figure 4. 4 2-D Chessboard's Layout for Phase one of a Small problem solved using Simple Matching Similarity Coefficient

From the machine part matrix, it can be seen that there are 2 part families with 3 exceptional elements. Therefore, the variables can be set as:

n_m=5

n_p=11

M_1=3

M_2=2

N_1=7

N_2=4

e_d=20

e_o=3

k=2

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{22}{29} = 0.759$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{3}{26} = 0.885$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1 - w)\eta_2 = 0.821$$

Therefore, the simple matching Similarity Coefficient gives a grouping efficiency of 82.1% at the end of phase 1.

Small Problem: Relative Matching Similarity Coefficient

Solving the small example's first phase with the Relative Matching Similarity Coefficient, the machine part matrix seen in Table 4.5 is obtained.

Table 4. 5 Machine-Part matrix for Phase one of a Small problem solved using Relative Matching Similarity Coefficient

		Parts										
		9	11	4	1	5	2	7	6	10	8	3
Machines	4	1	1	1			1	1				
	2	1	1	1	1	1						
	1				1	1	1	1	1	1	1	
	3								1	1	1	1
	5								1	1	1	1

The two-dimensional chessboard's layout for the small example can be seen in figure 4.5 is obtained from the first phase of clustering.

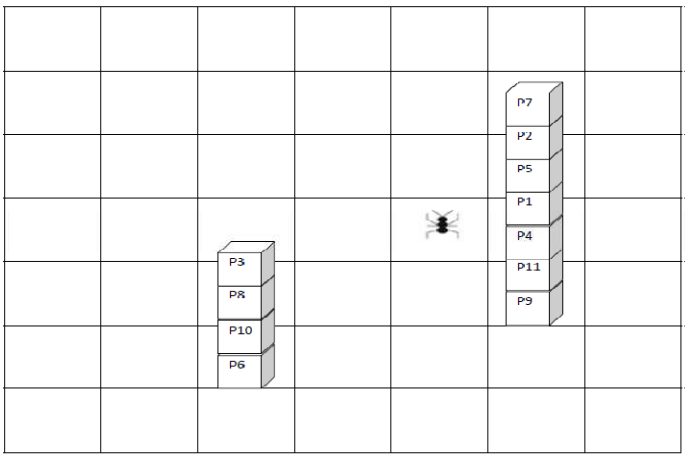


Figure 4. 5 2-D Chessboard's Layout for Phase one of a Small problem solved using Relative Matching Similarity Coefficient

From the machine part matrix, it can be seen that there are 2 part families with 3 exceptional elements. Therefore, the variables can be set as:

n_m=5

n_p=11

M_1=3

M_2=2

N_1=7

N_2=4

e_d=20

e_o=3

k=2

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{22}{29} = 0.759$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{3}{26} = 0.885$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1-w)\eta_2 = 0.821$$

Therefore, the Relative Matching Similarity Coefficient gives a grouping efficiency of 82.1% at the end of phase 1.

<u>Small Problem: Baroni-Urbani and Buser Similarity Coefficient</u>

Solving the small example's first phase with the Relative Matching Similarity Coefficient, the machine part matrix seen in Table 4.6 is obtained.

Table 4. 6 Machine-Part matrix for Phase one of a Small problem solved using Baroni-Urbani and Buser Matching Similarity Coefficient

		Parts										
		11	9	4	1	5	2	7	6	10	3	8
Machines	2	1	1	1	1	1						
	4	1	1	1			1	1				
	1				1	1	1	1	1	1		1
	5								1	1	1	1
	3								1	1	1	1

The two-dimensional chessboard's layout for the small example can be seen in Figure 4.6 is obtained from the first phase of clustering.

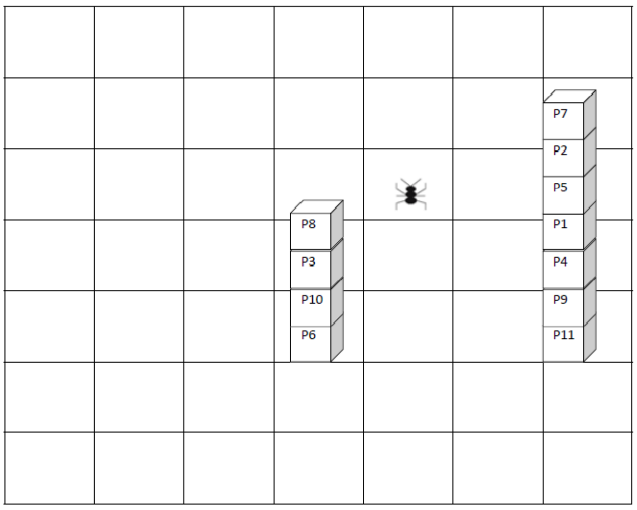


Figure 4. 6 2-D Chessboard's Layout for Phase one of a Small problem solved using Relative Matching Similarity Coefficient

From the machine part matrix, it can be seen that there are 2 part families with 3 exceptional elements. Therefore, the variables can be set as:

n_m=5

n_p=11

M_1=3

M_2=2

N_1=7

N_2=4

e_d=20

e_o=3

k=2

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{22}{29} = 0.759$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{3}{26} = 0.885$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1-w)\eta_2 = 0.821$$

Therefore, the Baroni-Urbani and Buser Similarity Coefficient gives a grouping efficiency of 82.1% at the end of phase 1.

Refining the Part Families for a Small Example

Refining the part families using K-means refining, the layout in Figure 4.7 is obtained:

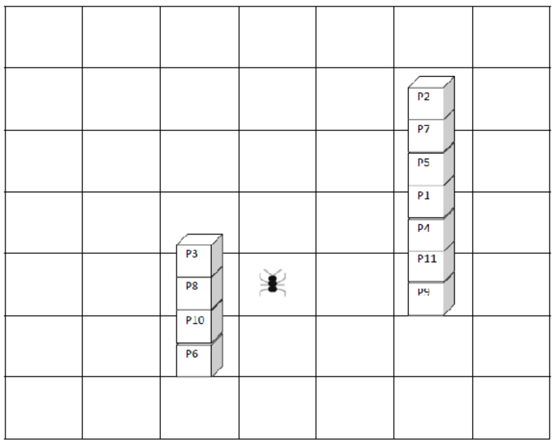


Figure 4. 7 2-D Chessboard for the solution to the small example (End of Phase 2)

It should be noted that this is the same layout obtained at the end of phase 1. Therefore the machine-part matrix can be seen in Table 4.7 and the maximum grouping efficiency which can be obtained is 82.1% . It is not necessary to perform the 3rd phase of the ant-based algorithm due to the size of the problem, and the nature of the testing.

Table 4. 7 Machine-Part Matrix for the solution to the small example (End of Phase 2)

	Parts											
Machines		9	11	4	1	5	7	2	6	10	8	3
	2	1	1	1	1	1						
	4	1	1	1			1	1				
	1				1	1	1	1	1	1	1	
	5								1	1	1	1
	3								1	1	1	1

<u>Medium Problem</u>

This example is a cellular manufacturing problem with 20 parts manufactured on 8 machines. For the small example, the artificial ants will perform initial clustering with 10, 000 cycles. The machine-part matrix can be seen in table 4.8

Table 4. 8 Initial Medium Machine Part Matrix

		Parts																			
Machines		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	1		1	1					1	1		1		1	1		1	1		1	
	2			1	1		1	1							1				1		1
	3		1						1	1		1		1	1		1	1	1	1	
	4			1	1		1	1			1							1			
	5	1				1	1				1		1			1		1			
	6	1				1				1	1		1			1					1
	7			1	1		1	1				1	1			1			1		1
	8			1	1		1	1											1		1

The number of cells and artificial ants on the two dimensional chess board can be calculated as:

$$m = \sqrt{n \times 4} = \sqrt{20 \times 4} \approx 9 \qquad n_{ants} = \frac{n_{parts}}{10} = \frac{20}{10} \approx 2$$

Therefore there are 2 ants and 20 parts randomly distributed on a 2-dimensional Chessboard of 9 x 9 size.

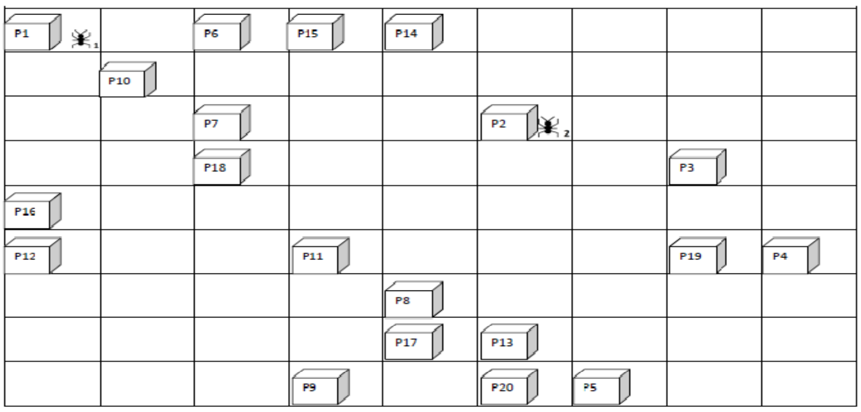


Figure 4. 8 Initial Random Layout for the medium size problem

Medium Problem: Jaccard Similarity Coefficient

Solving the medium example's first phase with the Jaccard Similarity Coefficient, the machine part matrix seen in Table 4.9 is obtained.

Table 4. 9Phase 1 of the medium example solved with the Jaccard Coefficient

	Parts																				
Machines		2	8	13	12	16	19	11	17	4	7	18	20	10	14	9	3	6	1	15	5
	1	1	1	1		1	1	1	1						1	1	1				
	2	1	1	1		1	1	1	1			1			1	1					
	3									1	1	1	1		1		1	1			
	4								1	1	1			1			1	1			
	5				1				1					1				1	1	1	1
	6				1			1		1	1	1	1				1	1		1	
	7									1	1	1	1				1	1			
	8				1								1	1		1			1	1	1

The two-dimensional chessboard's layout for the small example can be seen in figure 4.9 is obtained from the first phase of clustering with the Jaccard Similarity Coefficient.

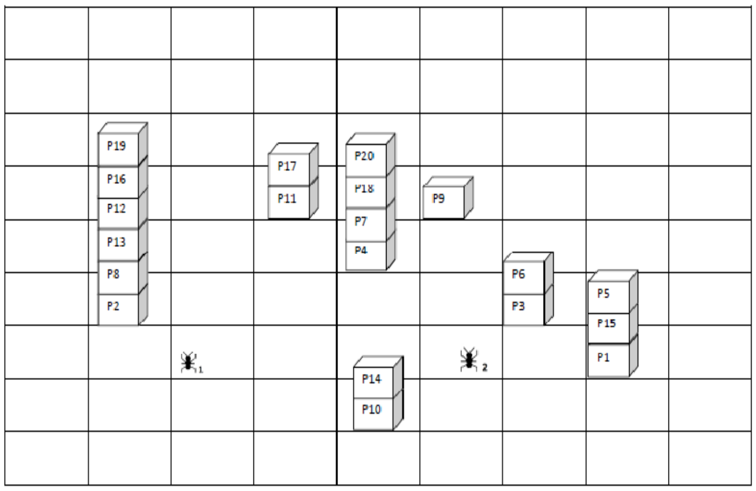


Figure 4. 9 Medium Layout for Phase 1 using the Jaccard Similarity Coefficient

From the machine part matrix, it can be seen that there are 3 part families with 20 exceptional elements. Therefore, the variables can be set as:

n_m=8

n_p=20

M_1=2

M_2=5

M_3=1

N_1=8

N_2=7

N_3=5

e_d=42

e_o=20

k=3

Both left and right side “partial-grouping” efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{44}{62} = 0.7097$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{18}{98} = 0.8163$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1-w)\eta_2 = 0.763$$

Therefore, the Jaccard Similarity Coefficient gives a grouping efficiency of 76.3% at the end of phase 1.

Medium Problem: Russel and Rao’s Similarity Coefficient

Solving the medium example’s first phase with Russel and Rao’s Similarity Coefficient, the machine part matrix seen in Table 4.10 is obtained.

Table 4. 10 Phase 1 of the medium example solved with the R and R Similarity Coefficient

	Parts																				
Machines		2	8	13	18	16	3	11	17	10	14	6	7	4	20	12	9	19	1	15	5
	1	1	1	1		1	1	1	1		1						1	1			
	3	1	1	1	1	1		1	1		1						1	1			
	2				1		1				1	1	1	1	1						
	4						1		1	1		1	1	1							
	5								1	1		1				1			1	1	1
	7				1		1	1				1	1	1	1	1				1	
	8				1		1					1	1	1	1						
	6									1					1	1	1		1	1	1

The two-dimensional chessboard’s layout for the small example can be seen in figure 4.10 is obtained from the first phase of clustering with Russel and Rao’s Similarity Coefficient.

