Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

On solving manufacturing cell formation via Biclister Editing

Rian G. S. Pinheiro[a,b], Ivan C. Martins[a], Fábio Protti[a,*], Luiz S. Ochi[a], Luidi G. Simonetti[a], Anand Subramanian[c]

[a] *Fluminense Federal University Niterói, RJ, Brazil*
[b] *Federal Rural University of Pernambuco Garanhuns, PE, Brazil*
[c] *Federal University of Paraíba João Pessoa, PB, Brazil*

ARTICLE INFO

Article history:
Received 11 July 2014
Accepted 6 May 2016
Available online 14 May 2016

Keywords:
Combinatorial optimization
Biclusterization
Graph partitioning
Manufacturing cell formation

ABSTRACT

This work investigates the Bicluster Graph Editing Problem (BGEP) and how it can be applied to solve the Manufacturing Cell Formation Problem (MCFP). We develop an exact method for the BGEP with a new separation algorithm. We also describe a new preprocessing procedure for the BGEP derived from theoretical results on vertex distances in the input graph. Computational experiments performed on randomly generated instances with various levels of difficulty show that our separation algorithm accelerates the convergence speed, and our preprocessing procedure is effective for low density instances. Another contribution of this work is to take advantage of the fact that the BGEP and the MCFP share the same solution space. This leads to the proposal of two new exact approaches for the MCFP that are based on mathematical formulations for the BGEP. Both approaches use the grouping efficacy measure as the objective function. Up to the authors' knowledge, these are the first exact methods that employ such a measure to optimally solve instances of the MCFP. The first approach is based on a new ILP formulation for the MCFP, and the second consists of iteratively running several calls to a parameterized version of the BGEP. Computational experiments performed on instances of the MCFP found in the literature show that our exact methods for the MCFP are able to prove several previously unknown optima.

1. Introduction

The Bicluster Graph Editing Problem (BGEP) is described as follows: given a bipartite graph $G = (U, V, E)$, where U and V are nonempty stable sets (called the "parts" of the bipartition (U, V)) and E is a set of unweighted edges linking vertices in U and a vertices in V, the goal is to transform G into a disjoint union of complete bipartite graphs (or *bicliques*) by performing a minimum number of *edge editing operations*. Each edge editing operation consists of either removing an existing edge in E or adding to E a new edge between a vertex in U and a vertex in V. Note that, in a feasible solution, each edge and each vertex must belong to exactly one biclique.

A *bicluster* is a subgraph of G isomorphic to a biclique. In some graph theoretical models used in Computational Biology and other areas, the existence of biclusters indicates a high degree of similarity between the data (vertices). In particular, a perfectly clustered bipartite graph is called a *bicluster graph*, i.e., a bipartite graph in which each of its connected components is a bicluster. Hence, we can alternatively define the goal of the BGEP, as stated by Amit (2004), as follows: "find a minimum number of edge editing operations in order to transform an input bipartite graph into a bicluster graph".

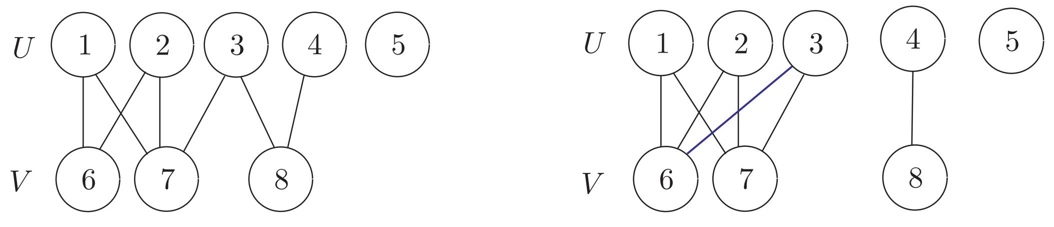
Fig. 1 shows an example where adding an edge between vertices 3, 6 and deleting the edge between vertices 3, 8 transforms G into a bicluster graph. Note that this does not correspond to an optimal solution, since G can also be transformed into a bicluster graph by simply removing the edge between 3 and 7. We remark that a single vertex is considered as a bicluster (e.g., vertex 5 in Fig. 1).

The Cluster Graph Editing Problem (CGEP) is a clustering problem similar to the BGEP. The CGEP was first studied by Gupta and Palit (1979) and its goal is to transform a (not necessarily bipartite) graph G into a disjoint union of complete graphs (cliques). The CGEP and the BGEP can also be viewed as important examples of partition problems in graphs.

The concept of biclustering was introduced in the mid-70s by Hartigan (1975). In 2000, Cheng and Church (2000) use biclustering within the context of Computational Biology. Since then, algorithms for biclustering have been proposed and used in various applications, such as multicast network design

* Corresponding author. Tel.: +55 2126295669; fax: +55 2126295669.
E-mail addresses: rgpinheiro@ic.uff.br (R.G.S. Pinheiro), imartins@ic.uff.br (I.C. Martins), fabio@ic.uff.br (F. Protti), satoru@ic.uff.br (L.S. Ochi), luidi@ic.uff.br (L.G. Simonetti), anand@ct.ufpb.br (A. Subramanian).

http://dx.doi.org/10.1016/j.ejor.2016.05.010
0377-2217/© 2016 Elsevier B.V. All rights reserved.



(a) Instance. (b) Solution.

Fig. 1. BGEP example.

$$\begin{array}{c|ccccc|}
 & M_1 & M_2 & M_3 & M_4 & M_5 \\ \hline
P_1 & 0 & 1 & 1 & 0 & 1 \\
P_2 & 1 & 0 & 0 & 1 & 0 \\
P_3 & 0 & 1 & 1 & 0 & 0 \\
P_4 & 1 & 0 & 0 & 1 & 0 \\
P_5 & 1 & 0 & 0 & 0 & 1 \\
P_6 & 1 & 0 & 1 & 1 & 0 \\
P_7 & 0 & 0 & 1 & 0 & 1 \\ \hline
\end{array}
\quad \Rightarrow \quad
\begin{array}{c|ccc|cc|}
 & M_2 & M_3 & M_5 & M_1 & M_4 \\ \hline
P_1 & 1 & 1 & 1 & 0 & 0 \\
P_3 & 1 & 1 & 0 & 0 & 0 \\
P_7 & 0 & 1 & 1 & 0 & 0 \\ \hline
P_2 & 0 & 0 & 0 & 1 & 1 \\
P_4 & 0 & 0 & 0 & 1 & 1 \\
P_5 & 0 & 0 & 1 & 1 & 0 \\
P_6 & 0 & 1 & 0 & 1 & 1 \\ \hline
\end{array}$$

Fig. 2. MCFP example. The matrix on the right represents a feasible solution, viewed as a permutation of rows/columns of the input matrix on the left. Cells are shown on the right as submatrices delimited by rectangles.

(Faure, Chretienne, Gourdin, & Sourd, 2007) and analysis of biological data (Abdullah & Hussain, 2006; Bisson & Hussain, 2008).

In Biology, concepts such as co-clustering, two-way clustering, among others, are often used in the literature to refer to the same problem. Matrices are used instead of graphs to represent relationships between genes and characteristics, and their rows/columns represent graph bipartitions; in this case, the goal is to find significant submatrices having certain patterns. The BGEP can be used to solve any problem whose goal is to obtain a biclusterization with *exclusive* rows and columns, i.e., each gene (characteristic) must be associated with only one submatrix.

Amit (2004) proved the $\mathcal{NP}$-hardness of the BGEP via a polynomial reduction from the 3-Exact 3-Cover Problem; in the same work, a binary integer programming formulation and an 11-approximation algorithm based on the relaxation of a linear program are described. The work by Protti, Dantas da Silva, and Szwarcfiter (2006) describes an algorithm for the parameterized version of the BGEP that uses a strategy based on modular decomposition techniques. Guo, Hffner, Komusiewicz, and Zhang (2008) developed a randomized 4-approximation algorithm for the BGEP. More recently, Sousa Filho, dos Anjos F. Cabral, Ochi, and Protti (2012) proposed a GRASP-based heuristic for the BGEP.

A new application of the BGEP, introduced in this work, is related to the Manufacturing Cell Formation Problem (MCFP). The input of the MCFP is given as a binary product-machine matrix M such that each entry $M(i, j)$ has value 1 if product i is manufactured by machine j, and 0 otherwise. Any feasible solution of the MCFP consists of a product-machine cell assignment, i.e., a collection of product-machine cells where every product (or machine) is allocated to exactly one cell. Hence, for each cell C, machines allocated to C are exclusively dedicated to manufacture products also allocated to C. In an ideal solution of the MCFP, for each cell C there must be a high similarity between products and machines allocated to it. Fig. 2 shows an example of the MCFP solved as a block diagonalization problem. Note that a solution of the MCFP can be viewed as a permutation of rows/columns of the input matrix yielding a new matrix M' where diagonal block submatrices represent cells. Of course, the permutation is not needed to obtain a solution, it only helps to make it more visual. In the figure, products P_1, P_3, P_7 and machines M_2, M_3, M_5 are gathered to form a cell, while the remaining products/machines form another cell.

