Top (2012) 20:679–706
DOI 10.1007/s11750-010-0159-3

ORIGINAL PAPER

A GRASP heuristic for the manufacturing cell formation problem

Juan A. Díaz · Dolores Luna · Ricardo Luna

Received: 15 December 2009 / Accepted: 9 September 2010 / Published online: 1 October 2010

Abstract In this work, we address the Manufacturing Cell Formation Problem (*MCFP*). Cellular Manufacturing is a production strategy that has emerged to reduce materials handling and set up times in order to reduce lead times in production systems and to improve customer's service levels while reducing costs. We propose a GRASP heuristic to obtain lower bounds for the optimal solution of the problem. To evaluate the performance of the proposed method, we test the heuristic with different instances from the literature and compare the results obtained with those provided by other heuristic methods from the literature. According to the obtained results, the proposed GRASP procedure provides good quality lower bounds with reasonable computational effort.

Keywords Group technology · Manufacturing cell formation · Metaheuristics

Mathematics Subject Classification (2000) 90C59

1 Introduction

Nowadays, there is a large quantity of production systems that effectively use Cellular Manufacturing. This strategy has been used to significantly reduce materials handling, set up costs and work in progress, especially in batch and job shop production systems. This strategy may also help to improve customer's service levels.

J.A. Díaz · D. Luna (✉) · R. Luna
Dpt. Ingeniería Industrial y Mecánica, Universidad de las Américas, Puebla, México
e-mail: dolorese.luna@udlap.mx

J.A. Díaz
e-mail: juana.diaz@udlap.mx

R. Luna
e-mail: ricardo.luna@udlap.mx

One of the basic problems in Cellular Manufacturing is the identification of parts that share similar processes and to group them into families. In the ideal case each part-family is processed by a single cell. However, in practice, often this is not the case, since total reduction of inter-cell movement (parts processed by two or more machine-cells) may imply considerable investment in machines that process parts grouped in different families. In practice, part-families are assigned to machine-cells in order to minimize inter-cell movements or to maximize efficiency or efficacy measures related with both, inter-cell movements and families homogeneity. Therefore, in Manufacturing Cell Formation Problems (*MCFP*) we have to identify disjoint machine-cells and part-families.

The problem has been extensively studied in the literature. We can find an excellent recent survey of solution methods in Gonçalves and Resende (2004) where the authors propose an evolutionary algorithm for *MCFP*. They also present a classification of the different solution methods from the literature (i.e. cluster analysis, graph partitioning, etc.)

Recently Wu et al. (2009) proposed a hybrid heuristic that combines the Boltzmann function from simulated annealing with a mutation operator from genetic algorithms. The proposed heuristic is an iterative approach that starts with two part-families and successively increments the number of part-families until no improvement is obtained. At each iteration, they construct an initial solution with a fixed number of part-families using the Jaccard's similarity measure. They first construct a set of part-families placing the parts with higher similarity coefficients in the same family. Then, each machine is assigned to the part-family where the sum of exceptional elements and voids is the smallest. Ties are broken assigning the machine to the part-family with smaller number of voids. The initial solution is improved combining the Boltzmann function of simulated annealing and a mutation operator from genetic algorithms.

In this paper we consider a GRASP heuristic for finding lower bounds for the MCFP. The construction phase of the GRASP heuristic alternates between the construction of machine-cells and part-families. The construction phase is a randomized greedy algorithm that identifies seed machine-cells (or seed part-families) using Kusiak's (see Kusiak 1987) similarity coefficient. Several similarity (or dissimilarity) coefficients have been extensively used in literature to form initial machine-cells or part-families. These methods compute similarity coefficients between machines or parts and group them together using cluster analysis or hierarchical methods (see for example, McAuley 1972; Carrie 1973; Waghodekar and Sahu 1984; Mosier and Taube 1985a, 1985b; Seifoddini and Wolfe 1986; Seifoddini 1989; Mosier 1986; Gupta and Seifoddini 1990; Khan et al. 2000; Yasuda and Yin 2001). In our approach, after seed machine-cells (seed part-families) are identified, unassigned machines (unassigned parts) are assigned to the seed machine-cell (seed part-family) that maximizes a greedy function. The improvement phase consists of two stages. In the first stage parts (machines) are assigned to machine-cells (part-families) using the procedure used by Gonçalves and Resende (2004). The solution obtained is improved using a local search procedure that explores several neighborhoods. To evaluate the quality of the feasible solutions explored by the heuristic we use the efficacy measure proposed by Kumar and Chandrasekhran (1990). To the best of our knowledge, GRASP has not been used for this version of the MCFP. Adil and Ghosh (2005) propose a GRASP heuristic for a different version of the problem where the conversion of a plant to group technology is only partial.

Different instances of the *MCFP* from the literature are used to test the performance of the proposed GRASP procedure. We compare the obtained results with other heuristic methods. According to the results obtained we observe that the GRASP heuristic provides good quality lower bounds with small computational effort.

This paper is structured as follows. Section 2 gives some notation, defines the problem and presents a mathematical programming formulation for the *MCFP*. Next, in Section 3 the construction and improvement phase of the proposed GRASP heuristic are described in detail. Section 4 describes the computational tests used to evaluate the proposed heuristic and compares its performance with other heuristic methods from the literature. Finally, in Section 5 we draw some conclusions.

2 Problem formulation

In this section we describe the Manufacturing Cell Formation Problem. We introduce some useful notation and formulate the related optimization problem considered for grouping machines into machine-cells and parts into part-families.

In cellular manufacturing, parts with similar processing requirements are grouped together in homogeneous clusters in order to reduce manufacturing costs. Also, the set of machines is partitioned into clusters (machine-cells) and part-families are assigned to this clusters to be processed. Ideally, a part-family is processed in a single machine-cell. However, in practice there are many parts that need to be processed outside of their parent cell. Typically, parts are partitioned into part-families and machines into machine-cells according to similarity measures. Commonly, the different parts of a part-family need to be processed in almost all machines of a machine-cell. Similarly, the machines grouped into a machine-cell are needed to produce almost all parts of a part-family. Typically, an instance of the *MCFP* is represented by a machine-part incidence matrix $A = (a_{ij})$ where

$$a_{ij} = \begin{cases} 1, & \text{if part } i \text{ is processed on machine } j, \\ 0, & \text{otherwise.} \end{cases}$$

We have to partition the machine set into machine-cells and the part set into part-families in order to maximize a given efficiency or efficacy measure. The most extended technique to tackle this problem is block diagonalization of the machine-part incidence matrix. Let $P = \{1, \ldots, m\}$ denote the parts set and $M = \{1, \ldots, n\}$ the machines set. An ideal solution is obtained if a perfect block diagonal matrix is obtained. That is, all $a_{ij} = 1, i \in P, j \in M$ are inside diagonal blocks and all $a_{ij} = 0, i \in P, j \in M$ are outside of the diagonal blocks. In ill-structured machine-part incidence matrices, all elements $a_{ij} = 1, i \in P, j \in M$ outside diagonal blocks are called *exceptional elements*, and these parts have operations on machines outside their parent cell. The machines that process exceptional elements are called *bottleneck machines*. Also, all elements $a_{ij} = 0, i \in P, j \in M$ inside diagonal blocks are usually called *voids*.

A large number of methods to perform the block diagonalization of the machine-part incidence matrix have been proposed in the literature. Different measures have also been proposed to evaluate the block diagonalization of ill-structured machine-part incidence matrices. These measures include grouping measures and performance measures (see for example Sarker 2001; Sarker and Mondal 1999.) However, there is not an unified way to evaluate the goodness of block diagonalization. Sarker and Mondal (1999) conduct a critical review and a comparative study of different quantitative measures of group efficiency of cell formation solutions. In this work we use the efficacy measure proposed by Kumar and Chandrasekhran (1990) since this measure has been extensively used to evaluate solution quality for the *MCFP*. The grouping efficacy μ proposed by Kumar and Chandrasekhran (1990) can be defined as:

$$\mu = \frac{N_1 - N_1^{\text{out}}}{N_1 + N_0^{\text{in}}},$$

where N_1 is the total number of 1's in the machine-part incidence matrix, N_1^{out} is the total number of 1's outside the diagonal blocks, N_0^{in} is the total number of 0's inside the diagonal blocks.