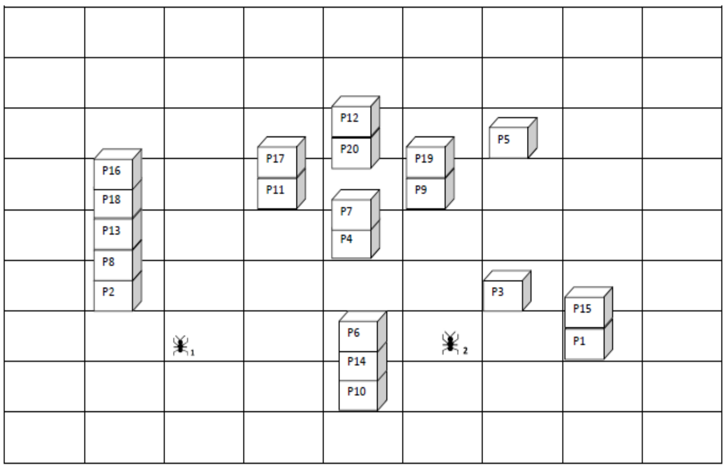


Figure 4. 10 Medium Layout for Phase 1 using the R and R Similarity Coefficient

From the machine part matrix, it can be seen that there are 3 part families with 20 exceptional elements. Therefore, the variables can be set as:

n_m=8

n_p=20

M_1=2

M_2=5

M_3=8

N_1=7

N_2=5

N_3=3

e_d=42

e_o=20

k=3

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{42}{56} = 0.0.75$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{20}{104} = 0.808$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1 - w)\eta_2 = 0.779$$

Therefore, Russel and Rao's Similarity Coefficient gives a grouping efficiency of 77.9% at the end of phase 1.

Medium Problem: Simple Matching Similarity Coefficient

Solving the medium example’s first phase with the Simple Matching Similarity Coefficient, the machine part matrix seen in Table 4.11 is obtained.

Table 4. 11Phase 1 of the medium example solved with the Simple Matching Similarity Coefficient

		Parts																			
Machines		8	13	18	17	10	14	6	5	20	2	12	9	19	3	16	1	15	11	4	7
	3	1	1	1	1		1				1		1	1		1			1		
	1	1	1		1		1				1		1	1	1	1			1		
	2			1			1	1		1					1					1	1
	7			1				1		1		1			1			1	1	1	1
	8			1				1		1					1					1	1
	4				1	1		1							1					1	1
	5				1	1		1	1			1					1	1			
	6					1			1	1		1	1				1	1			

The two-dimensional chessboard’s layout for the small example can be seen in figure 4.11 is obtained from the first phase of clustering with the Simple Matching Similarity Coefficient.

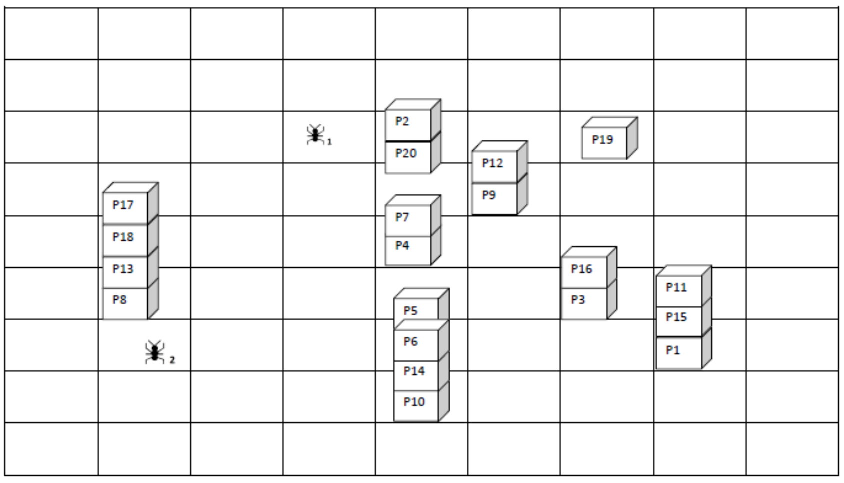


Figure 4. 11 Medium Layout for Phase 1 using the Simple Matching Similarity Coefficient

From the machine part matrix, it can be seen that there are 3 part families with 34 exceptional elements. Therefore, the variables can be set as:

n_m=8

n_p=20

M_1=2

M_2=5

M_3=1

N_1=6

N_2=8

N_3=6

e_d=28

e_o=34

k=3

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{28}{58} = 0.483$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{34}{102} = 0.667$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1 - w)\eta_2 = 0.575$$

Therefore, the Simple Matching Coefficient gives a grouping efficiency of 57.5% at the end of phase 1.

Medium Problem: Relative Matching Similarity Coefficent

Solving the medium example's first phase with the Relative Matching Similarity Coefficient, the machine part matrix seen in Table 4.12 is obtained.

Table 4. 12 Phase 1 of the medium example solved with the Relative Matching Similarity Coefficient

		Parts																			
Machines		12	15	8	9	20	16	11	17	6	5	7	3	18	4	13	19	1	10	2	14
	1			1	1		1	1	1				1			1	1			1	1
	3			1	1		1	1	1					1		1	1			1	1
	5	1	1						1	1	1							1	1		
	6	1	1		1	1					1							1	1		
	7	1	1			1		1		1		1	1	1	1						
	2					1				1		1	1	1	1						1
	8					1				1		1	1	1	1						
	4								1	1		1	1		1				1		

The two-dimensional chessboard's layout for the small example can be seen in figure 4.12 is obtained from the first phase of clustering with the Relative Matching Similarity Coefficient.

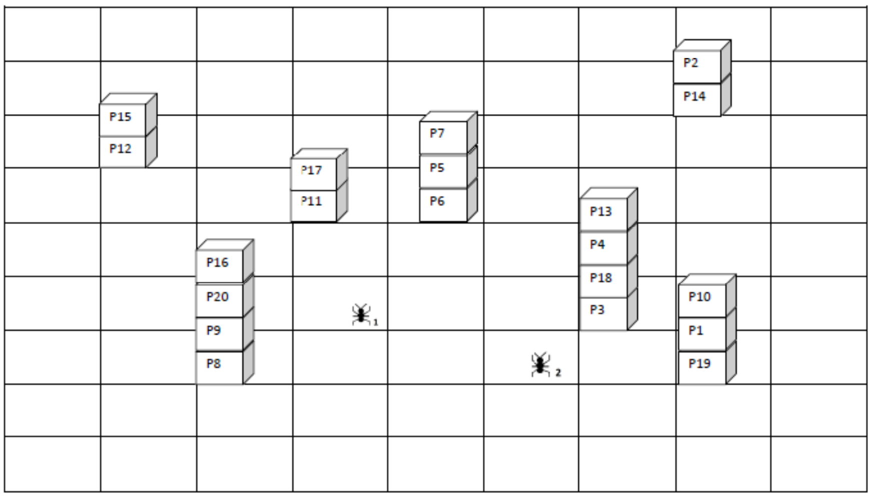


Figure 4. 12 Layout for Phase 1 using the Relative Matching Similarity Coefficient

From the machine part matrix, it can be seen that there are 3 part families with 29 exceptional elements. Therefore, the variables can be set as:

n_m=8

n_p=20

M_1=2

M_2=2

M_3=4

N_1=8

N_2=6

N_3=6

e_d=33

e_o=29

k=3

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{33}{52} = 0.635$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{29}{108} = 0.731$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1-w)\eta_2 = 0.683$$

Therefore, the Relative Matching Similarity Coefficient gives a grouping efficiency of 68.3% at the end of phase 1.

Medium Problem: Baroni-Urbani and Buser Similarity Coefficient

Solving the medium example's first phase with the Baroni-Urbani and Buser Similarity Coefficient, the machine part matrix seen in Table 4.13 is obtained.

Table 4. 13 Phase 1 of the medium example solved with the Baroni-Urbani and Buser Similarity Coefficient

		Parts																			
Machines		3	18	4	8	13	20	17	11	16	2	19	1	15	5	7	6	10	12	9	14
	2	1	1	1			1									1	1				1
	4	1		1				1								1	1	1			
	7	1	1	1			1		1					1		1	1		1		
	8	1	1	1			1									1	1				
	1	1			1	1		1	1	1	1	1								1	1
	3		1		1	1		1	1	1	1	1								1	1
	6						1						1	1	1			1	1	1	
	5							1					1	1	1		1	1	1		

The two-dimensional chessboard's layout for the small example can be seen in figure 4.13 is obtained from the first phase of clustering with the Baroni-Urbani and Buser Similarity Coefficient.

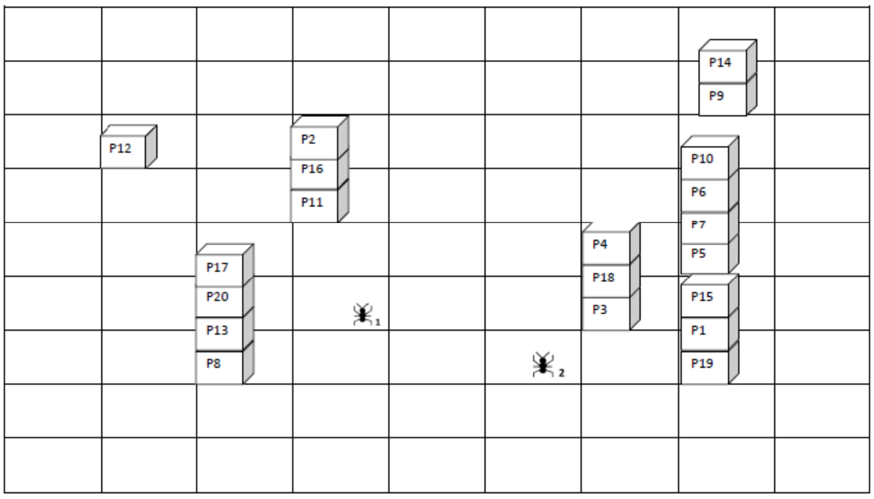


Figure 4. 13 Layout for Phase 1 using the Baroni-Urbani and Buser Similarity Coefficient

From the machine part matrix, it can be seen that there are 3 part families with 25 exceptional elements. Therefore, the variables can be set as:

n_m=8

n_p=20

M_1=4

M_2=2

M_3=2

N_1=3

N_2=8

N_3=9

e_d=37

e_o=25

k=3

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{37}{46} = 0.804$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{25}{46} = 0.781$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1-w)\eta_2 = 0.792$$

Therefore, the the Baroni-Urbani and Buser Similarity Coefficient gives a grouping efficiency of 79.2% at the end of phase 1.

Refining Part Families for Phase 2

Refining the part families using K-means refining, the machine part matrix in table 4.14 below is obtained.

Table 4. 14 Medium Problems Result of K-Means Clustering: Machine-Part Matrix

		Parts																		
Machines		4	7	18	20	10	12	11	17	1	5	15	2	8	9	13	14	16	19	3	6
	2	1	1	1	1												1			1	1
	4	1	1			1			1											1	1
	7	1	1	1	1		1	1				1								1	1
	8	1	1	1	1															1	1
	1							1	1				1	1	1	1	1	1	1	1	
	3			1				1	1				1	1	1	1	1	1	1		
	5					1	1		1	1	1	1									1
	6				1	1	1			1	1	1			1						

Figure 4.14 below shows the layout of the two dimensional chessboard at the end of phase 2.

