An Automated Workforce Clustering Method for Business Process Reengineering in Research and Development Organizations

Madjid Tavana, School of Business, La Salle University, Philadelphia, PA, USA

Alex F. Sisti, Air Force Research Laboratory, Rome, NY, USA

Dawn A. Trevisani, Air Force Research Laboratory, Rome, NY, USA

ABSTRACT

Business process reengineering (BPR) involves the radical redesign of functional organizations into cross-functional teams to achieve dramatic improvements in productivity. Redesigning complex and dynamic processes in research and development (R&D) organizations with multi-layer projects is a difficult task. Previous researchers have proposed many intuitive approaches for BPR utilizing intuition and subjective judgment from "experts" at various stages of their implementation. However, the complexities inherent in large R&D organizations have restricted the effectiveness of their use in practice. The authors present an automated and structured analytic algorithm that eliminates the need for subjective human judgment. The proposed method facilitates the reorganization of R&D processes into cross-functional work units and projects. The efficacy of the algorithm is confirmed with a small problem and the applicability of the proposed method is demonstrated at the Air Force Research Laboratory.

Keywords: *Air Force Research Laboratory, Business Process Reengineering, Clustering Algorithm, Cross-Functional Teams, Organizational Design*

INTRODUCTION

Innovation plays a prominent role in the current turbulent economy. For innovating organizations, workforce creativity is a crucial determinant of competitiveness and productivity (Lovelace et al., 2001). Team building is a strategy for combining the creativity of employees and promoting innovation (Hoegl & Gemuenden, 2001). Innovation requires teamwork and teams need to have high levels of collaboration to synergistically combine their

DOI: 10.4018/jitpm.2012100101

skills and successfully cope with the complex and dynamic nature of innovative projects (Stewart & Barrick, 2000; Hoegl & Praveen Parboteeah, 2007; Okhuysen & Eisenhardt, 2002; Seijts & Gandz, 2009; Thompson, 2003).

The ability to form effective teams to take on challenges in this turbulent environment is a source of competitive advantage for most research and development (R&D) organizations. Several approaches have been proposed for team formation in the literature. For example, the team management wheel proposed by Margerison and McCann (1991) or the team role model proposed by Belbin (1991) can be used to form balanced and complementary workgroups. Other methods include multi-functional work-force team formation (Fitzpatrick & Askin, 2005; Tseng et al., 2004), project management team formation (Wi et al., 2009a, 2009b), and task force team formation (Tavana et al., 2007). The main objective of these methods is to organize a group of individuals into a useful set of teams such that the similarity of the individual members within a team is maximized while the similarity of the individual members between different teams is minimized (Wi et al., 2009a). More specifically, the staff in R&D oriented organizations is generally structured in work units and projects, which have an enormous influence on the performance of the entire organization (Wi et al., 2009a).

Previous researchers have proposed many intuitive approaches for team formation utilizing intuition and subjective judgment from "experts" at various stages of their implementation. However, the complexities inherent in large R&D organizations have restricted the effectiveness of their use in practice. In order to achieve maximum benefits, teams have to be formed carefully and placing individuals randomly in a group and assigning them a task is not sufficient (Soller, 2001). We propose an alternative method for business process reengineering (BPR) and multi-functional team formation in R&D organizations based on group technology (GT) problems in manufacturing. GT is a methodology for organizing work into independent groups each responsible for the production of a given family of products (Burbridge, 1979). GT simply states that similar things should be done similarly. These problems are also known as the machine-part cell formation (CF) problems where parts and machines in a manufacturing process are assigned to independent cells so that the machine utilization within a cell is maximized and the between-cell movement of parts is minimized (Ahi et al., 2009; Bajestani et al., 2009; De Lit et al., 2000; James et al., 2007).

We extend the machine-part CF method to model the team formation problem in R&D organizations. Our model is simple, easily implemented, and it is based on well-known and widely used GT concepts. A typical R&D organization can be explained in a similar manner by replacing the word *machine* with *employee* and *part* with *job*. A business process requires skills from employees in many different departments. By analyzing a business system from a process point of view, it becomes evident that it is logical to form business process cells. These business process cells are essentially process teams described by Hammer and Champy (1993, p. 66) as work units that naturally fall together to complete a process. Hammer and Champy (1993) also suggested replacing functional departments with process teams to take advantage of GT benefits.

This paper is organized into eight sections. In the next section, we present a brief review of the clustering literature. Following this brief review, we discuss the criteria for evaluation of clustering solutions. In the following section, we present the business process team formation approach proposed in this study. We then present the details of our clustering algorithm and discuss the complexity of our algorithm. Next, we present a case example at the Air Force Research Lab and we summarize with our conclusions and future research directions.

LITERATURE REVIEW

Clustering problems arise in numerous domains ranging from cellular manufacturing and data mining to weather forecasting and epidemiology. In different contexts, clustering takes different names such as partitioning, typology, or numerical taxonomy. Clustering methodologies originate in equally diverse disciplines including operations research, biology, neural networks, mathematical programming, and fuzzy sets among others (Halkidi et al., 2001). Clustering is the process of organizing a set of objects (operating units) into a useful set of mutually exclusive clusters such that the similarity of the objects *within* a cluster is maximized while the similarity of the objects *between* different clusters is minimized (e.g., Jain et al., 1999; Okazaki, 2006; Rai et al., 2006; Samoilenko & Osei-Bryson, 2010; Wallace et al., 2004). Generally, clustering methods are grouped into hierarchical, learning network, and distance-based clustering.

Hierarchical clustering groups the objects by creating a cluster tree called dendrogram. Clusters are then formed by either the agglomerative approach or the divisive approach (Johnson, 1967; Kaufman & Rousseeuw, 1990). Agglomerative methods assume that each object is its own cluster and then these clusters are combined to form larger clusters with each step of the process. Eventually, these clusters are combined to form a single cluster. Divisive methods assume a single cluster encompassing all the objects within the sample and then proceeds to divide this cluster into smaller dissimilar clusters.

Learning network clustering is a neural network based method where high dimensional data is mapped into a discrete one or two-dimensional space. Learning network clustering performs clustering through a competitive learning mechanism (Bu et al., 2003; Choi & Yoo, 2001; Du, 2010; Harb & Chen, 2005; Kohenen, 1990, 2001; Mahdavi et al., 2009).

Distance-based clustering is a partitioning method which creates an initial cluster and then uses an iterative relocation technique to maximize total similarity or minimize total dissimilarity by moving objects from one cluster to another. *K*-means (McQueen, 1967), fuzzy *c*-means (Yang, 1993; Wu & Yang, 2002) and possibilistic c-means (Krishnapuram & Keller 1993) are various forms of distance-based clustering.

Several researchers have studied the team formation problem in the literature. Süer (1996) used integer and mixed integer programming models to assign workers to labor-intensive process teams. Tseng et al. (2004) proposed a novel approach based on fuzzy sets theory and grey decision theory to solve the multi-functional team formation problem by matching customers' requirements with engineers' characteristics. Min and Shin (1993) and Fitzpatrick and Askin (2005) proposed multi-objective models for assigning multi-skilled workers to process teams. Ebeling and Lee (1994) and Suresh and Slomp (2001) developed cross-training models to link team formation with the process team configuration problem. Askin and Huang (2001) proposed an integer programming model to convert a functionally organized production facility into a process team organized facility.

More recently, Dereli and Baykasoglu (2007) proposed a fuzzy mathematical-programming model and a solution algorithm for team formation based on simulated annealing. Wi et al. (2009a) proposed a genetic algorithm model using social network measures to choose team members in an R&D organization. Strnad and Guid (2010) showed that the problem of optimal team formation is beyond manual implementation when the pool of available personnel grows into several tens and proposed a new fuzzy-genetic analytical model for this problem. Agustín-Blas et al. (2011) presented a new model for team formation based on GT in an R&D organization. They also presented a parallel hybrid grouping genetic algorithm to solve the team formation problem. While these methods have made significant contributions to the team formation literature, most of them are either intuitive or very difficult to implement without technical help. In the next section, we present the details of our model and show that

our model is (1) simple to understand and use; (2) automated and does not require human expert intervention; (3) easily implementable with minimum computing power, and (4) based on well-known and widely used GT concepts.