Among several measures of performance used as objective functions for the MCFP, Sarker and Khan (2001) evaluated the quality of a solution using different measures reported in the literature as: *Grouping Efficiency* (Chandrasekharan & Rajagopalan, 1986); *Grouping Efficacy* (Kumar & Chandrasekharan, 1990); *Grouping Capability Index* (Hsu, 1990); and *Grouping Measure* (Miltenburg & Zhang, 1991). The *grouping efficacy* μ is considered in the literature as a standard measure to represent the quality of solutions. It is defined as:

$$\mu = \frac{N_1 - N_1^{out}}{N_1 + N_0^{in}}\ , \tag{1}$$

where N_1 is the total number of 1's in the input matrix, and N_1^{out} (N_0^{in}) is the total number of 1's outside (respectively, 0's inside) diagonal blocks in the solution matrix. In Fig. 2, $\mu = \frac{16-2}{16+3} = 0.7368$.

Some works define a minimum value for the size of the cells. For instance, in (Chandrasekharan & Rajagopalan, 1987; Gonçalves & Resende, 2004; Srinivasan & Narendran, 1991), cells with less than two products or machines are not allowed; such cells are called *singletons*. However, there is no consensus with respect to the size of the cells. Other studies do not consider any size constraint, allowing the existence of empty cells, such as the work by Pailla, Trindade, Parada, and Ochi (2010). An example of a solution with an empty cell is shown in Fig. 3.

In the present work, we deal with two versions of the MCFP found in the literature:

1. unrestricted version, allowing singletons and empty cells;
2. with cell size constraints (the minimum size of each cell is 2 $\times$ 2).

Cellular manufacturing is an application of the Group Technology concept (Goldengorin, Krushinsky, & Pardalos, 2013). The goal is to identify and cluster similar parts in order to optimize the manufacturing process. Such a concept was originally introduced by Flanders (1924) and formally described by Mitrofanov (1966) in 1966. In the early 70s, Burbidge (1971) presented one of the first techniques for creating a system of cellular manufacturing. Since this work, several approaches have been proposed to the MCFP,

$$\begin{array}{c|ccccc|}
 & M_1 & M_2 & M_3 & M_4 & M_5 \\ \hline
P_1 & 0 & 1 & 1 & 0 & 1 \\
P_2 & 1 & 0 & 0 & 0 & 0 \\
P_3 & 0 & 1 & 1 & 0 & 0 \\
P_4 & 1 & 0 & 0 & 0 & 0 \\
P_5 & 1 & 0 & 0 & 0 & 1 \\
P_6 & 0 & 0 & 1 & 1 & 0 \\
P_7 & 0 & 0 & 1 & 0 & 1 \\
P_8 & 0 & 0 & 0 & 0 & 1 \\ \hline
\end{array}
\Rightarrow
\begin{array}{c|ccc|c|c|}
 & M_2 & M_3 & M_5 & M_1 & M_4 \\ \hline
P_1 & 1 & 1 & 1 & 0 & 0 \\
P_3 & 1 & 1 & 0 & 0 & 0 \\
P_7 & 0 & 1 & 1 & 0 & 0 \\ \hline
P_2 & 0 & 0 & 0 & 1 & 0 \\
P_4 & 0 & 0 & 0 & 1 & 0 \\
P_5 & 0 & 0 & 1 & 1 & 0 \\ \hline
P_6 & 0 & 1 & 0 & 0 & 1 \\ \hline
P_8 & 0 & 0 & 1 & 0 & 0
\end{array}$$

Fig. 3. Example with a singleton and an empty cell.

whose goal is to create the cells in order to optimize the manufacturing process, as described in Section 5. The work by Boutsinas (2013) applies biclustering techniques to the MCFP, but in a very different way from our approach: in his work, data mining techniques are directly applied to a machine-part matrix in order to identify the cells (diagonal blocks). Our approach consists of exact methods applied to a corresponding bipartite graph, and make strong use of graph theoretical developments. We show that the BGEP and MCFP have a high degree of similarity, and that good solutions for the BGEP are close to good solutions for the MCFP.

1.1. Similarity between the BGEP and the MCFP

Consider an instance $G = (U, V, E)$ of the BGEP, and let M be the $|U| \times |V|$ adjacency matrix of G where rows correspond to elements of U and columns to elements of V. Then, it is clear that M is an instance of the MCFP. Conversely, if M is an instance of the MCFP, we can easily define an instance G of the BGEP by setting U as the set of products, V as the set of machines, and E as the set of edges such that ij is an edge of G if and only if $M(i, j) = 1$. In addition, a feasible solution G' of the BGEP (i.e., a biclusterization of G) can be easily transformed into a machine-product cell assignment M', and vice-versa: if vertices $i \in U$ and $j \in V$ belong to the same bicluster in G' then product i and machine j are gathered inside the same cell in M' (biclusters with only one element are mapped into empty cells); conversely, if a product i and a machine j are in the same cell in M' then the corresponding vertices i and j belong to the same bicluster in G' (empty cells are mapped into biclusters with one element).

The above discussion implies that instances of the BGEP can be encoded as instances of the MCFP, i.e., 0–1 matrices. In both problems, the objective is to find a biclustering of rows and columns. In other words, if $R_1, \dots, R_k$ is a partition of the rows and $C_1, \dots, C_k$ a partition of the columns of the input matrix, then a biclustering associated with the sub-matrices $(R_1, C_1), \dots, (R_k, C_k)$ is a solution for both the BGEP and the MCFP. The only difference between them lies in the objective function. The BGEP objective function is given by

$$N^1_{out} + N^0_{in} \tag{2}$$

whereas the MCFP objective function is given by (1).

An optimal solution G^* of the BGEP does not necessarily correspond to an optimal solution of the MCFP, because the objective functions are different; however, G^* corresponds to a feasible solution of the MCFP. For example, Fig. 4(b) shows an optimal solution of the BGEP, but in Fig. 4(d) the corresponding solution of the MCFP is not optimal. This happens because an edge deletion, informally, corresponds to a ‘1’ outside cells, and an edge addition corresponds to a ‘0’ inside a cell. That is, for the BGEP, additions and deletions have the same weight, but for the MCFP, a ‘0 inside’ is preferable than a ‘1 outside’ (using the objective function (1)). In Fig. 4(a), for instance, adding an edge between vertices 5 and 8 is better than deleting the edge between 4 and 9, in terms of the corresponding solutions of the MCFP.

We remark that the BGEP can be viewed as a generic clustering framework whose primary goal is to obtain a partition of the graph that optimizes a natural, simple optimization criterion (namely, number of editing operations). Due to its generic nature, it can be used to model several real-world problems (e.g., the MCFP) that use the same set of valid clustering solutions but are equipped with more sophisticated objective functions (e.g., the grouping efficacy).

1.2. Contributions of this work

Our contributions can be summarized as follows. In Section 2, we propose two new variants for the BGEP and their corresponding mathematical formulations. The first variant is the Bicluster Graph Editing Problem with Size Restriction (BGEPS), which considers bicluster size constraints, and the second is a parameterized version of the BGEPS denoted by λ-BGEPS, which focuses on finding a solution with exactly λ editing operations and minimum number of edge deletions. In Section 3, we develop an exact method for the BGEP consisting of a Branch-and-Cut approach combined with a separation algorithm, and describe a new preprocessing procedure for the BGEP derived from theoretical results on vertex distances in the input graph. In Section 4, we explore the similarity between the BGEP and the MCFP. Since both problems share the same solution space, we describe (Section 4.2) a new ILP formulation for the MCFP based on the formulation for the λ-BGEPS. In Section 5, we present a novel approach to solve the MCFP using the grouping efficacy measure as the objective function. It consists of iteratively running several calls to a λ-BGEPS solver. In Section 6, we apply our exact method for the BGEP to solve randomly generated instances with various levels of difficulty. Experimental results show that our separation algorithm is able to accelerate the convergence speed, and our preprocessing procedure for the BGEP is effective for low density instances. In addition, computational experiments are performed on instances of the MCFP found in the literature. We compare our new ILP formulation (Section 4.2), a standard linearization (Section 4.1.1), our exact iterative method (Section 5.1), and Dinkelbach's algorithm (Section 5.2). The reported results reveal that our exact methods for the MCFP are able to prove several previously unknown optima. Section 7 contains our conclusions. We remark that, up to the authors' knowledge, our exact methods for the MCFP are the first that employ the grouping efficacy measure to optimally solve instances of the problem.