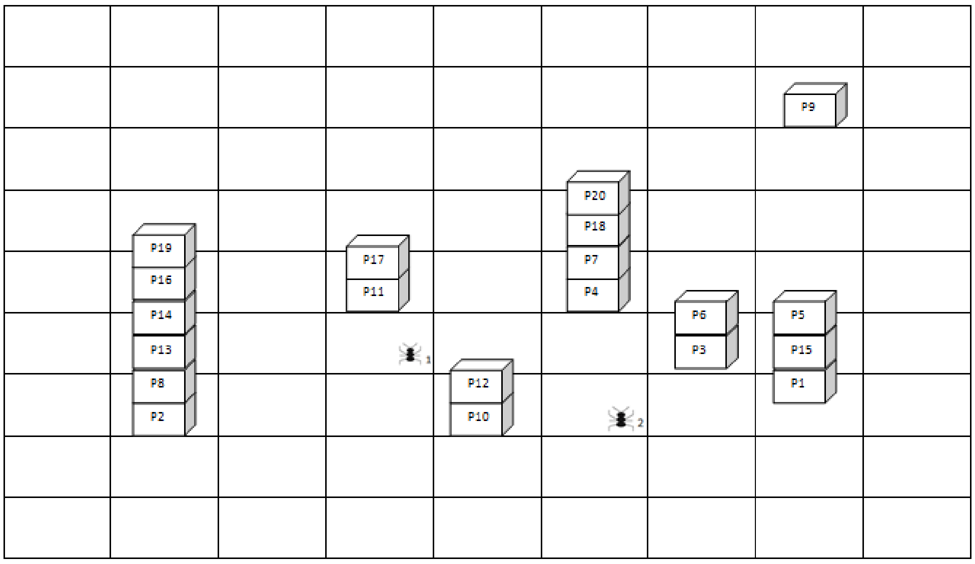


Figure 4. 14 Medium Problems Result of K-Means Clustering: 2-D Chessboard

Combining heaps into Part Families

Since the first phase is based on random clustering, there is a possibility that the first phase will produce too many part families. This excess in families will therefore result a high number of exceptional elements. This phase is done at the discretion of the facility designer. After performing the third phase of combining the heaps, the following machine part matrix is obtained:

Table 4. 15 Final machine part matrix for the medium sized example

	Parts																				
Machines		3	4	6	7	18	20	2	8	9	13	14	16	17	19	11	1	5	10	12	15
	2	1	1	1	1	1	1					1									
	4	1	1	1	1									1					1		
	7	1	1	1	1	1	1									1				1	1
	8	1	1	1	1	1	1														
	1	1						1	1	1	1	1	1	1	1	1					
	3					1		1	1	1	1	1	1	1	1	1					
	5			1										1			1	1	1	1	1
	6						1			1							1	1	1	1	1

The layout of the two dimensional chessboard is shown below:

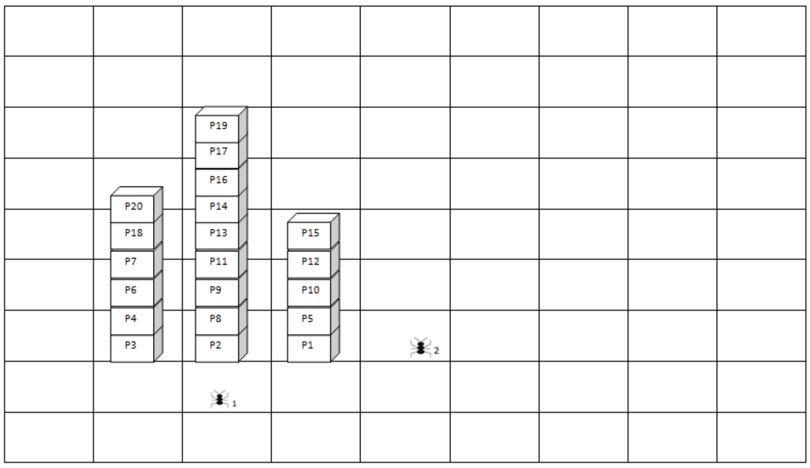


Figure 4. 15 Final layout of the medium sized problem

From the machine part matrix shown in table 4.15, it can be seen that the last phase gives 3 distinct part families with 11 exceptional elements. The total number of operations is 62 operations done by 8 machines to the 20 parts. Therefore, the variables can be set as:

n_m=8

n_p=20

M_1=4

M_2=2

M_3=2

N_1=6

N_2=9

N_3=5

e_d=51

e_o=11

k=3

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{51}{52} = 0.981$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{11}{52} = 0.898$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1-w)\eta_2 0.940$$

Therefore, after refining the part heaps using K-means refining, and combining the excess part families, the grouping efficiency is improved to 94.0%.

Large Problem

This problem is a cellular manufacturing problem with 40 parts being manufactured on 24 machines. For the initial machine – part matrix please refer to Appendix B.1.

Large Problem: Jaccard Similarity Coefficient

From the results generated by the ant based algorithm, the Jaccard similarity coefficient gives a solution which generated the following variables:

n_m=24	N_2=5
n_p=40	N_3=3
M_1=4	N_4=7
M_2=4	N_5=5
M_3=2	N_6=3
M_4=4	N_7=3
M_5=4	e_d=112
M_6=2	e_o=19
M_7=2	k=7
N_1=7	

The sum of machines multiplied by parts in a heap is:

$$\sum_{r=1}^{k} M_r N_r = 132$$

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \quad = \frac{112}{132} = 0.848$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{19}{828} = 0.977$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1 - w)\eta_2 = 0.913$$

Therefore, the Jaccard Similarity Coefficient gives a grouping efficiency of 91.3% at the end of phase 1.

Large Problem: Russel and Rao's Similarity Coefficient

From the results generated by the ant based algorithm, Russel and Rao's similarity coefficient gives a solution which generated the following variables:

n_m=24	N_2=2
n_p=40	N_3=5
M_1=4	N_4=11
M_2=2	N_5=3
M_3=3	N_6=8
M_4=2	N_7=7
M_5=8	e_d=66
M_6=2	e_o=65
M_7=3	k=7
N_1=4	

The sum of machines multiplied by parts in a heap is:

$$\sum_{r=1}^{k} M_r N_r = 118$$

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \quad = \frac{65}{118} = 0.559$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{65}{842} = 0.922$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1 - w)\eta_2 = 0.741$$

Therefore, Russel and Rao's Coefficient gives a grouping efficiency of 74.1% at the end of phase 1.

<u>Large Problem: Simple Matching Similarity Coefficient</u>

From the results generated by the ant based algorithm, the Simple Matching similarity coefficient gives a solution which generated the following variables:

n_m=24

n_p=40

M_1=2

M_2=2

M_3=2

M_4=1

M_5=5

M_6=7

M_7=7

N_1=4

N_2=4

N_3=5

N_4=4

N_5=8

N_6=8

N_7=7

e_d=68

e_o=63

k=7

The sum of machines multiplied by parts in a heap is:

$$\sum_{r=1}^{k} M_r N_r = 159$$

Both left and right side "partial-grouping" efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \frac{68}{159} = 0.428$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{63}{801} = 0.921$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1-w)\eta_2 = 0.675$$

Therefore, the Simple Matching Similarity Coefficient gives a grouping efficiency of 67.5% at the end of phase 1.

Large Problem: Relative Matching Similarity Coefficient

From the results generated by the ant based algorithm, the Relative Matching similarity coefficient gives a solution which generated the following variables:

n_m=24	N_2=3
n_p=40	N_3=3
M_1=2	N_4=6
M_2=2	N_5=3
M_3=3	N_6=3
M_4=2	N_7=13
M_5=2	e_d=77
M_6=5	e_o=54
M_7=8	k=7
N_1=3	

The sum of machines multiplied by parts in a heap is:

$$\sum_{r=1}^{k} M_r N_r = 167$$

Both left and right side “partial-grouping” efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \quad = \frac{77}{167} = 0.461$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{54}{793} = 0.932$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1 - w)\eta_2 = 0.696$$

Therefore, the Relative Matching similarity coefficient gives a grouping efficiency of 69.6% at the end of phase 1.

Large Problem: Baroni-Urbani and Buser Matching Similarity Coefficient

From the results generated by the ant based algorithm, the Baroni-Urbani and Buser similarity coefficient gives a solution which generated the following variables:

n_m=24

n_p=40

M_1=5

M_2=3

M_3=4

M_4=4

M_5=2

M_6=2

M_7=4

N_1=6

N_2=5

N_3=5

N_4=3

N_5=7

N_6=8

N_7=6

e_d=112

e_o=19

k=7

The sum of machines multiplied by parts in a heap is:

$$\sum_{r=1}^{k} M_r N_r = 131$$

Both left and right side “partial-grouping” efficiencies can be obtained:

$$\eta_1 = \frac{e_d}{\sum_{r=1}^{k} M_r N_r} = \quad = \frac{112}{131} = 0.855$$

and

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right] = 1 - \frac{19}{829} = 0.977$$

Having both η_1 and η_2, the grouping efficiency (η) can be calculated as:

$$\eta = w\eta_1 + (1-w)\eta_2 = 0.916$$

Therefore, the Baroni-Urbani and Buser similarity coefficient gives a grouping efficiency of 91.6% at the end of phase 1.

Refining the Part Families in the large example

Performing the k means refining on the solutions generated in phase 1, the machine component matrix in Appendix B, Table B.2 is generated. As there are no exceptional elements in this solution, the heaps do not need to be combined to form part families. The lack of exceptional elements in the solution also yields a grouping efficiency of 100%. Therefore the solution presented in table B.2 is the optimum solution for the problem presented in table B.1.

CHAPTER V

COMPARING THE RESULTS OF THE TESTS

<u>Results of the solution qualities yielded from different similarity coefficients</u>

The results in the small example all yielded the same grouping efficiency. This may have occurred due to the size of the problem and the low number of part families. The compared grouping efficiencies and exceptional elements for the small sized example can be seen in Figure 4.16 and figure 4.17 respectively.

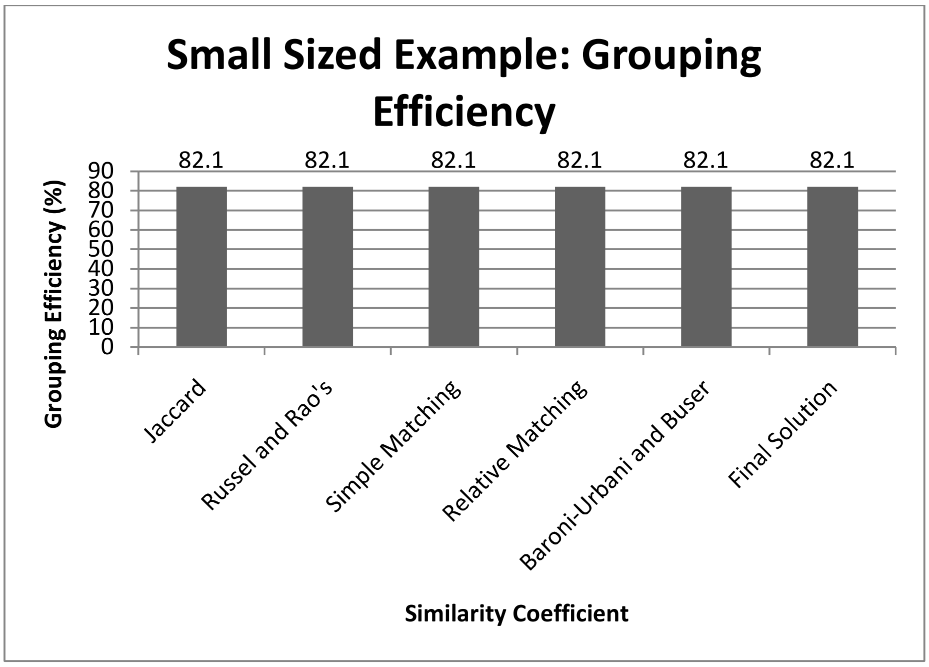


Figure 5. 1 Small Sized Example: Grouping Efficiency

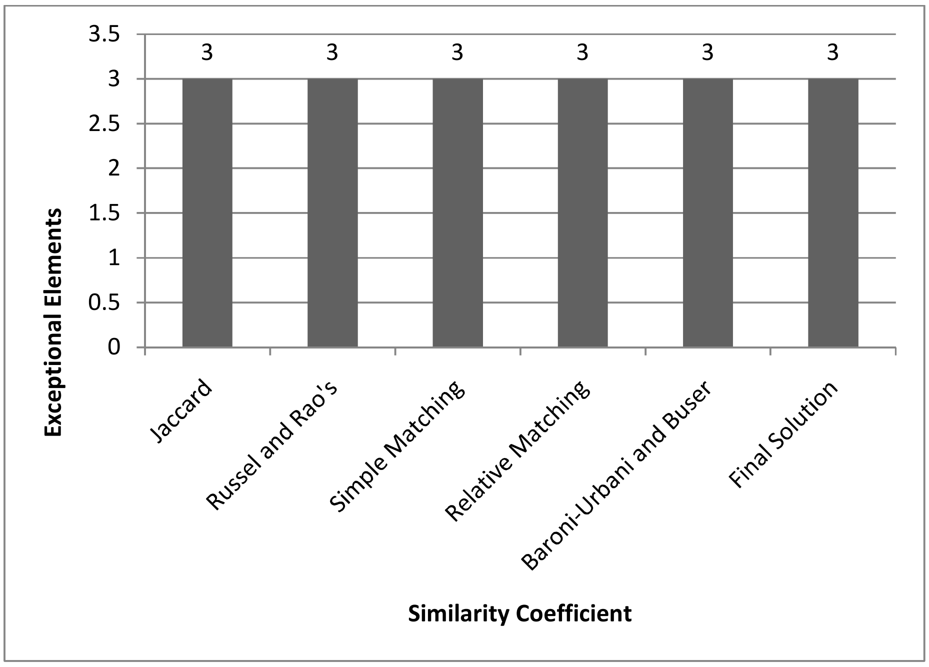


Figure 5. 2 Small Sized Example: Exceptional Elements

The compared grouping efficiencies and exceptional elements for the medium sized example can be seen in Figure 4.18 and figure 4.19 respectively.

