# On Solving Manufacturing Cell Formation via Bicluster Editing

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

[a]*Fluminense Federal University*
*Niterói, RJ - Brazil*
[b]*Federal University of Paraíba*
*João Pessoa, PB - Brazil*

**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 that consists of a Branch-and-Cut approach combined with a special separation algorithm based on dynamic programming. 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. Other contribution of this work is to reveal the similarities between the BGEP and the MCFP. We show that the BGEP and the MCFP have the same solution space. This fact leads to the proposal of two new exact approaches for the MCFP 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 consists of iteratively running several calls to a parameterized version of the BGEP, and the second is a linearization of a new fractional-linear model for the MCFP. 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.

*Keywords:* Biclusterization, Manufacturing Cell Formation, Graph Partitioning

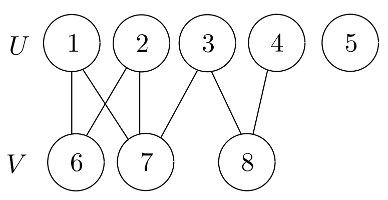
## 1. Introduction

The Bicluster Graph Editing Problem (BGEP) is described as follows: given a bipartite graph $G = (U, V, E)$, where $U$ and $V$ are non-empty stable sets of vertices and $E$ is a set of edges linking vertices in $U$ to 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 $U$ to a vertex in $V$.

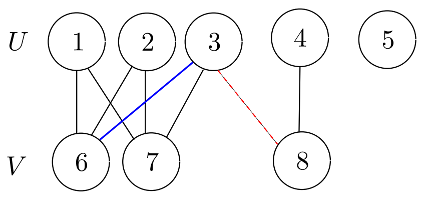
In a bipartite graph $G$, a *bicluster* is a subgraph of $G$ isomorphic to a biclique. The existence of biclusters indicates a high degree of similarity between the data (vertices). In particular, a perfectly clustered bipartite graph is called a *bicluster graph*, i.e., a bipartite graph in which each of its connected components is a biclique. Hence, we can alternatively define the goal of the BGEP, as stated by Amit [1], as follows: "find a minimum number of edge editing operations in order to transform an input bipartite graph into a bicluster graph".

Figure 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,

*Corresponding author. Tel. +55 83 3216-7549; Fax +55 83 3216-7179.
*Email addresses:* `rian.gabriel@ic.uff.com` (Rian G. S. Pinheiro), `imartins@ic.uff.br` (Ivan C. Martins), `fabio@ic.uff.br` (Fábio Protti), `satoru@ic.uff.br` (Luiz S. Ochi), `luidi@ic.uff.br` (Luidi G. Simonetti), `anand@ct.ufpb.br` (Anand Subramanian)



(a) Instance.



(b) Solution.

Figure 1: BGEP Example.

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 Figure 1).

A problem similar to the BGEP is the Cluster Graph Editing Problem (CGEP), first studied by Gupta and Palit [2]. Its goal is to transform $G$ into a disjoint union of complete graphs (cliques). The CGEP and the BGEP are important examples of partition problems in graphs.

The concept of biclustering was introduced in the mid-70s by Hartigan [3], but its first use appeared in a paper by Cheng and Church [4], within the context of Computational Biology. Since then, algorithms for biclustering have been proposed and used in various applications, such as multicast network design [5] and analysis of biological data [6, 7].

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 and columns represent graph partitions; 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 [1] 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. Protti et al. [8] proposed an algorithm for the parameterized version of the BGEP that uses a strategy based on modular decomposition techniques. Guo et al. [9] developed a randomized 4-approximation algorithm for the BGEP. More recently, Sousa Filho et al. [10] proposed a GRASP-based heuristic for the BGEP.

Other important application of the BGEP, introduced in this work, is related to the Manufacturing Cell Formation Problem (MCFP). We show that such problems have a high degree of similarity, and that good solutions for the BGEP are close to good solutions for the MCFP. Cellular manufacturing is an application of the Group Technology concept. The goal is to identify and cluster similar parts in order to optimize the manufacturing process. Such a concept was originally proposed by Flanders [12] and formally described by Mitrofanov [13] in 1966. In the early 70s, Burbidge [14] proposed one of the first techniques for creating a system of cellular manufacturing. Since this work, several approaches have been proposed to the MCFP, whose goal is to create the cells in order to optimize the manufacturing process, as described in Section 3.

Our contributions can be summarized as follows. In Section 2, we develop an exact method for the BGEP consisting of a Branch-and-Cut approach combined with a special separation algorithm based on dynamic programming, and we describe a new preprocessing procedure for the BGEP derived from theoretical results on vertex distances in the input graph. In Section 3, we explore the similarity between the BGEP and the MCFP. We show that the BGEP and the MCFP have the same solution space, and due to this fact we propose two new exact approaches for the MCFP 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 (Section 3.3) consists of iteratively running several calls to a parameterized version of the BGEP, and the second (Section 3.4) is a linearization of a new fractional-linear model for the MCFP. In Section 4, we apply our Branch-and-Cut method for the BGEP to randomly generated BGEP 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. Our exact methods for the MCFP are able to prove several previously unknown optima. Section 5 contains our conclusions.

## 2. Branch-and-Cut Approach for the BGEP

A mathematical model for the BGEP is described in Amit [1]. It relies on the simple fact that the graph $P_4$ (a path with four vertices, shown in Figure 2) is a forbidden induced subgraph for a bicluster graph. More precisely, for a bipartite graph $G$, $G$ is a bicluster graph if and only $G$ does not contain $P_4$ as an induced subgraph.

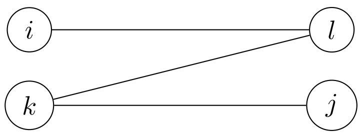


Figure 2: Graph $P_4$.

The formulation proposed in Amit [1] is as follows:

$$\min \quad \sum_{+(ij)} (1 - y_{ij}) + \sum_{-(ij)} y_{ij} \tag{1}$$

$$\text{s.t.} \quad y_{il} + y_{kj} + y_{kl} \leq 2 + y_{ij} \qquad \forall\ i \neq k \in U \text{ and } j \neq l \in V \tag{2}$$

$$y_{ij} \in \{0, 1\} \qquad \forall\ i \in U \text{ and } j \in V \tag{3}$$

where: (a) $y_{ij}$ are binary variables such that $y_{ij} = 1$ if and only if the solution contains edge $ij$; (b) $+(ij) = \{ij \mid ij \in E\}$ is the set of edges; (c) $-(ij) = \{ij \mid ij \notin E\}$ the set of non-edges. The objective function (1) counts how many edge editing operations are made. The first and second sums represent the number of edge deletions and edge additions, respectively. Constraints (2) eliminate induced subgraphs isomorphic to $P_4$. Constraints (3) define the domain of the variables.

