The k-Regular Induced Subgraph Problem

Agostinho Agra[1], Geir Dahl[2], Torkel A. Haufmann[2], Sofia J. Pinheiro[1]

[1] *CIDMA, Dept. of Mathematics, University of Aveiro, Portugal*
[2] *Dept. of Mathematics, University of Oslo, Norway*

To appear in *Discrete Applied Mathematics*.

Abstract

We consider the problem of finding a maximum k-regular induced subgraph of a graph G. Theoretical results are established to compare upper bounds obtained from different techniques, including bounds from quadratic programming, Lagrangian relaxation and integer programming.

This general problem includes well-known subproblems as particular cases of k. In this paper we focus on two particular cases. The case $k = 1$ which is the maximal cardinality strong-matching and the case of finding the maximal cardinality family of induced cycles ($k = 2$). For each one of the two cases, combinatorial algorithms are presented to solve the problem when graphs have particular structures and polyhedral descriptions of the convex hull of the corresponding feasible set are given. Computational tests are reported to compare the different upper bounds with the optimal values for different values of k, and to test the effectiveness of the inequalities introduced.

keyword: regular induced graphs; strong-matchings; combinatorial optimization; integer programming;

1 Introduction

Let $G = (V, E)$ be a simple undirected graph where $V = V(G)$ denotes the vertex set and $E = E(G)$ is the nonempty edge set. We denote an edge by $e = uv$, where u and v are the end vertices. The neighborhood of a vertex $v \in V(G)$, that is, the set of vertices adjacent to v, is denoted by $N(v)$. The adjacency matrix of a graph G is denoted by A_G and its eigenvalues are denoted, in nonincreasing order, by $\lambda_1 \geq \cdots \geq \lambda_n$. The subgraph of G induced by the vertex subset $S \subseteq V(G)$ is denoted by $G[S]$. A graph is said to be *k-regular* when all its vertices have degree k.

In this paper we study the problem of finding a maximum k-Regular Induced Subgraph (k-RIS) of a graph G.

The k-RIS problem can be modeled as the following integer programming problem whose optimal value equals the maximum size of a k-regular subgraph of G. The variable x_v indicates whether vertex v is selected or not, and y_e indicates whether the edge e is selected or not.

$$
\begin{array}{lll}
(IP) & \max \sum_{u \in V} x_u & \\
& \text{subject to} & \\
(i) & \sum_{v:uv \in E} y_{uv} = k x_u & (u \in V), \\
(ii) & y_{uv} \leq x_u & (uv \in E), \\
(iii) & y_{uv} \leq x_v & (uv \in E), \\
(iv) & x_u + x_v \leq 1 + y_{uv} & (uv \in E), \\
(v) & y_{uv} \in \{0,1\} & (uv \in E), \\
(vi) & x_u \in \{0,1\} & (u \in V).
\end{array} \tag{1}
$$

The objective function is to maximize the number of selected vertices. The equations $\sum_{v:uv \in E} y_{uv} = k x_u$ ensure that there are exactly k edges incident to each vertex u, if the vertex is selected. The constraints $y_{uv} \leq x_u$, and $y_{uv} \leq x_v$ ensure that if an edge uv is selected then vertex u and vertex v must also be selected. The constraints $x_u + x_v \leq 1 + y_{uv}$ ensure that the selected subgraph is induced, that is, if vertices u and v are selected then the edge uv must be selected.

For some particular values of k one obtains well known problems. A vertex subset inducing a 0-regular subgraph is called an *independent (or stable) set*. A maximum independent set is an independent set of maximum cardinality and its cardinality is called the *independence number* and denoted by $\alpha(G)$. Thus, for $k = 0$, finding the maximum 0-RIS of G is to find the independence number of G. For $k = 1$ a 1-RIS is known as a *strong-matching* or *induced matching*. A strong-matching is an edge subset $M \subseteq E$ such that the induced subgraph $G[M]$ is a matching. For a matching M this is equivalent to saying that for any pair of distinct $e,\ f \in M$ there is no edge in E which joins an end vertex of e and an end vertex of f. For $k = 2$ the 2-RIS problem is to find a set of chordless cycles. When $k = |V|$ a graph G is k-RIS if and only if it is complete.

The problem of finding a maximum stable set has many real-life applications in areas like computer vision, economics, biomedical engineering (see [12] for some of these applications). Strong-matchings are studied in [11] in connection with an optimization problem concerning mobile communication systems and channel minimization. The 2-RIS is closely related to practical problems that can be modeled as finding cycles in direct graphs, such as the kidney exchange problem (see [8]), where vertices represent donors (each donors is associated to a patient) and arcs represent compatibility between the donors and the patient associated to other donors. The goal is to involve the maximum number of donors in directed cycles. Creating large cycles should be avoided since any failure will break the kidney exchange chain. Therefore, induced cycles would have a practical mean.

Cardoso, Kamiǹski, and Lozin [4] proved that finding a maximum k-RIS is NP-hard for any fixed value of k. A similar result appears in [28].

Due to the NP-hardness of the k-RIS problem, polynomial time bounds were deduced for some particular cases. There exists many bounds on the independence number. The most popular is the ratio bound obtained by Hoffman (unpublished) and published by Lovász in [22] for regular graphs. Other bounds on the independence number were introduced, for some upper bounds see [9, 14, 17, 23, 24] and for some lower bounds see [16, 18, 19, 33]. The study of the general case of k-regular induced subgraph problem is more recent. Some upper bounds on the order of k-regular induced subgraphs appear in [4, 5, 6, 17, 25].