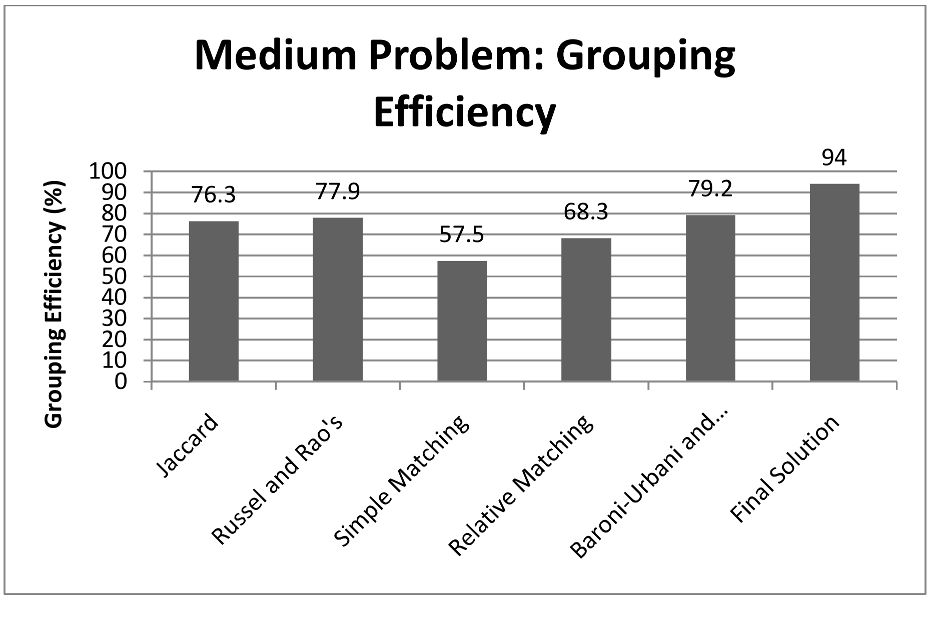


Figure 5. 3 Medium Problem: Grouping Efficiency

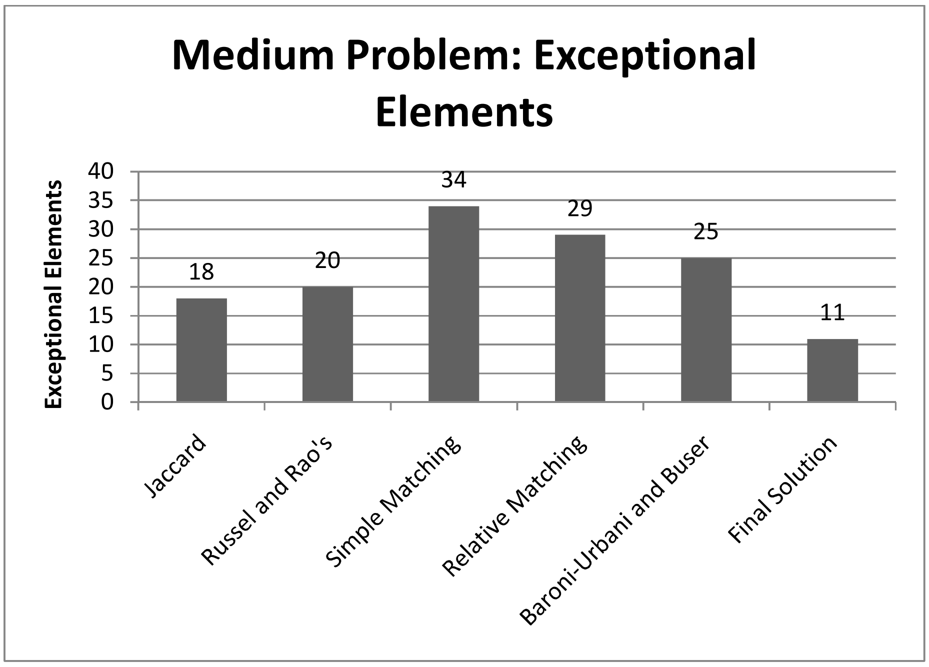


Figure 5. 4 Medium Problem: Exceptional Elements

The compared grouping efficiencies and exceptional elements for the large sized example can be seen in Figure 4.20 and figure 4.21 respectively.

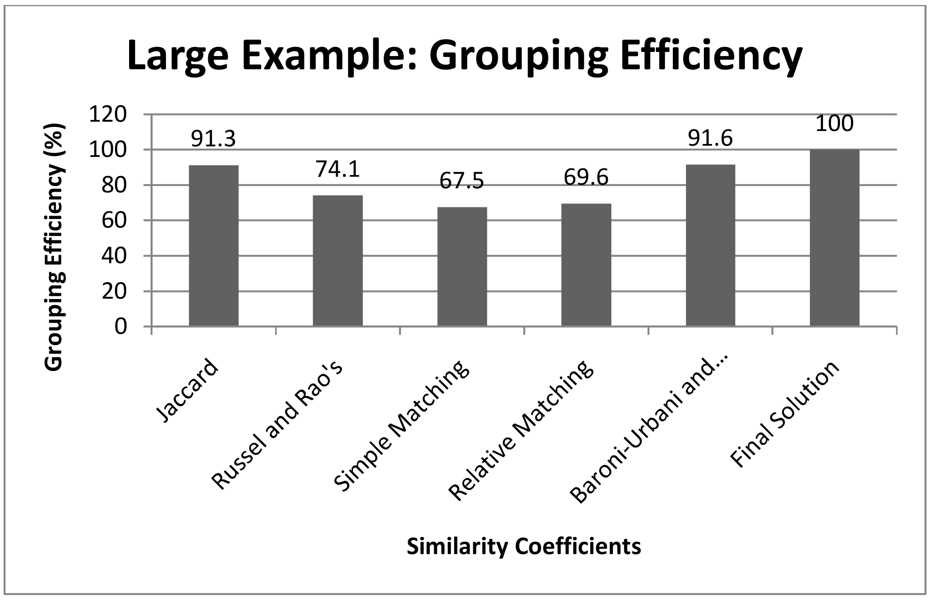


Figure 5. 5 Large Example: Grouping Efficiency

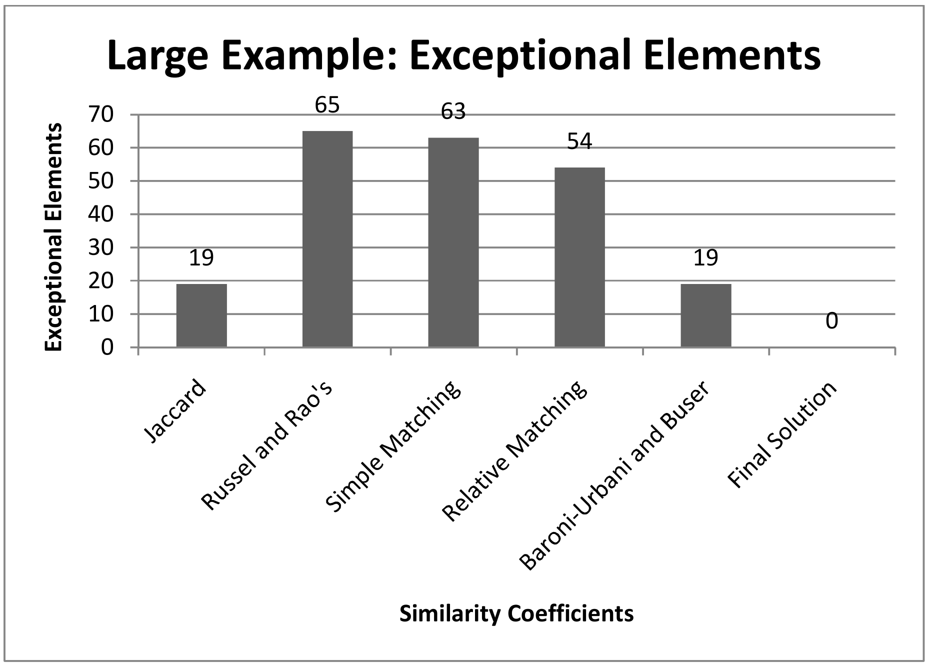


Figure 5. 6 Large Example: Exceptional Elements

In most cases, the lower number of exceptional elements results in a higher grouping efficiency. Although the exact relationship between the exceptional elements and the grouping efficiency is unknown, they seem to be inversely proportionate. In all cases the Baroni-Urbani and Buser Similarity generates the highest quality of solutions from the first phase, followed by the Jaccard similarity coefficient. This results in a faster phase 2.

Comparison of algorithm efficiency between different similarity coefficients

As previously mentioned there is a relationship between the grouping efficiency in the solution produced from the first phase of the artificial ant-based algorithm. It is important to note that all processes in this investigation were conducted under identical conditions. The computer that produced these solutions is a Hewlett-Packard company model p6152f desktop model, with an AMD Phenom™ 8450 Triple Core processor. It is a 64-bit operating system with 4.00 GB of ram. It should be noted that one of the processors on this system was dedicated to running the ant based algorithm, and all other windows idle processes with done on the remaining processors. This was done so that the idle processes that the Windows Vista generates do not interfere with the process of timing the duration of the artificial ant-based algorithm.

In Figures 4.22-4.26, on the following pages, the average length of the 3 examples for each similarity coefficient is shown. Each example was run ten times per similarity coefficient under identical conditions.

