On the maximum edge biclique packing problem: formulations and computational experiments

V. Acuña*, C. E. Ferreira†, A. S. Freire† and E. Moreno‡

∗ *Université Claude Bernard, Lyon I, France*

† *IME-USP, São Paulo, Brazil*

‡ *Universidad Adolfo Ibáñez, Santiago, Chile*

Abstract

A *biclique* is a complete bipartite graph. Given a (L, R)-bipartite graph $G = (V, E)$ and a positive integer k, the *maximum edge biclique packing* (MEBP) problem consists in finding a set of at most k bicliques, subgraphs of G, such that the bicliques are vertex disjoint with respect to a subset $S \subseteq V$ and the number of edges inside the bicliques is maximized. The *maximum edge biclique* (MEB) problem is a special case of the MEBP problem in which k is fixed as 1.

Several applications of the MEB problem have been studied and, in this paper, we describe applications of the MEBP problem in *metabolic networks* and *product bundling*. In these applications the input graphs are very unbalanced (i.e. $|R|$ is considerably greater than $|L|$), thus we consider carefully this property in our models. We introduce a new formulation for the MEB problem and a branch-and-price scheme for the MEBP problem. Finally, we present computational experiments with instances that come from the described applications and also with randomly generated instances.

Keywords: maximum edge biclique packing, branch-and-price, metabolic networks, product bundling.

1. Introduction

A *biclique* is a complete bipartite graph. Given a (L, R)-bipartite graph $G = (V, E)$ and a positive integer k, the *maximum edge biclique packing* (MEBP) problem consists in finding a set of at most k bicliques, subgraphs

of G, such that the bicliques are vertex disjoint with respect to a subset $S \subseteq V$ and the number of edges inside the bicliques is maximized. In this paper, which is an extension of the work presented in [1], we study the MEBP problem and also the *maximum edge biclique* (MEB) problem, a special case of the MEBP problem in which k is fixed as 1.

As shown in 2003 by R. Peeters [14], the MEB problem is NP-hard. In 2004, U. Feige and S. Kogan [8] conjectured that the MEB problem is hard to approximate within a factor of $O(n^{\epsilon})$, for some $\epsilon > 0$. In the cited work, the authors prove that the MEB problem is hard to approximate under the plausible assumption that 3-SAT has no sub-exponential algorithm, and they also show in [7] that the above conjecture is valid under certain other plausible assumption. M. Dawande et al. introduced in [5] an algorithm with expected approximation ratio of 2, for sufficiently dense random bipartite graphs. For convex bipartite graphs the problem is polynomially solvable [13]. There are many applications of the MEB problem in biological data analysis and other areas (see [11] for a very comprehensive survey, and see [6] for a slightly different variant of the problem with application in multicast network design).

Different variants of the MEB problem can be found in the literature. Consider a function associating a non-negative value to each edge of a bipartite graph G. The *maximum weight biclique* problem consists in finding a biclique, subgraph of G, with maximum weight, where the weight of a biclique is the sum of the weights of its edges. As shown in 2008 by Jinsong Tan [17], the maximum weight biclique problem is hard to approximate. We say that a (L, R)-bipartite graph B is *balanced* if $|R| = |L|$. U. Feige and S. Kogan showed in [8] that the *maximum balanced biclique* problem is hard to approximate. The problem of finding a biclique with maximum number of vertices in a bipartite graph can be solved in polynomial time [9]. Although there is a wide literature on the MEB problem and some variants of it, we could not find any previous result on the MEBP problem.

One of the motivations of this work is data analysis in Bioinformatics. Consider L as a set of *individuals* and R as a set of *conditions*. An edge uv means that individual u satisfy condition v. The aim is to cluster individuals based on the common conditions satisfied by the *entire* cluster. In particular, this method is been used in the study of *metabolic networks*, to improve the interpretation of the data given by the analysis of *Elementary Flux Modes* (EFMs). The EFMs of a metabolic network, correspond to minimal sets of reactions that can be used in steady state (for precise definitions of EFMs see Schuster and Hilgetag [16]). However, the number of EFMs obtained in

real networks is, in general, huge and thus impossible in practice to analyze "by hand". For instance, the central metabolism of *E.Coli* corresponds to a network of 106 reactions and about 26.000.000 EFMs (see Terzer and Stelling [18]). In this context a way to obtain more significant biological information is dividing the metabolic network into k sets of reactions that belong to many EFMs (see Notebaart et al. [12]). This correspond to solve the MEBP problem where L is the set of reactions, R is the set of EFMs (with $|L| \ll |R|$) and bicliques are the clusters with non overlapping reactions (i.e., $S = L$).

Another application of the MEBP problem is in the context of consumer products bundling. The concept of *product bundling* is the sale of two or more separate goods in a single package. This marketing strategy is massively utilized, especially in retail products. We study the problem of selecting an optimal set of k product bundles that maximize the total number of products sold via bundles, subject to client's demand of products. The motivation of this problem came from a major food company in Chile that was evaluating the capability to deliver its products directly to small grocers, avoiding its current wholesaler and distributors. This problem can be modeled as an instance of MEBP problem by constructing a bipartite graph, where vertices in L represent clients and vertices in R products, and an edge uv correspond to a client u that consumes a particular product v. In this case, a biclique on this graph corresponds to a set of clients (potential consumers) that buy a set of products (product bundle), and the goal is to find k product bundles that maximize the supplied demand (i.e., the number of the edges inside the bicliques). In this application $S = L$ means that each client is supposed to buy at most one bundle. On the other hand, $S = R$ means that one product cannot appear in more than one bundle.

This paper is organized as follows. In Section 2 we study the MEB problem and in Section 2.1 a new integer linear formulation is introduced for the problem. In Section 3 we present a branch-and-price scheme for the MEBP problem. The computational experiments are shown in Section 4. In Section 4.1 we compare two different approaches for solving the MEB problem and in Section 4.2 we evaluate our implementation of the branch-and-price procedure proposed in Section 3. Finally, we present the concluding remarks and future work in Section 5.

2. On the maximum edge biclique problem

Let $G = (V, E)$ be a simple (L, R)-bipartite graph. For simplicity of notation, given an edge $uv \in E$, we assume as a convention that $u \in L$ and $v \in R$ (i.e. the label in the left side corresponds to the vertex in the class L, and the label in the right side corresponds to the vertex in the class R). We assume, without loss of generality, that $|L| \leq |R|$. In particular, we are interested in very unbalanced bipartite graphs (in some instances from the application in metabolic networks we have that $\frac{|R|}{|L|} \geq 26.000$).

Given a subgraph B of G, we denote its vertex set by V_B and its edge set by E_B. We say that B is a (L_B, R_B)-bipartite graph, where $L_B = V_B \cap L$ and $R_B = V_B \cap R$. A *biclique* is a complete bipartite graph, and the MEB problem can be stated as follows.

Problem 2.1. *Given a bipartite graph G, find a biclique B, subgraph of G, such that $|E_B|$ is maximum.*

We say that two edges uv and pq of E are *incompatible* if $u \neq p$, $v \neq q$ and $uq \notin E$ or $pv \notin E$. Let $\mathcal{I} := \{\{uv, pq\} \in E \times E \mid uv \text{ and } pq \text{ are incompatible}\}$ be the set of all pairs of incompatible edges of E.