This paper makes the following contributions: (i) A theoretical result is given to compare different upper bounds for the k-RIS problem. This result compares the polynomial time bound resulting from spectral graph theory with bounds resulting from quadratic programming and Lagrangian relaxation. (ii) for strong-matchings, a combinatorial upper bound is given and, when the graph is a tree, the bound is proved to be tight and a proof of the integrality of an integer formulation is provided. Additionally, valid inequalities are discussed for the general case. (iii) A polynomial time algorithm is presented for the 2-RIS problem when G has a special structure, and valid inequalities are introduced for the general case. (iv) A computational study is presented to test the discussed formulations for the particular cases of $k = 1$ and $k = 2$ and, for the general case, the exact value of k-RIS is compared against several upper bounds.

The remaining document is organized as follows: In Section 2 we do a comparison between the optimal solution of (1) and several upper bounds resulting from different techniques. In Section 3 the case of strong-matchings is discussed. Then, the case $k = 2$ is considered in Section 4. Finally, in Section 5 some computational experiments are presented and analysed.

2 A theoretical comparison of upper bounds for the k-RIS problem

In this section we introduce a relation between the optimal solution of (1) and some upper bounds that are presented next. Cardoso *et al.* [4] introduced the following family of convex quadratic programming problems depending on a parameter k:

$$\upsilon_k(G) = \max_{x \geq 0} \; 2\hat{e}^T x - \frac{\tau}{k+\tau} x^T \left(\frac{A_G}{\tau} + I_n \right) x, \tag{2}$$

where $\hat{e}$ is the all ones vector, I_n the identity matrix of order n, k is a nonnegative integer and $\tau = -\lambda_n(A_G)$. It was proved that $\upsilon_k(G)$ is an upper bound on the size of a k-regular induced subgraph, and that if (2) has a binary optimal solution, then the corresponding characteristic vector induces a maximum k-regular subgraph. Note that problem (2) is a convex optimization problem.

A better bound can be obtained from the quadratic problem by restricting the variables to be binary.

$$(QIP) \max_{x\in\{0,1\}^n} \left\{ 2\hat{e}^T x - \frac{\tau}{k+\tau} x^T \left(\frac{A_G}{\tau} + I_n \right) x \right\}. \tag{3}$$

Notice that $\upsilon_k(G)$ is obtained by relaxing both the integrality constraints and the upper bound constraints $x_j \leq 1$ from (QIP).

Another upper bound for the k-RIS problem can be obtained through the following Lagrangian relaxation obtained by relaxing constraint (1.i):

$$\begin{aligned} (LRIP_\lambda) \max \quad & \sum_{u\in V} x_u + \sum_{u\in V} \lambda_u (k x_u - \sum_{v:uv\in E} y_{uv}) \\ s.t. \quad & (1.ii) - (1.vi). \end{aligned}$$

Let $(DLIP)$ define the dual problem

$$(DLIP) \min_{\lambda\in\mathbb{R}^n} LRIP_\lambda$$

Next we relate these bounds. Let $V(P)$ denote the value of a problem P.

Theorem 2.1 *$V(IP) \leq V(DLIP) \leq V(LRIP_{\lambda^*}) = V(QIP) \leq \upsilon_k(G)$, where λ^* are the multipliers defined by $\lambda^*_u = \frac{1}{k+\tau}, u = 1, \ldots, n$.*

Proof. The first inequality $V(IP) \leq V(DLIP)$ follows from weak duality. The inequality $V(DLIP_\lambda) \leq V(LRIP_{\lambda^*})$ is trivial since $DLIP_\lambda = \min_\lambda LRIP_\lambda$. Next we show $V(LRIP_{\lambda^*}) = V(QIP)$.

Since

$$\begin{aligned} \hat{e}^T x &= \sum_{u\in V(G)} x_u \\ x^T A_G x &= \sum_{u\in V(G)} x_u \left(\sum_{v\in N(u)} x_v \right) \\ x^T x &= \sum_{u\in V(G)} x_u^2 \end{aligned}$$

we may rewrite (QIP) as follows:

$$\max_{x\in\{0,1\}^n} \left\{ 2 \sum_{u\in V(G)} x_u - \frac{1}{k+\tau} \sum_{u\in V(G)} x_u \sum_{v\in N(u)} x_v - \frac{\tau}{k+\tau} \sum_{u\in V(G)} x_u^2 \right\}.$$

As $x_u \in \{0,1\}$, $x_u^2 = x_u$ and (QIP) becomes

$$\max_{x\in\{0,1\}^n} \left\{ (2 - \frac{\tau}{k+\tau}) \sum_{u\in V(G)} x_u - \frac{1}{k+\tau} \sum_{u\in V(G)} x_u \sum_{v\in N(u)} x_v \right\}$$

We introduce an intermediate model denoted by (LQIP), which is a linearization of (QIP). Define variables y_{uv} which are 1 if x_u and x_v are 1 and 0 otherwise. Then $y_{uv} = x_u x_v$, so

$$\sum_{u \in V} x_u \sum_{v \in N(u)} x_v = \sum_{u \in V} \sum_{v \in N(u)} y_{uv} = \sum_{uv \in E} 2y_{uv},$$

and (LQIP) is defined as follows:

$$\begin{array}{ll} (LQIP)\max & \left\{ (2 - \frac{\tau}{k+\tau}) \sum_{u \in V} x_u - \dfrac{2}{k+\tau} \sum_{uv \in E} y_{uv} \right\}. \\ s.t. & (1.ii) - (1.vi). \end{array}$$

There is a one-to-one correspondence between an incidence vector $x \in \{0,1\}^n$ and the vector (x, y) satisfying (1.ii) - (1.vi), such that the objective function value of x in (QIP) coincides with the objective function value of (x, y) in $(LQIP)$, so $V(LQIP) = V(QIP)$.