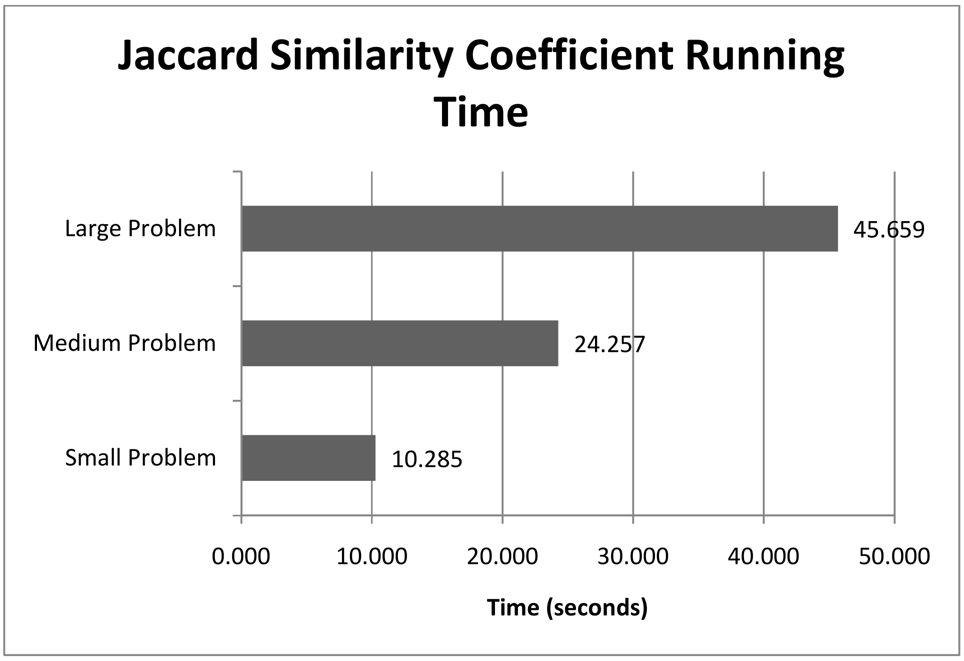


Figure 5. 7 Jaccard Similarity Coefficient Running Time

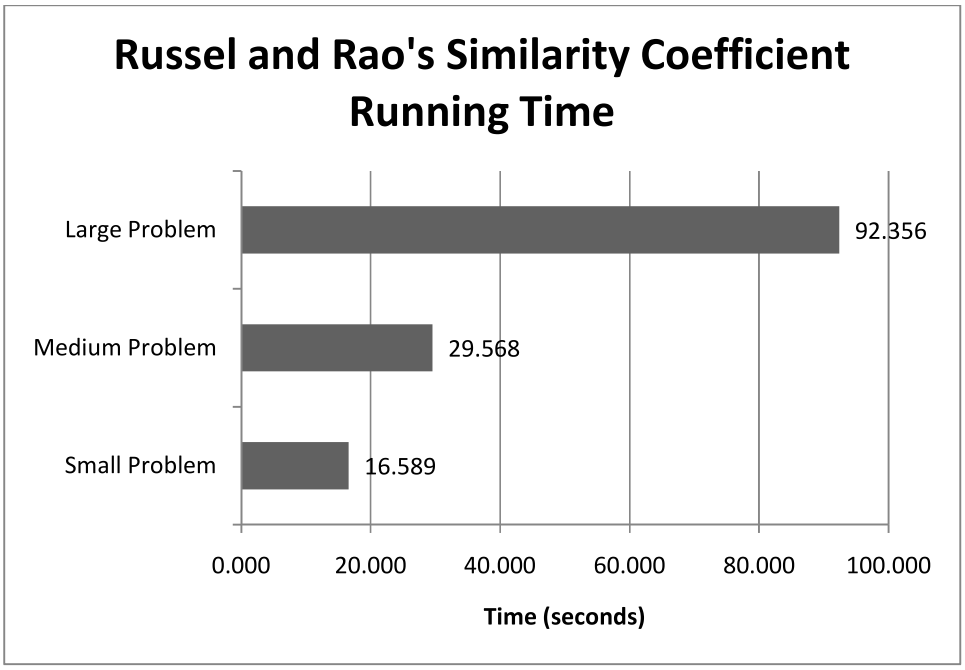


Figure 5. 8 Russel and Rao's Similarity Coefficient Running Time

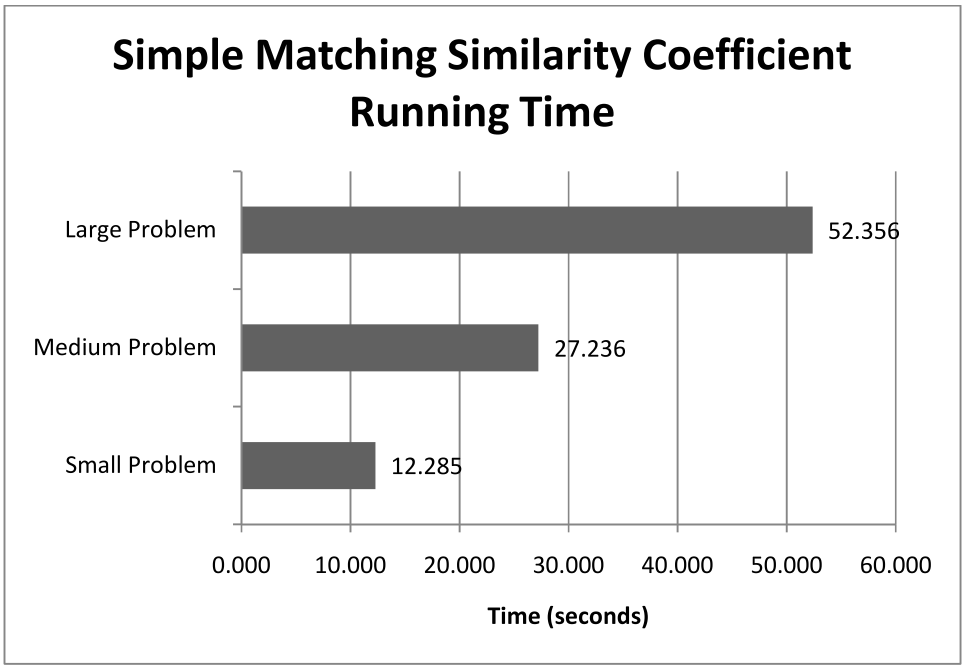


Figure 5. 9 Simple Matching Similarity Coefficient Running Time

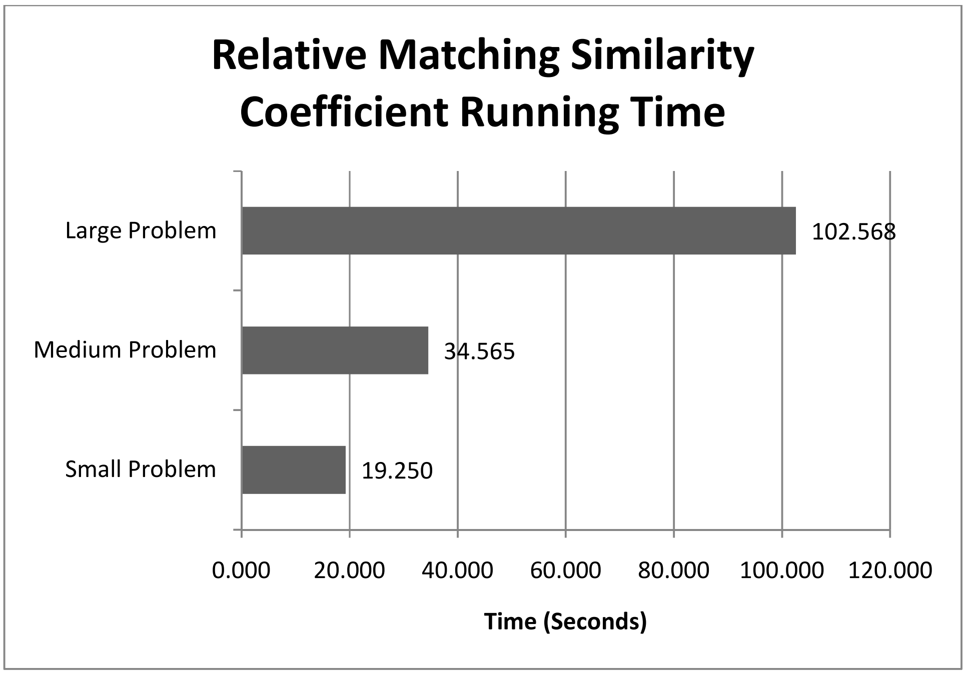


Figure 5. 10 Relative Matching Similarity Coefficient Running Time

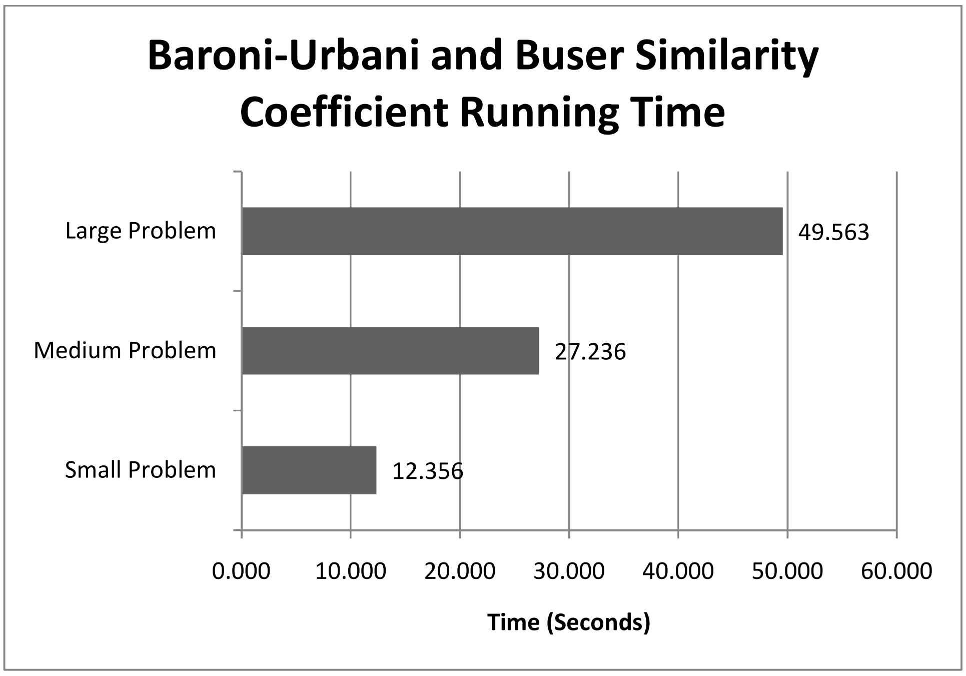


Figure 5. 11 Baroni-Urbani and Buser Similarity Coefficient Running Time

Examing the process run times shown in Figures 4.22 to 4.26, the similarity coefficient which reaches the solution as quickly as possible is the Jaccard similarity coefficient. This could be because the Jaccard similarity coefficient, unlike the others, does not include dissimilarity factors (or *d* values) in their calculations. This is advantageous because the artificial ants in this algorithm cannot process dissimilarity coefficients, and some similarity coefficients such as Russel and Rao’s Siimilarity coefficient and the relative matching similarity coefficient include dissimilarity factors in similarity coefficients (different from similarity/dissimilarity coefficients) without rationalizing them. Not rationalizing the dissimilarity factors can cause the process to become unstable, and this instability is likely the cause of longer process run times.

CHAPTER VI

CONCLUSIONS AND RECOMMENDATIONS

Concluding Remarks

Most of the ant algorithm models in literature focus on developing part families to be as optimal as possible, rather than focusing on the efficiency of the algorithm itself. Many of the ant algorithm optimization techniques in literature are also developed into software, which are used by practitioners in the industry. The operational requirements of these ant algorithm software are very demanding. Thus, there exists a need for an efficient, easy to program ant algorithm that would create the optimal cellular manufacturing problem solution, and in doing so would use minimum resources. In order to satisfy the need for an artificial ant-based algorithm that is efficient as well as easy to program, an existing ant algorithm was modified so that it could be used to solve the cellular manufacturing problem. The original algorithm, AntClass uses Euclidean vectors to measure the similarity between parts. The modified version used in this research, due to the fact that similarity is used to group parts together instead of distances, the modified version uses similarity coefficients. The concept of heaping clusters was also introduced to ant algorithms for cellular manufacturing. Instead of using Euclidean vectors to measure the distance to the center of a heap, as is such in the AntClass algorithm, an average similarity was introduced to measure the similarity between a part and a heap, therefore allowing the easy rebuilding of clusters in order to compare the effects of different similarity coefficients on the ant-based algorithm.

In the literature, we have seen that there are several ways in which to solve the cellular manufacturing problem. Many of these existing methods however, are not as flexible as the swarm intelligence methods. Even though there are existing ant algorithm models, there has been limited comparison of the processes within the ant algorithm, with other replaceable processes. In order to determine a similarity coefficient, 5 similarity coefficients were selected. Of the five compared similarity coefficients, two of the similarity coefficients worked well with the algorithm. The Jaccard similarity coefficient produces slightly lower quality results in a slightly shorter time than the Baroni-Urbani and Buser similarity coefficient. The Simple Matching, Relative Matching and Russel and Rao's Similarity coefficient are not recommended for this algorithm. The simple matching similarity coefficient does not produce high quality solutions in phase 1, therefore leading to longer than necessary remaining phases. The Relative Matching and Russel and Rao's Similarity coefficients are not suggested because they seem to be unstable with this type of algorithm, therefore causing process lockups, and longer process run times. Similarity/Dissimilarity Coefficients, which differ from similarity coefficients because they have a range of -1 to 1 instead of 0 to 1, will not work with this algorithm, because the artificial ant's logic does not consider negative values.

Recommendations for future research

With the modifications and comparisons within this artificial ant-based clustering algorithm for cellular manufacturing, several windows of opportunity for new research open up. The relationship between exceptional elements and grouping efficiency can now be formally investigated. The algorithm can be additionally modified so that it considers negative values and therefore some similarity/dissimilarity coefficients can be used, or tested on this algorithm. There are also additional factors that can now be considered. Product volume and demand can be taken into account in the artificial ant-based algorithm. Other factors, such as operator considerations and setup times, or idle time for machine repairs can also be taken into account.