Observation 2.1. *Two incompatible edges cannot be in the same biclique and every maximal edge set with no incompatibles edges is a biclique.*

Based on this observation, M. Dawande et al introduced in [5] the following formulation for the MEB problem.

$$
\begin{array}{lllll}
 & \max & \displaystyle\sum_{uv \in E} x_{uv} & & \\
(\mathrm{P_{MEB}}) & \text{s.t.} & x_{uv} + x_{pq} \leq 1, & \text{for each } \{uv, pq\} \in \mathcal{I} & (1) \\
 & & x_{uv} \in \{0, 1\}, & \text{for each } uv \in E & (2)
\end{array}
$$

The interpretation for the binary variables x_{uv} is the following: $x_{uv} = 1$ if and only if the edge uv is in the biclique (i.e. x is the characteristic vector of the biclique's edge set). Let $H = (V_H, E_H)$ be the graph defined as follows: $V_H := \{v_e \mid e \in E\}$ and $E_H := \{v_e v_f \mid \{e, f\} \in \mathcal{I}\}$. In this construction, there is a one-to-one correspondence between each *independent set* in H and a feasible solution to ($\mathrm{P_{MEB}}$). In fact, as Berman and Schnitger show in [4], the independent set problem is NP-hard to approximate by a factor of $O(n^\epsilon)$, for some constant $\epsilon > 0$, where $n = |V_H|$ is the number of vertices in the

input graph. Despite the hardness of solving, and even approximating, the independent set problem, there are methods for solving this problem in a reasonable running time, for considerable large graphs (see [2, 19, 21, 10]). Thus, one approach that we consider in Section 4.1 for solving the MEB problem is to apply one of these methods for finding a maximum independent set in the graph H (or equivalently, finding a maximum clique in the complement of H).

2.1. A new formulation for the MEB *problem*

Given a vertex $i \in V$, we denote by $\delta(i) \subset V$ the set of vertices adjacent to i, and we define $\overline{\delta}(i) := \{j \in Q \mid j \notin \delta(i)\}$, where if $i \in L$ then $Q = R$, otherwise $Q = L$.

Observation 2.2. *Given a subset $L' \subseteq L$, we have that $B = (L_B, R_B)$, where $L_B := L'$ and $R_B := \{v \in R \mid \delta(v) \cap L' = L'\}$, is a biclique and $|E_B| \geq |E_{B'}|$, for each biclique B' such that $L_{B'} = L'$.*

From observation 2.2 follows that we can focus only on finding the appropriate subset of L (typically, $|L|$ is far smaller than $|E|$, even when the input graph is well balanced). Below, we introduce a new formulation for the MEB problem based on this idea.

$$
\begin{array}{lll}
 & \max & \displaystyle\sum_{v \in R} y_v \\
(\mathrm{P}^2_{\mathrm{MEB}}) & \text{s.t.} & \\
 & y_v \leq \displaystyle\sum_{u \in \delta(v)} z_u, & \text{for each } v \in R \qquad (3) \\
 & \displaystyle\sum_{v \in \overline{\delta}(u)} y_v \leq (1 - z_u) \sum_{v \in \overline{\delta}(u)} |\delta(v)|, & \text{for each } u \in L \qquad (4) \\
 & y_v \geq 0, & \text{for each } v \in R \qquad (5) \\
 & z_u \in \{0, 1\}, & \text{for each } u \in L \qquad (6)
\end{array}
$$

The binary variables z_u and the continuous variables y_v are interpreted in the following way: $z_u = 1$ if and only if the vertex u is in the biclique; and y_v represents the number of edges incident to v which are in the biclique, thus a vertex $v \in R$ is in the biclique if and only if $y_v > 0$ (note that since z is integral, thus in any optimal solution we have that y is integral as well).

Lemma 2.1. $(\mathrm{P}^2_{\mathrm{MEB}})$ *is a formulation for the* MEB *problem.*

PROOF. Given a biclique B', subgraph of G, let $z' \in \{0,1\}^{|L|}$ be the characteristic vector of the vertex set $L_{B'}$, and let $y' \in \mathbb{R}_+^{|R|}$ be a vector such that, for each $v \in R$, if $v \in R_{B'}$, then $y'_v = |L_{B'}|$, otherwise $y'_v = 0$. Since B' is a biclique, then we have that $u \in \delta(v)$, for each $u \in L_{B'}$ and $v \in R_{B'}$, thus inequalities (3) are satisfied. Moreover, for each $u \in L_{B'}$ and $v \in \overline{\delta}(u)$, we have that $y'_v = 0$, thus inequalities (4) are satisfied. Constraints (5) and (6) are clearly satisfied. Therefore, (z', y') is a feasible solution to $(\mathrm{P}^2_{\mathrm{MEB}})$ and $\sum_{v \in R} y_v = |E_{B'}|$.

Conversely, given an optimal solution (z, y) to $(\mathrm{P}^2_{\mathrm{MEB}})$, let B be the subgraph of G induced by $V_B \subseteq V$, where $L_B := \{u \in L \mid z_u = 1\}$ and $R_B := \{v \in R \mid y_v > 0\}$. Suppose, by contradiction, that B is not a biclique and, consequently, V_B contains a pair of vertices $u \in L_B$ and $v \in R_B$ such that $v \in \overline{\delta}(u)$. Since $z_u = 1$ and (z, y) satisfies the inequalities (4), then $\sum_{p \in \overline{\delta}(u)} y_p = 0$, in particular $y_v = 0$. But, since $v \in R_{B'}$, then $y_v > 0$, which is a contradiction. Therefore, B is a biclique. By constraints (3) and (4) and by the optimality of (z, y), we have that for each $v \in R$, if $v \in R_B$, then $y_v = |L_B|$, otherwise $y_v = 0$. Thus, the value of the objective function for (z, y) is $|E_B|$. □

Note that $(\mathrm{P}^2_{\mathrm{MEB}})$ can be strengthened if we replace constraints (4) by the following constraints.

$$y_v \leq |\delta(v)|(1 - z_u), \text{ for each } u \in L \text{ and } v \in \overline{\delta}(u). \tag{7}$$

We denote by $(\mathrm{P}^3_{\mathrm{MEB}})$ the formulation which is the same as $(\mathrm{P}^2_{\mathrm{MEB}})$, but replacing constraint (4) by constraint (7). As one can observe, $(\mathrm{P}^2_{\mathrm{MEB}})$ and $(\mathrm{P}^3_{\mathrm{MEB}})$ are equivalent, in the sense that both of them are formulations for the MEB problem, but $(\mathrm{P}^2_{\mathrm{MEB}})$ has many less constraints than $(\mathrm{P}^3_{\mathrm{MEB}})$. As a side effect, $(\mathrm{P}^2_{\mathrm{MEB}})$ is weaker than $(\mathrm{P}^3_{\mathrm{MEB}})$ (i.e., the feasible set of $(\mathrm{P}^3_{\mathrm{MEB}})$ is strictly included into the feasible set of $(\mathrm{P}^2_{\mathrm{MEB}})$). In Section 4.1 we compare the performance of these two formulations in practice.

3. On the maximum edge biclique packing problem

Let $S \subseteq V$ be any subset of the vertices of the input graph $G = (V, E)$. Given a biclique B, we denote by S_B the set of vertices in $V_B \cap S$. A *biclique packing* in G is a set $\mathcal{H}$ of bicliques, subgraphs of G, such that, for each pair B, B' of distinct bicliques in $\mathcal{H}$, we have that $S_B \cap S_{B'} = \emptyset$. The *size* of $\mathcal{H}$ is given by $s(\mathcal{H}) = \sum_{B \in \mathcal{H}} |E_B|$ and the MEBP problem can be stated as follows.

Problem 3.1. *Given a bipartite graph G and a positive integer k, find a biclique packing $\mathcal{H}$ in G such that $|\mathcal{H}| \leq k$ and $s(\mathcal{H})$ is maximum.*

Note that if $k \geq |S|$ and $S \subseteq L$, then the problem becomes trivial, since each star rooted in L is a biclique (the case in which $S = R$ is analogous). On the other hand, if $k = 1$, then we have the MEB problem.