### *2.1. Separation Algorithm*

Note that the number of constraints (2) in the above formulation is $|U|^2|V|^2$ and therefore it is not advisable to consider all these constraints *a priori*. This becomes computationally expensive for exact methods, especially when dealing with large instances. Alternatively, we start without such constraints and we add them in a cutting plane fashion as they are violated according to the separation algorithm described below:

Algorithm 1 works with a linear relaxation as input. Its main objective is to find the *most violated constraint* (2) for each pair $(i, j)$ of vertices, and then add it to the model. The idea is to use an auxiliary complete bipartite graph $G'(U, V, E)$ where each edge $ij$ has a nonnegative weight $w_{ij}$; the weights are defined according the values $y^*_{ij}$ obtained by the linear relaxation, i.e., $w_{ij} = y^*_{ij}$. Note that an edge may have a zero weight.

After the construction of $G'$, a dynamic programming approach is used to find the constraints. It calculates the values $d^s_{ij}$, where $d^s_{ij}$ is the *maximum cost* between $i$ and $j$ considering paths with $s - 1$ internal vertices. This is explained below in detail.

In line 2, $d^1_{ij}$ is initialized with the value of the linear relaxation $y^*_{ij}$. In line 3, for each pair $(i, k)$ of vertices, the maximum cost $d^2_{ik}{}'$ between them considering paths with a single internal vertex is calculated; also, the internal vertex by which such a cost is achieved is saved in variable $l'_{ik}$. Line 4 is similar to line 3, but instead of calculating the maximum cost, it calculates the second maximum cost. Line 5 verifies, for all

**Algorithm 1** Separation algorithm

```
1: procedure Separation(relaxation $y^*$)
2:     $\forall i \in U$ and $j \in V$        $d^1_{ij} = y^*_{ij}$
3:     $\forall i, k \in U$        $d^2_{ik}{}' = \max_{l \in V}\{d^1_{il} + d^1_{kl}\}$        $l'_{ik} = \arg\max_{l \in V}\{d^1_{il} + d^1_{kl}\}$
4:     $\forall i, k \in U$        $d^2_{ik}{}'' = \max_{l \in V \setminus \{l'_{ik}\}}\{d^1_{il} + d^1_{kl}\}$        $l''_{ik} = \arg\max_{l \in V \setminus \{l'_{ik}\}}\{d^1_{il} + d^1_{kl}\}$
5:     $\forall i, k \in U\ (i \neq k)$ and $j \in V$        $d^2_{ikj} = \begin{cases} d^2_{ik}{}', & \text{if } j \neq l'_{ik} \\ d^2_{ik}{}'', & \text{if } j = l'_{ik} \end{cases}$        $l_{ik} = \begin{cases} l'_{ik}, & \text{if } j \neq l'_{ik} \\ l''_{ik}, & \text{if } j = l'_{ik} \end{cases}$
6:     $\forall i \in U$ and $j \in V$        $d^3_{ij} = \max_{k \in U \setminus \{i\}}\{d^2_{ikj} + d^1_{kj}\}$        $k_{ij} = \arg\max_{k \in U \setminus \{i\}}\{d^2_{ikj} + d^1_{kj}\}$
7:     $\forall i \in U$ and $j \in V$        if $d^3_{ij} - d^1_{ij} > 2$ then add cut $y_{il} + y_{kj} + y_{kl} \leq 2 + y_{ij}$
                                           ( for $k = k_{ij}$ and $l = l_{ik_{ij}}$)
8: end procedure
```

$i, k \in U$ $(i \neq k)$ and $j \in V$, if $j$ belongs to the maximum cost path, and chooses to use the maximum cost path or the second maximum cost path. In this case, $d^2_{ikj}$ represents the maximum cost between $i$ and $k$ using a path that avoids $j$, and $l_{ik}$ stores the corresponding internal vertex. Line 6 calculates the maximum cost between $i$ and $j$ using two internal vertices, and stores it in $d^3_{ij}$; it represents the "$P_4$ of maximum cost"; variable $k_{ij}$ saves the internal vertex $k$. Finally, in line 7, for each pair $(i, j)$, if the constant is violated $(d^3_{ij} - d^1_{ij} > 2)$ then the cut $y_{il} + y_{kj} + y_{kl} \leq 2 + y_{ij}$ for $k = k_{ij}$ and $l = l_{ik_{ij}}$ is added to the model.

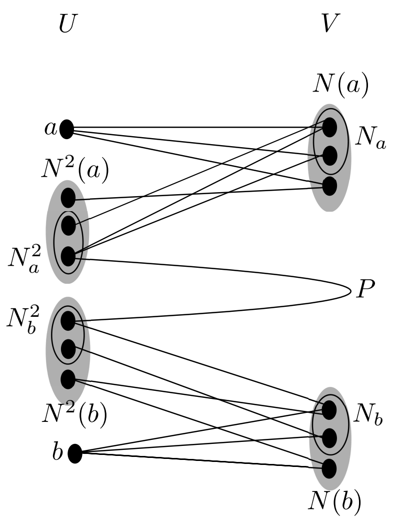
### *2.2. Preprocessing Procedure*

In this section, we propose 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.

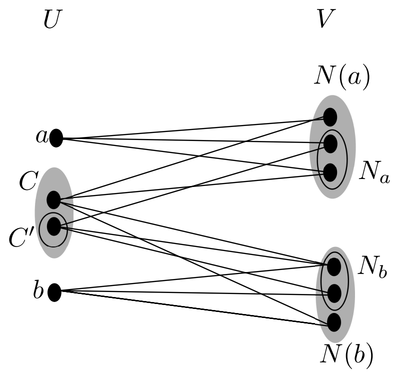
**Theorem 1.** *Let $a, b$ be vertices of a bipartite graph $G(V, U, E)$, and let $d(a, b)$ be the distance between $a$ and $b$ in $G$. If $d(a, b) \geq 4$ then there is an optimal solution in which $a, b$ belong to distinct biclusters.*

*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. The following notation is useful for the proof. Let $X \subseteq V$ and $Y \subseteq U$. Let $rem(X, Y) = |\{xy \in E \mid x \in X \text{ and } y \in Y\}|$; informally, $rem(X, Y)$ is the cost of removing all edges in $E$ between $X$ and $Y$. Also, let $add(X, Y) = |X||Y| - rem(X, Y)$, i.e., $add(X, Y)$ is the cost of adding all the missing edges between $X$ and $Y$ in order to create a bicluster $B$ with vertex set $V(B) = X \cup Y$. If $X = \{x\}$, we simply write $add(x, Y)$ and $rem(x, Y)$ instead of $add(\{x\}, Y)$ and $rem(\{x\}, Y)$, and similarly if $Y = \{y\}$. Denote by $N(a)$ the neighborhood of $a$, and let $N^2(a) = \{v \in V \cup U \mid d(a, v) = 2\}$.