Now, we show that the objective function of $V(LQIP)$ coincides with the objective function of $V(LRIP_{\lambda^*})$.

$$\begin{aligned} & (2 - \frac{\tau}{k+\tau}) \sum_{u \in V} x_u - \frac{2}{k+\tau} \sum_{uv \in E} y_{uv} \\ = & \sum_{u \in V} x_u + \frac{k}{k+\tau} \sum_{u \in V} x_u - \frac{2}{k+\tau} \sum_{uv \in E} y_{uv} \\ = & \sum_{u \in V} x_u + \sum_{u \in V} \frac{1}{k+\tau} \left(k x_u - \sum_{v : uv \in E} y_{uv} \right) \end{aligned}$$

Finally, the inequality $V(QIP) \leq \upsilon_k(G)$ is due to the fact that $\upsilon_k(G)$ is obtained from (QIP) by relaxing the integrality constraints and the upper bound. □

Note that there are examples for which the inequalities given in the statement of Theorem 2.1 are strict, as we will see in Section 5.

The bound $\upsilon_k(G)$ can be obtained in polynomial time, while the complexity of problems $(LRIP_\lambda)$ and (QIP) is still open. Given our computational experiments we conjecture that these problems are NP-hard.

Next we discuss the computational complexity of the Lagrangian relaxation (LRIP$_\lambda$), so consider this problem for a given multiplier vector λ. The variables are (x, y) and the constraints are (1.ii)-(1.vi). Due to these constraints, as explained above, the edge variables y are determined by the vertex variables x as follows

$$y_{uv} = x_u x_v \quad (uv \in E).$$

So we eliminate y, and then the remaining variables x are only constrained to be boolean, and (LRIP$_\lambda$) becomes

$$\max_{x\in\{0,1\}^n}\left\{\sum_{u\in V}x_u+\sum_{u\in V}\lambda_u(kx_u-\sum_{v:uv\in E}x_ux_v)\right\} \tag{4}$$

or, equivalently,

$$\max_{x\in\{0,1\}^n}\left\{\sum_{u\in V}(1+k\lambda_u)x_u-\sum_{u\in V}\lambda_u\sum_{v:uv\in E}x_ux_v\right\}.$$

This may be seen (by switching to minimization) as a boolean quadratic program, i.e., a problem of the form

$$\min_{x\in\{0,1\}^n}\left\{\sum_{i=1}^{n}a_ix_i-\sum_{i\neq j}b_{ij}x_ix_j\right\}. \tag{5}$$

It is well-known (see [30]) that (5) can be solved in polynomial time, as a certain minimum cut problem, provided that $a_i\geq 0$ for each i and $b_{ij}\geq 0$ for each $i\neq j$. Unfortunately, these sign conditions do not hold in (4) as each component in λ^* is $1/(k+\tau)$ which is positive. Actually, the reformulation of the boolean quadratic problem into an *st*-cut problem in [30] then gives a minimum *st*-cut problem with negative edge weights. This means that we have a max cut problem. Although this argument does not prove that (LRIP$_{\lambda^*}$) is NP-hard (as there is some structure in this max cut reformulation), it still suggests that this may be the case. Perhaps it even may be proved by investigating the complexity of certain restricted max cut problems.

3 Strong-matchings

In this section we discuss the case $k=1$, which corresponds to finding the largest induced matching in a graph G.

Let the *star* $\delta(v)$ denote the set of edges incident to v, i.e., having v as an end vertex. For each edge $e=uv\in E$ we define the *2-star* $\delta^2(e)=\delta(u)\cup\delta(v)$. Then an edge subset M is a strong-matching if and only if the 2-stars $\delta^2(e)$ ($e\in M$) are pairwise disjoint. Thus, the problem of finding a maximum size strong-matching in G is equivalent to the following *packing* problem of 2-stars:

$$\begin{array}{l}\text{find } M\subseteq E \text{ of maximum size such that the 2-star}\\ \text{family } \delta^2(e)\ (e\in M) \text{ is pairwise disjoint.}\end{array} \tag{6}$$

The Strong-Matching (SM) problem can be modeled as a binary program as

follows.

$$
\begin{array}{lll}
(SM) & \max \sum_{uv \in E} y_{uv} & \\
& \text{subject to} & \\
(i) & \sum_{uv \in \delta^2(f)} y_{uv} \leq 1 & (f \in E), \\
(ii) & y_{uv} \in \{0,1\} & (uv \in E),
\end{array}
\tag{7}
$$

Except for the objective function, the SM model results from k-RIS by eliminating the binary variables x_u. Maximizing the number of edges of the selected subgraph is equivalent, for $k = 1$, to maximizing the number of vertices, since the number of vertices of a strong-matching is twice the number of edges.