Let $\mathcal{B}$ be the set of all bicliques which are subgraphs of G and let $\mathcal{B}(v) := \{B \in \mathcal{B} \mid v \in V_B\}$ be the set of all bicliques containing the vertex v, for each $v \in V$. We define variables $x_B \in \{0, 1\}$, for each $B \in \mathcal{B}$, with the following interpretation: $x_B = 1$ if and only if the biclique B is in the biclique packing. Now, we introduce a formulation for the MEBP problem.

$$
\begin{array}{lll}
 & \max & \displaystyle\sum_{B \in \mathcal{B}} |E_B| \cdot x_B \\
(\mathrm{P_{MEBP}}) & \text{s.t.} & \displaystyle\sum_{B \in \mathcal{B}} x_B \leq k \qquad (8) \\
 & & \displaystyle\sum_{B \in \mathcal{B}(u)} x_B \leq 1, \quad \text{for each } u \in S \qquad (9) \\
 & & x_B \in \{0, 1\}, \quad \text{for each } B \in \mathcal{B} \qquad (10)
\end{array}
$$

Constraint (8) guarantees that at most k bicliques are chosen and constraints (9) guarantee that the chosen bicliques are vertex disjoint with respect to S. The objective is to maximize the size of the biclique packing.

Let ($\mathrm{L_{MEBP}}$) be the linear relaxation of ($\mathrm{P_{MEBP}}$), where constraints (10) are replaced by $x_B \geq 0$, for each $B \in \mathcal{B}$. The number of variables in ($\mathrm{L_{MEBP}}$) is exponential in the size of the input, thus we propose here a *column generation* algorithm (see [3] for a nice survey on the object) for solving ($\mathrm{L_{MEBP}}$). For this algorithm, we need to investigate the corresponding so-called *pricing problem* (i.e. find a variable x_B with negative reduced price or give a proof that no such variable exists). First, consider the dual of ($\mathrm{L_{MEBP}}$).

$$
\begin{array}{lll}
 & \min & \displaystyle\sum_{u \in S} \alpha_u + k\beta \\
(\mathrm{D_{MEBP}}) & \text{s.t.} & \beta + \displaystyle\sum_{u \in S_B} \alpha_u \geq |E_B|, \quad \text{for each } B \in \mathcal{B} \qquad (11) \\
 & & \alpha_u \geq 0, \quad \text{for each } u \in S \qquad (12) \\
 & & \beta \geq 0 \qquad (13)
\end{array}
$$

The dual variable β is associated to the constraint (8) and the dual variables α are associated to the constraints (9). For simplicity, we assume that

$\alpha_v = 0$, for each $v \in V \setminus S$. For a given biclique B, subgraph of G, the reduced price of a variable x_B is given by $\hat{x}_B := \beta + \sum_{v \in V_B} \alpha_v - |E_B|$. Let $f : \mathcal{B} \times \mathbb{R}_+^{|V|} \to \mathbb{R}$ be the function defined as follows: $f(B, \alpha) = |E_B| - \sum_{v \in V_B} \alpha_v$. Usually, the pricing problem is strongly related with the problem of finding a variable with minimum reduced price, which in this case corresponds to solving the following optimization problem.

Problem 3.2. *Given a bipartite graph G and a vector $\alpha \in \mathbb{R}_+^{|V|}$, find a biclique B, subgraph of G, such that $f(B, \alpha)$ is maximum.*

Clearly, the MEB problem is a special case of the problem 3.2 in which $\alpha_v = 0$, for each $v \in V$. Hence, the problem 3.2 is NP-hard and, moreover, adapting the formulation $(\mathrm{P}^2_{\mathrm{MEB}})$ for solving it seems a natural idea.

Observation 3.1. *Given a vector $\alpha \in \mathbb{R}_+^{|V|}$ and a subset $L' \subseteq L$, we have that $B = (L_B, R_B)$, where $L_B := L'$ and $R_B := \{v \in R \mid \delta(v) \cap L' = L'$ and $\alpha_v < |L'|\}$, is a biclique and $f(B, \alpha) \geq f(B', \alpha)$, for each biclique B' such that $L_{B'} = L'$.*

Therefore, for the problem 3.2 we also have the nice property that we can focus only on finding the appropriate subset of L in order to find a biclique B which maximizes the objective function (in this case $f(B, \alpha)$ instead of $|E_B|$). But, the problem 3.2 is a little more tricky to be formulated using only $O(|L|)$ integer (binary) variables as in $(\mathrm{P}^2_{\mathrm{MEB}})$, when $S \cap R \neq \emptyset$. Below, we introduce a formulation for the problem 3.2 and later on we discuss what can be simplified in it when $S \cap R = \emptyset$.

We define variables $z_u \in \{0, 1\}$, for each $u \in L$, $l_i \in \{0, 1\}$, for $i = 0, \ldots |L|$, and $y_v, r_v \in \mathbb{R}_+$, for each $v \in R$, with the following interpretation: $z_u = 1$ if and only if u is in the biclique; $r_v = 1$ if and only if v is in the biclique; $l_i = 1$ if and only if $|L_B| = i$, where $L_B \subseteq L$ is the set of vertices in L which are in the biclique; and y_v represents the number of edges incident to v which are in the biclique. We define $\Gamma_{\mathrm{MEB}} := \{(z, y) \in \{0, 1\}^{|L|} \times \mathbb{R}_+^{|R|} \mid (z, y)$ satisfies the constraints $(3), (4), \ldots, (6)$ from $(\mathrm{P}^2_{\mathrm{MEB}})\}$. Now, we introduce a formulation for the problem 3.2.

$$
\begin{aligned}
& \max \quad \sum_{v \in R} (y_v - \alpha_v r_v) - \sum_{u \in L} \alpha_u z_u \\
(\mathrm{P_{PRC}}) \quad & \text{s.t.} \quad \sum_{i=0}^{|L|} l_i = 1 && (14)\\
& \sum_{u \in L} z_u = \sum_{i=0}^{|L|} i l_i && (15)\\
& y_v \leq |\delta(v)| \sum_{i=\lfloor \alpha_v + 1 \rfloor}^{|L|} l_i, \quad \text{for each } v \in R && (16)\\
& \sum_{i=\lfloor \alpha_v + 1 \rfloor}^{|L|} l_i \leq r_v + \sum_{u \in \bar{\delta}(v)} z_u, \quad \text{for each } v \in R && (17)\\
& (z, y) \in \Gamma_{\mathrm{MEB}} && (18)\\
& r_v \geq 0, \quad \text{for each } v \in R && (19)\\
& l_i \in \{0, 1\}, \quad \text{for } i = 1, \ldots, |L| && (20)
\end{aligned}
$$

Given an optimal solution (z^*, y^*, r^*, l^*) to $(\mathrm{P_{PRC}})$, let B^* be the biclique such that $u \in L_{B^*}$ if and only if $z_u^* = 1$, and $v \in R_{B^*}$ if and only if $y_v^* > 0$. Constraints (14), ..., (17) together with constraint (18) guarantee that $v \in R_{B^*}$ if and only if (i) $\alpha_v < \sum_{u \in L} z_u^*$ and (ii) $u \in \delta(v)$, for each $u \in L$ such that $z_u^* = 1$. Thus, the vertices from R are chosen correctly, according to observation 3.1. More precisely, constraints (14) and (15) are introduced just for "keeping track" of the cardinality of L_B^*, while constraints (16) are introduced for preventing the vertices in R which do not satisfy the property of observation 3.1 of being included in the biclique. Constraints (17) guarantee that if v is chosen to be in the biclique, then $r_v = 1$, thus the value of $f(B^*, \alpha)$ is calculated correctly.