APPENDICES

APPENDIX A

Layout Presentation and Similarity Coefficient Calculations

Output for two dimensional chessboards

The algorithm does not output pictures; it outputs a set of coordinates for the parts and the ants. If two or more parts have the same coordinates, they are considered heaped together. An example of a layout for the medium sized problem is:

```
Ants' initial positions:

Ant[ 0]: ( 0 ,  0)

Ant[ 1]: ( 2 ,  5)

Parts' initial positions:

Part[ 0]: ( 0 ,  0)                  Part[10]: ( 5 ,  3)

Part[ 1]: ( 2 ,  5)                  Part[11]: ( 5 ,  0)

Part[ 2]: ( 3 ,  7)                  Part[12]: ( 7 ,  5)

Part[ 3]: ( 5 ,  8)                  Part[13]: ( 0 ,  4)

Part[ 4]: ( 8 ,  6)                  Part[14]: ( 0 ,  3)

Part[ 5]: ( 0 ,  2)                  Part[15]: ( 4 ,  0)

Part[ 6]: ( 2 ,  2)                  Part[16]: ( 7 ,  4)

Part[ 7]: ( 6 ,  4)                  Part[17]: ( 3 ,  2)

Part[ 8]: ( 8 ,  3)                  Part[18]: ( 5 ,  7)

Part[ 9]: ( 1 ,  1)                  Part[19]: ( 8 ,  5)
```

Similarity Coefficient Calculations

The similarity coefficients are calculated according to their corresponding formulas and then stored in a table, such as the one for the Medium sized Problem using the Baroni-Urbani and Buser Similarity coefficient shown in table A.1.

Table A. 1 Similarity Coefficient Calculation and Storage

		Parts																			
Parts		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	1	0.48	0.00	0.00	0.00	0.48	0.26	0.00	0.00	0.30	0.46	0.00	0.46	0.00	0.00	0.48	0.00	0.28	0.00	0.00	0.28
	2	0.00	0.48	0.26	0.00	0.00	0.00	0.00	0.48	0.46	0.00	0.46	0.00	0.48	0.46	0.00	0.48	0.45	0.28	0.48	0.00
	3	0.00	0.26	0.75	0.65	0.00	0.63	0.65	0.26	0.22	0.22	0.40	0.22	0.26	0.40	0.00	0.26	0.36	0.52	0.26	0.52
	4	0.00	0.00	0.46	0.67	0.00	0.65	0.67	0.00	0.00	0.26	0.26	0.26	0.00	0.26	0.00	0.00	0.22	0.55	0.00	0.55
	5	0.48	0.00	0.00	0.00	0.48	0.26	0.00	0.00	0.30	0.46	0.00	0.46	0.00	0.00	0.48	0.00	0.28	0.00	0.00	0.28
	6	0.26	0.00	0.63	0.65	0.26	0.75	0.65	0.00	0.00	0.40	0.22	0.40	0.00	0.22	0.26	0.00	0.36	0.52	0.00	0.52
	7	0.00	0.00	0.65	0.67	0.00	0.65	0.67	0.00	0.00	0.26	0.26	0.26	0.00	0.26	0.00	0.00	0.22	0.55	0.00	0.55
	8	0.00	0.48	0.26	0.00	0.00	0.00	0.00	0.48	0.46	0.00	0.46	0.00	0.48	0.46	0.00	0.48	0.45	0.28	0.48	0.00
	9	0.30	0.46	0.22	0.00	0.30	0.00	0.00	0.46	0.48	0.28	0.45	0.28	0.46	0.45	0.30	0.46	0.43	0.26	0.46	0.26
	10	0.46	0.00	0.22	0.26	0.46	0.40	0.26	0.00	0.28	0.58	0.00	0.45	0.00	0.00	0.46	0.00	0.43	0.00	0.00	0.26
	11	0.00	0.46	0.40	0.26	0.00	0.22	0.26	0.46	0.00	0.00	0.58	0.28	0.46	0.45	0.00	0.46	0.43	0.43	0.46	0.26
	12	0.46	0.00	0.22	0.26	0.46	0.40	0.26	0.00	0.45	0.45	0.28	0.58	0.00	0.00	0.46	0.00	0.26	0.26	0.00	0.43
	13	0.00	0.48	0.26	0.00	0.00	0.00	0.00	0.48	0.00	0.00	0.46	0.00	0.48	0.46	0.00	0.48	0.45	0.28	0.48	0.00
	14	0.00	0.46	0.40	0.26	0.00	0.22	0.26	0.46	0.00	0.00	0.45	0.00	0.46	0.58	0.00	0.46	0.43	0.43	0.46	0.26
	15	0.48	0.00	0.00	0.00	0.48	0.26	0.00	0.00	0.46	0.46	0.00	0.46	0.00	0.00	0.48	0.00	0.28	0.00	0.00	0.28
	16	0.00	0.48	0.26	0.00	0.00	0.00	0.00	0.48	0.00	0.00	0.46	0.00	0.48	0.46	0.00	0.48	0.45	0.28	0.48	0.00
	17	0.28	0.45	0.36	0.22	0.28	0.36	0.22	0.45	0.43	0.43	0.43	0.26	0.45	0.43	0.28	0.45	0.67	0.22	0.45	0.00
	18	0.00	0.28	0.52	0.55	0.00	0.52	0.55	0.28	0.00	0.00	0.43	0.26	0.28	0.43	0.00	0.28	0.22	0.67	0.28	0.55
	19	0.00	0.48	0.26	0.00	0.00	0.00	0.00	0.48	0.00	0.00	0.46	0.00	0.48	0.46	0.00	0.48	0.45	0.28	0.48	0.00
	20	0.28	0.00	0.52	0.55	0.28	0.52	0.55	0.00	0.26	0.26	0.26	0.43	0.00	0.26	0.28	0.00	0.00	0.55	0.00	0.67

APPENDIX B: Data from Large Example

Table B. 1 Initial Machine-Part Matrix

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	25	25	26	27	28	29	30	31	32	33	34	36	37	38	39	40	65
1	1								1							1	1																1							
2										1			1	1								1													1					1
3		1									1	1			1								1	1							1			1						
4								1											1		1							1								1	1	1		
5										1			1	1								1													1					1
6				1	1													1								1	1			1										
7			1																						1							1								
8				1	1													1								1	1			1										
9						1	1													1									1										1	
10						1	1													1									1										1	
11										1			1	1								1													1					1
12				1	1													1								1	1			1										
13	1								1							1	1																1							
14			1																						1							1								
15				1	1													1								1	1			1										
16								1											1		1							1								1	1	1		
17						1	1													1									1										1	
18				1	1													1								1	1			1										
19										1			1	1								1													1					1
20		1									1	1			1								1	1							1			1						
21	1								1							1	1																1							
22	1								1							1	1																1							
23			1																						1							1								
24			1																						1							1								

Table B. 2 Machine Part Matrix For the Large Problem with a 100% grouping efficiency

	1	9	16	17	33	10	13	14	22	35	36	2	11	12	15	23	25	31	34	8	19	21	28	37	38	39	4	5	18	26	27	30	3	25	32	6	7	20	29	40
1	1	1	1	1	1																																			
13	1	1	1	1	1																																			
21	1	1	1	1	1																																			
22	1	1	1	1	1																																			
2						1	1	1	1	1	1																													
5						1	1	1	1	1	1																													
11						1	1	1	1	1	1																													
19						1	1	1	1	1	1																													
3												1	1	1	1	1	1	1	1																					
20												1	1	1	1	1	1	1	1																					
4																				1	1	1	1	1	1	1														
16																				1	1	1	1	1	1	1														
6																											1	1	1	1	1	1								
8																											1	1	1	1	1	1								
12																											1	1	1	1	1	1								
15																											1	1	1	1	1	1								
18																											1	1	1	1	1	1								
7																																	1	1	1					
14																																	1	1	1					
23																																	1	1	1					
24																																	1	1	1					
9																																				1	1	1	1	1
10																																				1	1	1	1	1
17																																				1	1	1	1	1

APPENDIX C

Equations and Variables

List of Equations

Equation 15: Calculating the size of the 2 dimensional chessboard

$$m^2 = n_p \times 4 \textbf{ OR } m = \sqrt{n_p \times 4}$$

Equation 16: Calculating the number of artificial ants

$$n_{ants} = \frac{n_p}{10}$$

Equation 17: Similarity density function to measure the similarity of a part P_k with its surroundings

$$f(P_k) = \frac{\sum s\,(P_k, P_l)}{nP_l}, P_l \in \, n^2$$

Equation 18: Probability transfer function for an artificial ant to pick up a part

$$P_{pick}(P_k) = \left(\frac{k_p}{k_p + f(P_k)}\right)^2$$

Equation 19: Probability transfer function for an artificial ant to drop a part

$$P_{drop}(P_k) = \begin{cases} 2f(P_k) & if\ f(P_k) < k_d \\ 1 & otherwise \end{cases}$$

Equation 20: Average similarity between a part and a heap

$$\bar{s}(P_k, H_j) = \sum_{i=1}^{h} \frac{s(P_k, P_i)}{nP_h + 1}$$

Equation 21: Jaccard Similarity Coefficient

$$\sigma = \frac{a}{a + b + c}$$

Equation 22: Russel and Rao’s Similarity Coefficient

$$\sigma = \frac{a}{a+b+c+d}$$

Equation 23: Simple Matching Coefficient

$$\sigma = \frac{a+d}{a+b+c+d}$$

Equation 24: Relative Matching Coefficient

$$\sigma = \frac{a+\sqrt{ad}}{a+b+c+d+\sqrt{ad}}$$

Equation 25: Baroni-Urbani and Buser Similarity Coefficient

$$\sigma = \frac{a+\sqrt{ad}}{a+b+c+d+\sqrt{ad}}$$

Equation 26: Grouping Efficiency for a Machine-Part Matrix

$$\eta = w\eta_1 + (1-w)\eta_2$$

Equation 27: Left side partial grouping efficiency

$$\eta_1 = \frac{e_c}{\sum_{r=1}^{k} M_r N_r}$$

Equation 28: Right Side grouping efficiency

$$\eta_2 = 1 - \left[\frac{e_o}{n_m n_p - \sum_{r=1}^{k} M_r N_r}\right]$$

List and definition of Variables

1. n_p is the number of parts
2. H_j is the current heap
3. nP_h is the number of parts in H_j
4. $\bar{s}(P_k, H_j)$ is the average similarity between a part and a heap
5. *a* is the number of machines which produce both components *i* and *j*
6. *b* is the number of machines which produce only component *i*
7. *c* is the number of machines which produce only component *j*
8. *d* is the number of machines which produce neither components *i* or *j*
9. e_c is the number of non-exceptional elements
10. e_e is the number of exceptional elements
11. $f(P_k)$ similarity density function to measure the similarity of a part P_k with its surroundings
12. h is the number of parts in the H_j
13. *k* is the number of diagonal blocks on the machine-part matrix
14. k_d is a constant value with a range of $0 < k_d < 1$
15. k_p constant value with their range between $0 < k_p < 1$
16. *m* is the length and width of the two dimensional chessboard,
17. M_r is the number of machines in the r^{th} cell
18. *n*2 is the surrounding area that is recognizable to an artificial ant (3-8 cells)
19. n_m is the number of machines
20. n_p is the number of parts
21. N_r is the number of components in the rth family
22. is the number of operations in the machine part matrix
23. $P_{drop}(P_k)$ probability transfer function for an artificial ant to lay aside the Part P_k
24. P_k is the part held or encountered by an artificial ant
25. P_l is the part located in one of the 3-8 surrounding cells on the 2-dimensional chessboard
26. $P_{pick}(P_k)$ is the probability transfer function for an artificial ant to pick up the part P_k

27. $S(P_k, P_l)$ is the similarity between parts P_k and P_l
28. v is the number of voids in the solution
29. w is a constant relating the importance of intercellular movement (equal to 0.5 in the study)
30. $s(P_k, P_i)$ is the similarity between part k and part i