Criteria for Evaluation of Clustering Solutions

Several authors have discussed the criteria for good clustering in data modeling (Akoka & Comyn-Wattiau 1996; Feldman & Miller 1986; Francalanci & Pernici 1994; Huffman & Zoeller 1989). Moody and Flitman (1999) proposed a set of nine principles and associated measurable criteria for clustering. Our algorithm is designed to meet the following seven criteria mainly drawn from the data modeling and GT literature:

1. **Semantically meaningful:** People familiar with the task domain should find the clusters logical and coherent;
2. **Completeness:** Decomposition should cover all of the employees and jobs and no employees or jobs should be left out;
3. **Non-redundancy:** Each employee and job should be in one, and only one, cluster;
4. **Fully connected:** All the employees in a cluster should be connected to each other, via job paths that are *within* the cluster;
5. **Maximal cohesion within clusters:** To the extent possible, all employees within a cluster should be closely related to each other;
6. **Minimal coupling between clusters:** To the extent possible, employees in different clusters should not be closely related to each other; and
7. **High degree of modularity:** Provided all of the other criteria are satisfied, a solution with a greater number of clusters is preferred to a solution with smaller number of clusters.

Attaining any one of our criteria 1, 2, 3, 4, and 7 does not detract from attaining other criteria. However, simultaneous attainment of maximal cohesion (Criterion 5) and minimal coupling (Criterion 6) is impossible. Since every employee directly or indirectly interacts with one or more employees, invariably, when decomposition increases within-cluster cohesion, it also increases inter-cluster coupling. This is because when each cluster contains a large number of employees and jobs, many employees are only indirectly connected to one another. Consequently, each cluster is not very cohesive. At the same time, since the total number of employees and jobs outside a cluster is relatively small, very few employees are connected directly to employees outside their clusters. Thus, inter-cluster coupling is also small. In a properly clustered team formation problem, as the average cluster size decreases, since there are not as many indirect interactions between employee pairs within each cluster, within-cluster cohesion increases. At the same time, inter-cluster coupling increases since now there are many more employees outside each cluster that may be directly related to the employees in that cluster.

Thus, we need a metric that properly trades off the attainment of maximal cohesion against the attainment of minimal coupling. GT authors have studied this problem of trade-off between cohesion and coupling for quite some time. Hence, we use one of the metrics from the GT literature to determine the goodness of fit of our algorithm. In the next section, we present this metric and its properties along with other relevant concepts from GT.

BUSINESS PROCESS TEAM FORMATION APPROACH

In GT, a manufacturing system is organized into temporary work-cells to exploit the advantages of a mass production system. Each work-cell consists of a number of dissimilar machines clustered together to produce a set of parts (called a "part family") with similar processing requirements (Burbidge, 1963; Chandrasekharan & Rajagopalan, 1986; Vakharia & Wemmerlov, 1995). Miltenburg and Zhang (1991) explain

that a good CF solution is such that: *within a cell, each machine processes many parts, and few parts require processing on machines outside the cell*. If we recognize that employees are similar to machines and jobs are similar to parts, the two properties of a good CF solution are precisely the objectives of maximizing cohesion and minimizing coupling in team formation clustering.

In the earliest CF research, Burbidge (1963) develops a binary machine-part incidence matrix [a_{ij}], where an entry of "1" indicates that machine i is used to process part j, while "0" indicates that machine i is not used to process part j. When an initial machine-part incidence matrix is constructed, using the similarity in the machines required for processing various parts and the similarity in the parts processed on the same machines, Burbidge's (1963) method rearranges the rows and columns of the initial incidence matrix to identify clusters of highly compatible parts and machines. The machine-part incidence matrix has become a standard method of representing a CF problem. In our algorithm, we also represent a team formation problem by a binary matrix where employees are represented by rows and jobs by columns. A "1" in a team formation matrix shows that an employee processes the corresponding job, and a "0" shows that an employee does not process the corresponding job.

In GT, a "distance" metric is often defined to rearrange the rows of the matrix to bring together parts that are processed by similar machines. The distance metric measures the lack of similarity between a pair of rows. Our algorithm uses the distance metric in Equation (1) to rearrange the employees of a team formation matrix. The distance between two employees, j and k, is given by:

$$d_{jk} = 4n - \sum_{i=1}^{n}(a_{ij} + a_{ik})^2 + 5\sum_{i=1}^{n}(a_{ij} - a_{ik})^2 \quad (1)$$

Where n is the number of columns (jobs) in the team formation matrix and a_{ij} is the 1 (or 0) value of the entry representing the processing (or the lack of processing) of the j-th job by the i-th employee.

Initially used by Joglekar et al. (1994), this distance metric has several desirable properties for assessing the relative "closeness" of employees. First, since in a binary matrix, a_{ij} and a_{ik} each can be either 1 or 0, all the distances calculated by this formula are in [0, $8n$]. If each one of a pair of employees processed a job in the matrix (i.e., when all $a_{ij} = 1$), the calculated distance would be smallest ($d_{jk} = 0$). When a pair of employees is such that whenever one employee processes a job, the other does not, and vice versa (i.e., when the n entries in Row j are never identical with the n entries in Row k), the distance is the largest ($d_{jk} = 8n$). If a pair of employees processes many matching jobs, the calculated distance is smaller than the distance for a pair of employees that processes only a few matching jobs. Thus, the smaller the calculated distance between two employees, the closer they are to one another in terms of their processing and non-processing of the jobs in a team formation problem.

In short, our distance metric is designed to indicate that employees that are processing the largest number of same jobs are the closest. While the distance metric helps us in the rest of the clustering process, it sometimes fails to keep the *dependent* (an employee depending on an *adopting* employee) and *singular* (an employee processing a job with only one other employee) employees in their appropriate clusters. Yet, to comply with the criterion of semantic meaningfulness, dependent employees must be clustered with their respective adopting employees. Similarly, singular employees must be clustered with their respective employees. Hence, our algorithm first removes those two types of employees (and associated jobs) before applying the above distance metric to juxtapose closely related employees and cluster them together. The *dependent* and *singular* employees (and associated jobs) are then added back to the matrix in their appropriate employee clusters.

In CF, often machines are also rearranged using a similar distance metric. Once the rows

and columns are rearranged, machine-part cells are identified with the dual objectives of maximizing within-cell cohesion and minimizing inter-cell coupling. As in team formation clustering, in GT, typically, when a clustering arrangement is modified to increase within-clusters cohesion, the modification also results in an increase in inter-cluster coupling. Hence, it is advantageous to seek a goodness of fit metric that provides a balanced trade-off between these two objectives.

GT researchers (Chandrasekharan & Rajagopalan, 1986; Miltenburg & Zhang, 1991; Mosier et al., 1997; Seifoddini, 1989) have proposed several different metrics to judge the goodness of fit of a CF solution. Among those metrics, Joglekar et al. (2001) found that Chandrasekharan and Rajagopalan's (1986) "grouping efficiency measure" and Miltenburg and Zhang's (1991) "primary performance measure" balanced the trade-off between the two objectives of CF, cohesion and coupling, reasonably well. However, Chandrasekharan and Rajagopalan's (1986) metric involved a weighting factor between [0,1]. Although the recommended default value for the weighting factor is 0.5, Chandrasekharan and Rajagopalan (1986) suggested that an analyst might want to change that value depending on the problem addressed. In contrast, Miltenburg and Zhang's (1991) metric does not change from problem to problem. Furthermore, using some simple examples, Seifoddini (1989) and Ng (1996) have pointed out that, with the default value of 0.5 for the weighting factor, Chandrasekharan and Rajagopalan's (1986) metric under-emphasised inter-cluster coupling. In their evaluation of the two metrics, Joglekar et al. (2001) found that for some problems, Miltenburg and Zhang's (1991) metric also under-emphasized inter-cluster coupling. However, in some other problems solved by maximizing one of these metrics at a time, while the solutions were equally cohesive, the solutions maximizing the Miltenburg and Zhang (1991) metric displayed a smaller number of inter-cluster couplings than the solutions yielded by maximizing the Chandrasekharan and Rajagopalan (1986) metric. Hence, we believe that Miltenburg and Zhang's (1991) "primary performance measure" represents the best available metric for the tradeoff between the objectives of maximal cohesion and minimal coupling. As such, our algorithm uses this metric to assess the goodness of fit of a team formation clustering solution.