Note that we define in $(\mathrm{P_{PRC}})$ only $2|L| + 1$ integer (binary) variables. But, since l and z are integral, then in any optimal solution r and y are integral as well. This observation is very important when $|L|$ is considerably smaller than $|R|$. If $S \cap R = \emptyset$ we can remove the variables r and l, and also remove all the constraints, except (18) (for this case we have the same formulation as $(\mathrm{P^2_{MEB}})$, except for the objective function).

One can solve $(\mathrm{L_{MEBP}})$ by column generation, using a MIP solver for solving $(\mathrm{P_{PRC}})$ at each pricing step. But, possibly the optimal solution found to $(\mathrm{L_{MEBP}})$ is not integral. Now, we propose a *branch-and-price* scheme for solving $(\mathrm{P_{MEBP}})$.

3.1. A branch-and-price scheme for the MEBP *problem*

Given an optimal solution x to ($\mathrm{L_{MEBP}}$), let $\lambda_{uv} := \sum_{B \in \mathcal{B}(u) \cap \mathcal{B}(v)} x_B$ be an implicit variable, for each pair of distinct vertices $\{u, v\} \subset S$, with the following interpretation: $\lambda_{uv} = 1$ if and only if the vertices u and v are in the same biclique. Given a variable λ_{uv} with fractional value μ, one can branch into the following two possibilities: $\lambda_{uv} \leq \lfloor \mu \rfloor = 0$ or $\lambda_{uv} \geq \lceil \mu \rceil = 1$. Thus, we have to care about the following two issues: how to select a variable λ_{uv} with fractional value (a candidate for branching) and how to solve the problem 3.3 stated below.

Problem 3.3. *Given two sets $F_1 \subset S \times S$ and $F_0 \subset S \times S$, find a solution to* ($\mathrm{L_{MEBP}}$) *subject to $\lambda_{uv} = 1$, for each $\{u, v\} \in F_1$, and $\lambda_{uv} = 0$, for each $\{u, v\} \in L_0$.*

We define $\mathcal{B}_{F_0} := \{B \in \mathcal{B} \mid \{u, v\} \not\subseteq V_B$, for each $\{u, v\} \in F_0\}$ and $X_{\mathrm{MEBP}} := \{x \in \mathbb{R}^{\mathcal{B}}_{+} \mid x$ satisfies (8) and (9) from ($\mathrm{P_{MEBP}}$) and $x \geq 0\}$. Below, we introduce a formulation for the problem 3.3.

$$
(\mathrm{L^2_{MEBP}}) \quad
\begin{array}{lrcll}
\max & \displaystyle\sum_{B \in \mathcal{B}_{F_0}} |E_B| \cdot x_B & & & \\
\text{s.t.} & \displaystyle\sum_{B \in \mathcal{B}_{F_0}(u) \cap \mathcal{B}_{F_0}(v)} x_B & \geq & 1, & \text{for each } \{u, v\} \in F_1 \quad (21) \\
& x & \in & X_{\mathrm{MEBP}} & (22)
\end{array}
$$

Since we include in ($\mathrm{L^2_{MEBP}}$) only the columns in $\mathcal{B}_{F_0}$, then we have that $\lambda_{uv} = 0$, for each $\{u, v\} \in F_0$. By constraints (21) and (22), we have that $\lambda_{uv} = 1$, for each $\{u, v\} \in F_1$. Given a feasible solution x to ($\mathrm{L^2_{MEBP}}$), we define $\omega(x) := \sum_{B \in \mathcal{B}} |E_B| x_B$. Below, in Lemma 3.3, we prove that if x is fractional, then there exists an implicit variable λ_{uv} with fractional value (a candidate for branching). First, we prove the following two auxiliary lemmas.

Lemma 3.1. *Given an optimal solution x to* ($\mathrm{L^2_{MEBP}}$)*, such that x is fractional, we have that x has at least two entries with fractional value.*

Proof. Since the right hand side of all constraints in ($\mathrm{L^2_{MEBP}}$) are integral, then the result follows immediately.

Lemma 3.2. *Let x be an optimal solution to* ($\mathrm{L^2_{MEBP}}$)*, such that, for each pair of distinct variables x_{B_1} and x_{B_2} with fractional values, we have that either $S_{B_1} \cap S_{B_2} = \emptyset$ or $S_{B_1} = S_{B_2}$. Thus, there exists an optimal solution x^* to* ($\mathrm{L^2_{MEBP}}$)*, such that x^* is integral and $\omega(x) = \omega(x^*)$.*

PROOF. Let $\Psi : \mathbb{R}_+^{|\mathcal{B}|} \times \mathcal{B} \times \mathcal{B} \times \mathbb{R}_+ \Rightarrow \mathbb{R}_+^{|\mathcal{B}|}$ be the function defined as follows: $\Psi(x, B_1, B_2, \epsilon) = x'$, where $x'_{B_1} := x_{B_1} + \epsilon$, $x'_{B_2} := x_{B_2} - \epsilon$ and $x'_B := x_B$, for each $B \in \mathcal{B} \backslash \{B_1, B_2\}$. We prove the lemma by induction on n, where n is the number of variables with fractional value. If $n = 0$, then the result follows immediately. Now, suppose that $n > 0$. By Lemma 3.1, we have that $n \geq 2$. Let $x_{B_{max}}$ be a variable with fractional value, such that $|E_{B_{max}}|$ is maximum. Note that one of the following two cases must occur: (i) $S_{B_{max}} = S_{B_{eq}}$, for some other variable $x_{B_{eq}}$ with fractional value; (ii) $S_{B_{max}} \cap S_B = \emptyset$, for each other variable x_B with fractional value.

For the case (i), consider the vector $x' := \Psi(x, B_{max}, B_{eq}, \epsilon)$, where $\epsilon := \min\{1 - x_{B_{max}}, x_{B_{eq}}\}$. Since $S_{B_{max}} = S_{B_{eq}}$, then x' does not violate any constraint of ($\mathrm{L}^2_{\mathrm{MEBP}}$). Moreover, we have that $x'_{B_{max}} = 1$ or $x'_{B_{eq}} = 0$ (i.e. x' has less than n variables with fractional values), and also $\omega(x') \geq \omega(x)$, which implies $\omega(x') = \omega(x)$, since x is optimal. Thus, by the induction hypothesis, we have that there is a feasible solution x^* such that $\omega(x^*) = \omega(x') = \omega(x)$ and x^* is integral.

For the case (ii), let $x_{B_{neq}}$ be a variable with fractional value, such that $B_{max} \neq B_{neq}$. Since constraints (21) are satisfied with equality, then we have that $\{u, v\} \nsubseteq S_{B_{max}}$, for each $\{u, v\} \in F_1$. By (ii), we have that the value of the slack variable in row corresponding to u in constraints (9) is $\tau_u = 1 - x_{B_{max}}$, for each $u \in S_{B_{max}}$. Therefore, $x' := \Psi(x, B_{max}, B_{neq}, \epsilon)$ does not violate any constraint of ($\mathrm{L}^2_{\mathrm{MEBP}}$), where $\epsilon := \min\{1 - x_{B_{max}}, x_{B_{neq}}\}$. As in case (i), we have that $x'_{B_{max}} = 1$ or $x'_{B_{neq}} = 0$ and, by the induction hypothesis, there is a feasible solution x^* such that $\omega(x^*) = \omega(x') = \omega(x)$ and x^* is integral. □

Lemma 3.3. *Given an optimal solution x to ($\mathrm{L}^2_{\mathrm{MEBP}}$), such that x is fractional, we have that λ_{uv} is fractional for some $u, v \in S$ or there exists a feasible solution x^* to ($\mathrm{L}^2_{\mathrm{MEBP}}$), such that $\omega(x) = \omega(x^*)$ and x^* is integral.*

PROOF. By Lemma 3.1, there are at least two entries in x with fractional value, thus one of the following two cases must occur: (i) for each pair of distinct variables x_{B_1} and x_{B_2} with fractional values, we have that either $S_{B_1} \cap S_{B_2} = \emptyset$ or $S_{B_1} = S_{B_2}$; (ii) there are two distinct variables x_{B_1} and x_{B_2} with fractional values, such that $u \in S_{B_1} \cap S_{B_2}$ and $v \in S_{B_1} \nabla S_{B_2}$, for some $u, v \in S$, where $S_{B_1} \nabla S_{B_2} = (S_{B_1} \cup S_{B_2}) \setminus (S_{B_1} \cap S_{B_2})$.