APPENDIX D

Source Code

```
#include<stdio.h>

#include<math.h>

#include<stdlib.h>

#define PARTS 20

#define MACHINES 8

#define ANTS 2

#define MEM 8

#define TRUE 1

#define FALSE 0


#define SUCCESS 0

#define FAIL 1


//------------------------------------------------------------------------


int CM[MACHINES][PARTS];                    // the components matrix

double SM[PARTS][PARTS];                    // the similarity matrix


int C = ceil(sqrt((double) PARTS * 4.0));  // Dynamic chessboard dimension
```

```
//------------------------------------------------------------------------


int InRange(int x, int y)
{
  if(x>=0 && x<C && y>=0 && y<C) return TRUE;
  return FALSE;
}


//------------------------------------------------------------------------


#include "T_position.hpp"
#include "T_que.hpp"
#include "T_part.hpp"


//------------------------------------------------------------------------


TPart PartList[PARTS];


//------------------------------------------------------------------------


void InitiatePartLocations(void)
{
  for(int i=0; i<PARTS; i++)
  {
    PartList[i].SetId(i);  PartList[i].TakeALocation();
  }
}

//----------------------------------------------------------------------------

int PartsAtXY(int x, int y, int *found_parts)
{
  int found=0;

  for(int i=0; i<PARTS; i++)
    if(PartList[i].AreYouAt(x, y) == TRUE)
      found_parts[found++] = PartList[i].GetId();
  return found;
}

//----------------------------------------------------------------------------

int HeapPart(int *FoundParts, int Prts)
{
  double Similarity(TPart, TPart); // just a prototype
  double Sum=0, *sum = new double[Prts];
```

```
for(int i=0; i<Prts; i++) sum[i]=0;  // resetting sums


int count = 0;

for(int i=0; i<Prts-1; i++)

  for(int j=i+1; j<Prts; j++)

  {

    ++count;

    Sum += Similarity(PartList[FoundParts[i]], PartList[FoundParts[j]]);

  }

Sum /= count;


for(int i=0; i<Prts; i++)

{

  for(int j=0; j<Prts; j++)

    if(i==j) continue;

    else

      sum[i] += Similarity(PartList[FoundParts[i]], PartList[FoundParts[j]]);

  sum[i] /= (Prts-1);

}


double MaxValue, Value; int BestPart;

for(int i=0; i<Prts; i++)

{
    Value = fabs(Sum-sum[i]);
    if(i==0)
    {
      MaxValue = Value;  BestPart = i;
    }
    else
        if(Value > MaxValue)
        {
          MaxValue = Value;  BestPart = i;
        }
  }

  delete[] sum;
  return BestPart;
}

//----------------------------------------------------------------------------

#include "T_ant.hpp"

//-------------------------------------------------------------------------------

TAnt AntList[ANTS];
```

```
//-----------------------------------------------------------------------------

void InitiateAntLocations(void)
{
  int occupied;
  for(int i=0; i<ANTS; i++)
  {
    AntList[i].SetId(i);
    do
    {
      occupied = FALSE;
      AntList[i].TakeALocation();
      for(int j=0; j<i; j++)
        if(AntList[j].AreYouAt(AntList[i].GetXLocation(),  AntList[i].GetYLocation())  ==
TRUE)
        {
          occupied = TRUE;  break;
        }
    } while(occupied == TRUE);
  }
}
```

```
//-----------------------------------------------------------------------------


double Similarity(TPart p1, TPart p2)
{
  int a, b, c, d, k=0;

  a = b = c = d = 0;
  for(int i=0; i<MACHINES; i++)
    if(CM[i][p1.GetId()] == 1 && CM[i][p2.GetId()] == 1)
      ++a;
    else
      if(CM[i][p1.GetId()] == 1 && CM[i][p2.GetId()] == 0)
        ++b;
      else
        if(CM[i][p1.GetId()] == 0 && CM[i][p2.GetId()] == 1)
          ++c;
        else
          if(CM[i][p1.GetId()] == 0 && CM[i][p2.GetId()] == 0)
            ++d;

  double Num = (a-k) + sqrt((a-k)*d);
  double Den = Num + b + c + d;
  return Num/Den;
```

```
}

//--------------------------------------------------------------------------

void ComputeSimilarityMatrix(void)
{
  for(int i=0; i<PARTS; i++)
    for(int j=0; j<PARTS; j++)
      SM[i][j] = Similarity(PartList[i], PartList[j]);
}

//--------------------------------------------------------------------------

int LoadComponentMatrix(char *file_name)
{
  FILE *f = fopen(file_name, "r");
  if(!f) return FALSE;
  for(int i=0; i<MACHINES; i++)
    for(int j=0; j<PARTS; j++)
      fscanf(f, "%d", &CM[i][j]);
  fclose(f);

  ComputeSimilarityMatrix();
  return TRUE;
}

//--------------------------------------------------------------------------


void PrintParts(char *f_name)
{
  FILE *f = fopen(f_name, "a");

  fprintf(f, "Parts:\n======\n");
  for(int i=0; i<PARTS; i++)
    fprintf(f,   "P%d   -->   (%d,   %d)\n",   i,   PartList[i].GetXLocation(),
PartList[i].GetYLocation());
  fprintf(f, "\n\n");
  fclose(f);
}

//--------------------------------------------------------------------------


void PrintAnts(char *f_name)
{
  FILE *f = fopen(f_name, "a");
  fprintf(f, "Ants:\n=====\n");
  for(int i=0; i<ANTS; i++)
  {
    fprintf(f, "Ant%d --> (%d, %d) --> [%d parts]:", i, AntList[i].GetXLocation(),
                AntList[i].GetYLocation(), AntList[i].GetParts());
    for(int j=0; j<AntList[i].GetParts(); j++)
      fprintf(f, "%d, ", AntList[i].GetPart(j));
    fprintf(f, "\n");
  }
  fprintf(f, "\n\n");
  fclose(f);
}

//---------------------------------------------------------------------------

int main(int argc, char* argv[])
{
  randomize();
  InitiatePartLocations();  PrintParts("d:\\Ant\\AntOutput.txt");
  InitiateAntLocations(); PrintAnts("d:\\Ant\\AntOutput.txt");
  LoadComponentMatrix("d:\\Ant\\CM01.txt");  // it does compute similarities as well
  for(int i=0; i<ANTS; i++)
  {
    AntList[i].Move();  PrintAnts("d:\\Ant\\AntOutput.txt");
  }

  return 0;
}
//---------------------------------------------------------------------------
```

REFERENCES

1. Abdule-Wahab, R. S., Monmarché, N., Silmane, M., Fahdil, M. A., & Saleh, H. H. (2006). A Scatter Search Algorithm for the Automatic Clustering Problem. *Lecture Notes in Computer Science* , 350-364.

2. Adil, G., Rajamani, D., & Strong, D. (1993). A mathematical model for cell formation considering investment and operational costs. *European Journal of Operational Research* , *69* (3), 330-341.

3. Al-Sultan, K. S. (1997). A hard clustering approach to the part family problem. *Production Planning and Control* , *8*, 231-236.

4. Arikan, F., & Güngör, Z. (2009). Modeling of a Manufacturing Cell Design Problem with Fuzzy Multi-objective Parametric Programming. *Mathematical and Computer Modelling* , *50* (3-4), 407-420.

5. Askin, R., & Chiu, K. (1990). A graph partitioning procedure for machine assignment and cell formation in group technology. *International Journal of Production Research* , *28* (8), 1555-1572.

6. Askin, S., Cresswell, S., Goldberg, J., & A.J., V. (1991). A Hamiltonian path approach to reordering the part-machine matrix for cellular manufacturing. *International Journal of Production Research* , *29* (6), 1081-1100.

7. Bajestani, M. A., Rabanni, M., Rahimi-Vahed, A. R., & Khoshkhou, G. B. (2009). A multi-objective scatter search for a dynamic cell formation problem. *Computers and Operations Research* , *36* (3), 777-794.

8. Baykasoglu, A. (2001). MOAPPS 1.0: aggregate production planning using the multiple-objective tabu search. *International Journal of Production Research, , 39* (16), 3685-3702.

9. Beaulieu, A., Gharbi, A., & Ait-Kadi. (1997). An algorithm for the cell formation and the machine selection problems in the design of a cellular manufacturing system. *International Journal of Production Research , 35* (7), 1857-1874.

10. Beni, G., & Wang, J. (1989). Swarm Intelligence in Cellular Robotic Systems. *NATO Advanced Workshop on Robots and Biological Systems.* Tuscany, Italy.

11. Bhide, P., Bhandwale, A., & Kesavadas, T. (2005). Cell formation using multiple process plans. *Journal of Intelligent Manufacturing , 16* (1), 53-65.

12. Black, J. (2000). Lean Manufacturing Implementation. *Innovations in competitive manufacturing* , 177-186.

13. Brandon, J. (1996). *Cellular Manufacturing: Integrating Technology and Management.* England: Somerset.

14. Brown, E. C., & Sumichrast, R. T. (2001). CF-CGA: a grouping genetic algorithm for the cell formation problem. *International Journal of Production Research , 36,* 3651-3669.

15. Burbridge, J. L. (1992). Change to group technology: process organisation is obsolete. *International Journal of Production Research , 30* (5), 1202-1219.

16. Caux, C., Bruniaux, R., & Pierreval, H. (2000). Cell formation with alternative process plans and machine capacity constraints: A new combined approach. *International Journal Production Economics , 64* (1-3), 279-284.

17. Chan, H. M., & Milner, D. A. (1982). Direct Clustering Algorithm for Group Formation in cellular Manufacture. *Journal of Manufacturing Systems , 1* (1), 65-75.

18. Chandrasekharan, M. P., & Rajagopalan, R. (1986a). An ideal seed non-hierarchical clustering algorithm for cellular manufacturing. *International Journal of Production Research , 24* (2), 451-464.

19. Chandrasekharan, M. P., & Rajagopalan, R. (1989). GROUPABILITY: an analysis of the properties of binary data matrices for group technology. *International Journal of Production Research , 27* (6), 1035-1052.

20. Chandrasekharan, M. P., & Rajagopalan, R. (1986b). MODROC: An extension of rank order clustering for group technology. *International Journal of Production Research , 24* (5), 1221-1233.

21. Chang, P.-T., & Lee, E. S. (2000). A multisolution Methid for Cell Formation-Exploring Practical Alternatives in Group Technology Manufacturing. *Computers and Mathematics with Applications , 40* (10-11), 1285-1296.

22. Choobineh, F. (1988). A framework for the design of cellular manufacturing systems. *International Journal of Production Research , 26* (7), 1161-1172.

23. Chow, W. S., & Hawaleshka, O. (1993). Minimizing intercellular part movements in manufacturing cell formation. *International Journal of Production Research , 31* (9), 357-372.

24. Chu, C.-h., & Tsai, M. (1990). A comparison of three array-based clustering techniques for manufacturing cell formation. *International Journal of Production Research , 28* (8), 1417-1433.

25. Co, H., & Arrar, A. (1988). Configuring cellular manufacturing systems. *International Journal of Production Research , 26* (9), 1511-1522.

26. da Silveira, G. (1999). A methodology of implementation of cellular manufacturing. *International Journal of Production Research , 37* (2), 467-479.

27. Dahel, N., & Smith, S. (1993). Designing flexibility into cellular manufacturing systems. *International Journal of Production Research , 43* (5), 933-945.

28. Damodaran, V., Singh, N., & Lashkari, R. (1993). Design of cellular manufacturing systems with refixturing and material handling considerations. *Applied Stochastic Models and Data Analysis , 9* (2), 97-109.

29. De Witte, J. (1980). The use of similarity coefficients in production flow analysis. *International Journal of Production Research , 18* (4), 503-514.

30. Defersha, F. M., & Chen, M. (2006). A comprehensive mathematical model for the design og cellular manufacturing systems. *International Journal of Production Economics , 103* (2), 767-783.

31. Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics , 26* (1), 29-41.

32. Du, K. L. (2010). Clustering: A neural network approach. *Neural Networks , 23* (1), 89-107.

33. Dutta, S. P., Lashkari, R. S., Nadoli, G., & Ravi, T. (1986). A heuristic procedure for determining manufacturing families from design based grouping for FMS. *Computers and Industrial Engineering , 10* (3), 193-201.

34. El-Essawy, I., & Torrance, J. (1972). Component Flow Analysis. *The Production Engineer* , 165-70.

35. Gombinski, ,. J. (1969). Fundamental aspects of component classification. *Annals of the CIRP* , 367-374.

36. Gonçalves, J. F., & Resende, M. G. (2004). An evolutionary algorithm for manufacturing cell formation. *Computers and Industrial Engineering , 47* (2-3), 247-273.

37. Groover, M., & Zimmers, E. (1984). *CAD/CAM: Computer-aided design and manufacturing.* Englewood Cliffs, NJ, USA: Prentice-Hall.

38. Gunasingh, K. R., & Lashkari, R. S. (1989). Machine grouping problem in cellular manufacturing systems— an integer programming approach Machine grouping problem in cellular manufacturing systems-an integer programming approach. *International Journal of Production Research , 27* (9), 1465-1473.

39. Gupta, T. (1993). Design of manufacturing cells for flexible environment considering alternative routeing. *International Journal of Production Research , 31* (6), 1259-1273.

40. Gupta, T., & Seifoddinni, H. (1990). Production data based similarity coeffcient for machine-component grouping decisions in the design of a cellular manufacturing system. *InternationalJournal of Production Research , 28* (7), 1247-1269.

41. Hartuv, E., & Shamir, R. (2000). A clustering algorithm based on graph connectivity. *Information Processing Letters , 76* (4-6), 175-181.