Adapted for the purposes of team formation clustering, this metric is shown in Equation (2) and will be referred to as "the goodness of fit" metric, G.

$$G = G_1 - G_2 \,,\ -1 \le G \le 1 \,, \tag{2}$$

where a binary team formation matrix, A, is partitioned into k clusters $\{D_r \mid r = 1, 2, \ldots, k\}$; Cluster D_r, consisting of employees E_r and jobs R_r, is the r-th cluster; and G_1 and G_2 are:

$$G_1 = \left(\sum_r \sum_{\substack{i \in E_r \\ j \in R_r}} a_{ij} \right) \Bigg/ \left(\sum_r \left| E_r \right| \left| R_r \right| \right),$$

$$0 \le G_1 \le 1 \,, \tag{3}$$

$$G_2 = 1 - \left(\sum_r \sum_{\substack{i \in E_r \\ j \in R_r}} a_{ij} \right) \Bigg/ \left(\sum_{i,j} a_{ij} \right),$$

$$0 \le G_2 \le 1 \,. \tag{4}$$

where $a_{ij} = 1$, if employee i processes job j, and 0 otherwise; $|E_r|$ denotes the number of employees, and $|R_r|$ denotes the number of jobs in D_r.

G's cohesion clause, G_1, is the sum of a_{ij} in each cluster divided by the sum of the number of elements in each cluster. It measures a cluster's density, i.e., how many of the employees process each of the jobs within their clusters. G_1's value is normalized by its denominator - the sum of the number of elements in each cluster ($0 \le G_1 \le 1$). A high value of G_1 indicates high cohesion, which is desirable.

G's coupling clause, G_2, is 1 minus the sum of a_{ij} within each cluster divided by the total number of 1s in the full matrix. G_2 measures inter-cluster jobs, normalized in terms of the total number of 1s in the full matrix ($0 \leq G_2 \leq 1$). A high value of G_2 indicates high coupling, which is undesirable. To seek a balance between the objectives of maximizing cohesion and minimizing coupling, in the algorithm presented in this paper, we seek to maximize G. In short, we have borrowed the following concepts from GT:

- A binary matrix representation of the employees and jobs;
- A distance metric to assess the similarities and dissimilarities between pairs of employees;
- Rearrangement of employee rows to juxtapose the most similar employee pairs; and
- The goodness of fit metric to assess alternative clustering arrangements.

The implementation of our algorithm first requires a binary matrix representation of a team formation problem. It is an easy, mechanical, and automable task to convert any team formation problem into a representative binary matrix, since it requires no information that is not already contained in a team formation problem. For example, Table 1 ($\boldsymbol{M_0}$) presents a binary matrix representation of a team formation problem with 11 employees as the rows and 14 jobs as the columns. Wherever an employee processes a job, we have put down a 1 in the corresponding cell, otherwise a 0. In addition, our algorithm requires a Table 2 (T_0) listing all the *dependent* employees in a team formation problem along with their corresponding adopting employees and respective jobs. In this problem, we assume employee J is dependent

Table 1. M_0 for introductory example

Employee	Job													
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
A	0	0	0	1	0	1	1	0	0	1	0	0	0	0
B	0	0	1	0	0	0	0	0	1	0	1	0	0	0
C	0	0	1	0	0	0	0	0	0	0	1	0	0	0
D	1	1	0	0	1	0	0	0	0	0	0	0	0	0
E	0	1	0	0	0	0	0	1	0	0	0	0	1	0
F	0	0	0	1	0	0	1	0	0	1	0	0	0	0
G	0	0	0	0	0	0	1	0	0	1	0	0	0	0
H	0	0	1	0	0	0	0	0	1	0	1	0	0	0
I	0	0	0	0	0	1	0	0	0	0	0	0	0	0
J	0	0	0	0	0	0	0	0	0	0	0	1	0	1
K	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table 2. T_0 for introductory example

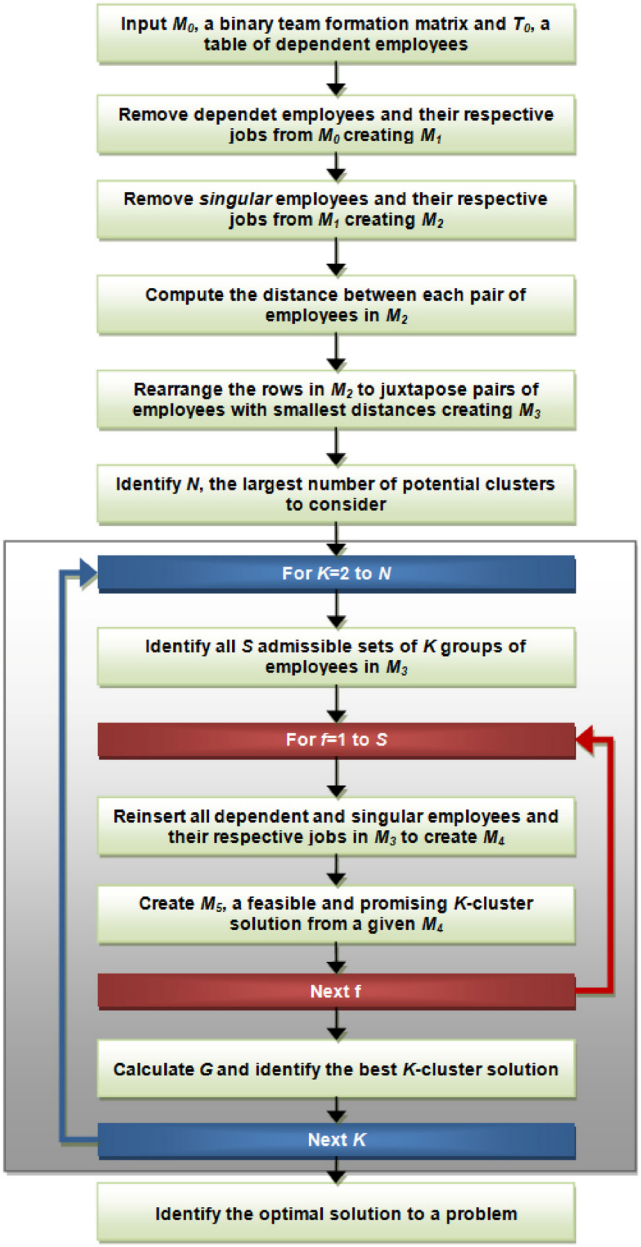
Dependent employee	Adopting employee
J	E
K	B

on employee *E* and employee *K* is dependent on employee *B*. Employees *J* and *K* are called dependent employees and employees *E* and *B* are called adopting employees.

CLUSTERING ALGORITHM

Our algorithm consists of eight major steps presented in Figure 1 and described in detail:

Figure 1. Overview of major steps in our algorithm



- Input M_0, a binary team formation matrix and T_0, a table of dependent employees.

The algorithm begins with the input of a p x q binary matrix, M_0, with employees as rows and jobs as columns. In M_0, a '1' in a cell indicates that the corresponding employee processes the corresponding job. When the corresponding employee does not process the corresponding job, the cell entry is 0. The information in M_0 allows the algorithm to automatically identify several types of employees and jobs. For example, if an employee row has a '1' in only one column, he/she is a *singular* employee; if a job column has 1s in only two rows, it is a binary job; and so on. Thus, the connectivity dimension of the employees and jobs is fully captured by the binary matrix representation of the team formation problem. On the other hand, M_0 does not help the algorithm to identify dependent employees and their respective adopting employees. Hence, that information must be input separately, through a table, T_0.