3.1 A combinatorial upper bound

Here we show a result providing a combinatorial upper bound for (6). We introduce a closely related problem which is a *covering* problem for 2-stars:

$$
\begin{array}{l}
\text{find } F \subseteq E \text{ of minimum size such that the 2-star} \\
\text{family } \delta^2(f) \ (f \in F) \text{ covers } E, \text{ i.e, } \cup_{f \in F} \delta^2(f) = E.
\end{array}
\tag{8}
$$

Let $\gamma^p(G)$ denote the maximum in (6) and $\gamma^c(G)$ denote the minimum in (8). Then we obtain the following relation between the two problems.

Lemma 3.1 *In any graph G,*

$$\gamma^p(G) \leq \gamma^c(G). \tag{9}$$

Proof. Let $M \subseteq E$ be such that $\delta^2(e)$ $(e \in M)$ are pairwise disjoint. Assume that $F \subseteq E$ satisfies $\cup_{f \in F} \delta^2(f) = E$. Then each $e \in M$ lies in a 2-star $\delta^2(f)$ for some $f = f(e) \in F$, and therefore $f \in \delta^2(e)$. Since these sets $\delta^2(e)$ $(e \in M)$ are pairwise disjoint, it follows that the edges $f(e)$ $(e \in M)$ are distinct, so $|M| \leq |F|$. □

We may have strict inequality in (9), even for very small graphs. For instance, if C_4 is the 4-cycle (a cycle with four vertices), then $\gamma^p(C_4) = 1$ and $\gamma^c(C_4) = 2$.

One may ask for graphs such that equality holds in (9). The C_4 example shows that this is not the case for bipartite graphs. However, for trees we have a positive answer, as the next result says. This result may be viewed as a combinatorial min-max theorem for this situation.

Theorem 3.1 *Let G be a tree. Then*

$$\gamma^p(G) = \gamma^c(G).$$

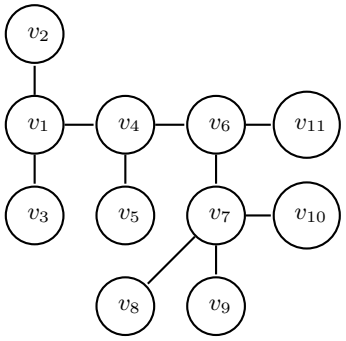
Proof. Recall that a leaf is a vertex of degree one. Let v be a vertex. If all neighbors of v, except at most one vertex, are leaves, we call v a *near-leaf*. It is easy to show that every nontrivial tree has at least two near-leaves (by looking at a longest path in the tree).

Consider the following algorithm:

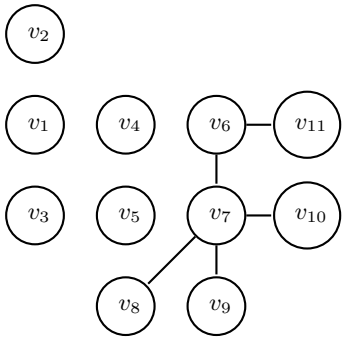
1. Initially, let $T = G$ and $M = F = \emptyset$.
2. Choose a connected component T' of T and
   (i) select a near-leaf v in T', and among the neighbors of v choose a leaf u and a non-leaf w, if it exists; otherwise use $w = u$;
   (ii) update M and F by $M := M \cup \{uv\}$ and $F := F \cup \{vw\}$, and
   (iii) update T by removing the 2-star $\delta^2(vw)$ from T.

Note that in Step 2(iii) the number of connected components of T may increase (so T is a forest). The algorithm is well-defined and when it terminates, clearly $|M| = |F|$. In each iteration the 2-star $\delta^2(vw)$ contains the edges removed, including in the last iteration. So, at termination $\cup_{f \in F} \delta^2(f) = E$. Moreover, the constructed M is a strong-matching since, in each iteration, the edge uv we add to M is one step further away from other edges in M than vw is, and we remove $\delta^2(vw)$ from T. Therefore, the result now follows from Lemma 3.1. □

Example: Let us consider the graph G depicted at left, in Figure 1a.



(a) Graph G.



(b) Graph T: G without edges of $\delta^2(v_1v_4) = \{v_1v_2, v_1v_3, v_1v_4, v_4v_5, v_4v_6\}$.

Figure 1: Graph G and the updated graph T following the algorithm given in proof of of Theorem 3.1.

Let $T = G$, $M = F = \emptyset$. Since T is connected we set $T' = T$. Then we select the near-leaf $v = v_1$ in T' and, among the neighbors of v, the leaf

$u = v_2$ and the non-leaf $w = v_4$. Then $M = \{v_2v_1\}$ and $F = \{v_1v_4\}$. Since $\delta^2(v_1v_4) = \{v_1v_2, v_1v_3, v_1v_4, v_4v_5, v_4v_6\}$ we get the graph depicted in Figure 1b.