42. Hinchey, M. G., Sterritt, R., & Rouff, C. (2007). Swarms and Swarm Intelligence. *Computer , 40* (4), 111-113.

43. Hothersall, D. (2004). *History of Psychology.* Boston: McGraw-Hill.

44. Hsu, C.-m., & Su, C.-t. (1998). Multi-objective machine-component grouping in cellular manufacturing: a genetic algorithm. *Production Planning and Control , 9* (2), 155-166.

45. Hyde, W. F. (1981). *Improving productivity by classification, coding, and data base standardization.* New York: Marcel Dekker.

46. Hyer, N. L., & Wemmerlöv, U. (1984). Group technology and productivity. *Harvard Business Review* , 140-149.

47. Hyer, N. L., Paulson, J., & Handfield, R. (1989). An effective implementation of parts coding system: a case study of PACCAR, Inc. *Proceedings of the Decision Sciences Institute Annual Conference* , 1005-1007.

48. Islam, K., & Sarker, B. (2000). A similarity coefficient measure and machine-parts grouping in cellular manufacturing systems. *International Journal of Production Research , 38* (3), 699-720.

49. Islier, A. A. (1998). A genetic algorithm approach for multiple criteria facility layout design. *International Journal of Production Research , 36* (6), 1549 - 1569.

50. Islier, A. A. (2005). Group technology by an ant system algorithm. *International Journal of Production Research , 43* (5), 913-932.

51. Jeon, G., & Leep, H. (2006). Forming part families by using genetic algorithm and designing machine cells under demand changes. *Computers & Operations Research , 33* (1), 263-283.

52. Jeon, G., Leep, H. R., & Parsaei, H. (1998). A cellular manufacturing system based on new similarity coefficient which considers alternative routes during machine failure. *Computers & Industrial Engineering , 34* (1), 21-36.

53. Kamrani, A., & Parsaei, H. (1993). A group technology based methodology for machine cell formation in a computer integrated manufacturing environment. *Computers and Industrial Engineering , 24* (3), 431-447.

54. Kandiller, L. (1998). A cell formation algorithm: Hypergraph approximation- Cut tree. *European Journal of Operational Research , 109* (3), 686-702.

55. Kao, Y., & Fu, S. C. (2006). An ant-based clustering algorithm for manufacturing cell design. *International Journal of Advanced Manufacturing Technology , 28* (11-12), 1182-1189.

56. Kao, Y., & Li, Y. L. (2008). Ant colony recognition systems for part clustering problems. *International Journal of Production Reseach , 46* (15), 4237-4258.

57. King, J. R. (1980). Machine-component grouping in production flow analysis an approach. *International Journal of Production Research , 18* (2), 117-133.

58. Kumar, A., & Vannelli, A. (1987). Strategic subcontracting for efficient disaggregated manufacturing. *International Journal of Production Research , 9* (6), 1715-1728.

59. Kumar, A., Kusiak, A., & Vannelli, A. (1986). Grouping of parts and components in flexible manufacturing systems. *European Journal of Operational Research , 24* (3), 387-397.

60. Kusiak, A. (1987). The generalized group technology concept. *International Journal of Production Research , 25* (4), 561-569.

61. Kusiak, A., & Cho, M. (1992). Similarity coefficient algorithms for solving the group technology problem . *International Journal of Production Research* , 2633-2646.

62. Kusiak, S., Kaparthi, S., Suresh, N., & Cerveny, R. (1993). An improved neural network leader algorithm for part-machine grouping in group technology. *European Journal of Operational Research , 69* (3), 342-356.

63. Labroche, N., Monmarché, N., & Venturini, G. (2003). AntClust: Ant Clustering and Web Usage Mining. *Lecture Notes in Computer Science , 2723* (201), 25-36.

64. Lesmana, S. (2002). *Formation of flexible manufacturing cells with human lifting consideration (Master's Thesis).* University of Windsor: Windsor, Ontario.

65. Liang, M., & Taboun, S. (1992). Part selection and part assignment in flexible manufacturing systems with cellular layout. *Computers and Industrial Engineering , 23* (1-4), 63-67.

66. Lin, S.-W., Ying, K.-C., & Lee, Z.-J. (2010). Part-machine cell formation in group technology using a simulated annealing-based meta-heuristic. *International Journal of Production Research , 48* (12), 3579-3591.

67. Logendran, R. (1993). A Binary Integer Programming Approach for Simultaneous Machine-Part Grouping in. *Computers and Industrial Engineering , 24* (3), 329-336.

68. McAuley, J. (1972). Machine grouping for efficient production. *The Production Engineer , 51* (2), 52-53.

69. McCormick, W. T., Schweitzer, ,. P., & White, ,. T. (1972). Problem Decomposition and Data Reorganization by a Clustering Technique. *Operations Research , 20* (5), 993-1009.

70. Miltenburg, J., & Zhang, W. (1991). A Comparative Evaluation of Nine Well-Known algorithms for Solving the Cell Formation Problem in Group Technology. *Journal of Operations Management , 10* (1), 44-72.

71. Monmarché, N., Slimane, M., & Venturini, G. (1999). *AntClass: découverte de classes dans des données numériques grâce à l'hybridation d'une colonie de fourims avec l'algorithme des centres mobiles.* Tours, France: Université de Tours.

72. Moon, C., & Gen, M. (1999). A genetic algorithm-based approach for design of independent manufacturing cells. *International Journal of Production Economics , 60-61*, 421-426.

73. Mosier, C. T. (1989). An experiment investigating the application of clustering procedures and similarity coefficients to the GT machine cell formation problem. *International Journal of Production Research , 48* (12), 1811-1835.

74. Moussa, S. E., & Kamel, M. (1996). A Direct Method for Cell Formation and Part-Machine Assignment Based on Operation Sequences and Processing Time Similarity. *Engineering Design and Automation , 35* (3-4), 141-155.

75. Onwubolu, G. C., & Mutingi, M. (2001). A genetic algorithm approach to cellular manufacturing systems. *Computers and Industrial Engineering , 39* (1), 125-144.

76. Opitz, H. (1970). *A Classification System to Describe Work-Pieces.* New York: Pergamon Press.

77. Opitz, H., & Wiendahl, H. P. (1971). Group technology and manufacturing systems for small and medium quantity production. *International Journal of Product Research , 9* (1), 181-203.

78. Ozdemir, A. A. (1995). *Simultaneous part family and machine cell formations in cellular manufacturing systems: an analytical and algorithmic approach (Master's Thesis).* University of Windsor: Windsor, Ontario.

79. Rajagopalan, R., & Batra, J. (1975). Design of cellular production systems A graph-theoretic approach. *International Journal of Production Research , 13* (6), 567-579.

80. Rajamani, N., Singh, N., & Aneja, Y. (1992). A model for cell formation in manufacturing systems with sequence dependence. *International Journal of Production Research , 30* (6), 1227-1235.

81. Rajamani, N., Singh, N., & Aneja, Y. (1990). Integrated design of cellular manufacturing systems in the presence of alternate process plans. *International Journal of Production Research , 28* (8), 1541-1554.

82. Satoglu, S., & Suresh, N. (2009). A goal-programming approach for design of hybrid cellular manufacturing systems in dual resource constrained environments. *Computers and Industrial Engineering , 56* (2), 560-575.

83. Seifoddini, H., & Hsu, C.-P. (1994). Comparitive study of similarity coefficients and clustering algorithms in cellular manufacturing. *Journal of Manufacturing Systems , 13* (2), 119-127.

84. Selvam, R. P., & Balasubramanian, K. N. (1985). Algorithmic grouping of operation sequences. *Engineering Costs and Production Economics , 9* (1-3), 125-134.

85. Shafer, S. M., & Merideth, J. R. (1990). A comparison of selected manufacturing cell formation techniques. *Ineternational Journal of Production Research , 28* (4), 661-673.

86. Shafer, S., & Rogers, D. (1992). A mathematical programming approach for dealing with exceptional elements in cellular manufacturing. *International Journal of Production Research , 30* (5), 1029-1036.

87. Shafer, S., & Rogers, D. (1993a). Similarity and distance measures for cellular manufacturing. Part I. A survey. *International Journal of Production Research , 31* (5), 1133-1142.

88. Shafer, S., & Rogers, D. (1993b). Similarity and distance measures for cellular manufacturing. Part II. An extension and comparison. *International Journal of Production Research , 31* (6), 1315-1326.

89. Singh, N., & Mohanty, B. (1991). Fuzzy multi-objective routing problem with applications to process planning in manufacturing systems. *International Journal of Production Research , 29* (6), 1161-1170.

90. Slomp, J., Bokhorst, J., & Molleman, E. (2005). Cross-training in a cellular manufacturing environment. *Computers & Industrial Engineering , 48* (3), 609-624.

91. Slomp, J., Chowdary, B. V., & Suresh, N. C. (2005). Design of virtual manufacturing cells: a mathematical programming approach. *Robots and Computer-Integrated Manufacturing , 21* (3), 273-288.

92. Sofianopoulou, S. (2006). Manufacturing cells efficiency evaluation using data envelopment analysis. *Journal of Manufacturing Technology Management , 17* (2), 224-238.

93. Tam, K. Y. (1990). An operation sequence based similarity coefficient for part families formations. *Journal of Manufacturing Systems , 35* (3-4), 55-68.

94. Tariq, A., Hussain, I., & Ghafoor, A. (2009). A hybrid genetic algorithm for machine-part grouping. *Computers and Industrial Engineering , 56* (1), 347-356.

95. Tatikonda, M. V., & Wemmerlöv, U. (1992). Adoption and implementation of group technology classification and coding systems. *International Journal of Production Research , 30* (9), 2087-2110.

96. Tsai, C.-C., & Lee, C.-y. (2006). Optimization of manufacturing cell formation with a multi-functional mathematical programming model. *International Journal of Advanced Manufacturing Technology , 30* (3-4), 309-318.

97. Uddin, M. K., & Shanker, K. (2002). Grouping of parts and machines in presence of alternative process routes by genetic algorithm. *International Journal of Production Economics , 76* (3), 219-228.

98. Vakharia, A. J., & Wemmerlöv, U. (1995). A comparitive investigation of hierarchical clustering techniques and dissimilarity measures applied to the cell formation problem. *Journal of Operations Management , 13* (2), 117-138.

99. Venugopal, V., & Narendran, T. T. (1992). A genetic algorithm approach to the machine-component grouping problem with multiple objectives. *Computers and Industrial Engineering , 22* (4), 469-480.

100. Vohra, T., Chen, D., Chang, J., & Chen, H. (1990). A network approach to cell formation in cellular manufacturing. *International Journal of Production Research , 28* (11), 2075-2084.

101. Waghodekar, P. H., & Sahu, S. (1984). Machine-component cell formation in group technology. *International Journal of Production Research , 28* (6), 937-948.

102. Wemmerlöv, U., & Johnson, D. J. (1997). Cellular manufacturing at 46 user plants: implementation experiences and performance improvements. *International Journal of Production Research , 35* (1), 29-49.

103. Wu, N., & Salvendy, G. (1993). Modified network approach for the design of cellular manufacturing systems. *International Journal of Production Research , 31* (6), 1409-1421.

104. Yin, Y., & Yasuda, K. (2005). Similarity coefficient methods applied to the cell formation problem: a comparative investigation. *Computers and Industrial Engineering , 48* (3), 471-489.

105. Yin, Y., & Yasuda, K. (2006). Similarity coefficient methods applied to the cell formation problem: a taxonomy and review. *International Journal of Production Economics , 101* (2), 329-352.

106. Zhao, C., & Wu, Z. (2000). A genetic algorithm for cell formation with multiple routes and multiple objectives. *International Journal of Production Research , 38* (2), 385-396.

107. Zhao, F., Dong, J., Li, S., & Sun, J. (2008). An improved ant colony optimization algorithm with embedded genetic algorithm for the traveling salesman problem. *Intelligent control and automation, 2008 7th world congress on Intelligent Control and Automation* (pp. 7902-7906). Chongqing, China: Intelligent control and automation.

VITA AUCTORIS

NAME:	Mohammed Salem Taboun
PLACE OF BIRTH:	Windsor, Ontario
YEAR OF BIRTH:	1976
EDUCATION:	Kennedy Collegiate Institute Windsor, Ontario 2000-2004
	B.A.Sc., Industrial and Manufacturing Systems Engineering University of Windsor Windsor Ontario
	Currently a candidate for a Masters of Applied Science in Industrial and Manufacturing Systems Engineering with hopes of graduating in the fall of 2010.