- Remove dependent employees and their respective jobs from M_0 creating M_1.

For the semantic meaningfulness of team formation decomposition, it is essential that a dependent employee is always grouped with his/her adopting employee. Unfortunately, our distance metric would not necessarily find dependent employees to be closest to their adopting employees. Hence, using the input table identifying the dependent employees, the algorithm's first step is to remove from M_0 all dependent employees as well as the job connecting the dependent employee to the adopting employees. In this step, it is important to ensure that the respective adopting employee is now connected to all of the other jobs that were originally connected to its dependent employee. This procedure produces a new matrix, M_1.

- Remove singular employees and their respective jobs from M_1, creating M_2.

Our distance metric also cannot ensure that a *singular* employee is always grouped with the only employee with whom he or she shares a job. If it were not grouped with that employee, the "fully connected" principle would be violated. Hence, the next step of the algorithm is to identify and remove each singular employee along with his/her associated job from M_1. Sometimes, when a singular employee is removed from M_1, a previously non-singular employee becomes a singular employee in the reduced matrix. This happens when a singular employee's connected employee is related to only one other employee. Hence, the process of identification and removal of singular employees is repeated until there are no singular employees in the modified matrix. Simultaneously, in the order of their removal, a list of all the singular employees and their associated jobs is created, so that at the appropriate time, they can be reinserted in the matrix, in the reverse order of their removal. The resulting matrix is labeled M_2. Clearly, if one or more dependent or singular employees are removed, M_2's dimension, m x n, will be smaller than M_0's dimension.

- Compute the distance between each pair of employees in M_2.

Now, using Equation (1), we compute the distance between each possible pair of rows (employees) in M_2. As explained earlier, the shorter the distance between a pair, the more appropriate it is to put the two employees in the same cluster, and the larger the distance, the more appropriate it is to put the two employees in different clusters.

- Rearrange the rows in M_2 to juxtapose pairs of employees with smallest distances, creating M_3.

Next, our algorithm constructs a matrix, M_3, where we leave the job columns in the same order as in M_2, but rearrange the rows so that pairs of employees with least distances are closest to each other. The algorithm compares all pair-wise row distances, and copies the pair

of rows with the least distance next to each other in M_3 *(arbitrarily one above the other)*. Among the *UnusedRows,* the algorithm finds the one with the smallest distance from one of the *current edge rows.* The identified row is added to M_3 next to that edge row (above the top edge or below the bottom edge). This process is repeated until there are no *UnusedRows* in M_2. Thus, at this point, M_3 represents the rows in M_2 in an order such that the pairs of employee rows with the smallest distances are closest together.

- Identify N, the largest number of potential clusters to consider.

The next step of the algorithm is to find the range for *K*, the number of clusters to consider. As suggested earlier, every cluster must contain at least two employees. Hence, the maximum number of clusters, *N*, is the integer value of the number of employees divided by two. We also assume that, when a team formation matrix is to be clustered, an organization is looking for at least two clusters. Therefore, our algorithm assumes that the desired number of clusters, *K,* is in [2,*N*].

- Identify the "best" K-cluster solution for every possible value of K.

This is the most involved part of the algorithm with several sub-steps detailed below. First, for each value of *K* (where *K*=2 to *N*), the algorithm identifies all admissible sets (say *S*) of *K* groups of employees. Next, for each one of the *S* sets for a given value of *K*, the algorithm creates *S* matrices that are labeled M_4. Then, each one of the M_4s is used to create a feasible and promising *K*-cluster solution. Once all *S* feasible and promising *K*-cluster solutions are created, we calculate the goodness of fit, *G*, for each one of them. Among the *S* solutions, the solution with the largest value of *G* is identified as the best *K*-cluster solution.

- Identify all S admissible sets of K groups of employees in M_3.

Since a matrix's own boundaries (i.e., the first and the last rows) also serve as the boundaries for the first and the last sub-matrices, respectively, to construct a partition resulting in *K* groups of employees (rows), we need *K*-1 other horizontal dividers between rows. It is desirable to draw the dividers between the *K*-1 pairs of adjacent rows that have the greatest distances. However, since a cluster must have at least two employees, divider locations immediately after the first row, or immediately before the last row, cannot be considered selectable. Initially, all other possible divider locations are considered as *selectable*. The pair of adjacent rows with the largest distance is found and the first divider is placed between those two rows. Since an employee group must contain at least two employees, employee pairs involving the rows immediately adjacent (on either side) to a selected divider are no longer selectable. Among the remaining pairs of employees, the pair of adjacent rows with the largest distance is found and the next divider is placed between those two rows. This process is repeated *K*-2 times to find *all but the last* divider. Up until this point, ties in largest distances are broken arbitrarily by choosing the first of the tied divider locations.

For the $(K\text{-}1)^{th}$ divider, often, there are several selectable divider locations with the same largest distance between adjacent rows. Hence, the last divider is handled differently to ensure that arbitrary tie-breaking does not cause a better solution to be missed. Suppose that there are *S* candidates for the $(K\text{-}1)^{th}$ divider. Together, the first *K*-2 dividers and *one of* the *S* candidates produce an admissible set of *K* groups of employees in M_3. Thus, there are *S* sets of *K* groups of employees to consider. Once all *S* sets of *K* groups of employees are identified, for each one of these *S* sets, the algorithm reinserts all dependent and singular employees and creates a feasible and promising *K*-cluster solution. When, for a given value of *K*, it is *impossible* to select *any* admissible $(K\text{-}1)^{st}$ divider, that value of *K* is eliminated from further consideration.

- Reinsert all dependent and singular employees and their respective jobs in M_3 to create M_4

First, for each admissible set, $f \in S$, of K groups of employees, the algorithm constructs a new matrix M_4 by reinserting, in the reverse order of their removal, each dependent and singular employee immediately above its adopting or respective employee. If necessary, a divider location is adjusted to ensure that the reinserted employee and its relevant employee are in the same group. All the jobs are reinserted as the last columns of the matrix and column 1s of any altered job are adjusted back to their original employee rows. Thus, at the end of this procedure, for each one of the L sets, the corresponding M_4 is a p x q matrix consisting of all the employees and jobs in M_0 and for a given value of K, the total number of M_4s is S.

- Create M_5, a feasible and promising K-cluster solution from a given M_4.

Now, the columns in a given M_4 need to be rearranged so that each group of employees is clustered with suitable jobs. Our aim is to maximize the goodness of fit of the resulting K-cluster solution. Given the mathematical formula (Equation 2) for G, to maximize G, one should cluster the largest possible number of relevant jobs with the smallest group of employees.

Hence, for each M_4, the algorithm creates a blank matrix, M_5, of the same dimensions (p x q) as M_4. It copies into M_5 all the employee (row) names in the same order as in M_4. The chosen K-1 dividers are also copied in M_5. Along with the matrix boundaries, the dividers help identify the K groups of employees in M_5. Then, an iterative process of column rearrangement and cluster identification begins.

In each iteration, we first identify the smallest group of employees (rows) that has not yet been put into a cluster. In case of a tie in the employee group size, the first group is selected. The algorithm considers each job that has not already been copied to M_5. If a job has at least as many 1s in the rows of the selected group of employees as it has in the rows of any of the other un-clustered groups of employees, that job column is copied to Matrix M_5 in the next available column spot. When consideration of all jobs is done, the copied jobs and the selected group of employees are declared a cluster, and the algorithm moves on to the identification of the next cluster. When all K clusters are constructed, M_5 represents the feasible and promising K-cluster solution resulting from a given M_4.

- Calculate G and identify the best K-cluster solution.

Once all S feasible and promising K-cluster solutions have been created, we calculate the goodness of fit, G, for each one of them. The solution with the largest value of G is identified as the best K-cluster solution.

- Identify the optimal solution to a problem.

Finally, all the best K–cluster solutions for the entire range of Ks are compared based on their G values, and the solution with the highest G is chosen as the "optimal" clustering solution. Although we use the word "optimal" to refer to the best of the best solutions for various values of K, it should be clear that we are not claiming that our solution is globally optimal. Our algorithm is a heuristic that employs a greedy approach. Thus, we use the word "optimal" simply to avoid the use of the awkward phrase, "best of the best K-cluster solution."