Let $G = \{1, \dots, g_{\max}\}$ be the index set for the number of groups (i.e. machine-cells and part-families), where $g_{\max}$ is an upper bound on the number of groups to be formed. That is, $g_{\max} = \min\{\lceil \frac{|M|}{M_{\min}} \rceil, \lceil \frac{|P|}{P_{\min}} \rceil\}$, where $P_{\min}$ is a lower bound on the number of parts on each part-family and $M_{\min}$ is a lower bound on the number of machines on each machine-cell. We consider the following decision variables:

$$x_{ig} = \begin{cases} 1, & \text{if part } i \in P \text{ is assigned to part-family } g \in G, \\ 0, & \text{otherwise,} \end{cases}$$

$$y_{jg} = \begin{cases} 1, & \text{if machine } j \in M \text{ is assigned to machine-cell } g \in G, \\ 0, & \text{otherwise,} \end{cases}$$

and

$$z_g = \begin{cases} 1, & \text{if group (i.e. machine-cell and part-family) } g \in G \text{ is needed,} \\ 0, & \text{otherwise.} \end{cases}$$

The *MCFP* can be formulated in the following way:

$$\text{(MCFP)} \quad \max \quad \frac{\sum_{i\in P}\sum_{j\in M} a_{ij} - \sum_{g\in G}\sum_{i\in P}\sum_{j\in M} a_{ij}(1-x_{ig})y_{jg}}{\sum_{i\in P}\sum_{j\in M} a_{ij} + \sum_{g\in G}\sum_{i\in P}\sum_{j\in M}(1-a_{ij})x_{ig}y_{jg}} \tag{1}$$

$$\text{s.t.} \quad \sum_{g\in G} x_{ig} = 1 \qquad \forall i \in P, \tag{2}$$

$$\sum_{g\in G} y_{jg} = 1 \qquad \forall j \in M, \tag{3}$$

$$\sum_{i \in P} x_{ig} \geq P_{\min} z_g \qquad \forall g \in G, \tag{4}$$

$$\sum_{j \in M} y_{jg} \geq M_{\min} z_g \qquad \forall g \in G, \tag{5}$$

$$x_{ig} \leq z_g \qquad \forall i \in P, \forall g \in G, \tag{6}$$

$$y_{jg} \leq z_g \qquad \forall j \in M, \forall g \in G, \tag{7}$$

$$x_{ig} \in \{0, 1\} \qquad \forall i \in P, \forall g \in G, \tag{8}$$

$$y_{jg} \in \{0, 1\} \qquad \forall j \in M, \forall g \in G, \tag{9}$$

$$z_g \in \{0, 1\} \qquad \forall g \in G. \tag{10}$$

The nonlinear objective function (1) to be maximized is the efficacy measure proposed by Kumar and Chandrasekhran (1990). This measure penalizes both, exceptional elements and voids. Constraints (2) ensure that each part is allocated to one part-family and, similarly, constraints (3) ensure that each machine is assigned to a machine-cell. Constraints (4) and (5) force, respectively, to allocate at least $P_{\min}$ parts to each part-family and to assign at least $M_{\min}$ machines to each machine-cell. Finally, constraints (6) ensure that a part can only be assigned to a part-family if that part-family exists and (7) ensure that a machine can only be assigned to a machine-cell if that machine-cell exists. In our work $P_{\min} = M_{\min} = 2$, that is, we do not allow singletons. The Manufacturing Cell Formation Problem is a NP-hard optimization problem. To the best of our knowledge, no exact methods have been proposed in the literature to solve practical dimension instances of this problem with a reasonable computational effort. However, the formulation always helps to understand the problem and its complexity in order to develop heuristic methods capable to find good quality solutions within a reasonable time.

3 Heuristic algorithm

In this work we propose a GRASP Heuristic for the *MCFP*. The GRASP methodology was first introduced by Feo and Resende (1995). GRASP is an iterative method consisting of two phases: (1) Construction phase and (2) local search phase. The purpose of the construction phase is to build a solution from scratch by partially randomizing a greedy procedure, while the purpose of the local search phase is to improve the solution obtained in the first phase using a local search procedure that explores one or more neighborhoods.

We now describe both phases of the proposed GRASP heuristic for the *MCFP*.

3.1 Construction phase

In our approach we use similarity coefficients to form clusters of machines or parts. McAuley (1972) and McCormick et al. (1972) started the use of similarity coefficients to identify clusters of machines (i.e. machine-cells) or clusters of parts (i.e.

part-families). Since then, several approaches use similarity (or dissimilarity) coefficients to form initial machine-cells or part-families. These methods compute similarity coefficients between machines or parts and group them together using cluster analysis or hierarchical methods (see for example, Seifoddini and Wolfe 1986; Gupta and Seifoddini 1990; Wu et al. 2009). Mathematical programming methods treat the clustering problem as an optimization problem. For example, Kusiak (1987) suggested the P-median model to maximize the total sum of similarities to obtain part-families. Lee and García-Díaz (1993) use a capacitated circulation network problem to minimize dissimilarity between machines in order to find machine-cells. Srinivasan and Narendran (1991) also use the Kusiak's similarity coefficient and identify machine-cells solving an assignment problem. Recently, Wu et al. (2009) use Jaccard's similarity coefficient to form a fixed number of part-families. They start with two part-families, and successively increment by one the number of part-families while solution quality improves. In our approach Kusiak's similarity coefficient is used to form machine-cells and part-families. In order to get diverse initial solutions the construction phase of our GRASP alternates between forming machine-cells and part-families, identifying seed groups. Since the number of groups (machine-cells or part-families) to be formed is unknown, we use a randomized greedy procedure that allows obtaining different initial configurations.

To construct an initial solution the problem is formulated as a Graph Partitioning Problem. Let $G=(V,E)$, where V is a set of vertices and $E=\{\{i,j\},i,j\in V,i\neq j\}$ is a set of edges. Since the construction phase alternates between forming machine-cells and part-families, each vertex of set V represents a machine, if we want to form machine-cells, or a part, if we want form part-families. If we want to construct machine-cells, for each edge $e\in E$ we associate a weight $w_e=\sum_{h=1}^{m}\gamma(a_{h,j_1},a_{h,j_2})$ where

$$\gamma(a_{h,j_1},a_{h,j_2})=\begin{cases}1, & \text{if } a_{h,j_1}=a_{h,j_2},\\ 0, & \text{otherwise.}\end{cases}$$

Similarly, if we want to determine part-families, we associate the weight $w_e=\sum_{h=1}^{n}\gamma(a_{i_1,h},a_{i_2,h})$ for each $e\in E$, where

$$\gamma(a_{i_1,h},a_{i_2,h})=\begin{cases}1, & \text{if } a_{i_1,h}=a_{i_2,h},\\ 0, & \text{otherwise.}\end{cases}$$

These weights measure similarities between pairs of machines or pairs of parts. The idea of the construction phase is to partition the set of vertices so as to minimize the sum of the weights of those edges whose end points are assigned to different partition subsets.

In what follows, the construction phase is described regardless of the meaning of the elements of the vertex set V. The construction phase consists of two stages. In the first stage, seed groups of vertices are generated from a restricted candidate list, while in the second stage, unassigned vertices are allocated to one of the seed groups generated in the first stage.

During the first stage of the construction phase we consider a list that contains the edges with greater weights. Intuitively, it is desired that the end points of these edges belong to the same subset of the partition. In order to consider a large number of different solutions, the size of the list may variate depending on the value of a parameter $\alpha \in [0, 1]$. A *Restrictive Candidate List* of edges *RCL* is constructed in the following way. Let $w_{\max} = \max_{e \in E}\{w_e\}$ and $w_{\min} = \min_{e \in E}\{w_e\}$. We add to *RCL* those edges whose weight is greater than or equal to a threshold value $u = w_{\max} - \alpha(w_{\max} - w_{\min})$. The value of α is used to control the size of *RCL*. When we choose small values of α the list contains only those edges with the largest weight values. On the other hand, for larger values of α the cardinality of *RCL* will grow.

We now describe how to generate the initial partition subsets. It is an iterative procedure. At each iteration we randomly select one edge $e = \{i, j\} \in RCL$. Let A be the set of vertices already assigned to a subset of the partition and K the number of subsets in the partition. Initially $A = \emptyset$ and $K = 0$. At each iteration we randomly select and edge $e^* = \{i^*, j^*\}$ of *RCL* and eliminate it from *RCL*, that is, $RCL := RCL \setminus \{e^*\}$. If both end points of edge e^* have not been assigned to some subset, then, a new subset is created, $K := K + 1$, and both end points of the edge, i^* and j^* are assigned to the new subset, $C_K := \{i^*, j^*\}$ and the assigned vertices set is updated, $A := A \cup \{i^*, j^*\}$. On the other hand, if only one of the end points has not been assigned to any set, let v be the unassigned vertex, then $v = i^*$ if $i^* \in V \setminus A$ or $v = j^*$ if $j^* \in V \setminus A$. Then we evaluate in which of the K subsets vertex v should be assigned. Let $\varphi_k(v)$ be a greedy function that measures the benefit of assigning vertex v to subset C_k where

$$\varphi_k(v) = \frac{\sum_{e \in \delta(v, C_k)} w_e}{|C_k|}, \quad k \in 1, \ldots, K. \tag{11}$$

We assign vertex v to subset C_r where $r \in \arg\max_{\{k \in 1,\ldots,K\}} \varphi_k(v)$. Finally we update $A := A \cup \{v\}$ (see Algorithm 1). The iterative procedure is repeated until $RCL = \emptyset$.