Now, we select the connected component T' of T such that

$$V(T') = \{v_6, v_7, v_8, v_9, v_{10}, v_{11}\}$$

and

$$E(T') = \{v_6v_7, v_6v_{11}, v_7v_8, v_7v_9, v_7v_{10}\}.$$

Then we choose the near-leaf $v = v_6$ in T' and, among the neighbors of v, the leaf $u = v_{11}$ and the non-leaf $w = v_7$. Then $M = \{v_2v_1, v_6v_{11}\}$ and $F = \{v_1v_4, v_6v_7\}$. Since $\delta^2(v_6v_7) = \{v_6v_{11}, v_6v_7, v_7v_8, v_7v_9, v_7v_{10}\}$ we obtain a graph T with $V(T) = V(G)$ and $E(T) = \emptyset$. So, the algorithm stops. The strong-matching obtained is $M = \{v_2v_1, v_6v_{11}\}$ and $\gamma^p(G) = |M| = 2$, $\gamma^c(G) = |F| = 2$, that is $\gamma^p(G) = \gamma^c(G)$. □

Let

$$Y^{SM} = \{y \in \mathbb{R}^E : \sum_{uv \in \delta^2(f)} y_{uv} \leq 1 \ (f \in E), y_{uv} \in \{0,1\} \ (uv \in E)\}$$

and

$$Z^{SM} = \{z \in \mathbb{R}^E : \sum_{f \in \delta^2(uv)} z_f \geq 1 \ (uv \in E), z_f \in \{0,1\} \ (f \in E)\}.$$

We denote by Y_L^{SM} and Z_L^{SM} the sets obtained from Y^{SM} and Z^{SM}, respectively, by relaxing the integrality constraints. From linear programming theory, since the two problems in the middle are dual linear programming problems, we may establish the following relation between $\gamma^p(G)$ and $\gamma^c(G)$:

$$\begin{aligned} \gamma^p(G) = \max\{\sum_{uv \in E} y_{uv} : y \in Y^{SM}\} &\leq \max\{\sum_{uv \in E} y_{uv} : y \in Y_L^{SM}\} \\ = \min\{\sum_{f \in E} z_f : z \in Z_L^{SM}\} &\leq \min\{\sum_{f \in E} z_f : z \in Z^{SM}\} = \gamma^c(G) \end{aligned} \quad (10)$$

Theorem 3.1 implies the following equality when G is a tree:

$$\max\{\sum_{uv \in E} y_{uv} : y \in Y^{SM}\} = \max\{\sum_{uv \in E} y_{uv} : y \in Y_L^{SM}\}. \quad (11)$$

One may ask if this equality follows by total unimodularity of the coefficient matrix (see [32]). The situation is very special because the coefficient matrix, say A, is square and symmetric: it is the so-called edge-edge adjacency matrix of the given tree. However, A may not be totally unimodular, even for trees (see [27] for this fact, and for some properties of such matrices). For instance, it is shown that A is totally unimodular when T is a caterpillar. Thus, Theorem 3.1 does not follow from these general principles.

On the other hand, our direct proof (Theorem 3.1) is both short and constructive; it contains a simple combinatorial algorithm for solving both the packing and the covering problem.

3.2 Integrality of vertices of Y_L^{SM} when G is a tree

In this section we prove that all vertices in Y_L^{SM} are integral when the graph is a tree, hence $Y_L^{SM} = \text{conv}(Y^{SM})$ ($\text{conv}(Y^{SM})$ denotes the convex hull of Y^{SM}), using the theory of perfect graphs and exploring the relation between strong-matching in G and cliques in the graph $L(G)^2$ [3].

Recall that the *line-graph*, $L(G) = (E, E')$, of a graph G has vertex set E, with two vertices being adjacent if and only if the corresponding edges of G have a common vertex, and the square, G^2, of a graph G has vertex set V, and two vertices are joined in G^2 exactly when they are joined by an edge or a path of two edges in G. Hence $L(G)^2$ is a graph with vertex set E, and an edge between two vertices if they share a vertex or are connected by an edge in G, that is, two edges are adjacent whenever they lie in the same 2-star. The *clique matrix* of a graph G is the $(0,1)-$incidence matrix whose rows correspond to all of the cliques of G and whose columns correspond to the vertices of G.

The constraints $y_{uv} \leq 1$ are redundant in the description of Y_L^{SM}, and may be omitted.

Theorem 3.2 $Y_L^{SM} = \text{conv}(Y^{SM})$ *when the graph G is a tree.*

Proof. We assume that G is not a star (otherwise integrality is trivial). Delete the constraints corresponding to leaves (they are dominated by other constraints), and let A be the corresponding coefficient matrix. Then A is the (maximal) clique matrix of the graph $L(G)^2$. Thus, the columns of A correspond to edges of G and the rows to the maximal cliques. Now, $L(G)^2$ is a chordal (triangulated) graph: For if C is a cycle in $L(G)^2$ without a chord making a triangle, then C contains precisely two nonadjacent edges in every 2-star it intersects; this is impossible when G is a tree. But every chordal graph is a perfect graph, and then $\text{conv}(Y^{SM})$ equals the convex hull of its integral points (see Theorem 3.14 and Theorem 4.11 in [15]). □

We remark that we have a direct proof of this theorem, but it is rather long and technical.

We also note that Theorem 3.1 may be derived from this proof.

3.3 Valid inequalities for the set of strong-matchings

Now we investigate a class of valid inequalities for Y^{SM}, for a general graph G, using the theory of independence systems.

First we observe that the set of strong-matchings

$$\mathcal{F} = \{M \subseteq E : \text{ the induced subgraph } G[M] \text{ is a strong-matching}\}$$

is an Independence System. Recall that an *Independence System* (IS) is characterized by the following inclusion property:

(P) If $J \in \mathcal{F}$ and $I \subseteq J$, then $I \in \mathcal{F}$.

The elements in $\mathcal{F}$ are called *independent* and those $J \subseteq E$ such that $J \notin \mathcal{F}$ are called *dependent*.