In case (i), by Lemma 3.2, we have that there is a feasible solution x^* such that $\omega(x^*) = \omega(x)$ and x^* is integral. In case (ii), without loss of generality, suppose that $v \in S_{B_1}$ and $v \notin S_{B_2}$. Since $B_1 \in \mathcal{B}(u) \cap \mathcal{B}(v)$,

then $\sum_{B\in\mathcal{B}(u)\cap\mathcal{B}(v)} x_B \geq x_{B_1} > 0$. Since $B_2 \notin \mathcal{B}(u) \cap \mathcal{B}(v)$, $B_2 \in \mathcal{B}(u)$ and $\sum_{B\in\mathcal{B}(u)} x_B \leq 1$ (by constraints (9)), then we have that $\sum_{B\in\mathcal{B}(u)\cap\mathcal{B}(v)} x_B \leq \sum_{B\in\mathcal{B}(u)} x_B - x_{B_2} < 1$. Thus, we have that $0 < \lambda_{uv} < 1$. □

The branching rule proposed here is based on the one introduced by Ryan and Foster in [15] for the scheduling problem. For more general branching rules we refer to the work of Vanderbeck and Wolsey [20].

Now, we investigate the pricing problem with respect to ($\mathrm{L}^2_{\mathrm{MEBP}}$). The reduced price of a variable x_B is given by

$$\hat{x}_B := \beta + \sum_{v\in V_B} \alpha_v - |E_B| - \sum_{\{u,v\}\subseteq S_B\cap F_1} \gamma_{uv},$$

where γ is the vector of dual variables associated with constraints (21). Let $\phi(B,\alpha,\gamma) : \mathcal{B}_{F_0} \times \mathbb{R}_+^{|V|} \times \mathbb{R}_+^{|F_1|} \to \mathbb{R}$ be the function defined as follows: $\phi(B,\alpha,\gamma) := |E_B| + \sum_{\{u,v\}\subseteq S_B\cap F_1} \gamma_{uv} - \sum_{u\in V_B} \alpha_u$. The problem of finding a minimum reduced price variable x_B, with respect to ($\mathrm{L}^2_{\mathrm{MEBP}}$), can be stated as follows.

Problem 3.4. *Given a bipartite graph $G = (V,E)$, two sets $F_1 \subseteq S \times S$ and $F_0 \subseteq S \times S$, and two vectors $\alpha \in \mathbb{R}_+^{|V|}$ and $\gamma \in \mathbb{R}_+^{|F_1|}$, find a biclique B, subgraph of G, such that $B \in \mathcal{B}_{F_0}$ and $\phi(B,\alpha,\gamma)$ is maximum.*

We define binary variables z_u, not only for each $u \in L$ as before, but also for each $u \in R_F$, where $R_F := \{v \in R \mid v \in \{u,v\}, \text{for some } \{u,v\} \in F_1 \cup F_0\}$, and we denote by $z_{|L}$ the vector z restricted to the elements in L. On the other hand, we define variables r_v only for each $v \in \overline{R_F}$, where $\overline{R_F} := R \setminus R_F$. The variables y and l are defined exactly as in ($\mathrm{P}_{\mathrm{PRC}}$). Finally, we define $\Upsilon_{\mathrm{PRC}} := \{(z_{|L}, l, y, r) \in \{0,1\}^{|L|} \times \{0,1\}^{|L|} \times \mathbb{R}_+^{|R|} \times \mathbb{R}_+^{|\overline{R_F}|} \mid (z_{|L}, y, l, r)$ satisfies (14), (15), ..., (20) from ($\mathrm{P}_{\mathrm{PRC}}$), where (16), (17) and (19) are defined only for each $v \in \overline{R_F}\}$. Consider the following formulation for the problem 3.4.

$$
\begin{aligned}
& \max \quad \sum_{v \in R} y_v + \sum_{\{u,v\} \in F_1} \gamma_{uv} z_u - \sum_{v \in \overline{R_F}} \alpha_v r_v - \sum_{u \in L \cup R_F} \alpha_u z_u \\
(\mathrm{P}^2_{\mathrm{PRC}}) \quad & \text{s.t.} \quad z_u + z_v \leq 1, && \text{for each } \{u,v\} \in F_0 && (23)\\
& \qquad z_u = z_v, && \text{for each } \{u,v\} \in F_1 && (24)\\
& \qquad y_v \leq |\delta(v)| z_v, && \text{for each } v \in R_F && (25)\\
& \qquad \sum_{u \in \bar{\delta}(v)} z_u \leq (1 - z_v)|\bar{\delta}(v)|, && \text{for each } v \in R_F && (26)\\
& \qquad (z_{|L}, l, y, r) \in \Upsilon_{\mathrm{PRC}} && && (27)\\
& \qquad z_u \in \{0,1\}, && \text{for each } u \in L \cup R_F && (28)
\end{aligned}
$$

Given an optimal solution (z^*, y^*, r^*, l^*) to $(\mathrm{P}^2_{\mathrm{PRC}})$, let B^* be the biclique such that $u \in L_{B^*}$ if and only if $z^*_u = 1$, and $v \in R_{B^*}$ if and only if $y^*_v > 0$. For simplicity, we assume that if $v \in R_F$ and $z_v = 1$, then $y^*_v > 0$ (if it does not hold we would have $L_{B^*} = \emptyset$ and $R_{B^*} \neq \emptyset$, but the formulation works for this case as well). Constraints (23) guarantee that $B^* \in \mathcal{B}_{F_0}$. From constraints (21) in $(\mathrm{L}^2_{\mathrm{MEBP}})$ follows that $x_B = 0$ in any optimal solution, for each $B \in \mathcal{B}$ such that $u \in S_B$ and $v \notin S_B$, for some $\{u,v\} \in F_1$. Thus, we introduce constraints (24) in order to prevent the generation of such variables. Constraints (25) guarantee that if $z^*_v = 0$, then $y^*_v = 0$ as well. On the other hand, constraints (26) guarantee that $z^*_v = 1$ only if $\delta(v) \cap L_{B*} = L_{B*}$, for each $v \in R_F$. Since the value of the objective function for (z^*, l^*, y^*, r^*) is precisely $\phi(B^*, \alpha, \gamma)$, thus $(\mathrm{P}^2_{\mathrm{PRC}})$ is a formulation for the problem 3.4

Again, we can simplify the formulation of the pricing problem if $S \cap R = \emptyset$. In this case we have that $R_F = \emptyset$, thus we have no constraints (25) and (26). If for a given instance our branch-and-price procedure needs to investigate a huge number of nodes in the branch-and-bound tree, then we do not expect the program to stop in reasonable running time. Thus, we are interested in the cases in which the number of investigated nodes is relatively small (i.e. F_1 and F_0 are small and, consequently, R_F is small as well). Considering this assumption, the formulation $(\mathrm{P}^2_{\mathrm{PRC}})$ has almost as few integer (binary) variables as $(\mathrm{P}_{\mathrm{PRC}})$.