The second stage of the constructive phase is also an iterative procedure. During this stage, each unassigned vertex is assigned to one of the partition subsets. At each iteration, a new vertex is selected and assigned to one subset. Again, $\varphi_k(v)$ measures the benefit of assigning vertex v to subset C_k. For each vertex $v \in V \setminus A$, a subset $C_k, k = 1, \ldots, K$, is selected and the vertex is assigned to it, and the set A is updated until all vertices are assigned to one subset (see Algorithm 2).

During the constructive phase of the proposed GRASP heuristic, several values of parameter α are used to obtain diverse initial solutions. Intuitively, the larger the value of parameter α, the greater the number of groups to be formed. For this reason, in our implementation, the value of α varies from 0.1 to 0.5. This allows obtaining diverse solutions since the number of groups in the optimal solution is unknown. For each α value we apply the constructive phase to get initial groups of machine-cells and part-families. This is equivalent to use the similarity coefficients between columns or rows of the machine-part incidence matrix, respectively.

3.2 Improvement phase

The graph partitioning formulation used in the construction phase is an indirect formulation to identify initial machine-part clusters based on similarity measures. How-

Algorithm 1 First stage of the constructive phase

```
Let $A \leftarrow \emptyset$ and $K = 0$
$w_{\max} \leftarrow \max\{w_e : e \in E\}$
$w_{\min} \leftarrow \min\{w_e : e \in E\}$
$u \leftarrow w_{\max} - \alpha(w_{\max} - w_{\min})$
$RCL \leftarrow \{e \in E : w_e \geqslant u\}$
**repeat**
    select $e^* = \{i^*, j^*\}$ randomly from $RCL$
    **if** $i^* \in V \setminus A$ and $j^* \in V \setminus A$ **then**
        $K \leftarrow K + 1$
        $C_K \leftarrow \{i^*, j^*\}$
        $A \leftarrow A \cup \{i^*, j^*\}$
    **else**
        **if** $i^* \notin V \setminus A$ or $j^* \notin V \setminus A$ **then**
            $v \leftarrow \begin{cases} i^*, & \text{if } i^* \in V \setminus A \\ j^*, & \text{if } j^* \in V \setminus A \end{cases}$
            **for all** $k \in \{1, \ldots, K\}$ **do**
                $\varphi_k(v) = \frac{\sum_{e \in \delta(v, C_k)} w_e}{|C_k|}$
            **end for**
            $r \in \arg\max_{k \in 1,\ldots,K}\{\varphi_k(v)\}$
            $C_r \leftarrow C_r \cup \{v\}$
            $A \leftarrow A \cup \{v\}$
        **end if**
    **end if**
    $RCL \leftarrow RCL \setminus \{e^*\}$
**until** $RCL = \emptyset$
```

Algorithm 2 Second stage of the constructive phase

```
**while** $V \neq A$ **do**
    select $v \in V \setminus A$
    **for all** $k \in \{1, \ldots, K\}$ **do**
        $\varphi_k(v) = \frac{\sum_{e \in \delta(v, C_k)} w_e}{|C_k|}$
    **end for**
    $r \in \arg\max_{k \in 1,\ldots,K}\{\varphi_k(v)\}$
    $C_r \leftarrow C_r \cup \{v\}$
    $A \leftarrow A \cup \{v\}$
**end while**
```

ever, the use of similarity measures not always leads to an optimal solution with respect to the efficacy measure that depends on the number of exceptional elements and voids. The objective of the improvement phase is to maximize the efficacy measure.

Different approaches from the literature propose methods with two phases, similarly to the one proposed in this work. First, an indirect formulation is used to form initial machine-part clusters, then, local search procedures are used to maximize efficiency or efficacy measures. For example, Srinivasan and Narendran (1991) proposed a nonhierarchical clustering algorithm call GRAFICS that uses similarity coefficients and solves an assignment problem to maximize total similarity. Subtours in the solution represent initial machine-cells. Then an improvement phase is used to maximize the grouping efficiency. The local search consists of assigning alternatively parts to machine-cells and machines to part-families until no improvement is found. Lee and García-Díaz (1993) proposed a capacitated circulation network flow problem for the clustering problem identifying machine-cells, then parts are assigned to machine-cells. Adenso-Díaz et al. (2005) proposed a Tabu search algorithm for a different version of the problem that considers the maximization of an efficiency measure. They first construct an initial solution solving a quadratic programming problem that uses an objective function based on the maximization of weighted similarity coefficients. Then, parts are assigned to machine-cells solving a minimum cost network flow problem. Tabu search is used to improve the initial solution. The algorithm explores four different neighborhoods: exchange of two machines assigned to different machine-cells, reassignment of one machine from one cell to another, union of two cells and splitting of one cell. Recently, Wu et al. (2009) proposed a hybrid heuristic that combines the Boltzmann function from simulated annealing with a mutation operator from genetic algorithms. They construct an initial solution using the Jaccard's similarity coefficient that is then improved using local search, the Boltzmann function of simulated annealing and a mutation operator from genetic algorithms. The local search procedure explores the reassignment of one part from one family to another.

The improvement phase of the proposed GRASP consists of two stages: assignment stage and local search stage. In the assignment stage, to obtain feasible solutions to *MCFP*, parts are assigned to machine-cells or machines are assigned to part-families depending on the initial grouping. That is, when the constructive phase forms part-families, then machines are assigned to those families. On the other hand, when the constructive phase forms machine-cells, then parts are assigned to those cells. Then, a local search procedure is applied to improve the initial assignment. In the local search stage, several neighborhoods are explored to improve the solution obtained in the assignment stage.

We now explain the initial assignment stage of the improvement phase.

3.2.1 Assignment stage

To find an assignment of parts to machine-cells or machines to part-families the procedure used by Gonçalves and Resende (2004) is applied. This procedure is a modification of the one proposed by Srinivasan and Narendran (1991) and Adil et al. (1997). It alternates the assignment of machines to part-families with the assignment of parts to machine-cells. The procedure works as follows.

- Assignment of parts to machine-cells: Let $\sigma : P \longrightarrow \{1, \dots, K\}$, then $\sigma(i) = r$ if part i is assigned to cell C_r. For each $i \in P$, $\sigma(i) = r_i^*$, where,

$$r_i^* \in \underset{r \in \{1,\dots,K\}}{\operatorname{argmax}} \frac{\sum_{i \in P} \sum_{j \in M} a_{ij} - \sum_{j \in M \setminus C_r} a_{ij}}{\sum_{i \in P} \sum_{j \in M} a_{ij} + \sum_{j \in C_r} (1 - a_{ij})}. \tag{12}$$

- Assignment of machines to part-families: Let $\tau : M \longrightarrow \{1, \dots, K\}$, then $\tau(j) = s$ if machine j is assigned to part-family C_s. For each $j \in M$, $\tau(j) = s_j^*$, where,

$$s_j^* \in \operatorname*{argmax}_{s \in \{1,\dots,K\}} \frac{\sum_{i \in P} \sum_{j \in M} a_{ij} - \sum_{i \in P \setminus C_s} a_{ij}}{\sum_{i \in P} \sum_{j \in M} a_{ij} + \sum_{i \in C_s} (1 - a_{ij})}. \tag{13}$$

Singletons can appear during the assignment phase. If it is the case, singletons are assigned to the second best option. Once the assignment is completed we evaluate the objective function value with the following expression:

$$\text{efficacy} = \frac{\sum_{i \in P} \sum_{j \in M} a_{ij} - \sum_{i \in P} \sum_{j \notin C_{\sigma(i)}} a_{ij}}{\sum_{i \in P} \sum_{j \in M} a_{ij} + \sum_{i \in P} \sum_{j \in C_{\sigma(i)}} (1 - a_{ij})}. \tag{14}$$

During the first stage of the improvement phase we use a procedure which alternates between the assignment of parts to machine-cells and the assignment of machines to part-families. If the initial solution obtained in the constructive phase partitions the machine set into machine-cells, the first stage of the improvement phase starts assigning parts to those machine-cells. On the other hand, if the initial solution obtained in the constructive phase partitions the part set into part-families, the first stage of the improvement phase starts assigning machines to those part-families. The first stage of the improvement phase terminates when the solution efficacy of two consecutive partitions of machines into machine-cells, or two consecutive partitions of parts into part-families, does not improve. The procedure is depicted in Algorithm 3.