- A special feature of the algorithm.

While our algorithm is designed to consider all possible values of the number of clusters, we have also incorporated a special feature in it. A manager may ask for just the best solution for a given value of K. This is a useful feature for a manager who has a given number of work units and projects in mind.

Consider the introductory example presented earlier with 11 employees (A through

Table 3. 2-cluster solution for the introductory example (G=.36)

Employee	Job													
	1	2	5	8	12	13	14	3	4	7	9	10	11	6
I	0	0	0	0	0	0	0	0	0	0	0	0	0	1
A	0	0	0	0	0	0	0	0	1	1	0	1	0	1
F	0	0	0	0	0	0	0	0	1	1	0	1	0	0
G	0	0	0	0	0	0	0	0	0	1	0	1	0	0
C	0	0	0	0	0	0	0	1	0	0	0	0	1	0
B	0	0	0	0	0	0	0	1	0	0	1	0	1	0
H	0	0	0	0	0	0	0	1	0	0	1	0	1	0
J	0	0	0	0	1	0	1	0	0	0	0	0	0	0
K	0	0	0	0	1	0	1	0	0	0	0	0	0	0
D	1	1	1	0	0	0	0	0	0	0	0	0	0	0
E	0	1	0	1	0	1	0	0	0	0	0	0	0	0

K) and 14 jobs (1 through 14). Tables 3, 4 and 5 present the 2-cluster, 3-cluster and 4-cluster solutions using the algorithm presented in this study. The 4-cluster solution with *G*=.72 is the optional solution for this problem. According to this solution, employees *I*, *A*, *F* and *G* should be grouped into one work unit processing jobs 4,7,10, and 6. Employees *C*, *B* and *H* should be grouped into a second work unit processing jobs 3, 9, and 11. Employees *J* and *K* should be grouped into the third work unit working on jobs 12 and 14. Finally, employees *D* and *E* should form the fourth work unit and process jobs 1, 2, 5, 8, and 13.

ALGORITHM COMPLEXITY

While our algorithm is complex, it is polynomial in time. In terms of the number of solutions evaluated for a *p x q* matrix containing no dependent or singular employees, our algorithm's Big O efficiency is $O(p^2)$. Note that, *K*, the number of clusters in a solution can range from 2 to *p*/2. In the worst-case scenario (if all distances are tied), for each *K*, there are $S = (p - K)$ admissible sets of *K* employee groups. Although each one of these sets must go through the steps of reinserting dependent and singular employees followed by creating a feasible and promising solution by clustering each job with its appropriate employee group, these steps are carried out only once for each admissible set. Therefore, in terms of the number of solutions evaluated, the worst-case efficiency is $\sum_{K=2}^{p/2}(p-K)$. It can be shown that:

$$\sum_{K=2}^{p/2}(p-K) = \left(\frac{p}{2}-1\right)\left(\frac{3p}{4}-1\right) \quad (5)$$

Assuming that *p* is large, this is roughly equal to $(3/8)p^2$; hence, the order of magnitude is $O(p^2)$. Of course, the actual number of solutions evaluated is often considerably smaller. In the introductory example, with $p = 11$, the Big O analysis suggests that, in the worst case, we have to evaluate 121 possible solutions, or more precisely (using Equation 5) 33 possible solutions. In its actual execution for the introductory example, the algorithm evaluated only 4 possible solutions.

The complexity of our algorithm is kept manageable by a series of strategies including: definition of a distance metric to assess the closeness of employees, arbitrary tie-breaking

Table 4. 3-cluster solution for the introductory example (G=.44)

Employee	Job													
	12	**14**	**1**	**2**	**5**	**8**	**13**	**3**	**4**	**7**	**9**	**10**	**11**	**6**
I	0	0	0	0	0	0	0	0	0	0	0	0	0	1
A	0	0	0	0	0	0	0	0	1	1	0	1	0	1
F	0	0	0	0	0	0	0	0	1	1	0	1	0	0
G	0	0	0	0	0	0	0	0	0	1	0	1	0	0
C	0	0	0	0	0	0	0	1	0	0	0	0	1	0
B	0	0	0	0	0	0	0	1	0	0	1	0	1	0
H	0	0	0	0	0	0	0	1	0	0	1	0	1	0
J	1	1	0	0	0	0	0	0	0	0	0	0	0	0
K	1	1	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	1	1	1	0	0	0	0	0	0	0	0	0
E	0	0	0	1	0	1	1	0	0	0	0	0	0	0

in various steps of the algorithm, and using the properties of the *G* metric in clustering jobs with employee groups. Several of these strategies are greedy and global optimality of our solution is not guaranteed. However, our algorithm is practical even for large enterprise data models with hundreds of employees and jobs.

On the other hand, a brute force search for the optimal solution would be exponential in time. Remember that in a *p* x *q* matrix, there can be *K* (= 2 to *p*/2) clusters. For a given *K*, there are ${}_p\mathbf{C}_K$ employee groups and ${}_q\mathbf{C}_K$ job groups. Multiplying these independent possibilities, we get (${}_p\mathbf{C}_K\ {}_q\mathbf{C}_K$) possible clustering arrangements. Thus, the total number of clustering arrangements to consider is $\sum_{K=2}^{p/2}\left({}_pC_{Kq}C_k\right)$.

Table 5. 4-cluster solution for the introductory example (G=.72)

Employee	Job													
	12	**14**	**1**	**2**	**5**	**8**	**13**	**3**	**9**	**11**	**4**	**7**	**10**	**6**
I	0	0	0	0	0	0	0	0	0	0	0	0	0	1
A	0	0	0	0	0	0	0	0	0	0	1	1	1	1
F	0	0	0	0	0	0	0	0	0	0	1	1	1	0
G	0	0	0	0	0	0	0	0	0	0	0	1	1	0
C	0	0	0	0	0	0	0	1	0	1	0	0	0	0
B	0	0	0	0	0	0	0	1	1	1	0	0	0	0
H	0	0	0	0	0	0	0	1	1	1	0	0	0	0
J	1	1	0	0	0	0	0	0	0	0	0	0	0	0
K	1	1	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	1	1	1	0	0	0	0	0	0	0	0	0
E	0	0	0	1	0	1	1	0	0	0	0	0	0	0

We cannot present a simple formula for this total number. However, for the introductory example involving $p = 11$ and $q = 14$, according to this formula, a brute force approach will require an evaluation of 1,320,319 possible solutions! When p and q are larger than those in our introductory example, a brute force approach would require the evaluation of exponentially greater number of possible solutions. In comparison, our algorithm's worst-case scenario of evaluating 33 solutions for the introductory example seems to be considerably more desirable! The fact that our algorithm actually evaluated only 4 possible solutions for the introductory example shows the true efficiency of our algorithm.

Some readers may suggest that evaluating a couple of million solutions (required by a brute force approach to a small problem) would be feasible and would allow a comparison of our greedy solution with the globally optimal solution. However, by now, the reader also knows that significant computation is needed in creating and evaluating each one of millions of solutions required for the brute force approach. Hence, identifying the globally optimal solution for even a small problem is not practical.

THE AIR FORCE RESEARCH LAB CASE EXAMPLE[1]

In this example, the Air Force Research Lab facility in Rome, New York is planning to reorganize a functional organization with 38 employees and 27 jobs into a set of cross functional organization. We used the approach proposed in this study to divide this hierarchical organization into a set of modular cross-functional work units and projects using a computer system developed in JAVA for this project. Initially, we constructed the team formation matrix given in Figure 2 with 38 employees (rows) and 27 jobs (columns).