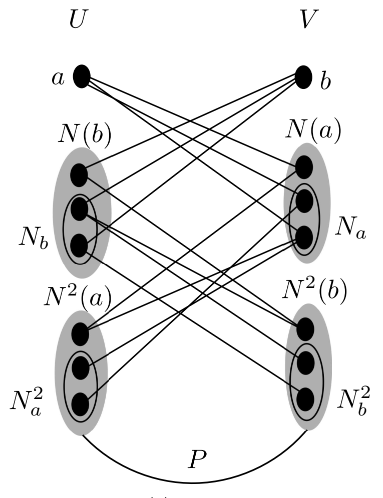
For the case $d(a, b) = \infty$, note that $a$ and $b$ lie in distinct connected components of $G$, and therefore will belong to distinct biclusters in any optimal solution. Now assume that $d(a, b) < \infty$ and there is an optimal solution $G^*$ in which vertices $a, b$ belong to a bicluster $B$ with vertex set $V(B) = X \cup Y$, where $X \subseteq V$ and $Y \subseteq U$. We analyze two cases: $a, b \in X$ (Case 1) and $a \in X, b \in Y$ (Case 2). Case 1 is divided in two sub-cases: $d(a, b) > 4$ (Case 1a) and $d(a, b) = 4$ (Case 1b).



(a) Case 1a



(b) Case 1b



(c) Case 2

Figure 3: Cases of Theorem 1

Let $N_a = N(a) \cap V(B)$, $N_b = N(b) \cap V(B)$, $N_a^2 = N^2(a) \cap V(B)$, and $N_b^2 = N^2(b) \cap V(B)$.

**Case 1a:** In this case, note that $N^2(a) \cap N^2(b) = \emptyset$. Figure 3a represents this situation, where $P$ is a path between $N^2(a)$ and $N^2(b)$. Let $X' \subseteq X$ and $Y' \subseteq Y$ defined as follows:

$$X' = \{a, b\} \cup N_a^2 \cup N_b^2 \cup (V(P) \cap U) \text{ and } Y' = N_a \cup N_b \cup (V(P) \cap V),$$

where $V(P)$ is the subset of vertices in path $P$. Since $X'$ and $Y'$ are completely connected by edges in $G^*$, the following cost $c_1$ is associated with the structure represented in Figure 3a:

$$\begin{aligned} c_1 \;=\;& rem(a, N(a) \setminus N_a) + rem(N_a, N^2(a) \setminus N_a^2) \;+\; rem(b, N(b) \setminus N_b) + rem(N_b, N^2(b) \setminus N_b^2) + \\ & add(N_a^2, N_a) + add(N_b^2, N_b) + rem(N_a^2, V \setminus Y') \;+\; rem(N_b^2, V \setminus Y') + \\ & add(a, N_b) + add(b, N_a) + add(N_a^2, N_b) + add(N_b^2, N_a) \;+ \\ & add(X', V(P) \cap V) + add(V(P) \cap U, N_a \cup N_b). \end{aligned}$$

Now consider the following subsets: $X_1' = \{a\} \cup N_a^2,\ Y_1' = N_a,\ X_2' = \{b\} \cup N_b^2,\ Y_2' = N_b$. Let $G^{**}$ be another solution with distinct biclusters $B_1$ and $B_2$ such that $X_1' \cup Y_1' \subseteq V(B_1)$ and $X_2' \cup Y_2' \subseteq V(B_2)$. The cost $c_2$ associated with this new situation is given by:

$$\begin{aligned} c_2 \;=\;& rem(a, N(a) \setminus N_a) + rem(N_a, N^2(a) \setminus N_a^2) \;+\; rem(b, N(b) \setminus N_b) + rem(N_b, N^2(b) \setminus N_b^2) + \\ & add(N_a^2, N_a) + add(N_b^2, N_b) + rem(N_a^2, V \setminus Y') \;+\; rem(N_b^2, V \setminus Y'). \end{aligned}$$

Since $c_1 \geq c_2$, the cost of keeping $a$ and $b$ in distinct biclusters is not greater than the cost of keeping them in the same bicluster.

**Case 1b:** If $d(a, b) = 4$ then $a$ and $b$ belong to the same part, say $U$, and $N^2(a) \cap N^2(b) = C \neq \emptyset$. Let $C' = C \cap V(B)$. Figure 3b depicts the subsets involved in Case 1b.

Let $X' \subseteq X$ and $Y' \subseteq Y$ defined as $X' = \{a, b\} \cup C'$ and $Y' = N_a \cup N_b$. Again, since $X'$ and $Y'$ are completely connected by edges in $G^*$, the cost $c_1$ associated with the structure in Figure 3b is given by:

$$\begin{aligned} c_1 \;=\;& add(b, N_a) + add(a, N_b) + rem(N_a, N^2(a) \setminus C') + rem(N_b, N^2(b) \setminus C') + add(C', N_a) + add(C', N_b) + \\ & rem(a, N(a) \setminus N_a) + rem(b, N(b) \setminus N_b) + rem(C', V \setminus (N_a \cup N_b)). \end{aligned}$$

Assume without loss of generality that $|N_b| \leq |N_a|$. Consider now the subsets $X_1' = \{a\} \cup C',\ Y_1' = N_a \cup N_b,\ X_2' = \{b\},\ Y_2' = \emptyset$. If $G^{**}$ is another solution with distinct biclusters $B_1$ and $B_2$ such that $X_1' \cup Y_1' \subseteq V(B_1)$ and $X_2' \cup Y_2' \subseteq V(B_2)$, the cost $c_2$ associated with $G^{**}$ is:

$$\begin{aligned} c_2 \;=\;& add(a, N_b) + rem(N_a, N^2(a) \setminus C') + rem(N_b, N^2(b) \setminus C') + add(C', N_a) + add(C', N_b) + \\ & rem(a, N(a) \setminus N_a) + rem(C', V \setminus (N_a \cup N_b)) + rem(b, N(b)). \end{aligned}$$

The difference of the costs is given by:

$$\begin{aligned} c_1 - c_2 \;&=\; add(b, N_a) + rem(b, N(b) \setminus N_b) - rem(b, N(b)) \\ &=\; |N_a| + |N(b)| - |N_b| - |N(b)| = |N_a| - |N_b|. \end{aligned}$$

Therefore $c_1 - c_2 \geq 0$, i.e., keeping $a$ and $b$ in distinct biclusters is not more costly.