3.2.2 *Local search stage*

The final solution obtained by the assignment stage is improved using a local search procedure. This solution consist of a set of K machine-cells, $C = \{C_1, \dots, C_K\}$ and K part-families, $F = \{F_1, \dots, F_K\}$. The local search procedure explores six different neighborhoods:

- $N_1(C, F)$: *Reassignment of one machine from one cell to another one with reassignment of parts*. This neighborhood considers all solution that can be reached from the current one with a machine reassignment movement. A machine reassignment movement is an operation that moves machine j from its current cell k to another cell k'. To evaluate the efficacy value of each neighbor solution in $N_1(C, F)$, first, each part is assigned to a machine-cell using (9) and solution efficacy is computed using (11). If K is the number of machine-cells, then in the worst case, when each cell has at least three machines (since no singletons are allowed), $|N_1(C, F)| = n(K - 1)$ and for each candidate in $N_1(C, F)$, mK evaluations are needed to reassign parts and compute the efficacy value.
- $N_2(C, F)$: *Reassignment of one part from one part-family to another one with reassignment of machines*. This neighborhood considers all solution that can be reached from the current one with a part reassignment movement. A part reassignment movement is an operation that moves part i from its current family k to another family k'. To evaluate the efficacy value of each neighbor solution in

Algorithm 3 First stage of the improvement phase

```
Let $K_t^{\text{Initial}}$ be the index set of the subsets of the initial partition in iteration $t$ and
$K_t^{\text{Final}}$ the index set of the subsets of the final partition in iteration $t$.
$t \leftarrow 0$
repeat
  $t \leftarrow t + 1$
  if $K_t^{\text{Initial}}$ represents a partition of the machines into machine-cells then
    for all $i \in P$ do
      $\sigma(i) = r_i^*$, where
        $r_i^* \in \operatorname*{argmax}_{r \in \{1,\dots,K\}} \frac{\sum_{i \in P} \sum_{j \in M} a_{ij} - \sum_{j \in M \setminus C_r} a_{ij}}{\sum_{i \in P} \sum_{j \in M} a_{ij} + \sum_{j \in C_r} (1 - a_{ij})}$
    end for
    Remove singletons
    {Compute $K_t^{\text{Final}}$}
    for all $j \in M$ do
      $\tau(j) = s_j^*$, where
        $s_j^* \in \operatorname*{argmax}_{s \in \{1,\dots,K\}} \frac{\sum_{i \in P} \sum_{j \in M} a_{ij} - \sum_{i \in P \setminus C_s} a_{ij}}{\sum_{i \in P} \sum_{j \in M} a_{ij} + \sum_{i \in C_s} (1 - a_{ij})}$
    end for
    Remove singletons
  else
    for all $j \in M$ do
      $\tau(j) = s_j^*$, where
        $s_j^* \in \operatorname*{argmax}_{s \in \{1,\dots,K\}} \frac{\sum_{i \in P} \sum_{j \in M} a_{ij} - \sum_{i \in P \setminus C_s} a_{ij}}{\sum_{i \in P} \sum_{j \in M} a_{ij} + \sum_{i \in C_s} (1 - a_{ij})}$
    end for
    Remove singletons
    {Compute $K_t^{\text{Final}}$}
    for all $i \in P$ do
      $\sigma(i) = r_i^*$, where
        $r_i^* \in \operatorname*{argmax}_{r \in \{1,\dots,K\}} \frac{\sum_{i \in P} \sum_{j \in M} a_{ij} - \sum_{j \in M \setminus C_r} a_{ij}}{\sum_{i \in P} \sum_{j \in M} a_{ij} + \sum_{j \in C_r} (1 - a_{ij})}$
    end for
    Remove singletons
  end if
  Compute the solution efficacy $K_t^{\text{Final}}$ using the expression:
    $\text{efficacy} = \frac{\sum_{i \in P} \sum_{j \in M} a_{ij} - \sum_{i \in P} \sum_{j \notin C_{\sigma(i)}} a_{ij}}{\sum_{i \in P} \sum_{j \in M} a_{ij} + \sum_{i \in P} \sum_{j \in C_{\sigma(i)}} (1 - a_{ij})}$
until $K_t^{\text{Final}} = K_t^{\text{Initial}}$
```

$N_2(C, F)$, first, each machine is assigned to a part-family using (10) and solution efficacy is computed using (11). If K is the number of part-families, then in the worst case when each family has at least three parts (since no singletons are allowed) $|N_2(C, F)| = m(K-1)$ and for each candidate in $N_2(C, F)$, nK evaluations are needed to reassign machines and compute the efficacy value.

- $N_3(C, F)$: *Cell creation*: This neighborhood considers all solutions that can be reached from the current one with a cell creation movement. A cell creation movement is an operation that creates a new machine-cell of two machines. Let $j_1, j_2 \in M$ be two different machine indices. Then a new cell $C_{K+1} = \{j_1, j_2)$ is created if j_1 and j_2 belongs to cell k and $|C_k| \geq 4$ or if j_1 belongs to cell k_1, j_2 belongs to cell k_2 and $|C_{k_1}| > 2$ and $|C_{k_2}| > 2$, since in any other case the pair (j_1, j_2) will produce an infeasible solution. To evaluate the efficacy value of each neighbor solution in $N_3(C, F)$, first, each part is assigned to a machine-cell using (9) and solution efficacy is computed using (11). In the worst case when all machine-cells have at least four machines $|N_3(C, F)| = n^2$ and for each solution in $N_3(C, F)$, mK evaluations are needed to reassign parts and compute the efficacy value.
- $N_4(C, F)$: *Family creation*: This neighborhood considers all solutions that can be reached from the current one with a family creation movement. A family creation movement is an operation that creates a new part-family of two parts. Let $i_1, i_2 \in P$ be two different part indices. Then a new family $F_{K+1} = \{i_1, i_2)$ is created if i_1 and i_2 belongs to family k and $|F_k| \geq 4$ or if i_1 belongs to family k_1, i_2 belongs to family k_2 and $|F_{k_1}| > 2$ and $|F_{k_2}| > 2$, since in any other case the pair (i_1, i_2) will produce an infeasible solution. To evaluate the efficacy value of each neighbor solution in $N_4(C, F)$, first, each machine is assigned to a part-family using (10) and solution efficacy is computed using (11). In the worst case when all part-families have at least four parts $|N_4(C, F)| = m^2$ and for each solution in $N_4(C, F)$, nK evaluations are needed to reassign machines and compute the efficacy value.
- $N_5(C, F)$: *Reassignment of one machine from one cell to another one*. This neighborhood is similar to $N_1(C, F)$ but without reassignment of parts. That is, the efficacy value of each neighbor solution in $N_5(C, F)$ is computed using (11). If K is the number of machine-cells, then in the worst case when each cell has at least three machines, $|N_5(C, F)| = n(K-1)$.
- $N_6(C, F)$: *Reassignment of one part from one family to another one*. This neighborhood is similar to $N_2(C, F)$ but without reassignment of machines. That is, the efficacy value of each neighbor solution in $N_6(C, F)$ is computed using (11). If K is the number of part-families, then in the worst case when each family has at least three parts, $|N_6(C, F)| = m(K-1)$.

The local search phase of the GRASP algorithm is summarized in Algorithm 4. It is worth noting that neighbor solutions on neighborhoods $N_1(C, F)$ and $N_2(C, F)$ have the same structure in the sense that the number of groups (machine-cells and part-families) does not change. Therefore, we explore these neighborhoods until no improvement is found as an intensification strategy. On the other hand, neighborhoods $N_3(C, F)$ and $N_4(C, F)$ are used as a diversification strategy, since they change the solution structure by adding a new group. Therefore, when an improvement move is found in $N_3(C, F)$ or $N_4(C, F)$ we restart the intensification strategy by exploring, respectively, $N_1(C, F)$ or $N_2(C, F)$. As can be observed neighborhoods $N_5(C, F)$ and $N_6(C, F)$ are explored at the end of the local search procedure, because assignment rules (9) and (10) do not consider the effect of an assignment in the global efficacy value (11). For example, consider the instance shown in Table 1. For the following machine-cells: $C_1 = \{1, 3, 5, 6, 12, 13, 18\}$, $C_2 = \{9, 15, 17\}$, $C_3 = \{2, 11, 16, 19\}$, $C_4 = \{7, 10\}$ and $C_5 = \{4, 8, 14, 20\}$, the assignment of parts to machine-cells using rule (9) is as shown in Table 2b. The solution efficacy of this solution is 49.64, but reassigning only part 20 from cell three to cell one, the efficacy improves to 49.65 (see Table 2a). This means that rules (9) and (10) not always choose the best assignment with respect to the solution efficacy.