2. Mathematical formulations for BGEP and variants

A mathematical model for the BGEP is described in Amit (2004). It relies on the simple fact that the graph P_4 (a path with

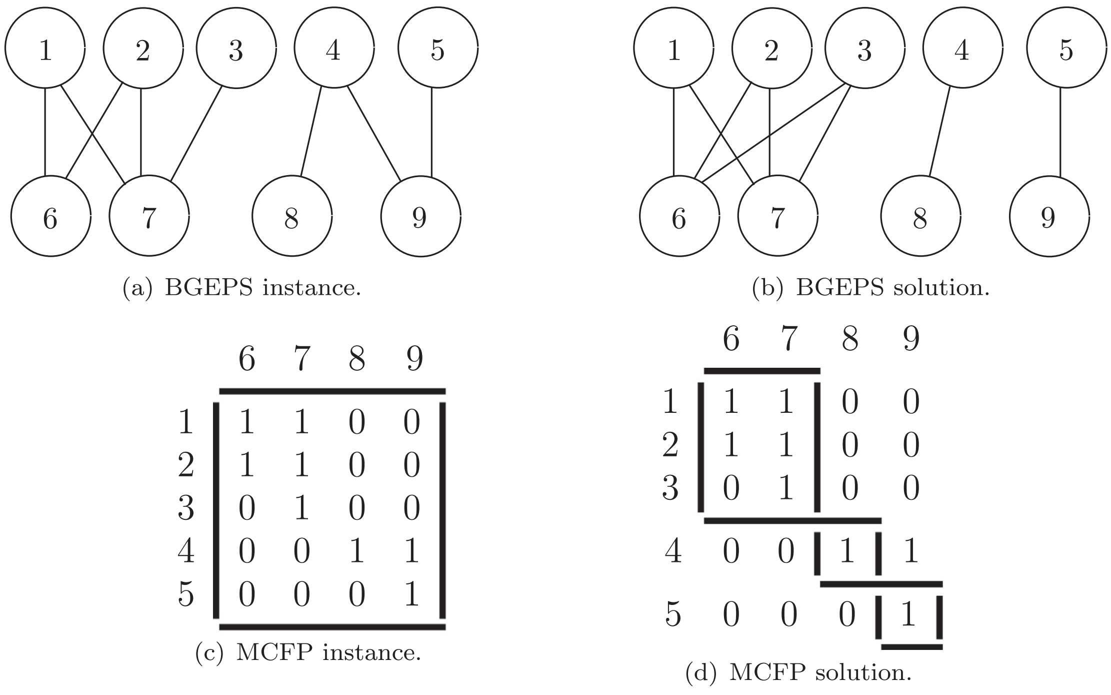


Fig. 4. BGEPS ↔ MCFP example.

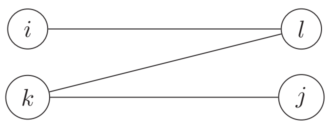


Fig. 5. Graph P_4.

four vertices, shown in Fig. 5) is a forbidden induced subgraph for a bicluster graph. More precisely, for a bipartite graph G, G is a bicluster graph if and only if G does not contain P_4 as an induced subgraph.

The formulation proposed in Amit (2004) is as follows:

$$\text{F-BGEP:}\quad \min \sum_{ij\in E}(1-y_{ij})+\sum_{ij\notin E}y_{ij} \tag{3}$$

$$\text{s.t.}\quad y_{il}+y_{kl}+y_{kj}\le 2+y_{ij}\quad \forall\, i,k\in U,\ i\neq k;\ l,j\in V,\ l\neq j \tag{4}$$

$$y_{ij}\in\{0,1\}\quad \forall\, i\in U,\ j\in V \tag{5}$$

where: (a) y_{ij} are binary variables such that $y_{ij}=1$ if and only if the solution contains edge ij; (b) $\{ij|ij\in E\}$ is the set of edges and $\{ij|ij\notin E\}$ the set of non-edges of G. The objective function (3) counts how many edge editing operations are made. The first and second sums represent the number of edge deletions and edge additions, respectively. Constraints (4) eliminate induced subgraphs isomorphic to P_4. Constraints (5) define the domain of the variables.

2.1. BGEP variants

We define a new BGEP variant, the BGEPS, to make a precise correspondence to the MCFP with cell size constraints. In brief, the BGEPS is defined by adding the following constraints to the BGEP: every bicluster in G must have at least s_r vertices of U and s_c vertices of V. In the translation from the BGEPS to the MCFP with cell size constraints, s_r is the minimum cell size for rows, and s_c is the minimum cell size for columns.

We propose the following formulation for the BGEPS by adding constraints (4) and (5):

$$\text{(F-BGEPS):}\quad \min\ (3)$$
$$\text{s.t.}\quad (4),(5)$$
$$\sum_{j\in V}y_{ij}\ge s_c\quad \forall\, i\in U \tag{6}$$
$$\sum_{i\in U}y_{ij}\ge s_r\quad \forall\, j\in V \tag{7}$$

In the above formulation, (6) and (7) are the bicluster size constraints.

We now define a parameterized variant of the BGEPS denoted by λ-BGEPS, which consists of finding a solution of the BGEPS with exactly λ edge editing operations, such that the number of edge deletions is minimized. A formulation for the λ-BGEPS is given by replacing the objective function (3) by (8) and adding the constraint (9):

$$\text{(F-}\lambda\text{-BGEPS):}\quad \min \sum_{ij\in E}(1-y_{ij}) \tag{8}$$
$$\text{s.t.}\quad (4),(5),(6),(7)$$
$$\sum_{ij\in E}(1-y_{ij})+\sum_{ij\notin E}y_{ij}=\lambda \tag{9}$$

3. Exact approach and preprocessing procedure for the BGEP

Branch-and-Cut combines the ideas of the well-known Branch-and-Bound implicit enumeration procedure with cutting planes. More precisely, a cutting plane approach is applied at each node of the Branch-and-Bound tree. The reader is referred to Wolsey (1998) for more details.

Note that the number of constraints (4) in the formulation by Amit is $|U|^2|V|^2$. Preliminary tests showed that considering all these constraints is not advisable – this would be computationally expensive for exact methods, especially when dealing with large instances. Alternatively, we start running the exact method with no such constraints, and add them in a cutting plane fashion as they are violated according to the separation algorithm described in the next section (Section 3.1).

In Section 6, we will compare two separation procedures: the one described in Section 3.1 and the default procedure used by a Mixed Integer Programming Solver (CPLEX), where all the constraints are added in advance and the software itself manages the pool of constraints by means of the LazyConstraints callback function.

3.1. Separation algorithm

In this section, we describe a separation algorithm for the BGEP. Its main objective is to find the "most violated constraint" (4) for each 4-tuple i, k, l, j of vertices, adding it to the model. The idea is to use an auxiliary complete bipartite graph $G'(U, V, E')$ where each edge $ij \in E'$ has a nonnegative weight; the weights are defined according to the linear relaxation values y^*_{ij}. Note that an edge in E' may have a zero weight.

After the construction of G', the constraints are determined by calculating the values c^3_{ij}, for $i \in U$ and $j \in V$, where c^3_{ij} is the *maximum cost* over all paths between i and j with exactly 3 edges (and thus 2 internal vertices). Calculating each value c^3_{ij} amounts to solving a 3-hop longest path problem in a bipartite graph, which can be done in polynomial time via a straightforward dynamic programming algorithm (Bellman, 1962). The algorithm first determines the values c^1_{ij} and $c^2_{i\langle j\rangle k}$. Each initial value c^1_{ij}, for $i \in U$ and $j \in V$, is set to y^*_{ij} (that is, linear relaxation values are used as input values to the algorithm). Next, each value $c^2_{i\langle j\rangle k}$ is set to the maximum cost between $i, k \in U$ ($k \neq i$) over all paths (i, l, k) such that $l \in V$ and $l \neq j$ (i.e., over all paths with 2 edges between i and k that avoid j as an internal vertex), in the following way:

$$c^2_{i\langle j\rangle k} = \max_{\substack{l\in V\\ l\neq j}}\{c^1_{il} + c^1_{lk}\}. \tag{10}$$

Finally, the values c^3_{ij} are given by:

$$c^3_{ij} = \max_{\substack{k\in U\\ k\neq i}}\{c^2_{i\langle j\rangle k} + c^1_{kj}\}. \tag{11}$$

If $c^3_{ij} - c^1_{ij} > 2$ then the cut $y_{il} + y_{kl} + y_{kj} \leq 2 + y_{ij}$ is added to the model, where k and l are suitably chosen from the solution of Eqs. (11) and (10), respectively.