**Case 2:** In this case, $a$ and $b$ belong to different parts. Assume $a \in U$ and $b \in V$, as shown in Figure 3c. Let $X' \subseteq X$ and $Y' \subseteq Y$ defined as $X' = \{a\} \cup N_b \cup N_a^2$ and $Y' = \{b\} \cup N_a \cup N_b^2$. The cost $c_1$ corresponding to the biclique with vertex set $X' \cup Y'$ is:

$$\begin{aligned} c_1 \;=\;& add(a, b) + add(a, N_b^2) + add(b, N_a^2) + add(N_a, N_b) + add(N_a, N_a^2) + add(N_b, N_b^2) + add(N_a^2, N_b^2) + \\ & add(X', V(P) \cap V) + add(V(P) \cap U, Y') + rem(a, N(a) \setminus N_a) + rem(b, N(b) \setminus N_b) + \\ & rem(N_a^2, V \setminus N_a) + rem(N_b^2, U \setminus N_b) + rem(N_a, N^2(a) \setminus N_a^2) + rem(N_b, N^2(b) \setminus N_b^2). \end{aligned}$$

Now let $X'_1 = \{a\} \cup N_a^2,\ Y'_1 = N_a,\ X'_2 = N_b,\ Y'_2 = \{b\}$. Also, let $G^{**}$ be another solution with distinct biclusters $B_1$ and $B_2$ such that $X'_1 \cup Y'_1 \subseteq V(B_1)$ and $X'_2 \cup Y'_2 \subseteq V(B_2)$. The cost $c_2$ associated with $G^{**}$ is:

$$\begin{aligned} c_2 \quad = \quad & rem(a, N(a) \setminus N_a) + rem(b, N(b) \setminus N_b) + rem(N_a, N^2(a) \setminus N_a^2) + rem(N_b, N^2(b) \setminus N_b^2) + \\ & rem(N_a^2, V \setminus N_a) + rem(N_b^2, U \setminus N_b) + add(N_a, N_a^2) + add(N_b, N_b^2). \end{aligned}$$

Again, the separation into two biclusters is not more costly.

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

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 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 $k \in V$.

### 2.3. Instances for the BGEP

To the best of our knowledge, no public sites contain instances for the BGEP. Thus, to evaluate the algorithms proposed in this work, we create random bipartite instances using the $\mathbb{G}(m,n,p)$ model, also known as binomial model, which is a particular case of the model proposed by Gilbert [11]. A bipartite instance $G(U,V,E)$ is created such that $|U| = m$, $|V| = n$, and each potential edge of $E$ is created with probability $p$, independently of the other edges.

## 3. Application to Manufacturing Cell Formation

The input of the MCFP is given as a binary product-machine matrix where each entry $(i,j)$ has value 1 if product $j$ is manufactured by machine $i$, and 0 otherwise. Any feasible solution of the MCFP consists of a collection of product-machine cells, where every product (or machine) is allocated to exactly one cell. Hence, in each cell $C$, machines allocated to it are exclusively dedicated to manufacture products in $C$. In an ideal solution of the MCFP, in each cell there must be a high similarity between the products and machines allocated to it. Figure 4 shows an example of the MCFP solved as a block diagonalization problem. Note that a solution for the MCFP is obtained by a permutation of rows/columns of the input matrix together with a cell assignment for products and machines. In Figure 4, 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.

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

Figure 4: MCFP example.

Among several measures of performance used as objective functions for the MCFP, the most used in literature is the *grouping efficacy* $\mu$, defined as:

$$\mu = \frac{N_1 - N_1^{out}}{N_1 + N_0^{in}}, \tag{4}$$

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, inside) diagonal blocks in the solution matrix.

### *3.1. The size of the cells*

Some works define a minimum value for the size of the cells. For instance, in [15, 16, 17], 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. Others studies do not consider any size constraint, allowing the existence of empty cells, such as the work by Pailla et al. [18]. An example of a solution with an empty cell is shown in Figure 5.

This work deals with two versions of the MCFP found in the literature:

1. unrestricted version, allowing singletons and empty cells;
2. with restrictions (minimum size $2 \times 2$ for each cell).

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

Figure 5: Example with a singleton and an empty cell.

### *3.2. Similarity between the BGEP and the MCFP*

Given an input for the MCFP, we can define an input $G$ for 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 the entry $(i, j)$ of the input matrix for the MCFP has value 1. In addition, a solution for the BGEP with input $G$ can be transformed into a cell assignment for machines/products.

The two problems (the BGEP and the MCFP) are very similar, but a point to note is that biclusters have no size limitation. Thus, we define a new BGEP variant, the Bicluster Graph Editing Problem with Size Restriction (BGEPS), to make a more precise correspondence with the MCFP. Informally, the BGEPS is defined by adding size restrictions to the BGEP: every bicluster in $G$ must now have at least $s_c$ vertices in $U$ and $s_r$ vertices in $V$. In the translation from the BGEPS to the MCFP, $s_r$ is the minimum cell size for rows and $s_c$ the minimum cell size for columns.

We propose the following formulation for the BGEPS:

$$\min \quad \sum_{+(ij)} (1 - y_{ij}) + \sum_{-(ij)} y_{ij} \tag{5}$$

$$st \quad y_{il} + y_{kj} + y_{kl} \leq 2 + y_{ij} \qquad \forall i, k \in U \textit{ and } j, l \in V \tag{6}$$

$$\sum_{j \in V} y_{ij} \geq s_r \qquad \forall i \in U \tag{7}$$

$$\sum_{i \in U} y_{ij} \geq s_c \qquad \forall j \in V \tag{8}$$

$$y_{ij} \in \{0, 1\} \qquad \forall i \in U \textit{ and } j \in V. \tag{9}$$

Lemma 2 makes the correspondence between the BGEPS and the MCFP.

**Lemma 2.** *There is a one-to-one correspondence from the feasible solution set of the BGEPS to the feasible solution set of the MCFP. In addition, for every pair of corresponding feasible solutions, the sizes of biclusters and cells are preserved.*

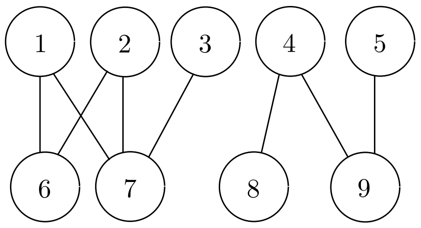
*Proof.* Consider an instance of the BGEPS consisting of a bipartite graph $G$ with parts $U$ and $V$. It is easy to see that it corresponds to an instance $M$ of the MCFP: Figures 6a and 6c show an example of a bipartite graph $G$ and a corresponding product-machine matrix $M$. Let $\mathcal{B}$ and $\mathcal{C}$ be the feasible solution sets of the BGEPS and the MCFP, respectively, associated with $G$ and $M$.

Let $f : \mathcal{B} \to \mathcal{C}$ be the transformation of a solution $G'$ in $\mathcal{B}$ to a solution $M'$ in $\mathcal{C}$ defined as follows. 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 $M'$, as shown in Figures 6b and 6d. Thus, each solution in $\mathcal{B}$ is uniquely mapped into one solution in $\mathcal{C}$.