As a consequence of this observation we can exploit the known polyhedral theory for Independence Systems (see [21] for details) applied to this particular problem. Notice that for general values of k this result does not hold since a subset of edges of a k-regular graph do not in general form a k-regular subgraph.

An interesting question is whether this IS is a matroid when G is a tree. The answer is negative, which reinforces the idea that the results in the previous sections for trees (the combinatorial algorithm for trees and the polyhedral description of the convex hull of X) are nontrivial.

An important concept for defining valid inequalities for the incidence vectors of IS is the rank function. The rank function for strong-matchings is given by

$$r_{SM}(S) = \max\{|I| : I \in \mathcal{F} \text{ and } I \subseteq S\} \quad \text{for all } S \subseteq E.$$

Since for IS the rank inequalities are valid for the set of the incidence vectors of the sets in $\mathcal{F}$, the following result follows

Proposition 3.3 *For every $S \subseteq E$ the rank inequality*

$$\sum_{uv \in S} y_{uv} \leq r_{SM}(S) \tag{12}$$

is valid for Y^{SM}.

We note that the inequality $\sum_{uv \in \delta^2(f)} y_{uv} \leq 1$ is the rank inequality for set S given by 2-stars $\delta^2(f)$.

As $r_{SM}(S)$ is the cardinality of the maximum cardinality of the strong-matching contained in $G[S]$, computing such value is in general NP-hard (as mentioned in Section 1).

Two well-known necessary conditions for a rank inequality associated with S to define facets of the independent system polytope (i.e., the convex hull of incidence vectors of the independent sets) are the following: (i) S is closed, and (ii) S is nonseparable. A subset $S \subseteq E$ is called *closed* if $r_{SM}(S \cup \{uv\}) \geq r_{SM}(S)+1$ for all $e \in E \setminus S$, and S is called *nonseparable* if $r_{SM}(S) < r_{SM}(T) + r_{SM}(S \setminus T)$ for all nonempty subsets T of S with $T \neq S$.

Lemma 3.4 *If $S \subseteq E$ is closed and nonseparable, then S defines a maximal (inclusion-wise) dependent set with rank $r_{SM}(S)$.*

Proof. By definition, S is closed if it defines a maximal set with rank $r_{SM}(S)$. If S is independent then $r_{SM}(S) = |S|$, thus S is separable. □

As a consequence one can easily see that 2-stars obtained from pendant edges are not maximal dependent sets, so they will not define facets. Also, all the other defining inequalities based on 2-stars should be lifted, when G is not a tree, in order to obtain maximal dependent sets with rank 1.

For a closed subset S construct a graph G_S where S is the vertex set and there is an edge linking $e = uv$ and e' if there is a strong-matching $M \subseteq S$ with $|M| = r_{SM}(S)$, $e \in M$, and $M - e + e'$ is a strong-matching.

As an immediate consequence of Theorem 3.2 in [21] we have the following result.

Corollary 3.5 *Let S be a closed subset of E. If the graph G_S is connected, then the rank inequality $\sum_{uv \in S} y_{uv} \leq r_{SM}(S)$ defines a facet of* $\text{conv}(Y^{SM})$.

Example: Consider the graph given in Figure 2.

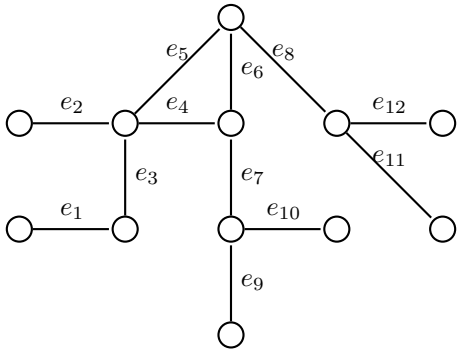


Figure 2: Graph G.

The rank inequality $\sum_{i=4}^{12} y_{e_i} \leq 2$ defines a facet of $\text{conv}(Y^{SM})$. □

Using the correspondence between strong-matchings in G and stable sets in $L(G)^2$, the inequality (12) is equivalent to the inequality

$$\sum_{uv \in S} y_{uv} \leq \overline{\alpha}(S), \tag{13}$$

where $\overline{\alpha}(S) = \max\{|I \cap S| : I \text{ is a stable set of } L(G)^2\}$.

For the particular case of $S \subseteq E$ with $r_{SM}(S) = 1$, S is closed in G if and only if S is a maximal clique in $L(G)^2$. Below we discuss what this condition corresponds to in G.

Proposition 3.6 *The inequality $\sum_{uv \in S} y_{uv} \leq 1$ defines a facet of* $\text{conv}(Y^{SM})$ *if and only if S is a clique in $L(G)^2$.*

Thus, to separate inequalities $\sum_{uv \in S} y_{uv} \leq 1$ is equivalent to finding the maximum weight clique in $L(G)^2$.

A particular case of rank 1 facet-defining inequalities of rank 1 is a direct generalization of 2-stars. A set $S \subseteq E$ is a *clique-star* if $S = \bigcup_{u \in C} \delta(u)$ where $C \subseteq V$ is a set of vertices defining a maximal clique in G. Thus for each edge $e \in S$ either one or both its end points belong to C. A 2-star $\delta^2(e)$, $e = uv$, is a clique-star where $C = \{u, v\}$.