3.2. Considerations about the implementation

There are many details that have to be carefully taken into account in order to get a successful implementation of a branch-and-price algorithm. Below, we comment the issues that have more impact in our implementation, as evaluated in various experiments (not reported in this paper):

- **How to construct a first set of columns for the master LP**: we use a greedy algorithm which takes a maximum biclique B in G, add x_B in the master LP and repeat (k times) recursively the process for $G[V \setminus S_B]$. Note that this procedure also leads to a feasible solution to ($\mathrm{P_{MEBP}}$) which can be used as a lower bound on the optimal value. We refer to this procedure as the *greedy heuristic* (GH).

- **Heuristics for solving the pricing problem**: usually, MIP solvers provide callback routines for getting feasible solutions found during the optimization process, thus we include all collected solutions with negative reduced price, not only the optimal one.

- **Heuristics for getting primal bounds**: at any time we can obtain a feasible solution by solving the master IP restricted to the columns added in the master LP up to the moment and, as we noted empirically, it takes a very short time to be done. Before doing that, we make the following pre-processing: for each pair of variables x_{B_1} and x_{B_2} with positive value in the current solution, we add to master LP the columns $x_{B'_1}$ and $x_{B'_2}$, where $V_{B'_1} = V_{B_1} \setminus (S_{B_1} \cap S_{B_2})$ and $V_{B'_2} = V_{B_2} \setminus (S_{B_1} \cap S_{B_2})$. We refer to this procedure applied at the first node of the branch-and-bound tree as the *post-processing heuristic* (PPH).

- **How to select the next node to be explored in the branch-and-bound tree and how to choose a variable with fractional value for branching**: since we are leading with a maximization problem, the node's selection policy affects more directly the upper bound. As it is usually done in this case, we select a node with the biggest upper bound and branch in a more fractional variable.

4. Computational experiments

In this section we present the computational experiments with randomly generated instances and also with instances that come from the applications in metabolic networks and product bundling. We used *CPLEX© 12.1* as the MIP solver and the machine configurations are the following: 8 processors Intel© Xeon© E5440 (2.83GHz) and 32GB of RAM.

4.1. Computational experiments with the MEB *problem*

We consider two approaches for solving the MEB problem. The first one is to solve ($\mathrm{P_{MEB}}$), ($\mathrm{P^2_{MEB}}$) and ($\mathrm{P^3_{MEB}}$) in a mixed integer programming

(MIP) solver, and the second one is to use algorithms for finding a maximum clique in the complement of the graph H described in Section 2. One of the fastest known algorithms for the maximum clique problem were developed in 2002 by Patric R. J. Östergård [21]. In 2007, Janez Konc introduced another very fast algorithm for the maximum clique problem [10] based on the algorithm developed by E. Tomita and T. Seki [19]. In our experiments, the formulation (P_{MEB}) and the Östergård's algorithm (the source code is available at `http://users.tkk.fi/pat/cliquer.html`) got really slow running times compared to the other methods that we tested, thus we decided not reporting the results of applying these two methods in our instances.

We organize the results of the experiments with random generated instances in Table 4.1 (for balanced bipartite graphs) and Table 4.2 (for unbalanced bipartite graphs). In each row of Table 4.1 and Table 4.2 we show the arithmetic means between the instances solved to optimality in less than 1 hour, where the number inside brackets indicates how many instances, among the 10 generated, were solved to optimality. The first column has the density of the input graph (i.e., $\frac{|E|}{|R|\cdot|L|}$) and the column (JK) shows the running time of the Janez Konc's algorithm (the source code is available at `http://www.sicmm.org/~konc/maxclique/`).

Table 4.1 Experiments with random balanced bipartite graphs.

	$\|L\| = \|R\| = 40$			$\|L\| = \|R\| = 50$		
Dns	(P^2_{MEB})	(P^3_{MEB})	(JK)	(P^2_{MEB})	(P^3_{MEB})	(JK)
0.1	0s (10)	0s (10)	0s (10)	0s (10)	1s (10)	0s (10)
0.2	1s (10)	2s (10)	0s (10)	4s (10)	7s (10)	0s (10)
0.3	2s (10)	5s (10)	0s (10)	8s (10)	19s (10)	0s (10)
0.4	11s (10)	12s (10)	0s (10)	35s (10)	1m (10)	0s (10)
0.5	22s (10)	27s (10)	0s (10)	1m33s (10)	2m58s (10)	2s (10)
0.6	42s (10)	1m01s (10)	2s (10)	4m01s (10)	8m25s (10)	6s (10)
0.7	1m18s (10)	1m48s (10)	9s (10)	18m49s (10)	27m09s (10)	25s (10)
0.8	3m20s (10)	2m38s (10)	4m05s (10)	34m35s (05)	28m01s (03)	43m38s (05)
0.9	2m11s (10)	1m45s (10)	- (0)	27m56s (05)	24m59s (06)	- (0)
	$\|L\| = \|R\| = 60$			$\|L\| = \|R\| = 70$		
Dns	(P^2_{MEB})	(P^3_{MEB})	(JK)	(P^2_{MEB})	(P^3_{MEB})	(JK)
0.1	2s (10)	6s (10)	0s (10)	3s (10)	15s (10)	0s (10)
0.2	9s (10)	20s (10)	0s (10)	29s (10)	46s (10)	0s (10)
0.3	28s (10)	1m12s (10)	0s (10)	48s (10)	3m03s (10)	0s (10)
0.4	1m50s (10)	3m57s (10)	1s (10)	5m36s (10)	13m11s (10)	4s (10)
0.5	5m58s (10)	17m32s (10)	8s (10)	27m28s (10)	57m44s (01)	2s (10)
0.6	35m10s (09)	48m09s (02)	7s (10)	- (0)	- (0)	1m04s (10)
0.7	- (0)	- (0)	5m27s (10)	- (0)	- (0)	40m09s (04)
0.8	- (0)	- (0)	- (0)	- (0)	- (0)	- (0)
0.9	- (0)	- (0)	- (0)	- (0)	- (0)	- (0)

Table 4.2 Experiments with random unbalanced bipartite graphs.

	$\|L\| = 10$, $\|R\| = 150 \cdot \|L\|$			$\|L\| = 10$, $\|R\| = 200 \cdot \|L\|$		
Dns	(P^2_{MEB})	(P^3_{MEB})	(JK)	(P^2_{MEB})	(P^3_{MEB})	(JK)
0.1	2s (10)	24s (10)	1s (10)	3s (10)	47s (10)	3s (10)
0.2	4s (10)	19s (10)	12s (10)	4s (10)	35s (10)	1s (10)
0.3	5s (10)	17s (10)	4s (10)	5s (10)	32s (10)	13s (10)
0.4	4s (10)	19s (10)	23s (10)	8s (10)	32s (10)	1m (10)
0.5	5s (10)	23s (10)	1m19s (10)	9s (10)	39s (10)	3m11s (10)
0.6	19s (10)	26s (10)	3m39s (10)	10s (10)	44s (10)	9m02s (10)
0.7	1m36s (10)	35s (10)	9m03s (10)	1m11s (10)	56s (10)	22m30s (10)
0.8	1m19s (10)	21s (10)	32m17s (02)	59s (10)	33s (10)	53m29s (02)
0.9	5s (10)	3s (10)	- (0)	9s (10)	5s (10)	- (0)
	$\|L\| = 10$, $\|R\| = 250 \cdot \|L\|$			$\|L\| = 10$, $\|R\| = 300 \cdot \|L\|$		
Dens.	(P^2_{MEB})	(P^3_{MEB})	(JK)	(P^2_{MEB})	(P^3_{MEB})	(JK)
0.1	4s (10)	1m4s (10)	6s (10)	5s (10)	1m28s (10)	10s (10)
0.2	7s (10)	54s (10)	3s (10)	10s (10)	1m11s (10)	7s (10)
0.3	8s (10)	51s (10)	28s (10)	12s (10)	1m11s (10)	53s (10)
0.4	11s (10)	52s (10)	1m59s (10)	16s (10)	1m16s (10)	3m26s (10)
0.5	14s (10)	1m07s (10)	6m16s (10)	20s (10)	1m38s (10)	11m21s (10)
0.6	17s (10)	1m15s (10)	18m50s (10)	24s (10)	1m48s (10)	31m39s (10)
0.7	1m59s (10)	1m23s (10)	44m54s (06)	2m33s (10)	1m53s (10)	- (0)
0.8	1m20s (10)	45s (10)	- (0)	1m58s (10)	1m04s (10)	- (0)
0.9	14s (10)	7s (10)	- (0)	23s (10)	10s (10)	- (0)