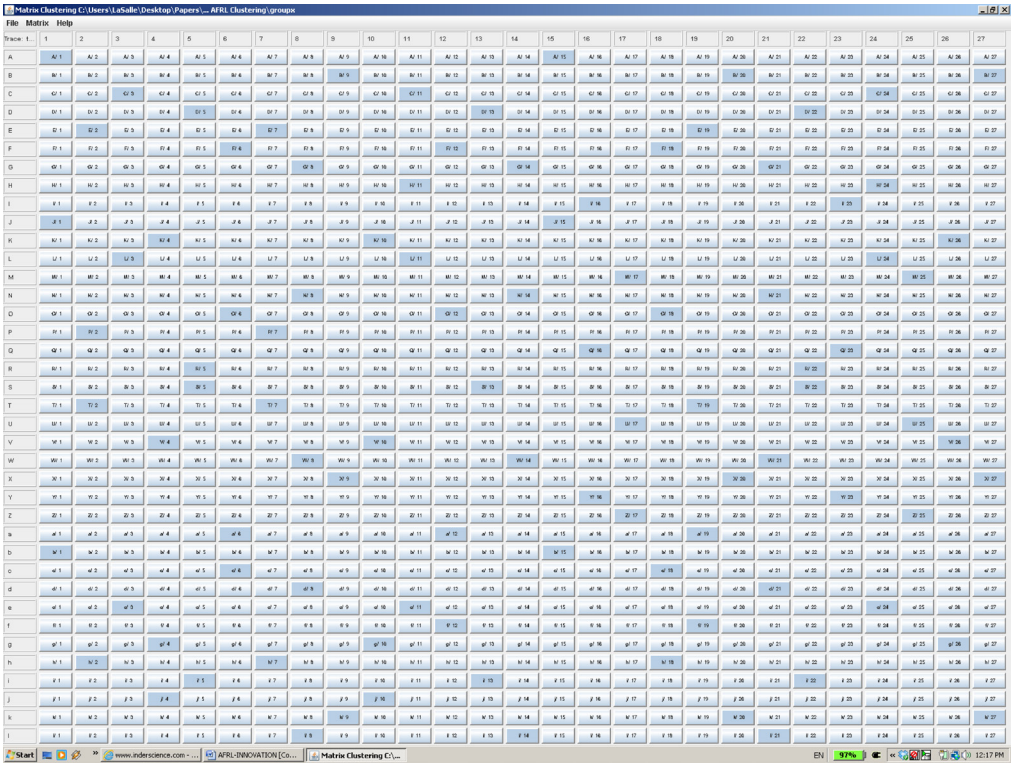
Figure 2 shows is a screenshot of the initial team formation matrix where a darkly-shaded cell in the matrix shows that an employee processes the corresponding job, and a lightly-shaded cell shows that an employee does not process the corresponding job. The management had indicated that because of contractual obligations, employees *I* and *Q* had to work on job 23 and employees *R* and *D* had to process job 13. These constraints were introduced in the model and the computer program as the *dependent* and *adopting* employees. We ran the algorithm and the computer program provided us a 2-cluster solution (G=.17), 3-cluster solution (G=.20), 4-cluster solution (G=.32), 5-cluster solution (G=.39), 6-cluster solution (G=.46), 7-cluster solution (G=.50), 8-cluster solution (G=.55), 9-cluster solution (G=.58), 10-cluster solution (G=.78) and 11-cluster solution (G=.94). The 11-cluster solution presented in Figure 3 was selected as the best K-cluster solution. According to this solution, the 27 jobs and 38 employees were organized into 11 projects and 11 work units presented in Table 6.

One employee from each work unit was designated as the team leader and 11 individuals were appointed to manage the projects. Next, similar projects were grouped together to form three programs. Program Centrifuge included 4 projects (9 jobs) and 4 work units (10 employees), Program Photovoltaic included 4 projects (9 jobs) and 4 work units (15 employees), and Program Nebula included 3 projects (9 jobs) and 3 work units (13 employees).

CONCLUSION AND FUTURE RESEARCH

Viewing the business from a business process perspective rather than a structural or departmental point of view underscores value-added activities and highlights where there are the most opportunities for improvement. Kim (1994) argues that one of the most important aspects of BPR is the decomposition of the business by cross-functional processes. The application of techniques from production management opens up a wide range of proven tools for the implementation of BPR. Specifically, GT can be used for team formation in cross-functional organizations. The parallels between a production system and a human system are many.

Figure 2. Initial work unit-project matrix for the Air Force Research Lab case



Through the development of business process teams, an organization can be studied from a business process perspective. The process teams are formed to perform jobs that once travelled between many departments before completion but now are accomplished without ever leaving the process team. The advantages of creating process teams include: streamlined process flow, employee flexibility, and work-in-process reduction.

- Streamlined process flow: process flows are more streamlined with process teams when compared to functional departments because the interdepartmental flow is eliminated or reduced significantly.
- Employee flexibility: process teams have a more diverse set of skills and can handle a greater number of jobs.
- Work-in-process reduction: the majority of the work is performed by a single process team and the business processes do not have to wait to be completed by different departments.

In this study, we proposed an algorithm for restructuring functional organizations into cross-functional units. In designing the algorithm, we have capitalized on certain concepts and metrics from research in GT. We have discussed how our algorithm fulfills a comprehensive set of criteria for a good decomposition of functional organizations. Our algorithm produces a cohesive set of work units and projects while keeping inter-cluster coupling small.

While our algorithm produces very good solutions, it cannot guarantee their global

Figure 3. Clustered work unit-project matrix for the Air Force Research Lab case

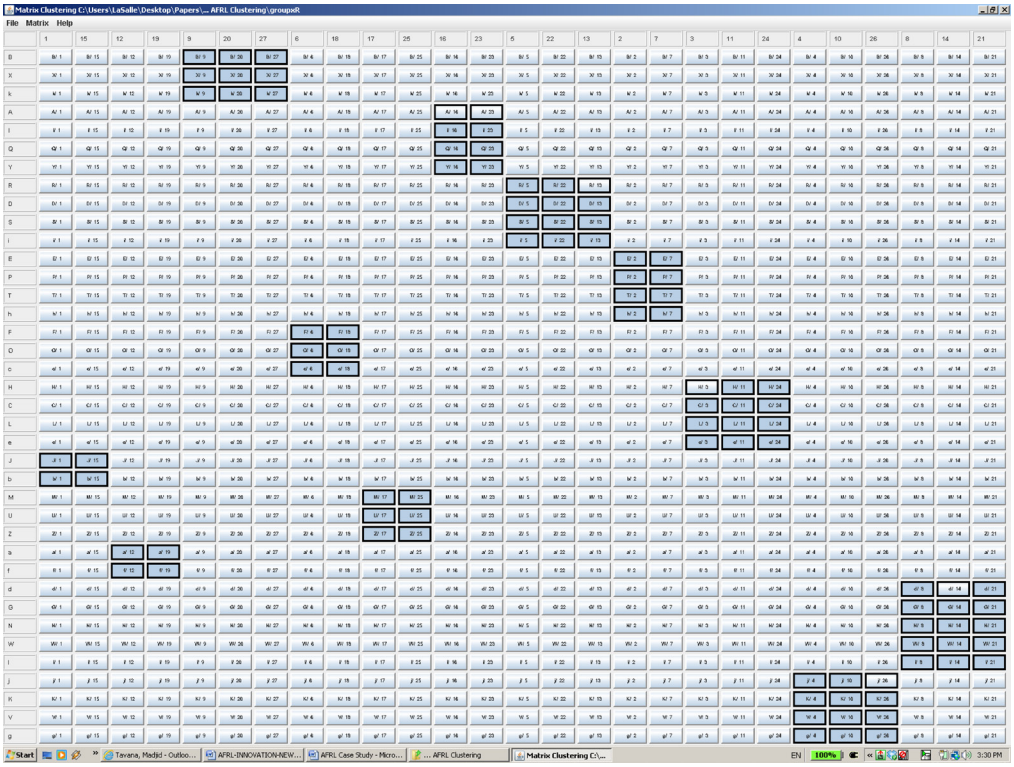


Table 6. Best K-cluster solution with 11 work units and 11 projects

Programs	Project (Jobs)	Work Unit (Employees)
Centrifuge	1-15	J-b
	12-19	a-f
	9-20-27	B-X-K
	6-18	F-O-c
Photovoltaic	17-25	M-U-Z
	16-23	A-I-Q-Y
	5-22-13	R-D-S-i
	2-7	E-P-T-h
Nebula	3-11-24	H-C-L-e
	4-10-26	j-K-V-g
	8-14-21	d-G-N-W-I

optimality since our goodness of fit metric is not necessarily perfect. It is the best of several available metrics that attempt to balance the two objectives of maximizing cohesion while minimizing coupling. Future research may produce a better metric that may lead to a better algorithm.