Algorithm 4 Local search stage of the improvement phase

```
Stop ← false
repeat
   repeat
      Explore $N_1(C, F)$
   until (no improvement move is found)
   Explore $N_3(C, F)$
   if (no improvement move is found) then
      Stop ← true
   end if
until (Stop)
Stop ← false
repeat
   repeat
      Explore $N_2(C, F)$
   until (no improvement move is found)
   Explore $N_4(C, F)$
   if (no improvement move is found) then
      Stop ← true
   end if
until (Stop)
repeat
   Explore $N_5(C, F)$
until (no improvement move is found)
repeat
   Explore $N_6(C, F)$
until (no improvement move is found)
```

4 Results

In order to evaluate the behavior of the proposed GRASP heuristic, it is tested on 35 instances from the literature. This set of instances has been used in several works of

Table 1 Instance example

	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			1	1	
2			1			1			1	1		1								
3	1			1				1										1	1	1
4					1								1							
5		1							1		1					1			1	
6										1							1			
7			1				1		1	1	1					1	1			
8							1			1										
9							1			1										
10	1				1	1			1			1	1					1		
11	1		1		1							1								
12				1				1						1	1					1
13		1						1									1			
14		1														1				
15	1	1	1		1	1		1				1	1	1			1	1		1
16								1		1				1						1
17				1						1										
18			1						1			1	1		1	1	1			
19									1						1		1			
20												1	1			1				
21	1			1					1						1					
22	1		1	1		1		1	1	1				1	1					1
23	1			1					1						1					

Table 2 Instance solutions

	1	3	5	6	12	13	18	4	8	14	20	2	11	16	19	7	10	9	15	17
1	1	1	1	1	1	1	1	1	1	1					1				1	
2		1		1	1												1	1		
4			1			1														
10	1		1	1	1	1	1											1		
11	1	1	1		1															
15	1	1	1	1	1	1	1		1	1	1	1								1
20					1	1								1						
3	1						1	1	1		1				1					
12								1	1	1	1								1	
16									1	1	1						1			
22	1	1		1				1	1	1	1						1	1	1	
5												1	1	1	1			1		
14												1		1						
6																	1			1
7		1											1	1		1	1	1		1
8																1	1			
9																1	1			
17								1									1			
13									1			1								1
18		1			1	1								1				1	1	1
19																		1	1	1
21	1							1										1	1	
23	1							1										1	1	

(a) Efficacy = 49.65

	1	3	5	6	12	13	18	4	8	14	20	2	11	16	19	7	10	9	15	17
1	1	1	1	1	1	1	1	1	1	1					1				1	
2		1		1	1												1	1		
4			1			1														
10	1		1	1	1	1	1											1		
11	1	1	1		1															
15	1	1	1	1	1	1	1		1	1	1	1								1
3	1						1	1	1		1				1					
12								1	1	1	1								1	
16									1	1	1						1			
22	1	1		1				1	1	1	1						1	1	1	
5												1	1	1	1			1		
14												1		1						
20					1	1								1						
6																	1			1
7		1											1	1		1	1	1		1
8																1	1			
9																1	1			
17								1									1			
13									1			1								1
18		1			1	1								1				1	1	1
19																		1	1	1
21	1							1										1	1	
23	1							1										1	1	

(b) Efficacy = 49.64

the *MCFP*. The size of the problems range from 7 parts and 5 machines to 100 parts and 40 machines. Except for instances 30 and 31, the rest of the instances were taken from Gonçalves and Resende (2004). Instances 30 and 31 were taken from Wu et al. (2009) since they outperform the results in Gonçalves and Resende (2004), however, the data for these two instances in Wu et al. (2009) are different from the original source. We compare the obtained results of the proposed GRASP heuristic with the evolutionary algorithm (HGA) proposed in Gonçalves and Resende (2004) and with the hybrid heuristic algorithm proposed by Wu et al. (2009) (SCF-BMCF), that uses Boltzmann function and a mutation operator of genetic algorithms.

The proposed heuristic was coded in C language and was run on a PC with a Pentium Core 2 Duo, 1.60 GHz, processor. For each instance, the algorithm was run 10 times and we report the best, the worst and the average efficacy. In each run five values of α are used and for each α value, five iterations are performed. We also report the average CPU time for each instance.

In order to tune up the α parameter used in the randomized greedy heuristic of the constructive phase of our GRASP, we performed a series of experiments. In every experiment, ten replicates were run for each instance. In each replicate 25 iterations of the GRASP were executed. In the first five experiments the value of α was fixed to 0.1, 0.2, 0.3, 0.4 and 0.5 respectively. In the sixth experiment, to obtain diverse initial

Table 3 α parameter tuning

	α parameter					
	0.1	0.2	0.3	0.4	0.5	from 0.1 to 0.5
Number of best known solutions found	24	30	30	30	32	35
Maximum difference between best known efficacy and worst efficacy	23.35	14.30	34.74	21.05	32.95	2.11
Maximum difference between best known efficacy and average efficacy	24.23	4.30	11.97	7.13	14.80	0.84
Average difference between best known efficacy and worst efficacy	1.78	0.99	1.46	0.98	1.40	0.23
Average difference between best known efficacy and average efficacy	1.23	0.39	0.57	o.41	0.30	0.10

solutions, the value of the α parameter varies from 0.1 to 0.5 with increments of 0.1. For each α value five iterations of GRASP were performed. To compare and evaluate the solution quality and the robustness obtained in each experiment, we compute the following measures: (1) number of best known solutions found, (2) maximum difference between best known efficacy and worst efficacy obtained, (3) maximum difference between best known efficacy and average efficacy obtained, (4) average difference between best known efficacy and worst efficacy obtained and (5) average difference between best known efficacy and average efficacy obtained. Table 3 shows the results obtained for each experiment.

As can be seen in Table 3 the results of the experiment where the α parameter varies in the range from 0.1 to 0.5 outperforms the other experiments. Small values of the α parameter tend to create a small number of groups (machine-cells and part-families) while large values of the α parameter tend to create more groups. As a consequence, the algorithm is not able to find the best known solutions when a fixed value of α is used, since the number of groups in the optimal solutions is not known beforehand. On the other hand, when the α value varies, a higher degree of diversification is achieved and this fact allows to find better quality solutions. Also, the algorithm is more robust when the α value varies in the range from 0.1 to 0.5. Maximum and average difference measures are significantly smaller than in the other experiments.

The obtained results are depicted in Table 4. The first three columns show the instance number, the source reference and the size ($m \times n$) of each instance. The next column shows the best known solution for each instance. The remaining columns show the results obtained by HGA, SCF-BMCF and the GRASP algorithm proposed in this work. For HGA and GRASP the table shows the worst, the average and the best solutions found in ten replicates. For the SCF-BMCF only the best solution is reported because minimum and average solutions within the ten replicates are not reported in Wu et al. (2009). Also, the columns under the heading "Dev" show the percentage deviation of the best known solution with respect to the best solution found by each method. Finally, the last column of Table 4 shows the average CPU time for the GRASP algorithm. The HGA efficacy values shown for test instances 7 and 14 are different than those reported in Gonçalves and Resende (2004). For these test instances the reported best, average and worst solutions have the same value, but the solutions shown in their appendix have smaller efficacy values than those reported in the results table of their paper. We therefore use the efficacy values of the solutions shown in the paper Appendix.