The number of edges in the input graph is directly related with the size of the linear programs matrices, speaking of (P^2_{MEB}) and (P^3_{MEB}), and with the size of the graph H, speaking of (JK) algorithm. This explains the fact that in Table 4.1 and Table 4.2 the smaller is the density of the graph, the easier it becomes to solve, except for instances with density 0.9, which are easier than the instances with density 0.8. Intuitively, it seems that density 0.8 is the threshold for the instance's difficulty (note that when density is 1.0 the problem becomes trivial). Comparing Table 4.1 with Table 4.2, we conclude that (JK) algorithm is much faster than the others for balanced graphs. On the other hand, the (JK) algorithm is much slower than the others when the input graphs become more unbalanced and dense. For unbalanced graphs with density ≤ 0.3 we observe that (JK) and (P^2_{MEB}) perform similarly. For graphs with density ≤ 0.6, (P^2_{MEB}) is faster than (P^3_{MEB}), but for graphs with density ≥ 0.8, (P^3_{MEB}) is faster than (P^2_{MEB}), while for graphs with density $\simeq 0.7$ these two formulations perform almost equivalently (this behavior remains the same for balanced and unbalanced graphs).

Below, in Table 4.3, we show the experiments with instances that come from the applications. Instances PB1 and PB2 are from the application in product bundling and instances MN1 and MN2 are from the application in

metabolic networks.

Table 4.3 Experiments with instances that come from the applications.

Instance	$\|L\|$	$\|R\|$	Dns	(P^2_{MEB})	(P^3_{MEB})	(JK)
PB1	32	760	0.4	57s	4m49s	4m09s
PB2	77	760	0.3	6m53s	5h17m20s	6m10s
MN1	16	4637	0.6	12m42s	29m20s	12h36m31s
MN2	40	4637	0.4	14m36s	8h12m33s	MLE

For instances in Table 4.3 we clearly note that (P^2_{MEB}) performs quite better than the other methods, except for instance PB2, in which the input graph is the more balanced one and, as a consequence, the (JK) algorithm performs slightly better than (P^2_{MEB}). In instances from the application in metabolic networks, which are the more unbalanced ones, the (JK) performs very poorly (in instance MN2 it got a *memory limit exceeded* - MLE).

4.2. Computational experiments with MEBP *problem*

Now, we show the experiments with MEBP problem. We organize the results of the experiments with random generated instances in Table 4.4. In each row of Table 4.4 we show the arithmetic means between the instances solved to optimality in less than 1 hour among the 10 generated. All instances were generated with the same number of vertices ($|L| = 10$ and $|R| = 100$). The first two columns show the parameters k and S (we tested several values for these parameters and selected the ones we consider more significant), while the third column has the density of the input graph. The next four columns have the time spent to run the greedy heuristic "GH", to solve the master linear program "LP", to run the post-processing heuristic "PPH" and to solve the integer program with branch-and-price "IP". The next two columns have the number of pricing problems solved "Prc" and the number of explored nodes in the branch-and-bound tree "Nd". The next two columns "GH" and "PPH" have the gap between the heuristics solutions and the optimal solution, respectively. Column "OPT" show the percentage of instances solved to optimality. The next two columns show the percentage of instances not solved to optimality, where column "LP" corresponds to instances for which even the LP was solved and column "IP" corresponds to instances for which the timeout occurred after solving the LP. Finally, the last column "Gap" shows the gap between the current upper and lower bounds when the timeout occurred.

Table 4.4 Experiments with random unbalanced bipartite graphs.

			Time				Number of		Gap			Timeout		
S	k	Dns	GH	LP	PPH	IP	Prc	Nd	GH	PPH	OPT	LP	IP	Gap
		0.3	0s	0s	0s	0s	2	0	0%	0%	100%	-	-	-
	3	0.5	1s	1s	0s	0s	3	0	0%	0%	100%	-	-	-
		0.8	0s	5s	0s	0s	17	0	15%	0%	100%	-	-	-
		0.3	1s	0s	0s	0s	2	0	0%	0%	100%	-	-	-
L	6	0.5	1s	1s	0s	0s	5	0	1%	0%	100%	-	-	-
		0.8	0s	2s	0s	0s	21	0	56%	0%	100%	-	-	-
		0.3	1s	0s	0s	0s	2	0	0%	0%	100%	-	-	-
	9	0.5	1s	1s	0s	0s	7	0	34%	0%	100%	-	-	-
		0.8	2s	2s	0s	0s	21	0	75%	0%	100%	-	-	-
		0.3	0s	41s	1s	4m19s	522	15	9%	1%	80%	20%	-	-
	3	0.5	1s	1m57s	1s	7m43s	365	13	8%	1%	90%	10%	-	-
		0.8	1s	13m14s	1s	9m15s	373	2	8%	0%	90%	-	10%	1%
		0.3	0s	1m11s	1s	5m14s	586	21	32%	2%	60%	10%	30%	2%
R	6	0.5	1s	5m18s	3s	14m46s	452	14	25%	0%	30%	-	70%	1%
		0.8	1s	9m42s	2s	24m27s	555	14	23%	1%	30%	-	70%	1%
		0.3	1s	1m12s	2s	8m10s	495	35	47%	2%	80%	-	20%	2%
	9	0.5	1s	6m56s	4s	18m40s	512	20	41%	0%	40%	-	60%	2%
		0.8	3s	15m8s	3s	22m18s	569	21	31%	1%	60%	10%	30%	1%
		0.3	0s	36s	1s	2m01s	387	11	9%	1%	80%	20%	-	-
	3	0.5	1s	1m33s	1s	10m37s	464	15	7%	1%	90%	-	10%	2%
		0.8	0s	37m49s	14s	16m31s	883	2	3%	0%	50%	10%	40%	3%
		0.3	0s	1m15s	3s	7m36s	633	24	27%	3%	50%	10%	40%	4%
V	6	0.5	0s	3m55s	1s	0s	270	0	7%	0%	10%	-	90%	6%
		0.8	0s	37m57s	18s	9m17s	941	0	1%	0%	30%	20%	50%	3%
		0.3	0s	1m21s	44s	8m15s	521	9	25%	3%	80%	-	20%	5%
	9	0.5	1s	3m25s	1s	0s	283	0	9%	0%	10%	-	90%	6%
		0.8	0s	43m13s	21s	7m18s	1006	0	1%	0%	30%	30%	40%	3%

As shown in Table 4.4, the results are excellent for $S = L$. In this case, all instances were solved to optimality at the root node of the branch-and-bound tree and the number of pricing problems solved as well as the total time needed for solving these instances was quite small. Intuitively, since $S = L$ is small, the number of intersecting bicliques tend to be smaller than in other cases. In all choices of S, the smaller is the density of the graph, the easier it becomes to solve. The case in which $S = V$ is slightly more difficult than when $S = R$. This can be better seen if we consider the last three columns. Some values in Table 4.4 should be carefully analyzed for not being misinterpreted. For example, when $k = 6$ (and $k = 9$), dens= 0.5 and $S = V$, just one instance was solved to optimality and in those cases the time spent to solve the IP (after column generation) was 0 seconds. In cases

like these the last three columns express better what happened in the tests.