3.2. Preprocessing procedure

In this section, we describe a preprocessing procedure to fix variables and/or generate new constraints to the BGEP. The procedure is a direct application of Theorem 1, presented below. New generated constraints will be added to the Branch-and-Cut algorithm.

Theorem 1. *Let a, b be vertices of a bipartite graph $G = (U, V, E)$, and let $d(a, b)$ be the distance (counted in number of edges) between a and b in G. If $d(a, b) \geq 4$ then there is an optimal solution in which a, b belong to distinct biclusters.*

Proof. The proof consists of showing that, when $d(a, b) \geq 4$, the cost of keeping a and b in the same bicluster is greater than or equal to the cost of keeping them in distinct biclusters.

For the case $d(a, b) = \infty$, note that a and b lie in distinct connected components of G, and therefore will belong to distinct biclusters in any optimal solution. Now assume that $d(a, b) < \infty$ and there is a solution G^* in which vertices a, b belong to a bicluster B with vertex set $V(B) = X \cup Y$, where $X \subseteq U$ and $Y \subseteq V$. We analyze two cases: $a, b \in X$ (Case 1) and $a \in X$, $b \in Y$ (Case 2).

Denote by $N(x)$ the neighborhood of a vertex x, and by $N^2(x)$ the subset $\{y \in U \cup V \mid d(x, y) = 2\}$.

Let $N_a = N(a) \cap V(B)$ and $N_b = N(b) \cap V(B)$.

Case 1: In this case, note that $N_a \subseteq Y$ and $N_b \subseteq Y$. In addition, since $d(a, b) \geq 4$, we have that $N_a \cap N_b = \emptyset$.

Let $N_0 = \{y \in Y \mid ay, by \notin E\}$. Notice that $N_0 \cup N_a \cup N_b$ is a partition of Y.

Assume without loss of generality that $|N_a| \geq |N_b|$. Consider another solution G^{**} constructed from G^* in which bicluster B is replaced by two biclusters B' and B'' where $V(B') = V(B) \setminus \{b\}$ and $V(B'') = \{b\}$ (the remaining biclusters other than B', B'' in G^{**} are exactly the same as in G^*). Let c^* and c^{**} be, respectively, the values of the solutions G^* and G^{**}. We analyze how c^{**} is expressed in terms of c^*. To construct G^{**} from G^*, we simply need to remove the edges between b and $Y = N_a \cup N_b \cup N_0$. But note that, in G, there are no edges between b and $N_0 \cup N_a$, and there are all edges between b and $N(b)$. Therefore, $c^{**} = c^* - |N_0| - |N_a| + |N_b|$. As $|N_a| \geq |N_b|$, this implies $c^{**} \leq c^*$, i.e., the cost of keeping a and b in distinct biclusters is not greater than the cost of keeping them in the same bicluster.

Case 2: In this case we have $N_a \subseteq Y$ and $N_b \subseteq X$. Also, since $d(a, b) \geq 4$, there are no edges between N_a and N_b in the input graph G.

Let $N^a_0 = \{y \in Y \mid y \neq b, ay \notin E\}$ and $N^b_0 = \{x \in X \mid x \neq a, bx \notin E\}$. Then $\{a\} \cup N_b \cup N^b_0$ is a partition of X, and $\{b\} \cup N_a \cup N^a_0$ is a partition of Y.

Let $b' \in N_b$. The proof in Case 2 is based on the fact that $d(a, b') \geq 4$ for every such b'. This is explained as follows. Since $a, b' \in U$, we know that $d(a, b')$ is even. Moreover, we cannot have $d(a, b') = 2$, for otherwise there is an induced path (a, a', b', b) with $a' \in N(a)$, contradicting the fact that $d(a, b) \geq 4$.

Now, since $a, b' \in X$, we can use Case 1 to conclude that replacing B by a pair of biclusters B', B'' with either $(V(B') = V(B) \setminus \{a\}$ and $V(B'') = \{a\})$) or $(V(B') = V(B) \setminus \{b'\}$ and $V(B'') = \{b'\})$) yields a solution G^{**} with value not higher than the value of G^*. If the former possibility occurs then we are done, because in this case a and b belong to distinct biclusters. Otherwise, we can apply Case 1 again to bicluster B'. This process continues until we get a solution G^*_1 with value not higher than the value of G^* such that a, b belong to a same bicluster B_1 with $V(B_1) \cap U = \{a\} \cup N^b_0$ and $V(B_1) \cap V = \{b\} \cup N_a \cup N^a_0$. Notice that there are no edges in the input graph G linking b to vertices in $V(B_1)$.

To conclude the proof, consider another solution G^{**} obtained from G^*_1 in which bicluster B_1 is replaced by two biclusters B'_1 and B''_1 where $V(B'_1) = V(B_1) \setminus \{b\}$ and $V(B''_1) = \{b\}$, similarly as in Case 1; in other words, the construction of G^{**} simply consists of removing the edges between b and $\{a\} \cup N^b_0$. It is easy to see that the values c^*_1, c^{**} of G^*_1, G^{**}, respectively, satisfy $c^{**} = c^*_1 - 1 - |N^b_0|$. This concludes Case 2.

For each case above, we have shown that there is another solution in which a and b belong to distinct biclusters and whose value is not greater. Then the theorem follows. □

Note that Theorem 1 is no longer valid when bicluster size constraints are added to the model. Based on the above theorem, after computing the distance between each pair of vertices, variables can be fixed and cuts can be generated as follows: if vertices i and j are in different parts of the bipartition and $d(i, j) > 3$ then variable y_{ij} is set to 0. Otherwise, i and j are in the same part, say U, and the cut $y_{ik} + y_{jk} \leq 1$ will be generated for every $k \in V$.

4. Mathematical formulations for the MCFP

In this section we describe a new ILP formulation for the MCFP (Section 4.2). Before its description, we present a natural linear-fractional programming formulation for the MCFP (Section 4.1) and a standard linearization of this formulation (Section 4.1.1).

4.1. Linear-fractional programming formulation

Note that the constraints of the BGEPS formulation can be used in a linear-fractional formulation for the MCFP. The objective function is set according to the grouping efficacy measure described in Section 1. Let $m = |E(G)|$. By putting $N_1^{out} = \sum_{ij\in E}(1-y_{ij})$ and $N_0^{in} = \sum_{ij\notin E} y_{ij}$, we derive the following formulation:

$$\text{(F-MCFP):} \quad \max \quad \frac{m - \sum_{ij\in E}(1-y_{ij})}{m + \sum_{ij\notin E} y_{ij}}$$
$$\text{s.t.} \quad (4), (5), (6), (7) \tag{12}$$

4.1.1. Linearization

Now we describe a standard linearization of the model described above. See Glover (1975) for details. Let $z = \frac{m-N_1^{out}}{m+N_0^{in}}$. Clearly, $z(m+\sum_{ij\notin E} y_{ij}) = m - \sum_{ij\in E}(1-y_{ij})$. Thus, $mz + \sum_{ij\notin E} zy_{ij} = m - \sum_{ij\in E}(1-y_{ij})$. Replacing zy_{ij} by a new variable z_{ij}, we obtain:

$$mz + \sum_{ij\notin E} z_{ij} = m - \sum_{ij\in E}(1-y_{ij}).$$

Let L be a lower bound and U an upper bound for the value of the objective function. Note that $L \le z \le U$ and $Ly_{ij} \le z_{ij} \le Uy_{ij}$, for all $ij \notin E$. Note also that suitable values of L and U can produce better results.

The proposed linearization is as follows:

$$\text{(L-MCFP):} \quad \max \quad z \tag{13}$$

$$\text{s.t.} \quad (4)$$

$$mz + \sum_{ij\notin E} z_{ij} = m - \sum_{ij\in E}(1-y_{ij}) \tag{14}$$

$$L \le z \le U \tag{15}$$

$$Ly_{ij} \le z_{ij} \le Uy_{ij} \quad \forall\, ij \notin E \tag{16}$$

$$z - U(1-y_{ij}) \le z_{ij} \le z - L(1-y_{ij}) \quad \forall\, ij \notin E \tag{17}$$

$$z \in \mathbb{R} \tag{18}$$

$$z_{ij} \in \mathbb{R} \quad \forall\, ij \notin E \tag{19}$$

The above linearization will be compared with our new exact methods in Section 6.