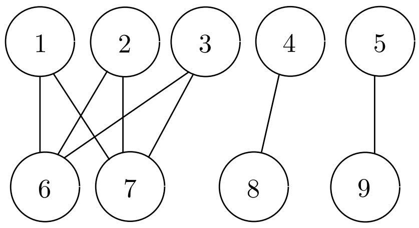
Similarly, let $g : \mathcal{C} \to \mathcal{B}$ be the inverse transformation of $f$ that maps each solution $M'$ in $\mathcal{C}$ into a solution $G'$ in $\mathcal{B}$, as follows: if a product $i$ and a machine $j$ are in the same cell then the corresponding vertices $i$ and $j$ belong to the same bicluster in $G'$.

Since $f$ and $g$ are injective functions, there is a one-to-one correspondence between $\mathcal{B}$ and $\mathcal{C}$. Moreover, if $G'$ and $M'$ are corresponding feasible solutions, it is easy to see that a bicluster $B$ in $G'$ corresponds to a cell containing exactly $|V(B) \cap U|$ products, $|V(B) \cap V|$ machines, and $|E(B)|$ entries; also, a cell with $pm$ entries in $M'$ corresponds to a bicluster with exactly $p + m$ vertices and $pm$ edges. $\square$

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, Figure 6b shows an optimal solution of the BGEP, but in Figure 6d 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 funtion (4)). In Figure 6a, 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.



(a) BGEPS instance.



(b) BGEPS solution.

$$\begin{array}{c|cccc|}
 & 6 & 7 & 8 & 9 \\
\hline
1 & 1 & 1 & 0 & 0 \\
2 & 1 & 1 & 0 & 0 \\
3 & 0 & 1 & 0 & 0 \\
4 & 0 & 0 & 1 & 1 \\
5 & 0 & 0 & 0 & 1 \\
\hline
\end{array}$$

(c) MCFP instance.

$$\begin{array}{c|cc|c|c|}
 & 6 & 7 & 8 & 9 \\
\cline{2-3}
1 & 1 & 1 & 0 & 0 \\
2 & 1 & 1 & 0 & 0 \\
3 & 0 & 1 & 0 & 0 \\
\cline{2-4}
4 & 0 & 0 & 1 & 1 \\
\cline{4-5}
5 & 0 & 0 & 0 & 1 \\
\cline{5-5}
\end{array}$$

(d) MCFP solution.

Figure 6: BGEPS $\leftrightarrow$ MCFP example.

## *3.3. A First Exact Algorithm for the MCFP*

In this section, we propose an exact iterative method for the MCFP. In Lemma 3 we describe upper/lower bounds for the MCFP. Next, we define a parameterized version of the BGEPS to be used in the exact algorithm.

**Lemma 3.** *Let $b^* = a^* + d^*$ be the optimal solution value of the BGEP 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 and $(m - d^*)/(m + a^*)$ is a lower bound for the optimal solution value of the MCFP with input $M$, where $m = |E(G)|$.*

*Proof.* Let $\mu(a, d) = (m - d)/(m + a)$ be the objective function of the MCFP, where $d$ and $a$ denote, respectively, the number of ones outside cells and zeros inside cells in $M$. Consider also that $a + d = k$, for a positive constant $k$.

Taking $\mu(a, d)$ as a function $f(a)$ of $a$, we obtain that

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

Calculating the derivative,

$$\frac{df}{da} = \frac{(m + a) - 1(m - k + a)}{m^2 + 2ma + a^2} = \frac{k}{m^2 + 2ma + a^2} > 0.$$

Since $\frac{df}{da} > 0$ for every $a$, $f(a)$ is an increasing function. Since $k \geq a$, we have $f(k) \geq f(a)$, and thus

$$\frac{m}{m + d + a} \geq \frac{m - d}{m + a}.$$

In other words, in the best case, the $k$ edge editing operations would correspond to $a = k$ edge additions and $d = 0$ edge deletions, since as $a$ increases, $f(a)$ increases as well.

Now consider that $\mu'(a, d) = a + d$ is a feasible solution value of the BGEP. It follows that $a^* + d^* \leq a + d$ and

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

Therefore, $m/(m + a^* + d^*)$ is an upper bound for the optimal solution value of the MCFP.

Showing that $(m - d^*)/(m + a^*)$ is a lower bound is trivial since $b^* = a^* + d^*$ is a feasible solution value of the MCFP, as shown in Lemma 2. $\square$

We now define a parameterized version of the BGEPS, the Bicluster Graph Editing Problem with Size Restriction($\lambda$) (BGEPS($\lambda$)), which consists of finding a solution of the BGEPS with exactly $\lambda$ edge editing operations, such that the number of edge deletions is minimized. A formulation for the BGEPS($\lambda$) is described below:

$$\min \quad \sum_{+(ij)} (1 - y_{ij}) \tag{10}$$

$$st \quad y_{il} + y_{kj} + y_{kl} \leq 2 + y_{ij} \qquad \forall i, k \in U \textit{ and } j, l \in V \tag{11}$$

$$\sum_{j \in V} y_{ij} \geq s_r \qquad \forall i \in U \tag{12}$$

$$\sum_{i \in U} y_{ij} \geq s_c \qquad \forall j \in V \tag{13}$$

$$\sum_{+(ij)} (1 - y_{ij}) + \sum_{-(ij)} y_{ij} = \lambda \tag{14}$$

$$\sum_{+(ij)} (1 - y_{ij}) \leq U_{opt} - 1 \tag{15}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in U \text{ and } j \in V. \tag{16}$$

We now describe the exact iterative method for the MCFP (Algorithm 2 below). The idea is to make several calls to the BGEPS($\lambda$). At each iteration, we seek for a solution with fewer deletions. Constraint (15) tells the model that the optimal value is less than $U_{opt}$, which is obtained using previous feasible solutions.

**Algorithm 2** Exact iterative method for the MCFP

```
 1: procedure ECM(instance $M$)
 2:     let $G$ be the instance of the BGEPS corresponding to $M$
 3:     $(a^*, d^*) \leftarrow BGEPS[G]$    ▷ Solve the BGEPS for input $G$, obtaining $a^*$ additions and $d^*$ deletions
 4:     $UB \leftarrow \frac{m}{m+a^*+d^*}$
 5:     $LB \leftarrow \frac{m-d^*}{m+a^*}$
 6:     $U_{opt} \leftarrow d^*$    ▷ A bound for the number of deletions already found
 7:     $cont \leftarrow 0$
 8:     while $UB > LB$ do
 9:         $(a, d) \leftarrow BGEPS(a^* + d^* + cont)[G]$    ▷ Solve the BGEPS($\lambda$) with $\lambda = a^* + d^* + cont$ for $G$
10:         if $U_{opt} > d$ then
11:             $U_{opt} \leftarrow d$
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:         $UB \leftarrow \frac{m}{m+a+d}$
18:         $cont \leftarrow cont + 1$
19:     end while
20:     return $(a^*, d^*)$
21: end procedure
```

The correctness of Algorithm 2 is shown in the next result.