The gap between GH and the optimal value is reasonable good in most cases, while the solutions produced by PPH are in all cases very close to optimal. On the other hand, the time needed by GH is quite shorter than the time needed by PPH (note that PPH can be executed only after column generation terminates). Moreover, in most cases the time needed to prove that the solution found by PPH is optimal is very higher than the time needed to find the solution itself. The value of k and the density of the input graph seem to have influence on the quality of the solutions found by GH. In all cases, the bigger is k, the worse is the gap. When $S = L$, GH obtains worse gaps when the input graph is more dense. On the other hand, when $S = V$ or $S = R$, GH obtains worse gaps when the input graph is less dense.

Below, in Table 4.5, we show the experiments with instances that come from the applications in product bundling and metabolic networks. In instances with $S = R$ or $S = V$ we could not solve even the master linear program for any value of k (we stopped the program after 48 hours of processing). Thus, we consider only the case in which $S = L$. We solved the instances for some values of k and calculated the arithmetic means of the results.

Table 4.5 Experiments with instances that come from the applications.

	Input Size			Time				Number of		Gap	
Inst	$\|L\|$	$\|R\|$	Dns	GH	LP	PPH	IP	Prc	Nd	GH	PPH
PB1	32	760	0.4	5m24s	19m59s	0s	23s	31	1	27%	0%
PB2	77	760	0.3	6h26m24s	30h58m37s	0s	0s	16	0	8%	0%
MN1	16	4637	0.6	28m50s	1h10m59s	0s	0s	11	0	17%	0%
MN2	40	4637	0.4	1h26m28s	11h29m9s	0s	0s	12	0	5%	0%

As shown in Table 4.5, the number of pricing problems solved in all instances were very small and the optimal solutions were found at the root node (except in instance PB1 for which it was necessary to explore only one more node). The hardest instance was PB2, in which $|L|$ is the biggest one.

The data for the application in product bundling comes from a major food company in Chile. It was constructed from the historical weekly demand of 760 small retailer over 497 SKU. After preprocessing the data, we obtain an instance of 77 clients and 760 products (instance PB2). The interest of the company was to construct at most five product bundles. The resulting five bundles correspond to 44% of the total demand.

5. Concluding remarks and future work

We presented new formulations for MEB and MEBP problems and evaluated them with computational experiments. Moreover, we described two important applications of the MEBP problem and solved instances that come from these applications. Our formulations were specially designed for instances in which the input graphs are very unbalanced and, as shown in the computational experiments, we obtained good results for this case. Our experiments also show that pure combinatorial algorithms for the maximum clique problem can be successfully applied in order to solve small instances of the MEB problem.

Future work can be done on the MEB problem by developing pure combinatorial algorithms for solving it (adapting the algorithms for the clique problem seems a promising idea). Perhaps, such algorithm can be adapted in order to improve the performance of our branch-and-price algorithm, since the MEB problem is strongly related with the pricing problem of our formulation to MEBP problem.

References

[1] Acuña, V., Ferreira, C., Freire, A., Moreno, E., 2010. The biclique k-clustering problem in bipartite graphs and its application in bioinformatics. Electronic Notes in Discrete Mathematics 36, 159 – 166, ISCO 2010 - International Symposium on Combinatorial Optimization.

[2] Balas, E., Ceria, S., Cornuéjols, G., Pataki, G., 1993. Cliques, Coloring and Satisfiability: Second DIMACS Implementation Challenge. Vol. 26 of DIMACS Series in Discrete Mathematics and Theoretical Computer Science. Ch. Polyhedral Methods for the Maximum Clique Problem.

[3] Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W., Vance, P. H., 1998. Branch-and-price: Column generation for solving huge integer programs. Operational Research (3), 316–329.

[4] Berman, P., Schnitger, G., 1992. On the complexity of approximating the independent set problem. Information and Computation 96 (1), 77 – 94.

[5] Dawande, M., Keskinocak, P., Tayur, S., 1997. On the biclique problem in bipartite graphs. Tech. rep., Carnegie-Mellon University.

[6] Faure, N., Chrétienne, P., Gourdin, E., Sourd, F., 2007. Biclique completion problems for multicast network design. Discrete Optimization 4 (3-4), 360 – 377.

[7] Feige, U., 2002. Relations between average case complexity and approximation complexity. In: STOC '02: Proceedings of the thiry-fourth annual ACM symposium on Theory of computing. ACM, New York, NY, USA, pp. 534–543.

[8] Feige, U., Kogan, S., 2004. Hardness of approximation of the balanced complete bipartite subgraph problem. Tech. rep.

[9] Garey, M. R., Johnson, D. S., 1979. Computers and Intractability, A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, New York.

[10] Konc, J., Janezic, D., 2007. An improved branch and bound algorithm for the maximum clique problem. MATCH Communications in Mathematical and in Computer Chemistry 58, 569–590.

[11] Madeira, S. C., Oliveira, A. L., 2004. Biclustering algorithms for biological data analysis: A survey. IEEE/ACM Transactions on Computational Biology and Bioinformatics 1 (1), 24–45.

[12] Notebaart, R. A., Teusink, B., Siezen, R. J., Papp, B., 2008. Coregulation of metabolic genes is better explained by flux coupling than by network distance. PLoS Comp. Biol. (4), 157–63.

[13] Nussbaum, D., Pu, S., Sack, J.-R., Uno, T., Zarrabi-Zadeh, H., 2010. Finding maximum edge bicliques in convex bipartite graphs. In: Thai, M., Sahni, S. (Eds.), Computing and Combinatorics. Vol. 6196 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 140–149.

[14] Peeters, R., 2003. The maximum edge biclique problem is NP-complete. Discrete Applied Mathematics, 651–654.

[15] Ryan, D. M., Foster, B. A., 1981. An integer approach to scheduling. Computer Scheduling of Public Transport, 269–280.

[16] Schuster, S., Hilgetag, C., 1994. On elementary flux modes in biochemical reaction systems at steady state. Journal of Biological Systems 2 (2), 165–182.

[17] Tan, J., 2008. Inapproximability of maximum weighted edge biclique and its applications. In: TAMC'08: Proceedings of the 5th international conference on Theory and applications of models of computation. Springer-Verlag, Berlin, Heidelberg, pp. 282–293.

[18] Terzer, M., Stelling, J., 2008. Large-scale computation of elementary flux modes with bit pattern trees. Bioinformatics 24 (19), 2229–2235.

[19] Tomita, E., Seki, T., 2003. Discrete Mathematics and Theoretical Computer Science. Vol. 2731/2003 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, Ch. An Efficient Branch-and-Bound Algorithm for Finding a Maximum Clique, pp. 279–289.

[20] Vanderbeck, F., Wolsey, L. A., 1996. An exact algorithm for IP column generation. Operations Research Letters 19 (4), 151 – 159.

[21] Östergård, P. R. J., 2002. A fast algorithm for the maximum clique problem. Discrete Applied Mathematics 120, 197–207.