4.2. New ILP formulation

We now describe a new ILP formulation for the MCFP that uses two sets of variables. The binary variables x_{da} assume value 1 if and only if the solution contains d deletions and a additions. Each variable x_{da} is associated with a cost $f(d,a) = \frac{m-d}{m+a}$ in the objective function. Let l_c and l_b be lower bounds for the MCFP and the BGEP, respectively. In this case, l_c can be obtained heuristicaly, whereas l_b, for example, can be equal to the ceiling of the linear relaxation of the BGEP. The proposed linear formulation is as follows:

$$\text{(F-MCFP}'\text{)}: \quad \max \quad \sum_{\substack{a\le|U||V|-m\\ d\le m}}\sum f(d,a)\, x_{da} \tag{20}$$

$$\text{s.t.} \quad (4), (5), (6), (7)$$

$$\sum_{\substack{a\le|U||V|-m\\ d\le m}}\sum x_{da} = 1 \tag{21}$$

$$\sum_{\substack{a\le|U||V|-m\\ d\le m}}\sum a\, x_{da} = \sum_{ij\notin E} y_{ij} \tag{22}$$

$$\sum_{\substack{a\le|U||V|-m\\ d\le m}}\sum d\, x_{da} = \sum_{ij\in E}(1-y_{ij}) \tag{23}$$

$$\sum_{\substack{a\le|U||V|-m\\ d\le m}}\sum f(d,a)\, x_{da} \ge l_c \tag{24}$$

$$\sum_{\substack{a\le|U||V|-m\\ d\le m}}\sum (d+a)\, x_{da} \ge l_b \tag{25}$$

$$x_{da} \in \{0,1\} \quad \forall\, a \le |U||V| - m,\ d \le m \tag{26}$$

The objective function (20) computes the maximum cost $f(d,a) = \frac{m-d}{m+a}$. Constraint (21) imposes that exactly one x_{da} variable should assume value 1. Constraints (22) and (23) state that the number of deletions and additions must be d and a, respectively. Constraints (24) and (25) define that the objective function and the number of editions must be greater than the lower bounds. Constraints (26) define the domain of the variables. In the computational experiments we discarded variables x_{ad} not complying with $\frac{m-d}{m+a} \ge l_c$ and $a + d \ge l_b$.

5. Exact iterative method for the Manufacturing Cell Formation Problem

In this section we describe an exact iterative method for the MCFP (Algorithm 1 below). First, we present a lemma that establishes an upper bound for the MCFP.

Lemma 2. *Let $b^* = a^* + d^*$ be the optimum of the BGEPS for an instance G, where a^* and d^* are the number of edge additions and deletions, respectively, and let M be the instance of the MCFP corresponding to G. Then $m/(m + a^* + d^*)$ is an upper bound for the optimum of the MCFP with input M, where $m = |E(G)|$.*

Algorithm 1 Exact iterative method for the MCFP.

Input: instance M
Output: number of zeros inside cells (a^*) and ones outside cells (d^*) in an optimal solution of the MCFP

```
1:  let G be the instance of the BGEPS corresponding to M
2:  $(a^*, d^*) \leftarrow$ BGEPS-solver    ▷ solve the BGEPS for input G, obtaining $a^*$ additions and $d^*$ deletions
3:  $UB \leftarrow \frac{m}{m+a^*+d^*}$
4:  $LB \leftarrow \frac{m-d^*}{m+a^*}$
5:  $d_{max} \leftarrow d^* - 1$
6:  $\lambda \leftarrow a^* + d^*$
7:  while $UB > LB$ do
8:      if there is a feasible solution of the $\lambda$-BGEPS with parameters $\lambda$ and $d_{max}$ then
9:          $(a, d) \leftarrow \lambda$-BGEPS-solver$(\lambda, d_{max})$    ▷ solve the $\lambda$-BGEPS
10:         if $d \le d_{max}$ then    ▷ a new bound on the number of deletions has been found
11:             $d_{max} \leftarrow d - 1$    ▷ the algorithm will try to find a solution with fewer deletions
12:         end if
13:         if $\frac{m-d}{m+a} > LB$ then
14:             $LB \leftarrow \frac{m-d}{m+a}$
15:             $(a^*, d^*) \leftarrow (a, d)$
16:         end if
17:     end if
18:     $\lambda \leftarrow \lambda + 1$
19:     $UB \leftarrow \frac{m}{m+\lambda}$
20: end while
21: return $(a^*, d^*)$
```

Proof. Let $\mu(a,d) = (m-d)/(m+a)$ be the objective function of the MCFP with input M, where d and a denote, respectively, the number of ones outside cells and zeros inside cells in a solution. First, note that

$$\mu(a,d) = \frac{m-d}{m+a} \leq \frac{m}{m+a+d}.$$

Now suppose that $\mu'(a,d) = a+d$ is a feasible solution value of the BGEPS. Clearly, $a^* + d^* \leq a + d$. Therefore,

$$\frac{m}{m+a^*+d^*} \geq \frac{m}{m+a+d} \geq \frac{m-d}{m+a},$$

that is, $m/(m+a^*+d^*)$ is an upper bound for the optimum of the MCFP. $\square$

In addition to the above result, we observe that $(m-d^*)/(m+a^*)$ is a lower bound for the optimum of the MCFP, since $b^* = a^* + d^*$ is a feasible solution value of the BGEPS and the MCFP.

5.1. Description of the exact iterative method for the MCFP

The idea is to make repeated calls to a solver of the λ-BGEPS, as follows.

In line 2 of Algorithm 1, a BGEPS solver calculates the number of edge editing operations in an optimal solution of the BGEPS, whose value is $a^* + d^*$. Such a value is not necessarily an optimum μ^* of the MCFP – it may correspond to a solution of the BGEPS with more editing operations. To find μ^*, we resort to the λ-BGEPS. Recall that the objective function of the λ-BGEPS leads to a solution with the maximum number of edge additions among all solutions with exactly λ editing operations.

The initial bounds (variables *UB* and *LB*) are calculated in lines 3 and 4. At each iteration the bounds are updated. Iterations are performed until the upper bound is less than or equal to the lower bound, meaning that from that point there is no better solution.

Each iteration first checks whether there is a feasible solution of the λ-BGEPS with parameters λ (specifying the number of edge editing operations) and d_{max} (which limits the search space to optimal solutions with at most d_{max} edge deletions); if so, the λ-BGEPS solver runs (line 9) with the addition of the following constraint to the λ-BGEPS model:

$$\sum_{ij \in E} (1 - y_{ij}) \leq d_{max}. \tag{27}$$

The purpose of parameter d_{max} is actually to speed up the execution of Algorithm 1 by forcing the λ-BGEPS solver to find a solution with $d \leq d_{max}$ edge deletions, if any. It is initialized in line 5 with $d^* - 1$, and is updated in lines 10–12.

By Lemma 2, line 3 indeed calculates an upper bound. Also, lines 4 and 14 clearly determine lower bounds. It remains to show that the assignment operation in line 19 is correct. Note that, in line 9, new values a and d such that $\lambda = a + d$ are calculated. Note also that *UB* values are strictly decreasing along the iterations. Then line 19 defines a *UB* value which is either a new upper bound or satisfies $UB \leq LB$. We conclude that Algorithm 1 works correctly:

Theorem 3. *Algorithm 1 returns the optimum of the MCFP.* $\square$

5.2. Dinkelbach's procedure for the MCFP

Another exact iterative method is Dinkelbach's algorithm (Dinkelbach, 1967), which solves nonlinear fractional programming problems by successively solving a sequence of linear problems. It will applied to the formulation described in Section 4.1 and compared with our exact methods for the MCFP in Section 6.

Table 1
Comparison of separation algorithms for 30 random instances.

Separation algorithm	Total time (s)	Geometric mean	Number of best results
Dynamic Programming	77235.28	15.92	16
CPLEX Default	131068.72	16.33	14

Table 2
Impact of the preprocessing procedure.

Instance density	Average percentage of fixed variables (%)	Average percentage of generated cuts (%)
0.2	19	51
0.4	1	11
0.6	0	0
0.8	0	0

Dinkelbach's algorithm for the problem P: $\max\{Q(x)|x \in S\}$ where $Q(x) = N(x)/D(x)$ and $D(x) > 0$ is as follows (You, Castro, & Grossmann, 2009):

1. choose $x_0 \in S$, set $q_1 = \frac{N(x_0)}{D(x_0)}$, and let $k = 1$;
2. solve the MILP problem $F(q_k) = \max\{N(x) - q_k D(x) \mid x \in S\}$, and let x_k be the optimal solution;
3. if $F(q_k) < \delta$ (tolerance), stop and return x_k as the optimal solution;
   otherwise, let $q_{k+1} = \frac{N(x_k)}{D(x_k)}$, go to Step 2, and replace k with $k+1$ and q_k with q_{k+1}.