**Theorem 4.** *Algorithm ECM (Algorithm 2) returns an optimal solution value for the MCFP.*

*Proof.* Algorithm 2 seeks the optimal solution value iteratively through several calls to the the BGEPS($\lambda$), starting with the number of edge editing operations of the BGEPS. At each iteration new bounds are calculated (variables $LB$ and $UB$). Iterations are performed until the upper bound is equal to the lower bound, meaning that from that point there is no better solution.

By Lemma 3, lines 4, 5, and 14 indeed calculate the lower and the upper bounds. It remains to show that line 17 actually calculates an upper bound.

The value of an optimal solution $G^*$ of the BGEPS is not necessarily optimal for the MCFP. In this case, the optimal solution value $\mu^*$ of the MCFP corresponds to a solution of the BGEPS with more editing operations than $G^*$. To find $\mu^*$, we use the BGEPS($\lambda$). The objective function of the BGEPS($\lambda$) leads to a solution with the maximum number of edge additions among all solutions with exactly $\lambda$ editing operations.

In line 9, a call to the BGEPS($\lambda$) is made, using $\lambda = a^* + d^* + cont$ editing operations, and new values $a$ and $d$ are calculated. Then line 17 actually defines a new upper bound, because if there were a solution value between the new bound and the previous bound, the algorithm would have already found such a value in a previous iteration. Since the upper bound decreases along the iterations, we conclude that Algorithm 2 works correctly. □

## *3.4. New linear-fractional model for the MCFP*

According to Lemma 2, the BGEPS and the MCFP have the same feasible solution space. Thus, the constraints of the BGEPS formulation can be used in a new formulation for the MCFP. The adaptation of the objective function is made according to the grouping efficacy described in Section 3. By setting $N_1^{out} = \sum_{+(ij)} (1 - y_{ij})$ and $N_0^{in} = \sum_{-(ij)} y_{ij}$, we propose a new formulation for the MCFP based on linear-fractional programming:

$$
\begin{array}{lll}
\max & \dfrac{m - \sum_{+(ij)} (1 - y_{ij})}{m + \sum_{-(ij)} y_{ij}} & \\
st & y_{il} + y_{kj} + y_{kl} \leq 2 + y_{ij} & \forall i, k \in U \textit{ and } j, l \in V \\
 & \displaystyle\sum_{j \in V} y_{ij} \geq s_r & \forall i \in U \\
 & \displaystyle\sum_{i \in U} y_{ij} \geq s_c & \forall j \in V \\
 & y_{ij} \in \{0, 1\} & \forall i \in U \textit{ and } j \in V.
\end{array}
$$

## *3.5. Linear Formulation for the MCFP*

In this section we propose a linearization for the linear-fractional model described above. The process consists of adding binary variables $x_{da}$ that assume value 1 if and only if the solution contains $d$ deletions and $a$ additions. Each variable $x_{da}$ is associated with a cost $c_{da} = \frac{m-d}{m+a}$ in the objective function. Let $l_c$ and $l_b$ be lower bounds for the MCFP and BGEP, respectively. Define a set $\mathcal{F} = \{(d, a) | c_{da} \geq l_c \text{ and } d + a \geq l_b\}$. In this case, $l_c$ can be obtained heuristicaly, whereas $l_b$ can be, for example, the value of the linear relaxation of the BGEP.

The linear formulation proposed is as follows:

$$
\begin{array}{llll}
\max & \displaystyle\sum\sum_{\forall\ (d,a) \in \mathcal{F}} (c_{da}\ x_{da}) & & (17) \\
\text{st} & y_{ik} + y_{lj} + y_{lk} \leq 2 + y_{ij} & \forall\ i, l \in U \text{ and } k, j \in V & (18) \\
 & \displaystyle\sum\sum_{\forall\ (d,a) \in \mathcal{F}} x_{da} = 1 & & (19) \\
 & \displaystyle\sum\sum_{\forall\ (d,a) \in \mathcal{F}} (d\ x_{da}) = \sum_{-(ij)} y_{ij} & & (20) \\
 & \displaystyle\sum\sum_{\forall\ (d,a) \in \mathcal{F}} (a\ x_{da}) = \sum_{+(ij)} (1 - y_{ij}) & & (21) \\
 & x_{da} \in \{0, 1\} & \forall\ (d, a) \in \mathcal{F} & (22) \\
 & y_{ij} \in \{0, 1\} & \forall\ i \in U \text{ and } j \in V & (23)
\end{array}
$$

The objective function (17) computes the maximum cost $c_{da} = \frac{m-d}{m+a}$. Constraints (18) eliminate induced subgraphs isomorphic to $P_4$. Constraint (19) imposes that exactly one $x_{da}$ variable should assume value 1. Constraints (20) and (21) state that the number of deletions and additions must be $d$ and $a$, respectively. Constraints (22) and (23) define the domain of the variables.

### 3.6. *MCFP Instances*

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 [17], which have been used in many other papers. In Table 1, we present the instances with its dimensions. To the best of our knowledge, this work is the first one that finds the optimal solutions of all such instances.

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

Table 1: Instances of the MCFP.

## 4. Computational Experiments

In this section we evaluate and compare the algorithms proposed in this work. We use the mixed linear optimization software CPLEX [19], 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 2.1, the preprocessing procedure in Section 2.2, and the exact iterative method for the MCFP in Section 3.3 have been implemented in C++. All the instances and solutions are available at `http://www.ic.uff.br/~fabio/instances.pdf`.

### 4.1. *Experimental Results for the BGEP*

The proposed Branch-and-Cut algorithm for the BGEP was applied to 30 randomly generated instances, as explained in Section 2.3, with $p \in \{0.2, 0.4, 0.6, 0.8\}$, $m \in [10, 21]$, and $n \in [11, 22]$.

We first compare the default separation algorithm incorporated in CPLEX with the separation algorithm described in Section 2.1. To evaluate these two running scenarios, we use the *geometric mean*. The geometric mean of a data set $\{t_1, t_2, \ldots, t_n\}$ is defined as:

$$G = \sqrt[n]{t_1 t_2 \cdots t_n}. \tag{24}$$

Each scenario is applied to all the 30 random instances. Table 2 shows the results. From left to right, the columns of the table show: separation algorithm, total running time over all the 30 instances, geometric

| **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: Comparison of separation algorithms for 30 random instances.

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 2.1 clearly influences the convergence speed.

We now analyze the impact of the preprocessing procedure described in Section 2.2. Results are presented in Table 3. Each row in the table shows results for instances generated with the same value of $p$. We remark that, in the $\mathbb{G}(m, n, p)$ model, the probability $p$ is the expected edge density of a random bipartite instance (i.e., the expected number of edges is $mnp$). From left to right, the columns show: instance density, average percentage of fixed variables, and average percentage of generated cuts. (The maximum number of cuts is $mn(n-1)/2 + nm(m-1)/2$, which is achieved by an edgeless bipartite graph.)

| **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% |

Table 3: Impact of the preprocessing procedure.