Table 4 Computational results

	Problems		Best	HGA				SCF-BMCF		GRASP				
			Known	Min	Avg	Max	Dev	Max	Dev	Min	Avg	Max	Dev	Avg CPU (sec)
1	King and Nakornchai (1982)	5X7	73.68	73.68	73.68	73.68	0.00	73.68	0.00	73.68	73.68	73.68	0.00	0.002
2	Waghodekar and Sahu (1984)	5X7	62.50	62.50	62.50	62.50	0.00	62.50	0.00	62.50	62.50	62.50	0.00	0.001
3	Seifoddini (1989)	5X18	79.59	79.59	79.59	79.59	0.00	79.59	0.00	79.59	79.59	79.59	0.00	0.014
4	Kusiak and Cho (1992)	6X8	76.92	76.92	76.92	76.92	0.00	76.92	0.00	76.92	76.92	76.92	0.00	0.003
5	Kusiak and Chow (1987)	7X11	53.13	53.13	53.13	53.13	0.00	53.13	0.00	53.13	53.13	53.13	0.00	0.015
6	Boctor (1991)	7X11	70.37	70.37	70.37	70.37	0.00	70.37	0.00	70.37	70.37	70.37	0.00	0.009
7	Seifoddini and Wolfe (1986)	8X12	68.29	68.29	68.29	68.29	0.00	68.29	0.00	68.29	68.29	68.29	0.00	0.014
8	Chandrasekharan and Rajagopalan (1986)	8X20	85.25	85.25	85.25	85.25	0.00	85.25	0.00	85.25	85.25	85.25	0.00	0.042
9	Chandrasekharan and Rajagopalan (1986)	8X20	58.72	58.72	58.72	58.72	0.00	58.72	0.00	58.72	58.72	58.72	0.00	0.034
10	Mosier and Taube (1985a)	10X10	70.59	70.59	70.59	70.59	0.00	70.59	0.00	70.59	70.59	70.59	0.00	0.015
11	Chan and Milner (1982)	10X15	92.00	92.00	92.00	92.00	0.00	92.00	0.00	92.00	92.00	92.00	0.00	0.031
12	Askin and Subramanian (1987)	14X24	69.86	69.86	69.86	69.86	0.00	69.86	0.00	69.86	69.86	69.86	0.00	0.263
13	Stanfel (1985)	14X24	69.33	69.33	69.33	69.33	0.00	69.33	0.00	69.33	69.33	69.33	0.00	0.225
14	McCormick et al. (1972)	16X24	51.96	51.96	51.96	51.96	0.00	51.96	0.00	51.40	51.90	51.96	0.00	0.406
15	Srinivasan et al. (1990)	16X30	67.83	67.83	67.83	67.83	0.00	67.83	0.00	67.83	67.83	67.83	0.00	0.352
16	King (1980)	16X43	56.52	54.86	54.86	54.86	2.94	54.60	3.40	55.83	55.97	56.52	0.00	1.399
17	Carrie (1973)	18X24	54.46	54.46	54.46	54.46	0.00	54.46	0.00	54.46	54.46	54.46	0.00	0.605
18	Mosier and Taube (1985b)	20X20	42.96	42.86	42.94	42.96	0.00	42.96	0.00	42.57	42.82	42.96	0.00	0.460
19	Kumar et al. (1986)	20X23	49.65	49.65	49.65	49.65	0.00	49.65	0.00	49.65	49.65	49.65	0.00	0.884
20	Carrie (1973)	20X35	76.54	76.22	76.22	76.22	0.42	76.22	0.42	76.54	76.54	76.54	0.00	0.986
21	Boe and Cheng (1991)	20X35	58.15	58.07	58.07	58.07	0.14	57.61	0.93	58.15	58.15	58.15	0.00	0.948
22	Chandrasekharan and Rajagopalan (1986)	24X40	100.00	100.00	100.00	100.00	0.00	100.00	0.00	100.00	100.00	100.00	0.00	1.465
23	Chandrasekharan and Rajagopalan (1986)	24X40	85.11	85.11	85.11	85.11	0.00	85.11	0.00	85.11	85.11	85.11	0.00	1.637

Table 4 (*Continued*)

Problems			Best	HGA				SCF-BMCF		GRASP				
			Known	Min	Avg	Max	Dev	Max	Dev	Min	Avg	Max	Dev	Avg CPU (sec)
24	Chandrasekharan and Rajagopalan (1986)	24X40	73.51	73.51	73.51	73.51	0.00	73.51	0.00	73.51	73.51	73.51	0.00	1.865
25	Chandrasekharan and Rajagopalan (1986)	24X40	51.97	51.85	51.88	51.97	0.00	51.97	0.00	51.88	51.90	51.97	0.00	3.100
26	Chandrasekharan and Rajagopalan (1986)	24X40	47.37	45.78	46.69	47.06	0.65	47.33	0.08	47.13	47.29	47.37	0.00	4.330
27	Chandrasekharan and Rajagopalan (1986)	24X40	44.87	44.51	44.75	44.87	0.00	44.87	0.00	44.87	44.87	44.87	0.00	4.445
28	McCormick et al. (1972)	27X27	54.27	54.27	54.27	54.27	0.00	54.27	0.00	52.16	53.51	54.27	0.00	0.990
29	Carrie (1973)	28X46	46.06	44.10	44.37	44.62	3.13	45.31	1.62	45.27	45.53	46.06	0.00	8.023
30	Kumar and Vannelli (1987)	30X41	59.52	n.a.	n.a.	n.a.	n.a.	59.52	0.00	58.06	59.00	59.52	0.00	6.719
31	Stanfel (1985)	30X50	60.00	n.a.	n.a.	n.a.	n.a.	60.00	0.00	60.00	60.00	60.00	0.00	9.053
32	Stanfel (1985)	30X50	50.51	50.25	50.48	50.51	0.00	50.51	0.00	50.51	50.51	50.51	0.00	10.583
33	Kusiak and Chow (1987)	36X90	45.93	41.48	42.12	42.64	7.16	44.59	2.92	44.27	45.09	45.93	0.00	25.223
34	McCormick et al. (1972)	37X53	59.85	56.42	56.42	56.42	5.73	59.04	1.35	59.85	59.85	59.85	0.00	2.857
35	Chandrasekharan and Rajagopalan (1986)	40X100	84.03	84.03	84.03	84.03	0.00	84.03	0.00	84.03	84.03	84.03	0.00	32.918
	Average						0.64		0.32				0.00	

As can be observed in Table 4 GRASP outperforms HGA and SCF-BMCF. It obtains very good quality solution with reasonable amount of computing effort. In all instances the proposed heuristic obtains the best known solution in at least one of the ten replicates, while HGA found the best known solution in 28 out of the 35 instances and SCF-BMCF also found the best known efficacy values only in 28 out of the 35 instances. In fact, the proposed GRASP heuristic improves the previous best known solutions in 7 out of the 35 test instances. It is important to notice that, as we mentioned before, test instances 30 and 31 differ from those used in Gonçalves and Resende (2004). To make a fair comparison our GRASP was also run with the original data and for these two instances we also outperform HGA. Although our maximum efficacy values are the same as those reported in Gonçalves and Resende (2004) our minimum and average efficacy are better. On average, the percentage deviation from the best known solutions is 0.64% and 0.32% for the HGA and the SCF-BMCF heuristics, respectively. Also, the largest percentage deviations of the best known solutions with respect to the best solution found for HGA and SCF-BMCF, respectively, are 7.16% (instance 33) and 3.40% (instance 16), while the largest percentage deviation of the best known solution with respect to the worst solution found by the GRASP heuristic never exceeds 3.89% (instance 18). The average percentage deviation of best known solutions with respect to the average solutions for the proposed GRASP algorithm is 0.2% which is even better than the 0.32% average percentage deviation of best solutions found by SCF-BMCF. CPU times of the proposed GRASP heuristic are very reasonable. Only in one out of the 35 instances (instance 35) average CPU time exceeds half minute and in all cases the required CPU time is less than one minute. It is worth noting that most of the CPU time used by the proposed GRASP heuristic is consumed within the improvement phase. On average, the improvement phase consumes 99% of CPU time.

The best solutions found for each instance are shown in Appendix.

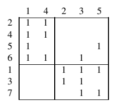
5 Conclusions

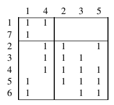
In this work we propose a GRASP heuristic to find lower bounds for the Manufacturing Cell Formation Problem. The proposed heuristics consists of two phases. In the first phase an initial partition of machines into machine-cells or parts into part-families is obtained, while in the second phase the assignment of parts to machine-cell or machines to part-families is considered. Also, during the second phase a local search heuristic is proposed to improve the efficacy measure of the solution.

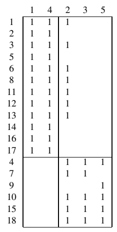
According to the obtained results we can observe that the proposed GRASP heuristic provides very good quality solutions with small computational effort. The proposed heuristic provides the best known solution for all of the test instances used to evaluate its performance and to compare it to other approaches from the literature. It is worth noting that for 7 out of the 35 test instances, the proposed method improves the previous best known solutions.

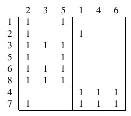
Acknowledgements The research of the third author was partially supported by a grant from the Consejo Nacional de Ciencia y Tecnología (CONACYT), México. The authors are also grateful to the reviewers for their valuable and insightful comments.