6. Computational experiments

In this section we evaluate and compare the algorithms proposed in this work. We use the mixed linear optimization software CPLEX (IBM, 2009), which is responsible for managing the Branch-and-Cut method, including:

- choice of variables for the branch;
- execution of the separation algorithm;
- addition of cuts generated by the preprocessing procedure.

All the algorithms have been run on an Intel Core i7-2600 3.40 GHz machine with 32 GB of RAM and Arch Linux 3.3.4 operating system. The separation algorithm in Section 3.1, the preprocessing procedure in Section 3.2, and the exact iterative method for the MCFP in Section 5.1 have been implemented in C++. All the instances and solutions are available at http://www.ic.uff.br/~fabio/instances.pdf.

6.1. Experimental results for the BGEP

To the best of our knowledge, no instances for the BGEP are available in the literature. Thus, we generate random bipartite instances using the $\mathbb{G}(n_1, n_2, p)$ model, also known as binomial model, which is a particular case of the model proposed by Gilbert (1959). Each bipartite instance $G = (U, V, E)$ is generated with $n_1 = |U|$ and $n_2 = |V|$. In addition, each potential edge of E is created with probability p, independently of the other edges.

The proposed Branch-and-Cut algorithm for the BGEP was applied to 30 randomly generated instances with $p \in \{0.2, 0.4, 0.6, 0.8\}$, $10 \leq n_1 \leq 21$, and $11 \leq n_2 \leq 29$.

We first compare the default algorithm incorporated in CPLEX (the lazy constraint pool management) with the separation algorithm described in Section 3.1. We recall that the number of constraints (4) is $|U|^2|V|^2$, and preliminary tests showed that considering all of them is impractical.

Table 3
Instances of the MCFP.

Instance	Dimension	Instance	Dimension
King1982	05 × 07	Kumar1986	20 × 23
Waghodekar1984	05 × 07	Carrie1973b	20 × 35
Seifoddini1989	05 × 18	Boe1991	20 × 35
Kusiak1992	06 × 08	Chandrasekharan1989_1	24 × 40
Kusiak1987	07 × 11	Chandrasekharan1989_2	24 × 40
Boctor1991	07 × 11	Chandrasekharan1989_3-4	24 × 40
Seifoddini1986	08 × 12	Chandrasekharan1989_5	24 × 40
Chandrasekaran1986a	08 × 20	Chandrasekharan1989_6	24 × 40
Chandrasekaran1986b	08 × 20	Chandrasekharan1989_7	24 × 40
Mosier1985a	10 × 10	McCormick1972b	27 × 27
Chan1982	10 × 15	Carrie1973c	28 × 46
Askin1987	14 × 24	Kumar1987	30 × 51
Stanfel1985	14 × 24	Stanfel1985_1	30 × 50
McCormick1972a	16 × 24	Stanfel1985_2	30 × 50
Srinivasan1990	16 × 30	King1982	30 × 90
King1980	16 × 43	McCormick1972c	37 × 53
Carrie1973a	18 × 24	Chandrasekharan1987	40 × 100
Mosier1985b	20 × 20		

To avoid biasing the results towards larger instances, we evaluate these two running scenarios using the *geometric mean*. The geometric mean of a data set $\{t_1, t_2, \dots, t_n\}$ is defined as:

$$G = \sqrt[\eta]{t_1 t_2 \cdots t_\eta}. \tag{28}$$

Each scenario is applied to all the 30 random instances. Table 1 shows the results. From left to right, the columns of the table show: separation algorithm, total running time over all the 30 instances, geometric mean of the 30 computational times, and number of times each scenario achieves the best running time. Note that the separation algorithm presented in Section 3.1 clearly influences the convergence speed.

The main benefits of using the separation algorithm are a best choice of the constraints and a quicker resolution of the problem. In fact, without using the separation algorithm, (i.e., by using CPLEX' lazy constraint pool management) medium-sized instances could not be resolved.

We now analyze the impact of the preprocessing procedure described in Section 3.2. Results are presented in Table 2. Each row in the table shows results for instances generated with the same value of p. We remark that, in the $\mathbb{G}(n_1, n_2, p)$ model, the probability p is the expected edge density of a random bipartite instance (i.e., the expected number of edges is $n_1 n_2 p$). From left to right, the columns show: expected instance density, average percentage of fixed variables, and average percentage of generated cuts. (The maximum number of cuts is $n_1 n_2 (n_1 + n_2 - 2)/2$, which is achieved by an edgeless bipartite graph.)

As the instances get denser, the effectiveness of the preprocessing procedure decreases. This is due to the fact that, in a sparse instance, the probability that two vertices are at a distance at least 4 (see Theorem 1) is greater than in a dense instance.

6.2. Experimental results for the MCFP

The MCFP has been explored in the literature for many years, and several works have proposed instances for this problem. In this paper, we used 35 instances available in Gonçalves and Resende (2004), which have been used in many other papers. In Table 3, we present the instances with its dimensions.

Table 4 shows the results of the application of the exact iterative method for the MCFP presented in Section 5.1 (with cell size constraints) on the instances shown in Table 3. From left to right, the columns show: instance name; best solution value found in

Table 4
Results of the exact iterative method for the MCFP with minimum size 2 × 2 for each cell.

Instance	Literature	Optimum	$N_1^{out} + N_0^{in}$	BGEPS Optimum	Number of iterations
King1982	73.68	73.68	**5**	**5**	19
Waghodekar1984	62.50	62.50	**9**	**9**	15
Seiffodini1989	79.59	79.59	**10**	**10**	12
Kusiak1992	76.92	76.92	**5**	**5**	26
Kusiak1987	53.13	53.13	**15**	**15**	21
Boctor1991	70.37	70.37	**7**	**7**	24
Seiffodini1986	68.30	68.30	**13**	**13**	25
Chandrasekaran1986a	85.25	85.25	**9**	**9**	33
Chandrasekaran1986b	58.72	58.72	45	43	69
Mosier1985a	70.59	70.59	**10**	**10**	**10**
Chan1982	92.00	92.00	**4**	**4**	**4**
Askin1987	69.86	69.86	22	21	28
Stanfel1985	69.33	69.33	23	22	28
McCornick1972a	51.96	51.96	49	46	83
Srinivasan1990	67.83	67.83	**46**	**46**	75
King1980	55.90	**56.52**	70	68	103
Carrie1973a	54.46	54.46	51	47	75
Mosier1985b	42.96				
Kumar1986	49.65	49.65	71	64	115
Carrie1973b	76.54	76.54	**37**	**37**	42
Boe1991	58.15	58.15	77	73	116
Chandrasekharan1989_1	100.00	100.00	**0**	**0**	**0**
Chandrasekharan1989_2	85.11	85.11	**21**	**21**	52
Chandrasekharan1989_3-4	73.51	73.51	**39**	**39**	67
Chandrasekharan1989_5	51.97	51.97	73	70	127
Chandrasekharan1989_6	47.37				
Chandrasekharan1989_7	44.87				
McCormick1972b	54.27				
Carrie1973c	46.06				
Kumar1987	58.58*	**58.94**	62	60	93
Stanfel1985_1	59.66*	59.65	**71**	**71**	105
Stanfel1985_2	50.51				
King1982	42.64				
McCormick1972c	59.85				
Chandrasekharan1987	84.03	84.03	**73**	**73**	106

Table 5
Results of the exact iterative method for the MCFP with no cell size constraints.