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.

### *4.2. Experimental results for the MCFP*

Table 4 shows the results of the application of the exact iterative method for the MCFP presented in Section 3.3 (with singleton constraints) on the instances shown in Table 1. From left to right, the columns show: instance name; best solution value found in the literature [17, 20, 21]; optimum value found by our iterative method (using the grouping efficacy as the objective function); the sum $N_1^{out} + N_0^{in}$; and the number of edge editing operations obtained by the application of our Branch-and-Cut method presented in Section 2 on the corresponding BGEPS instance (as explained in Lemma 2). We remark that instances marked with * have been erroneously encoded in some works.

Our method finds the optimum for 27 of 35 instances. These optima were previously unknown. For instances King1980 and Kumar1987, it finds solutions better than the existing solutions in the literature. A point to note is that the values in 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 without singleton constraints. Values in the second column are taken from [18, 22, 23]. As in the previous table, instances marked with * have been erroneously encoded in some works.

Again, our method finds the optimum for 27 of 35 instances. For three instances (King1980, Kumar1987, and Stanfel1985_1), it finds solutions better than the existing ones 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).

In Table 6 we compare the running times obtained by applying the two exact approaches for the MCFP proposed in this work (disregarding singleton constraints) on the instances of Table 1. The linear model described in Section 3.5 achieves running times faster than those by the exact iterative method in Section 3.3, with few exceptions. (The addition of singleton constraints produces similar results.)

| Instance | Literature | Optimum | $N_1^{out} + N_0^{in}$ | BGEPS Optimum |
|---|---|---|---|---|
| King1982 | 73.68 | 73.68 | **5** | **5** |
| Waghodekar1984 | 62.50 | 62.50 | **9** | **9** |
| Seiffodini1989 | 79.59 | 79.59 | **10** | **10** |
| Kusiak1992 | 76.92 | 76.92 | **5** | **5** |
| Kusiak1987 | 53.13 | 53.12 | **15** | **15** |
| Boctor1991 | 70.37 | 70.37 | **7** | **7** |
| Seiffodini1986 | 68.30 | 68.29 | **13** | **13** |
| Chandrasekaran1986a | 85.25 | 85.24 | **9** | **9** |
| Chandrasekaran1986b | 58.72 | 58.71 | 45 | 43 |
| Mosier1985a | 70.59 | 70.58 | **10** | **10** |
| Chan1982 | 92.00 | 92.00 | **4** | **4** |
| Askin1987 | 69.86 | 69.86 | 22 | 21 |
| Stanfel1985 | 69.33 | 69.33 | 23 | 22 |
| McCornick1972a | 51.96 | 51.96 | 49 | 46 |
| Srinivasan1990 | 67.83 | 67.83 | **46** | **46** |
| King1980 | 55.90 | **56.52** | 70 | 68 |
| Carrie1973a | 54.46 | 54.46 | 51 | 47 |
| Mosier1985b | 42.96 | | | |
| Kumar1986 | 49.65 | 49.64 | 71 | 64 |
| Carrie1973b | 76.54 | 76.54 | **37** | **37** |
| Boe1991 | 58.15 | 58.15 | 77 | 73 |
| Chandrasekharan1989_1 | 100.00 | 100.00 | **0** | **0** |
| Chandrasekharan1989_2 | 85.11 | 85.10 | **21** | **21** |
| Chandrasekharan1989_3-4 | 73.51 | 73.50 | **39** | **39** |
| Chandrasekharan1989_5 | 51.97 | 51.97 | 73 | 70 |
| Chandrasekharan1989_6 | 47.37 | | | |
| Chandrasekharan1989_7 | 44.87 | | | |
| McCormick1972b | 54.27 | | | |
| Carrie1973c | 46.06 | | | |
| Kumar1987* | 58.58 | **58.94** | 62 | 60 |
| Stanfel1985_1* | 59.66 | 59.65 | **71** | **71** |
| Stanfel1985_2 | 50.51 | | | |
| King1982 | 42.64 | | | |
| McCormick1972c | 59.85 | | | |
| Chandrasekharan1987 | 84.03 | 84.03 | **73** | **73** |

Table 4: Results of the exact iterative method for the MCFP with singleton constraints $2 \times 2$.

| Instance | Literature | Optimum | $N_1^{out} + N_0^{in}$ | BGEPS Optimum |
|---|---|---|---|---|
| King1982* | 75 | 75 | **4** | **4** |
| Waghodekar1984 | 69.57 | 69.56 | **7** | **7** |
| Seiffodini1989 | 80.85 | 80.85 | **9** | **9** |
| Kusiak1992 | 79.17 | 79.16 | **5** | **5** |
| Kusiak1987 | 60.87 | 60.86 | **9** | **9** |
| Boctor1991 | 70.83 | 70.83 | **7** | **7** |
| Seiffodini1986 | 69.44 | 69.44 | **11** | **11** |
| Chandrasekaran1986a | 85.25 | 85.24 | **9** | **9** |
| Chandrasekaran1986b | 58.72 | 58.71 | 45 | 43 |
| Mosier1985a | 75 | 75 | **7** | **7** |
| Chan1982 | 92 | 92 | **4** | **4** |
| Askin1987 | 74.24 | 74.24 | **17** | **17** |
| Stanfel1985 | 72.86 | 72.85 | 19 | 18 |
| McCornick1972a | 53.33 | 53.33 | **42** | **42** |
| Srinivasan1990 | 69.92 | 69.92 | 40 | 39 |
| King1980 | 57.96 | **58.04** | 60 | 59 |
| Carrie1973a | 57.73 | 57.73 | 41 | 40 |
| Mosier1985b | 43.97 | | | |
| Kumar1986 | 50.81 | 50.80 | 61 | 60 |
| Carrie1973b | 79.38 | 79.37 | 33 | 32 |
| Boe1991 | 58.79 | 58.79 | 75 | 70 |
| Chandrasekharan1989_1 | 100 | 100 | **0** | **0** |
| Chandrasekharan1989_2 | 85.11 | 85.10 | **21** | **21** |
| Chandrasekharan1989_3-4 | 73.51 | 73.50 | 40 | 39 |
| Chandrasekharan1989_5 | 53.29 | 53.28 | 71 | 64 |
| Chandrasekharan1989_6 | 48.95 | | | |
| Chandrasekharan1989_7 | 46.58 | | | |
| McCormick1972b | 54.82 | | | |
| Carrie1973c | 47.68 | | | |
| Kumar1987* | 62.86 | **63.04** | **51** | **51** |
| Stanfel1985_1* | 59.66 | **59.77** | 70 | 67 |
| Stanfel1985_2 | 50.83 | | | |
| King1982 | 47.93 | | | |
| McCormick1972c | 61.16 | | | |
| Chandrasekharan1987 | 84.03 | 84.03 | **73** | **73** |