Corollary 3.7 *If S is a clique-star in G then the inequality $\sum_{uv \in S} y_{uv} \leq 1$ defines a facet of* $\mathrm{conv}(Y^{SM})$.

For instance, in the graph depicted in Figure 2, the edge set $S = \{e_2, e_3, \ldots, e_8\}$ is a clique-star. Hence the inequality $\sum_{i=2}^{9} y_{e_i} \leq 1$ defines a facet of $\mathrm{conv}(Y^{SM})$.

However, there are different graphs in G that correspond to cliques in $L(G)^2$. In Figure 3 several such graphs are depicted. This means that it may be easier to identify rank 1 facet-defining inequalities using $L(G)^2$, than to identify rank 1 inequalities using G.

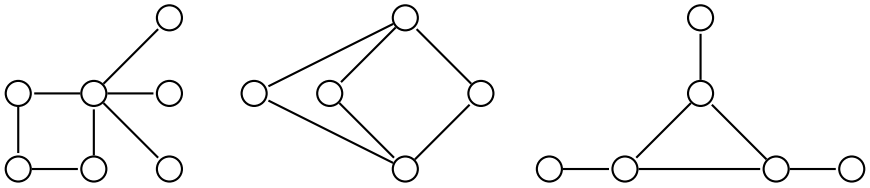


Figure 3: Examples of graphs G whose corresponding $L(G)^2$ is a clique.

Rank 1 inequalities can be regarded as clique inequalities derived from a conflict graph, where the set of nodes represent the variables and an edge represents an incompatibility between two variables. For SM, the conflict graph coincides with $L(G)^2$. Using the conflict graph concept one can improve the combinatorial upper bound given in Lemma 3.1 by considering as the upper bound the cardinality of a cover by cliques (which extends 2-stars):

Lemma 3.8 *Let $\mathcal{C}$ denote the set of cliques in $L(G)^2$ and c denote the minimum cardinality family of cliques $C_1, C_2, \ldots, C_c$ such that $\cup_{j=1}^{c} C_j = E$. Then $\gamma^p(G) \leq c$.*

4 Induced cycle family

In this section we consider the case $k = 2$. The 2-RIS problem can be modeled as follows by projecting out the x variables.

$$
\begin{aligned}
(IC) \qquad & \max \sum_{e \in E} y_e \\
& \text{subject to} \\
(i) \qquad & \sum_{f \in \delta^2(e) \setminus \{e\}} y_f \leq 2 && (e \in E), \qquad (14) \\
(ii) \qquad & y_e \leq \sum_{f \in \delta(v) \setminus \{e\}} y_f && (e \in \delta(v),\ v \in V), \\
(iii) \qquad & \sum_{e \in \delta(v)} y_e \leq 2 && (v \in V), \\
(iv) \qquad & y_e \in \{0, 1\} && (e \in E).
\end{aligned}
$$

4.1 Jungles of cycles

Note that any vertex of degree less than two cannot be a part of a 2-regular subgraph, so we may remove all such vertices from the graph (Possibly iterating if, after removal, some new vertex has degree less than two). Therefore we may assume that every edge in the graph belongs to at least one cycle.

We call a cycle C_i disjoint if it does not share any of its edges with other cycles. That is, all the edges in C_i belong to a unique cycle. When C_i is disjoint, then one can easily see that an edge in C_i is in the solution if and only if all the edges in C_i are in the solution. Hence, the following equations can be added to the formulation:

$$
y_e = y_{e'} \quad (e, e' \in E \text{ of the same disjoint cycle } C_i). \qquad (15)
$$

We introduce a class of graphs for which the RIS problem for $k = 2$ turns out to be well-behaved. This is shown by establishing a connection to stable sets in a certain graph.

A graph G is called a *jungle of cycles*, or a JC-graph, if it can be constructed recursively by starting with a cycle C_1 and (possibly) adding cycles as follows: if G consists of $C_1, C_2, \ldots, C_{i-1}$, then add a new cycle C_i and identify one of its vertices with a vertex in C_{i-1} $(i \leq n)$. We then write $G = J(C_1, \ldots, C_n)$. Note that the *only* cycles in $J(C_1, \ldots, C_n)$ are $C_1, C_2, \ldots, C_n$, see below. Associated to G we define another graph $G^* = (V^*, E^*)$ with vertices $v_{C_1}, v_{C_2}, \ldots, v_{C_n}$ corresponding to $C_1, C_2, \ldots, C_n$, and with an edge between v_{C_i} and v_{C_j} whenever C_i and C_j either have a common vertex or there is an edge in G joining them. We call G^* the *cycle-graph* of G.

Consider the JC-graph G with four cycles depicted in Figure 4.

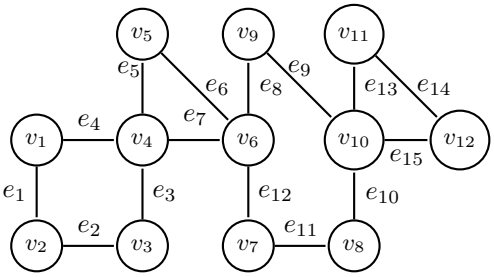


Figure 4: JC-graph G with cycles: $C_1 = \{e_1, e_2, e_3, e_4\}, C_2 = \{e_5, e_6, e_7\}, C_3 = \{e_8, e_9, e_{10}, e_{11}, e_{12}\}, C_4 = \{e_{13}, e_{14}, e_{15}\}$.