Instance	Literature	Optimum	$N_1^{out}+N_0^{in}$	BGEPS Optimum	Number of iterations
King1982	75*	75	**4**	**4**	19
Waghodekar1984	69.57	69.56	**7**	**7**	12
Seiffodini1989	80.85	80.85	**9**	**9**	12
Kusiak1992	79.17	79.17	**5**	**5**	6
Kusiak1987	60.87	60.87	**9**	**9**	16
Boctor1991	70.83	70.83	**7**	**7**	24
Seiffodini1986	69.44	69.44	**11**	**11**	17
Chandrasekaran1986a	85.25	85.25	**9**	**9**	31
Chandrasekaran1986b	58.72	58.72	45	43	65
Mosier1985a	75	75	**7**	**7**	8
Chan1982	92	92	**4**	**4**	**4**
Askin1987	74.24	74.24	**17**	**17**	22
Stanfel1985	72.86	72.86	19	18	23
McCornick1972a	53.33	53.33	**42**	**42**	86
Srinivasan1990	69.92	69.92	40	39	61
King1980	57.96	**58.04**	60	59	102
Carrie1973a	57.73	57.73	41	40	65
Mosier1985b	43.97				
Kumar1986	50.81	50.81	61	60	114
Carrie1973b	79.38	79.38	33	32	36
Boe1991	58.79	58.79	75	70	114
Chandrasekharan1989_1	100	100	**0**	**0**	**0**
Chandrasekharan1989_2	85.11	85.11	**21**	**21**	52
Chandrasekharan1989_3-4	73.51	73.51	40	39	67
Chandrasekharan1989_5	53.29	53.29	71	64	117
Chandrasekharan1989_6	48.95				
Chandrasekharan1989_7	46.58				
McCormick1972b	54.82				
Carrie1973c	47.68				
Kumar1987	62.86*	**63.04**	**51**	**51**	76
Stanfel1985_1	59.66*	**59.77**	70	67	104
Stanfel1985_2	50.83				
King1982	47.93				
McCormick1972c	61.16				
Chandrasekharan1987	84.03	84.03	**73**	**73**	106

the literature (Gonçalves & Resende, 2004; Wu, Chang, & Chung, 2008, 2010); optimum value found by our iterative method (using the grouping efficacy as the objective function); the sum $N_1^{out}+N_0^{in}$ corresponding to the optimum of the MCFP; the number of edge editing operations obtained by the application of our Branch-and-Cut method presented in Section 3 on the corresponding BGEPS instance; and the number of iterations executed by the iterative method. Blank entries mean that we could not find the optimum in such cases. We remark that instances marked with $*$ have been erroneously encoded in some works (probably due a typing error).

Our method finds the optimum for 27 of the 35 instances. Some of these optima are higher than previously reported results. For instances King1980 and Kumar1987, the optimal solutions found are better than the best known solutions reported in the literature. A point to note is that the values in the third and fourth columns are very close; the difference is only 1.81 on average. In particular, for 17 of the 27 optima in the fourth column, these values coincide. This shows that optimal solutions for the BGEPS correspond to quite good solutions for the MCFP.

Table 5 is similar to Table 4 and shows the results for the exact iterative method for the MCFP with no cell size constraints. Values in the second column are taken from (Elbenani, Ferland, & Bellemare, 2012; Pailla et al., 2010; Wu, Chang, & Y., 2009). As in the previous table, instances marked with $*$ have been erroneously encoded in some works.

Again, our method finds the optimum for 27 of the 35 instances. For three instances (King1980, Kumar1987, and Stanfel1985_1), the optimal solutions found are better than the best known solutions available in the literature. The difference between values in the third and fourth columns is only 1.13 on average (the difference is zero in 16 cases).

Table 6 compares the running times of the exact approaches for the MCFP described in this work (with no cell size constraints): Dinkelbach's algorithm (applied to the model of Section 4.1), the standard linearization model (Section 4.1.1), the new ILP model (Section 4.2), and the new iterative method (Section 5.1).

In the standard linearization model, two executions have been performed: the first uses the bounds obtained by the ILP model, and the second runs with no bounds. The two corresponding running times are shown in the column named 'Std. Linearization'.

The ILP model and the iterative method obtained the largest number of optimal solutions. On the other hand, Dinkelbach's algorithm achieves running times that are faster than those achieved by the others methods, with few exceptions. Note that Dinkelbach's algorithm had a good performance in some cases, but it failed to solve two instances the other approaches were able to solve to optimality. The standard linearization model had a good performance only using the bounds provided by the new ILP model; without using such bounds, it failed to solve six instances the other methods were able to solve. The addition of cell size constraints produces similar results.

7. Conclusions

This work has investigated the close relationships between the BGEP and the MCFP. This opens new possibilities of research, in the sense that any contributions to one of the problems may be applied to the other.

We have proposed a new Branch-and-Cut method for the BGEP based on a specific separation algorithm. Our method has been applied to 30 randomly generated instances with edge densities ranging from 0.2 to 0.8 and sizes from 110 to 560 edges.

Table 6
Comparison between running times (in seconds) of the exact methods for the MCFP. The symbol ' × ' means that the method was not able to find the optimum.

Instance	Dinkelbach's Alg.	Std. Linearization	ILP Model	Iterative Method
King1982	0.02	0.00–0.01	0.01	0.16
Waghodekar1984	0.01	0.00–0.01	0.01	0.07
Seifoddini1989	0.02	0.00–0.04	0.03	0.09
Kusiak1992	0.01	0.00–0.01	0.01	0.02
Boctor1991	0.01	0.01–0.03	0.01	0.14
Kusiak1987	0.04	0.05–0.13	0.06	0.29
Seifoddini1986	0.04	0.01–0.08	0.03	0.18
Chandrasekharan1986a	0.06	0.01–0.04	0.04	2.06
Chandrasekharan1986b	5.28	3.40–8455.81	4.94	81.46
Mosier1985a	0.04	0.01–0.03	0.01	0.03
Chan1982	0.05	0.01–0.01	0.02	0.01
Askin1987	0.38	0.05–0.24	0.09	0.49
Stanfel1985	0.4	0.06–0.48	0.11	0.49
McCormick1972	68.13	15.70–×	144.91	600.98
Srinivasan1990	1.17	0.47–12.08	0.54	7.24
King1980	137.63	16.63–×	125.62	1156.23
Carrie1973	12.44	7.26–23412.72	42.32	87.13
Mosier1985b	×	× – ×	×	×
Kumar1986	4391.3	349.29– ×	1771.99	23928.70
Boe1991	146.08	44.58– ×	305.48	2145.24
Carrie1973	1.72	0.75–0.38	14.55	1.78
Chandrasekharan1989_1	2.41	0.00–0.27	0.15	0.02
Chandrasekharan1989_2	2.45	0.33–0.46	0.44	10.08
Chandrasekharan1989_3-4	2.74	2.29–6.47	0.78	17.46
Chandrasekharan1989_5	×	× – ×	48743.90	371233.00
Chandrasekharan1989_6	×	× – ×	×	×
Chandrasekharan1989_7	×	× – ×	×	×
McCormick1972	×	× – ×	×	×
Carrie1973	×	× – ×	×	×
Kumar1987	14.09	4.06– ×	41.53	183.71
Stanfel1985_1	146.65	51.55–×	2622.06	13807.50
Stanfel1985_2	×	× – ×	×	×
King1982	×	× – ×	×	×
McCormick1972	×	× – ×	×	×
Chandrasekharan1987	×	8.03–14.19	18.22	325.53

Table 7
Problems that allow for a direct adaptation of the BGEP model.

Characteristic	Adaptation
a - Cost/operation time	Modify the weights of the edges
b - Capacity of the cells or parts	Impose lower and upper bounds on the size of the bicliques
c - Machine capacity and demand for a product	Modify the weights of the vertices
d - Machine proximity constraints	Given a minimum distance between each pair of vertices, constraints must be added to forbid the pairs of vertices that exceed such distance to be assigned to the same biclique

Experimental results show that this method outperforms the CPLEX standard algorithm.

We have also described a new preprocessing procedure for the BGEP based on theoretical developments related to vertex distances in the input graph (Theorem 1). The procedure is effective to fix variables and generate cuts for low density random instances.

The similarity between the BGEP and MCFP has been explored. Since these problems have the same feasible solution space, mathematical formulations for the BGEP can be used to solve the MCFP. It is worth remarking that the combinatorial structure of the BGEP can be directly applied to the solution of the MCFP. An example is the use of constraints (4) in two formulations for the MCFP described in this work; such constraints are used to eliminate induced subgraphs isomorphic to P_4, which are forbidden for bicluster graphs.

We have shown that good solutions of the BGEP correspond to good solutions of the MCFP, i.e., decreasing the number of editing operations corresponds to obtaining a matrix permutation with fewer 1's outside and 0's inside diagonal blocks.

Our contributions to the MCFP include:

- a new exact iterative method based on repeated calls to a parameterized version of the BGEP;
- a new ILP formulation;
- an experimental study of the impact of using a linear objective function for the MCFP;
- a separated analysis of problems with or without cell size constraints;
- exact resolution of most instances of the MCFP found in literature, some created almost 40 years ago.

At the time of revision of this paper, Bychkov, Batsyn, and Pardalos (2014) independently proved 14 of the 35 instances from the literature using a different formulation and objective function.

7.1. Solving variants of the MCF

Ongoing work includes the study of other variants of the MCFP and their correspondence with adapted versions of the BGEP.