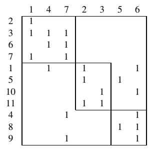
Appendix

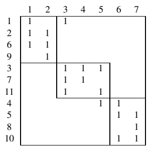

Problem 1

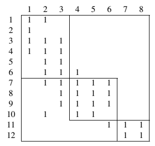

Problem 2

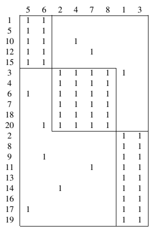

Problem 3

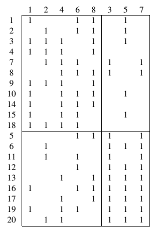

Problem 4

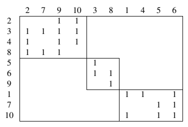

Problem 5

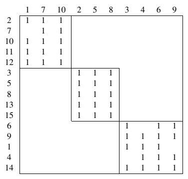

Problem 6

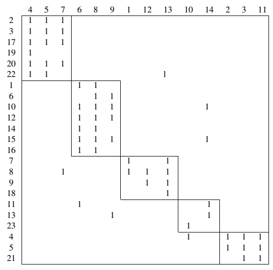

Problem 7

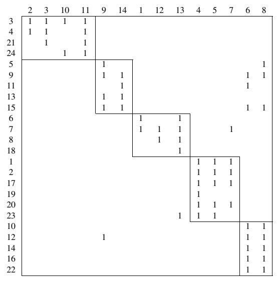

Problem 8

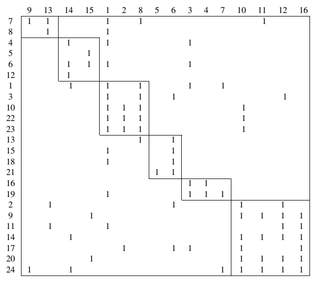

Problem 9

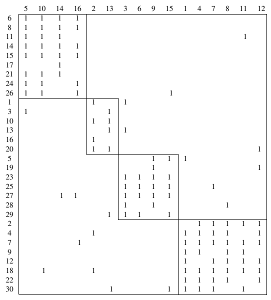

Problem 10

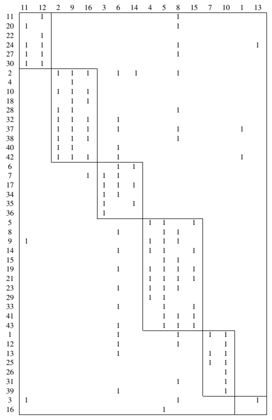

Problem 11

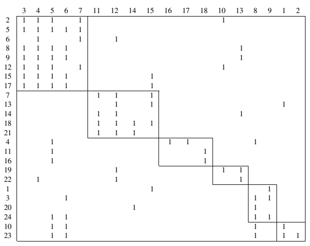

Problem 12


Problem 13


Problem 14


Problem 15


Problem 16


Problem 17

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

Problem 18

	1	3	5	6	12	13	18	4	8	14	20	2	11	16	19	7	10	9	15	17
1	1	1	1	1	1	1	1	1	1	1					1				1	
2		1		1	1												1	1		
4			1			1														
10	1		1	1	1	1	1											1		
11	1	1	1		1															
15	1	1	1	1	1	1	1		1	1	1	1								1
20					1	1								1						
3	1						1	1	1		1				1					
12								1	1	1	1								1	
16									1	1	1						1			
22	1	1		1				1	1	1	1						1	1	1	
5												1	1	1	1			1		
14												1		1						
6																	1			1
7		1											1	1		1	1	1		1
8																1	1			
9																1	1			
17								1									1			
13									1			1								1
18		1			1	1								1				1	1	1
19																		1	1	1
21	1							1										1	1	
23	1							1										1	1	

Problem 19

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

Problem 20

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

Problem 21

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

Problem 22

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

Problem 23

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

Problem 24

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

Problem 25

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

Problem 26

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

Problem 27

	6	17	20	21	23	24	26	27	9	10	11	12	1	2	3	4	5	7	8	13	14	16	19	22	15	18	25
6	1	1	1		1		1	1	1	1																	
17	1	1	1	1	1			1														1			1		
20	1	1	1	1		1	1				1																
21		1	1	1		1	1	1									1					1			1		
23	1	1			1			1																			
24			1	1		1	1	1																			
26	1		1	1		1	1																				
27	1	1		1	1	1		1																	1		1
9	1								1	1							1	1	1								
10	1								1	1	1	1					1	1	1	1	1		1				
11			1							1	1	1						1									1
12										1	1	1								1	1		1				
1													1	1		1	1	1	1	1				1		1	
2													1	1		1		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	1	1			1	1	1			
7									1	1	1		1	1	1		1	1	1	1	1	1		1			
8									1	1			1	1	1		1	1	1			1					
13										1		1		1		1		1		1		1	1	1			
14										1		1			1	1		1			1	1	1				
16		1		1										1	1	1	1	1	1	1	1	1	1	1	1	1	
19										1		1		1	1	1	1			1	1	1	1	1			
22													1	1			1	1		1		1	1	1	1	1	
15		1		1				1														1		1	1	1	
18													1	1								1		1	1		1
25								1			1															1	1

Problem 28

	23	25	28	11	13	14	1	8	9	10	15	6	18	26	19	20	21	22	3	4	5	2	16	17	12	24	7	27
34		1		1															1									
35		1										1							1									
13				1	1			1	1			1							1								1	
33						1											1				1							
39				1	1	1			1																1			
40		1		1	1				1																1			
41				1	1	1		1																				
43				1	1																					1		
44				1		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										
4	1						1	1	1	1																		
5	1							1	1	1					1													
7							1	1	1								1				1							
12							1	1		1													1					
15								1	1	1									1	1								
16								1	1	1		1							1									
24					1			1	1		1								1	1						1		
17								1				1	1	1					1		1							
31												1		1					1	1								
6								1	1						1		1				1							
8								1							1	1	1											
9								1	1						1	1												
10															1	1	1							1				
11															1		1	1			1							
32																1	1	1										
36															1	1				1								
14								1	1										1	1	1							1
18			1					1											1	1	1							
19										1			1						1	1	1							
20								1					1						1	1	1							
21												1	1						1	1	1							
22																			1	1	1					1		1
23								1											1	1	1					1		
25				1								1				1			1	1	1					1		
28																			1	1	1						1	
29												1							1		1						1	
30									1										1	1							1	
38							1															1	1					
45				1					1														1	1				
27												1							1						1	1	1	
42					1												1								1	1		
46		1						1																		1		
26								1				1							1								1	1
37															1				1								1	

Problem 29

	8	28	14	25	7	17	18	26	13	24	1	2	11	21	9	19	20	29	30	3	10	12	22	23	6	16	4	27	5	15
8	1	1																										1		
9	1																	1												
29	1	1																												
35		1																									1			
4			1	1																										
18			1									1																		
26			1																						1					
37			1																											1
14							1																							
16					1	1	1	1																						
27					1	1	1	1																						
34					1		1	1																					1	
36					1	1		1																					1	
19									1													1								
25									1																		1			
38									1	1																				
10											1		1	1									1							
11												1										1								
32											1	1		1						1			1							
33											1	1	1	1																
41											1	1	1																	
1	1															1	1	1	1											
3															1	1	1	1	1											
13															1			1												
21															1	1	1	1	1											
22																1	1		1											
30																1		1												
2																					1			1						
12												1		1						1	1	1	1	1						
20																					1	1								
23																				1	1	1	1	1						
31											1			1						1			1	1						
39											1	1	1							1	1	1	1	1						
40														1						1		1	1							
6																									1	1				
28																										1	1			
7						1																					1			
15																											1	1		
24																						1					1			
5			1																										1	
17																													1	1

Problem 30

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

Problem 31

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

Problem 32

	5	15	22	9	11	17	19	21	24	3	14	2	10	18	23	1	8	26	28	7	25	27	4	30	13	29	6	20	12	16
20			1																											
30			1		1																									
81	1	1	1		1					1											1			1						
83		1	1							1											1		1							1
85	1	1	1		1					1											1			1						
8			1		1		1	1	1															1						
15			1		1	1	1		1																					
21			1		1	1	1	1	1															1						
22			1		1	1	1	1	1															1						
40				1	1	1	1	1	1	1																				
51					1	1	1																							
56				1	1	1		1		1																				
64				1	1	1		1		1																				
68					1	1	1																							
73					1	1	1	1					1																	
76				1	1	1	1	1	1																					
77				1		1	1		1		1					1												1		
88		1				1	1	1			1																	1		
90						1	1	1	1		1	1						1										1		
14										1																1				
24										1																				
37										1								1												
44										1																				
47										1																1				
55											1							1			1					1		1		
58										1								1												
59										1	1																			1
65							1		1	1	1								1											1
66										1								1								1				
67						1				1	1													1						
70											1								1					1						
71										1	1								1					1						
87						1		1		1	1																	1		
89							1	1	1	1	1																			1
18													1																	
26														1																
16																1			1											
34																1														
50																1		1												
75																	1	1												
1																			1											
2																		1										1		
3																		1												
4																		1					1							
5																		1				1								
6																		1												
7																		1												
9			1				1			1								1	1					1						
13																		1	1			1								
19																		1			1									
25																		1	1											
27															1			1	1							1				
29																		1	1				1							
31																		1	1			1								
33			1																1											
42																		1										1		
43																		1	1				1							
46																		1				1								
54																		1	1			1								
57																			1							1				
60															1			1	1											
63					1	1												1										1		
69				1				1										1	1				1							
72																		1	1			1								
74			1																1				1							
78		1						1										1										1		
79						1		1	1									1	1					1				1	1	
84																		1	1			1					1	1		1
10																					1									
11																		1	1	1	1	1								
12																						1								
28																				1	1	1								
36																	1	1		1	1	1								
39																				1										
41																1		1	1	1	1	1								
49																		1			1	1	1							
61																					1	1								
23																							1							
38																							1							
45																								1						
48																							1	1				1		
52								1															1	1						
32																										1				
53																										1				
80									1																1	1				
82									1															1	1	1				
17																												1		
35																											1	1		
62					1		1																							1
86		1						1											1				1						1	1