ACKNOWLEDGMENT

This research was supported by the U.S. Air Force Research Laboratory grant number FA 9550-09-C-0114. The authors would like to thank the anonymous reviewers and the editor for their insightful comments and suggestions. The views expressed in this paper are those of the authors and do not reflect the official policy or position of the United States Air Force and the Department of Defense.

REFERENCES

Agustín-Blas, L. E., Salcedo-Sanz, S., Ortiz-García, E. G., Portilla-Figueras, A., Pérez-Bellido, A. M., & Jiménez-Fernández, S. (2011). Team formation based on group technology: A hybrid grouping genetic algorithm approach. *Computers & Operations Research*, *38*(2), 484–495. doi:10.1016/j.cor.2010.07.006

Ahi, A., Aryanezhad, M. B., Ashtiani, B., & Makui, A. (2009). A novel approach to determine cell formation, intracellular machine layout and cell layout in the CMS problem based on TOPSIS method. *Computers & Operations Research*, *36*(5), 1478–1496. doi:10.1016/j.cor.2008.02.012

Akoka, J., & Comyn-Wattiau, I. (1996). Entity-relationship and object-oriented model automatic clustering. *Data & Knowledge Engineering*, *20*(1), 87–117. doi:10.1016/S0169-023X(96)00007-9

Askin, R. G., & Huang, Y. (2001). Forming effective worker teams for cellular manufacturing. *International Journal of Production Research*, *39*(11), 2431–2451. doi:10.1080/00207540110040466

Bajestani, M. A., Rabbani, M., Rahimi-Vahed, A. R., & Khoshkhou, G. B. (2009). A multi-objective scatter search for a dynamic cell formation problem. *Computers & Operations Research*, *36*(3), 777–794. doi:10.1016/j.cor.2007.10.026

Belbin, M. (1991). *Management teams: Why they succeed or fail*. Oxford, UK: Butterworth-Heinemann.

Bu, N., Fukuda, O., & Tsuji, T. (2003). EMG-based motion discrimination using a novel recurrent neural network. *Journal of Intelligent Information Systems*, *21*(2), 113–126. doi:10.1023/A:1024706431807

Burbidge, J. L. (1963). Production flow analysis. *Production Engineering*, *42*(12), 742–752. doi:10.1049/tpe.1963.0114

Burbidge, J. L. (1979). *Group technology in the engineering industry*. New York, NY: John Wiley & Sons.

Chandrasekharan, M. P., & Rajagopalan, R. (1986). An ideal seed non-hierarchical clustering algorithm for cellular manufacturing. *International Journal of Production Research*, *24*(2), 451–464. doi:10.1080/00207548608919741

Choi, Y. S., & Yoo, S. I. (2001). Text database discovery on the web: Neural net based approach. *Journal of Intelligent Information Systems*, *16*(1), 5–20. doi:10.1023/A:1008708725988

De Lit, P., Falkenauer, E., & Delchambre, A. (2000). Grouping genetic algorithms: An efficient method to solve the cell formation problem. *Mathematics and Computers in Simulation*, *51*(3-4), 257–271. doi:10.1016/S0378-4754(99)00122-6

Dereli, T., Baykasoglu, A., & Das, G. S. (2007). Fuzzy quality-team formation for value added auditing: A case study. *Journal of Engineering and Technology Management*, *24*(4), 366–394. doi:10.1016/j.jengtecman.2007.09.005

Du, L.-L. (2010). Clustering: A neural network approach. *Neural Networks*, *23*(1), 89–107. doi:10.1016/j.neunet.2009.08.007

Ebeling, A. C., & Lee, C.-Y. (1994). Cross training effectiveness and profitability. *International Journal of Production Research*, *32*, 2843–2859. doi:10.1080/00207549408957104

Feldman, P., & Miller, D. (1986). Entity model clustering: Structuring a data model by abstraction. *The Computer Journal*, *29*(4), 348–360. doi:10.1093/comjnl/29.4.348

Fitzpatrick, E. L., & Askin, R. G. (2005). Forming effective worker teams with multi-functional skill requirements. *Computers & Industrial Engineering*, *48*(3), 593–608. doi:10.1016/j.cie.2004.12.014

Francalanci, C., & Pernici, B. (1994). Abstraction levels for entity-relationship schemas. In P. Loucopoulus (Ed.), *Proceedings of the 13th International Conference on the Entity-Relationship Approach* (LNCS 881, pp. 456-473).

Halkidi, M., Vazirgiannis, M., & Batistakis, Y. (2001). Clustering algorithms and validity measures. In *Proceedings of the Scientific and Statistical Database Management Conference* (pp. 3-22).

Hammer, M., & Champy, J. (1993). *Reengineering the corporation: A manifesto for business revolution*. New York, NY: HarperCollins. doi:10.1016/S0007-6813(05)80064-3

Harb, H., & Chen, L. (2005). Voice-based gender identification in multimedia applications. *Journal of Intelligent Information Systems*, *24*(2-3), 179–198. doi:10.1007/s10844-005-0322-8

Hoegl, M., & Gemuenden, H. G. (2001). Teamwork quality and the success of innovative projects: A theoretical concept and empirical evidence. *Organization Science*, *12*(4), 435–449. doi:10.1287/orsc.12.4.435.10635

Hoegl, M., & Praveen Parboteeah, K. (2007). Creativity in innovative projects: How teamwork matters. *Journal of Engineering and Technology Management*, *24*(1-2), 148–166. doi:10.1016/j.jengtecman.2007.01.008

Huffman, S. B., & Zoeller, R. V. (1989). A rule-based system tool for automated ER model clustering. In *Proceedings of the 8th International Conference on the Entity-Relationship Approach* (pp. 221-236).

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys*, *31*(3), 264–323. doi:10.1145/331499.331504

James, T. L., Brown, E. C., & Keeling, K. B. (2007). A hybrid grouping genetic algorithm for the cell formation problem. *Computers & Operations Research*, *34*, 2059–2079. doi:10.1016/j.cor.2005.08.010

Joglekar, P., Chung, Q., & Tavana, M. (2001). Note on a comparative evaluation of nine well-known algorithms for solving the cell formulation problem in group technology. *Journal of Applied Mathematics and Decision Sciences*, *5*(4), 253–268.

Joglekar, P., Tavana, M., & Banerjee, S. (1994). A clustering algorithm for identifying information subsystems. *Journal of International Information Management*, *3*, 129–141.

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, *32*(3), 241–254. doi:10.1007/BF02289588

Kaufman, L., & Rousseeuw, P. J. (1990). *Finding groups in data: An introduction to cluster analysis*. New York, NY: John Wiley & Sons. doi:10.1002/9780470316801

Kim, B.-O. (1994). Business process reengineering: Building a cross-functional information architecture. *Journal of Systems Management*, *45*(12), 30–36.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, *78*(9), 1464–1480. doi:10.1109/5.58325

Kohonen, T. (2001). *Self-organizing maps* (3rd ed.). Berlin, Germany: Springer-Verlag. doi:10.1007/978-3-642-56927-2

Krishnapuram, R., & Keller, J. M. (1993). A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems*, *1*, 98–110. doi:10.1109/91.227387

Lovelace, K., Shapiro, D. L., & Weingart, L. R. (2001). Maximizing cross-functional new product teams' innovativeness and constraint adherence: A conflict communications perspective. *Academy of Management Journal*, *44*, 779–793. doi:10.2307/3069415

Mahdavi, I., Paydar, M. M., Solimanpur, M., & Heidarzade, A. (2009). Genetic algorithm approach for solving a cell formation problem in cellular manufacturing. *Expert Systems with Applications*, *36*(3), 6598–6604. doi:10.1016/j.eswa.2008.07.054

Margerison, C., & McCann, D. (1991). *Team management: Practical approaches*. London, UK: Mercury.

McQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, CA (Vol. 1, pp. 281-297).

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. doi:10.1016/0272-6963(91)90035-V

Min, H., & Shin, D. (1993). Simultaneous formation of machine and human cells in group technology: A multiple objective approach. *International Journal of Production Research*, *31*, 2307–2318. doi:10.1080/00207549308956859

Moody, D., & Flitman, A. (1999). A methodology for clustering entity relationship models – A human information processing approach. In J. Akoka, M. Bouzeghoub, I. Comyn-Wattiau, E. Metais (Eds.), *Proceedings of 18th International Conference on the Entity-Relationship Approach* (LNCS 1728, pp. 114-130).

Mosier, C., Yelle, J., & Walker, G. (1997). Survey of similarity coefficient based methods as applied to the group technology configuration problem. *Omega*, *25*(1), 65–79. doi:10.1016/S0305-0483(96)00045-X

Ng, S. M. (1996). On the characterization and measure of machine cells in group technology. *Operations Research*, *44*(5), 735–744. doi:10.1287/opre.44.5.735

Okazaki, S. (2006). What do we know about mobile internet adopters? A cluster analysis. *Information & Management*, *43*, 127–141. doi:10.1016/j.im.2005.05.001

Okhuysen, G. A., & Eisenhardt, K. M. (2002). Integrating knowledge in groups: How formal interventions enable flexibility. *Organization Science*, *13*(4), 370–386. doi:10.1287/orsc.13.4.370.2947

Rai, A., Tang, X., Brown, P., & Keil, M. (2006). Assimilation patterns in the use of electronic procurement innovations: A cluster analysis. *Information & Management*, *43*(3), 336–349. doi:10.1016/j.im.2005.08.005

Samoilenko, S., & Osei-Bryson, K. M. (2010). Determining sources of relative inefficiency in heterogeneous samples: Methodology using cluster analysis, DEA and neural networks. *European Journal of Operational Research*, *206*, 479–487. doi:10.1016/j.ejor.2010.02.017

Seifoddini, H. (1989). A note on the similarity coefficient method and the problem of improper machine assignment in group technology applications. *International Journal of Production Research*, *27*(2), 1161–1165. doi:10.1080/00207548908942614

Seijts, G., & Gandz, J. (2009). One-teaming: Gaining a competitive edge through rapid team formation and deployment. *Organizational Dynamics*, *38*(4), 261–269. doi:10.1016/j.orgdyn.2009.07.005

Soller, A. (2001). Supporting social interaction in an intelligent collaborative learning system. *International Journal of Artificial Intelligence in Education*, *12*, 40–62.

Stewart, G. L., & Barrick, M. R. (2000). Team structure and performance: Assessing the mediating role of intrateam process and the moderating role of task type. *Academy of Management Journal*, *43*(2), 135–148. doi:10.2307/1556372

Strnad, D., & Guid, N. (2010). A fuzzy-genetic decision support system for project team formation. *Applied Soft Computing*, *10*(4), 1178–1187. doi:10.1016/j.asoc.2009.08.032

Süer, G. A. (1996). Optimal operator assignment and cell loading in labor-intensive manufacturing cells. *Computers & Industrial Engineering*, *26*, 155–158. doi:10.1016/0360-8352(96)00101-5

Suresh, N. C., & Slomp, J. (2001). A multi-objective procedure for labour assignments and grouping in capacitated cell formation problems. *International Journal of Production Research*, *39*(18), 4103–4131. doi:10.1080/00207540110072966

Tavana, M., Smither, J. W., & Anderson, R. V. (2007). D-side: A facility and workforce planning group multi-criteria decision support system for Johnson Space Center. *Computers & Operations Research*, *34*(6), 1646–1673. doi:10.1016/j.cor.2005.06.020

Thompson, L. (2003). Improving the creativity of organizational work groups. *The Academy of Management Executive*, *17*(1), 96–109. doi:10.5465/AME.2003.9474814

Tseng, T. L., Huang, C. C., Chu, H. W., & Gung, R. R. (2004). Novel approach to multi-functional project team formation. *International Journal of Project Management*, *22*(2), 147–159. doi:10.1016/S0263-7863(03)00058-9

Vakharia, A. J., & Wemmerlov, U. (1995). A comparative investigation of hierarchical clustering techniques and dissimilarity measures applied to cell formation problem. *Journal of Operations Management*, *13*(2), 117–138. doi:10.1016/0272-6963(95)00017-M

Wallace, L., Keil, M., & Rai, A. (2004). Understanding software project risk: A cluster analysis. *Information & Management*, *42*, 115–125. doi:10.1016/j.im.2003.12.007

Wi, H., Mun, J., Oh, S., & Jung, M. (2009b). Modeling and analysis of project team formation factors in a project-oriented virtual organization (ProVO). *Expert Systems with Applications*, *36*(3), 5775–5783. doi:10.1016/j.eswa.2008.06.116

Wi, H., Oh, S., Mun, J., & Jung, M. (2009a). A team formation model based on knowledge and collaboration. *Expert Systems with Applications*, *36*(5), 9121–9134. doi:10.1016/j.eswa.2008.12.031

Wu, K. L., & Yang, M. S. (2002). Alternative c-means clustering algorithms. *Pattern Recognition*, *35*, 2267–2278. doi:10.1016/S0031-3203(01)00197-2

Yang, M. S. (1993). A survey of fuzzy clustering. *Mathematical and Computer Modelling*, *18*, 1–16. doi:10.1016/0895-7177(93)90202-A

ENDNOTES

1 This is a fictitious example constructed by the authors at the Air Force Research Laboratory to demonstrate the efficacy of the algorithm and confirm the applicability of the method proposed in this study.

Madjid Tavana is a Professor of Business Systems and Analytics and the Lindback Distinguished Chair of Information Systems and Decision Sciences at La Salle University where he served as Chairman of the Management Department and Director of the Center for Technology and Management. He has been a Distinguished NASA Research Fellow at Kennedy Space Center, Johnson Space Center, Naval Research Laboratory - Stennis Space Center and Air Force Research Laboratory. He was recently honored with the prestigious Space Act Award by NASA. He holds an MBA, a PMIS and a PhD in Management Information Systems and received his post-doctoral diploma in strategic information systems from the Wharton School of the University of Pennsylvania. He is the Editor-in-Chief for Decision Analytics*, the* International Journal of Strategic Decision Sciences*, the* International Journal of Enterprise Information Systems *and the* International Journal of Applied Decision Sciences*. He has published over one hundred research papers in academic journals such as* Decision Sciences*,* Information Systems, Interfaces*,* Annals of Operations Research*,* Omega, Information and Management*,* Expert Systems with Applications*,* European Journal of Operational Research*,* Journal of the Operational Research Society*,* Computers and Operations Research*,* Knowledge Management Research and Practice*,* Computers and Industrial Engineering*,* Applied Soft Computing*,* Journal of Advanced Manufacturing Technology *and* Advances in Engineering Software*, among others.*

Alex F. Sisti is the chief of the Decision Support Systems Branch at the Air Force Research Laboratory in Rome, New York. He is in charge of planning, initiating and directing research and development of modeling, simulation and visualization techniques for exploiting, analyzing and processing of information in response to United States Air Force requirements and documented deficiencies. He has been actively applying his modeling and simulation research interests in a variety of Air Force programs over the last three decades. He holds a BS in Computer Science and received his MS in Computer Engineering from Syracuse University.

Dawn A. Trevisani is a Computer Scientist/Program Manager within the Decision Support Systems Branch at the Air Force Research Laboratory in Rome, New York. She manages modeling and simulation research, experimentation and analysis programs in response to United States Air Force requirements and documented deficiencies. Her focus has been in the area of Integrated Command and Control (IC2) architectures, decision support and course of action generation and assessment. She holds a BS in Computer Information Science and an MS in Computer Science from State University of New York – Institute of Technology at Utica/Rome.