In the design of manufacturing cell formation several production characteristics can be considered at the same time. Defersha and Chen (2006) discusses a number of such characteristics that can be found in the literature. In the following we show how our BGEP based algorithm for solving the MCFP can be extended to solve other variants of the problem. In fact, we can classify the problems according to the following characteristics:

Table 8
Problems that allow for a partial solution of the problem.

Characteristic	Adaptation
a - Workers and tools, e.g., allocation of operators, costs of tools, types of tools	First, we solve the problem associated with the workers/tools, that is, decide which operators/tools are going to be assigned to each machine, and then solve the MCFP itself using the result of such assignment as an input for the BGEP algorithm
b - Multiple planning processes, e.g., each part has several possible sequences of operations	We solve the problem of choosing a sequence of operations for each part, and then we use the BGEP algorithm to minimize moves of parts in these operations
c - Problems with factors that need extra structures, e.g. arrangement of machinery (space problem), load balancing and division of production lots (flow of operations between machines)	In this case, the resolution needs to be integrated (i.e., we cannot separate it in two subproblems); this prevents the use of the BGEP

1. Problems that allow for a direct adaptation of the BGEP model, in which only minor modifications have to be performed in our algorithm, as shown in Table 7.

2. Problems that allow for a partial solution. In this case, we first solve a subproblem considering the new production characteristic and then use its solution as an input to the BGEP algorithm, or vice-versa, as shown in Table 8. We remark that in some cases the optimal solution found by this two-phase method is not necessarily the one associated with the original problem.

References

Abdullah, A., & Hussain, A. (2006). A new biclustering technique based on crossing minimization. *Neurocomputing, 69*(1618), 1882–1896. doi:10.1016/j.neucom.2006.02.018.

Amit, N. (2004). *The bicluster graph editing problem*. Master's thesis. Tel Aviv University.

Bellman, R. (1962). Dynamic programming treatment of the travelling salesman problem. *J. ACM, 9*(1), 61–63. doi:10.1145/321105.321111.

Bisson, G., & Hussain, F. (2008). Chi-sim: A new similarity measure for the co-clustering task. In *Proceedings of the 2008 seventh international conference on machine learning and applications*. In *ICMLA '08* (pp. 211–217). IEEE Computer Society. doi:10.1109/ICMLA.2008.103.

Boutsinas, B. (2013). Machine-part cell formation using biclustering. *European Journal of Operational Reseach, 230*, 563–572.

Burbidge, J. L. (1971). Production flow analysis. *Production Engineer, 50*(4/5), 139–152.

Bychkov, I., Batsyn, M., & Pardalos, P. (2014). Exact model for the cell formation problem. *Optimization Letters, 8*(8), 2203–2210. doi:10.1007/s11590-014-0728-8.

Chandrasekharan, M. P., & Rajagopalan, R. (1986). Modroc: An extension of rank order clustering for group technology. *International Journal of Production Reseach, 24*(5), 1221–1233. doi:10.1080/00207548608919798.

Chandrasekharan, M. P., & Rajagopalan, R. (1987). Zodiac - an algorithm for concurrent formation of part-families and machine-cells. *International Journal of Production Research, 25*(6), 835–850. doi:10.1080/00207548708919880.

Cheng, Y., & Church, G. M. (2000). Biclustering of expression data. In *Proceedings of the eighth international conference on intelligent systems for molecular biology* (pp. 93–103). AAAI Press.

Defersha, F. M., & Chen, M. (2006). A comprehensive mathematical model for the design of cellular manufacturing systems. *International Journal of Production Economics, 103*(2), 767–783. doi:10.1016/j.ijpe.2005.10.008.

Dinkelbach, W. (1967). On nonlinear fractional programming. *Management Science, 13*(7), 492–498.

Elbenani, B., Ferland, J. A., & Bellemare, J. (2012). Genetic algorithm and large neighbourhood search to solve the cell formation problem. *Expert Systems with Applications, 39*(3), 2408–2414. doi:10.1016/j.eswa.2011.08.089.

Faure, N., Chretienne, P., Gourdin, E., & Sourd, F. (2007). Biclique completion problems for multicast network design. *Discrete Optimization, 4*(3-4), 360–377. doi:10.1016/j.disopt.2007.09.005.

Flanders, R. E. (1924). Design, manufature and production control of a standard machine. *Transactions of the American Society of Mechanical Enginneers, 46*, 691–738.

Gilbert, E. (1959). Random graphs. *Annals of Mathematical Statistics, 30*, 1141–1144. doi:10.1214/aoms/1177706098.

Glover, F. (1975). Improved linear integer programming formulations of nonlinear integer problems. *Management Science, 22*(4), 455–460.

Goldengorin, B., Krushinsky, D., & Pardalos, P. M. (2013). *Cell formation in industrial engineering: Theory, algorithms and experiments*. Springer.

Gonçalves, J., & Resende, M. (2004). An evolutionary algorithm for manufacturing cell formation. *Computers & Industrial Engineering, 47*(2-3), 247–273. doi:10.1016/j.cie.2004.07.003.

Guo, J., Hffner, F., Komusiewicz, C., & Zhang, Y. (2008). Improved algorithms for bicluster editing. In *Proceedings of 5th international conference on theory and applications of models of computation: vol. 4978* (pp. 445–456). Springer Berlin / Heidelberg. doi:10.1007/978-3-540-79228-4_39.

Gupta, A., & Palit, A. (1979). Clique generation using boolean equations. *Proceedings of The IEEE, 67*(1), 178–180.

Hartigan, J. A. (1975). *Clustering algorithms*. John Wiley & Sons.

Hsu, C. P. (1990). *Similarity coefficient approaches to machine component cell Formation in cellular manufacturing: A comparative study*. Ph.D. thesis. University of Wisconsin.

IBM (2009). *Ibm ilog cplex v12.1 user's manual for cplex*. IBM.

Kumar, C. S., & Chandrasekharan, M. P. (1990). Group efficacy: a quantitative criterion for goodness of block diagonal forms of binary matrices in group technology. *International Journal of Production Research, 28*(2), 233–243. doi:10.1080/00207549008942706.

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.

Mitrofanov, S. P. (1966). *The scientific principles of group technology*. National Lending Library for Science and Technology.

Pailla, A., Trindade, A. R., Parada, V., & Ochi, L. S. (2010). A numerical comparison between simulated annealing and evolutionary approaches to the cell formation problem. *Expert Systems With Applications, 37*, 5476–5483. doi:10.1016/j.eswa.2010.02.064.

Protti, F., Dantas da Silva, M., & Szwarcfiter, J. L. (2006). Applying modular decomposition to parameterized bicluster editing. In *Parameterized and exact computation*. In *Lecture Notes in Computer Science: vol. 4169* (pp. 1–12). Springer Berlin / Heidelberg. doi:10.1007/11847250_1.

Sarker, B., & Khan, M. (2001). A comparison of existing grouping efficiency measures and a new weighted grouping efficiency measure. *IIE Transactions, 33*, 11–27. doi:10.1023/A:1007633622787.

Sousa Filho, G. F., dos Anjos F. Cabral, L., Ochi, L. S., & Protti, F. (2012). Hybrid metaheuristic for bicluster editing problem. *Electronic Notes in Discrete Mathematics, 39*(0), 35–42. doi:10.1016/j.endm.2012.10.006.

Srinivasan, G., & Narendran, T. T. (1991). Grafics - a nonhierarchical clustering-algorithm for group technology. *International Journal of Production Research, 29*(3), 463–478. doi:10.1080/00207549108930083.

Wolsey, L. (1998). Integer programming. *Wiley Series in Discrete Mathematics and Optimization*. Wiley.

Wu, T. H., Chang, C. C., & Chung, S. H. (2008). A simulated annealing algorithm for manufacturing cell formation problems. *Expert Systems with Applications, 34*(3), 1609–1617. doi:10.1016/j.eswa.2007.01.012.

Wu, T. H., Chang, C. C., & Y. , Y. J. (2009). A hybrid heuristic algorithm adopting both Boltzmann function and mutation operator for manufacturing cell formation problems. *International Journal of Production Economics, 120*(2), 669–688. doi:10.1016/j.ijpe.2009.04.015.

Wu, T. H., Chung, S. H., & Chang, C. C. (2010). A water flow-like algorithm for manufacturing cell formation problems. *European Journal of Operational Research, 205*(2), 346–360. doi:10.1016/j.ejor.2010.01.020.

You, F., Castro, P. M., & Grossmann, I. E. (2009). Dinkelbach's algorithm as an efficient method to solve a class of MINLP models for large-scale cyclic scheduling problems. *Computers & Chemical Engineering, 33*(11), 1879–1889. doi:10.1016/j.compchemeng.2009.05.014.