Problem 33

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

Problem 34

	24	27	29	2	10	16	21	31	15	18	33	34	36	1	3	7	32	4	9	20	6	12	26	38	40	5	8	22	23	37	39	19	25	28	30	11	13	14	17	35
9	1	1	1																																					
12	1	1	1																																					
17	1	1	1																																					
19	1	1	1																																					
26	1	1	1																																					
30	1	1	1																																					
31	1	1	1																																					
40	1	1	1																								1													
43	1	1	1																																	1				
46	1	1	1																															1						
35				1	1	1	1																																	
47				1	1	1		1																																
53				1	1		1	1																																
77					1	1	1	1																					1											
78				1	1	1	1																		1															
79				1	1	1	1	1																																
83				1	1	1	1	1																											1					
88				1	1	1	1	1														1																		
93				1	1	1	1	1					1																											
39										1		1	1																								1			
45									1		1	1	1																											
67									1	1	1	1	1																											
71									1	1	1	1																					1							
73										1	1	1	1																			1								
91									1	1	1	1	1																						1					
10														1		1	1																							
18														1	1	1																								
29														1	1		1																							
33														1	1	1	1																							
34														1	1	1	1																							
37														1	1	1	1																							
41		1													1	1	1																							
44														1	1	1	1																							
49														1	1	1	1																							
50														1	1		1					1																		
54														1	1	1	1																							
55														1	1	1	1																							
8																		1	1	1																				
11																		1	1	1																				
13																		1	1	1																				
14																		1	1																	1					
20																		1	1	1																				
22																		1	1	1																				
23																		1	1	1																				
32																		1	1	1																				
36																						1	1	1	1															
38																					1	1	1	1																
42																					1		1	1	1															
51																					1	1	1	1																
52																						1	1	1	1															
64																					1	1	1	1	1															
65						1															1		1	1	1															
70																					1	1	1	1	1															
72							1														1	1	1		1															
74																					1	1	1	1	1															
75																					1	1	1		1															
76													1								1	1	1	1	1															
80																					1	1	1	1	1															
81																						1	1	1	1									1					1	
86						1															1	1	1	1			1													
87																					1	1	1	1	1	1														
63																										1	1			1	1									
66																										1	1	1		1	1									
68																										1	1	1	1	1										
69																										1		1	1	1	1									
82																										1	1	1	1	1	1									
84																			1							1	1	1	1	1										
85																										1	1	1	1	1	1									
89																										1	1	1	1	1	1									
90																										1	1	1	1	1	1									
92																										1	1	1	1	1	1									
94													1													1		1	1	1	1									
95																									1	1	1		1	1	1									
96																										1	1	1	1	1	1									
97																						1				1	1	1	1	1										
98																										1	1	1	1	1	1									
99																										1	1	1	1	1	1	1								
100										1																1	1	1	1	1	1									
56																																1	1	1	1					
57																																1	1	1	1					
58																																1	1	1	1					
59																																1	1	1	1					
61																																1	1	1	1					
62																																1	1	1	1					
1																																				1	1			
2																																				1	1			
3																																				1	1			
4																																				1	1			
5																																				1	1			
7																																				1	1			
21																																				1	1		1	
25																1																				1	1			
28																																				1	1	1		
48									1															1												1	1			
6																																						1		1
15																																						1	1	1
16																																						1	1	1
24																																						1	1	1
27																																				1		1		1
60																																					1	1	1	1

Problem 35

References

Adenso-Díaz B, Lozano S, Eguía I (2005) Part-machine grouping using weighted similarity coefficients. Comput Ind Eng 48:553–570

Adil GK, Ghosh JB (2005) Forming GT cells incrementally using GRASP. Int J Adv Manuf Technol 26(11):1402–1408

Adil GK, Rajamani D, Strong D (1997) Assignment allocation and simulated annealing algorithms for cell formation. IIE Trans 29:53–67

Askin RG, Subramanian S (1987) A cost based heuristic for group technology configuration. Int J Prod Res 25:101–113

Boctor F (1991) A linear formulation of the machine-part cell formation problem. Int J Prod Res 29(2):343–356

Boe W, Cheng CH (1991) A close neighbor algorithm for designing cellular manufacturing systems. Int J Prod Res 29(10):2097–2116

Carrie S (1973) Numerical taxonomy applied to group technology and plant layout. Int J Prod Res 11:399–416

Chan HM, Milner DA (1982) Direct clustering algorithm for group formation in cellular manufacture. J Manuf Syst 1:65–75

Chandrasekharan MP, Rajagopalan R (1986) An ideal seed non-hierarchical clustering algorithm for cellular manufacturing. Int J Prod Res 24(2):451–464

Feo TA, Resende MGC (1995) Greedy randomized adaptive search procedures. J Glob Optim 6:109–133

Gonçalves JF, Resende MGC (2004) An evolutionary algorithm for manufacturing cell formation. Comput Ind Eng 47:247–273

Gupta T, Seifoddini H (1990) Production data based similarity coefficient for machine-component grouping decisions in the design of a cellular manufacturing system. Int J Prod Res 28(7):1247–1269

Khan M, Islam S, Sarker B (2000) A similarity coefficient measure and machine-parts grouping in cellular manufacturing systems. Int J Prod Res 38(3):699–720

King JR (1980) Machine-components grouping in production flow analysis: an approach using a rank order clustering algorithm. Int J Prod Res 18(2):213–232

King JR, Nakornchai V (1982) Machine-component group formation in group technology: review and extension. Int J Prod Res 20(2):117–133

Kumar KR, Chandrasekhran MP (1990) A quantitative criterion for goodness of block diagonal forms of binary matrices in group technology. Int J Prod Res 28(2):233–234

Kumar KR, Vannelli A (1987) Strategic subcontracting for efficient disaggregated manufacturing. Int J Prod Res 25(12):1715–1728

Kumar KR, Kusiak A, Vannelli A (1986) Grouping of parts and components in flexible manufacturing systems. Eur J Oper Res 24:387–397

Kusiak A (1987) The generalized group technology concept. Int J Prod Res 25:561–569

Kusiak A, Cho M (1992) Similarity coefficient algorithm for solving the group technology problem. Int J Prod Res 30(11):2633–2646

Kusiak A, Chow W (1987) Efficient solving of the group technology problem. J Manuf Syst 6(2):117–124

Lee H, García-Díaz A (1993) A network flow approach to solve clustering problems in group technology. Int J Prod Res 31(3):603–612

McAuley J (1972) Machine grouping for efficient production. Prod Eng 51:53–57

McCormick WT, Schweitzer PJ, White TW (1972) Problem decomposition and data reorganization by a clustering technique. Oper Res 20:993–1009

Mosier CT (1986) An experiment investigating the application of clustering procedures and similarity coefficients to the GT machine-cell formation problem. Int J Prod Res 27:1811–1836

Mosier CT, Taube L (1985a) The facets of group technology and their impact on implementation. Omega 13(6):381–391

Mosier CT, Taube L (1985b) Weigthed similarity measure heuristics for the group technology machine clustering problem. Omega 13(6):577–583

Sarker BR (2001) Measures of grouping efficiency in cellular manufacturing systems. Eur J Oper Res 130:588–611

Sarker BR, Mondal S (1999) Grouping efficiency measures in cellular manufacturing: a survey and critical review. Int J Prod Res 37(2):285–314

Seifoddini H (1989) Single linkage versus average linkage clustering in machine cells formation applications. Comput Ind Eng 16(3):419–426

Seifoddini H, Wolfe PM (1986) Applications of the similarity coefficient method in group technology. IIE Trans 18(3):266–270

Srinivasan G, Narendran TT (1991) GRAFICS-A non hierarchical clustering-algorithm for group technology. Int J Prod Res 29(3):463–468

Srinivasan G, Narendran TT, Mahadevan B (1990) An assignment model for the part-families problem in group technology. Int J Prod Res 28:145–152

Stanfel L (1985) Machine clustering for economic production. Eng Costs Prod Econ 9:73–78

Waghodekar PH, Sahu S (1984) Machine-component cell formation in group technology MACE. Int J Prod Res 22:937–948

Wu TH, Chang CC, Yeh JY (2009) A hybrid heuristic algorithm adopting both Boltzmann function and mutation operator for manufacturing cell formation problems. Int J Prod Econ 120:669–688

Yasuda K, Yin Y (2001) A dissimilarity measure for solving the cell formation problem in cellular manufacturing. Comput Ind Eng 39(12):1–17