Table 5: Results of the exact iterative method for the MCFP without singleton constraints.

| **Instance** | **Linear Model** | **Iterative Model** |
|---|---|---|
| King1982 | 0.01 | 0.16 |
| Waghodekar1984 | 0.01 | 0.07 |
| Seifoddini1989 | 0.03 | 0.09 |
| Kusiak1992 | 0.01 | 0.02 |
| Boctor1991 | 0.01 | 0.14 |
| Kusiak1987 | 0.06 | 0.29 |
| Seifoddini1986 | 0.03 | 0.18 |
| Chandrasekharan1986a | 0.04 | 2.06 |
| Chandrasekharan1986b | 4.94 | 81.46 |
| Mosier1985a | 0.01 | 0.03 |
| Chan1982 | 0.02 | 0.01 |
| Askin1987 | 0.09 | 0.49 |
| Stanfel1985 | 0.11 | 0.49 |
| McCormick1972 | 144.91 | 600.98 |
| Srinivasan1990 | 0.54 | 7.24 |
| King1980 | 125.62 | 1156.23 |
| Carrie1973 | 42.32 | 87.13 |
| Mosier1985b | – | – |
| Kumar1986 | 1771.99 | 23928.70 |
| Boe1991 | 305.48 | 2145.24 |
| Carrie1973 | 14.55 | 1.78 |
| Chandrasekharan1989_1 | 0.15 | 0.02 |
| Chandrasekharan1989_2 | 0.44 | 10.08 |
| Chandrasekharan1989_3-4 | 0.78 | 17.46 |
| Chandrasekharan1989_5 | 48743.90 | 371233.00 |
| Chandrasekharan1989_6 | – | – |
| Chandrasekharan1989_7 | – | – |
| McCormick1972 | – | – |
| Carrie1973 | – | – |
| Kumar1987 | 41.53 | 183.71 |
| Stanfel1985_1 | 2622.06 | 13807.50 |
| Stanfel1985_2 | – | – |
| King1982 | – | – |
| McCormick1972 | – | – |
| Chandrasekharan1987 | 18.22 | 325.53 |

Table 6: Comparison between running times (in seconds) of the exact methods for the MCFP.

## 5. Conclusions

This work has investigated the close relationships between the BGEP and the MCFP. This opens new possibilities of research, in the sense that each every contribution 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 dynamic programming 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 462 vertices. Experimental results show that the proposed method outperforms the CPLEX standard separation 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. We have shown that these problems have the same feasible solution space. This fact allows the use of mathematical formulations for the BGEP 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 (11) and (18) 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 several calls to a parameterized version of the MCFP;
- a new linear-fractional formulation and its linearization;
- a experimental study of the impact of using a linear objective function for the MCFP;
- a separated analysis of problems with or without singleton constraints;
- exact resolution of most instances of the MCFP found in literature, some crated almost 40 years ago.

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

## References

[1] N. Amit, The Bicluster Graph Editing Problem, Master's thesis, Tel Aviv University, 2004.

[2] A. Gupta, A. Palit, Clique generation using boolean equations, Proceedings of The IEEE 67 (1979) 178–180.

[3] J. A. Hartigan, Clustering Algorithms, John Wiley & Sons, 1975.

[4] Y. Cheng, G. M. Church, Biclustering of expression data, in: Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, AAAI Press, 2000, pp. 93–103.

[5] N. Faure, P. Chretienne, E. Gourdin, F. Sourd, Biclique completion problems for multicast network design, Discrete Optimization 4 (2007) 360–377.

[6] A. Abdullah, A. Hussain, A new biclustering technique based on crossing minimization, Neurocomputing 69 (2006) 1882–1896.

[7] G. Bisson, F. Hussain, Chi-sim: A new similarity measure for the co-clustering task, in: Proceedings of the 2008 Seventh International Conference on Machine Learning and Applications, ICMLA '08, IEEE Computer Society, 2008, pp. 211–217.

[8] F. Protti, M. Dantas da Silva, J. L. Szwarcfiter, Applying modular decomposition to parameterized bicluster editing, in: Parameterized and Exact Computation, volume 4169 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 2006, pp. 1–12.

[9] J. Guo, F. Hffner, C. Komusiewicz, Y. Zhang, Improved algorithms for bicluster editing, in: 5th International Conference on Theory and Applications of Models of Computation, volume 4978, Springer Berlin / Heidelberg, 2008, pp. 445–456.

[10] G. F. Sousa Filho, L. dos Anjos F. Cabral, L. S. Ochi, F. Protti, Hybrid metaheuristic for bicluster editing problem, Electronic Notes in Discrete Mathematics 39 (2012) 35 – 42.

[11] E. Gilbert, Random graphs, Annals of Mathematical Statistics 30 (1959) 1141–1144.

[12] R. E. Flanders, Design, manufature and production control of a standard machine, Transactions of the American Society of Mechanical Enginneers 46 (1924) 691–738.
[13] S. P. Mitrofanov, The Scientific Principles of Group Technology, National Lending Library for Science and Technology, 1966.
[14] J. L. Burbidge, Production flow analysis, Production Engineer 50 (1971) 139–152.
[15] M. P. Chandrasekharan, R. Rajagopalan, Zodiac - an algorithm for concurrent formation of part-families and machine-cells, International Journal of Production Research 25 (1987) 835–850.
[16] G. Srinivasan, T. T. Narendran, Grafics - a nonhierarchical clustering-algorithm for group technology, International Journal of Production Research 29 (1991) 463–478.
[17] J. Gonçalves, M. Resende, An evolutionary algorithm for manufacturing cell formation, Computers & Industrial Engineering 47 (2004) 247–273.
[18] A. Pailla, A. R. Trindade, V. Parada, L. S. Ochi, A numerical comparison between simulated annealing and evolutionary approaches to the cell formation problem, Expert Syst. Appl. 37 (2010) 5476–5483.
[19] IBM, IBM ILOG CPLEX V12.1 User's Manual for CPLEX, IBM, 2009.
[20] T. H. Wu, C. C. Chang, S. H. Chung, A simulated annealing algorithm for manufacturing cell formation problems, Expert Systems with Applications 34 (2008) 1609–1617.
[21] T. H. Wu, S. H. Chung, C. C. Chang, A water flow-like algorithm for manufacturing cell formation problems, European Journal of Operational Research 205 (2010) 346–360.
[22] T. H. Wu, C. C. Chang, Y. J. Y., A hybrid heuristic algorithm adopting both boltzmann function and mutation operator for manufacturing cell formation problems, International Journal of Production Economics 120 (2009) 669–688.
[23] B. Elbenani, J. A. Ferland, J. Bellemare, Genetic algorithm and large neighbourhood search to solve the cell formation problem, Expert Systems with Applications 39 (2012) 2408–2414.