The graph G^* has four vertices $V^* = \{v_{C_1}, v_{C_2}, v_{C_3}, v_{C_4}\}$ and the edge set is given by $E^* = \{v_{C_1}v_{C_2}, v_{C_1}v_{C_3}, v_{C_2}v_{C_3}, v_{C_3}v_{C_4}\}$. See Figure 5.

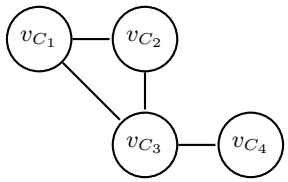


Figure 5: G^*.

Theorem 4.1 *Let $G = J(C_1, \ldots, C_n)$ be a JC-graph.*

(i) The only cycles in G are $C_1, C_2, \ldots, C_n$. Moreover, every edge in G lies in a unique cycle.

(ii) The 2-RIS problem in G is equivalent to the weighted stable set problem in G^* where the weight of vertex v_C equals the number of vertices in the cycle C.

(iii) G^* is a chordal graph, and its maximal cliques are $\{v_{C_i}, \ldots, v_{C_{j_i}}\}$ for certain integers $i \in I$ and where $j_1 < j_2 < \ldots$.

Proof. (i) In the recursive construction of G, when C_i is added to $J(C_1, C_2, \ldots, C_{i-1})$, the cycle C_i intersects C_{i-1} in a single vertex v which then is a cut vertex. So any cycle containing an edge in C_i must be equal to C_i. The first result then follows by induction. Since the cycles $C_1, C_2, \ldots, C_n$ are edge-disjoint, every edge in G lies in a unique cycle.

(ii) Consider a feasible solution of 2-RIS in G, i.e., a family of vertex-disjoint cycles not joined by any edge. By (i) this must be a subset of $\{C_1, C_2, \ldots, C_n\}$ so it corresponds to a subset $S \subseteq V^*$. By the definition of edges in G^*, S is a stable set precisely when the corresponding cycles in G are pairwise disjoint and no edge joins any pair of such cycles, i.e., we have a feasible solution of 2-RIS. Clearly, the weight of such a stable set S is the total number of vertices in the corresponding cycles, and this proves the desired equivalence.

(iii) To show that G^* is chordal it suffices to find a perfect elimination (PE) ordering of G^*. Actually, $v_n, v_{n-1}, \ldots, v_1$ is a PE ordering. To see this, let $i \leq n$. Let $j < i$ be smallest possible such that $v_{C_j} v_{C_i} \in E^*$. Then, by construction, $\{v_{C_j}, v_{C_{j+1}}, \ldots, v_{C_i}\}$ is a clique in G^*. This gives the desired PE ordering. By the way the cycles C_i are constructed, where C_i intersects with C_{i-1}, it follows that the cliques in G^* are of the form $v_{C_i}, v_{C_{i+1}}, \ldots, v_{C_j}$ for some $j > i$ (which depends on i). □

The following dynamic programming algorithm may be used to solve the 2-RIS problem in a JC-graph G. The number of vertices in a cycle C is denoted by $|C|$. We may think of α_i as the optimal value of the stable set problem in the subgraph induced by vertices $v_{C_1}, \ldots, v_{C_i}$.

Algorithm 1.

```
Input:  A JC-graph G = J(C_1, ..., C_n)
1.  Let α_0 = 0 and α_1 = |C_1|.
2.  for i = 2, ..., n do
        a) Let j ≥ 0 be smallest possible such that
           v_{C_s} v_{C_i} ∈ E* for s = j+1, ..., i-1.
        b) Let α_i = max{α_{i-1}, α_j + |C_i|}.
```

Theorem 4.2 *Algorithm* 1 *solves the 2-RIS problem in a JC-graph* $G = J(C_1, \ldots, C_n)$ *in* $O(n)$ *steps.*

Proof. By induction on i we see that α_i equals the maximum weight of a stable set in the subgraph $J(C_1, \ldots, C_i)$. This follows from the dynamic programming principle, by considering at stage i whether vertex v_{C_i} should be in the stable set or not. □

Example: Let us consider the graphs G and G^* depicted in Figure 4 and Figure 5, respectively.

We have $G = J(C_1, C_2, C_3, C_4)$ with $|C_1| = 4,\ |C_2| = |C_4| = 3,\ |C_3| = 5$.

Let $\alpha_0 = 0$ and $\alpha_1 = |C_1| = 4$.

For $i = 2$ and $j = 0$ we have $s = 1$. In this case $v_{C_1} v_{C_2} \in E^*$, therefore we set $j = 0$ and $\alpha_2 = \max\{\alpha_1, \alpha_0 + |C_2|\} = \max\{4, 3\} = 4$.

For $i = 3$ and $j = 0$ we have $s = 1, 2$. In this case $v_{C_1} v_{C_3} \in E^*$ and $v_{C_2} v_{C_3} \in E^*$, therefore we set $j = 0$ and $\alpha_3 = \max\{\alpha_2, \alpha_0 + |C_3|\} = \max\{4, 5\} = 5$.

For $i = 4$ and $j = 0$ we have $s = 1, 2, 3$. In this case $v_{C_1} v_{C_4} \notin E^*$ so we have to proceed for $j = 1$. In this case $s = 2, 3$ and $v_{C_2} v_{C_4} \notin E^*$ whereby we have to consider $j = 2$ and $s = 3$. In this case $v_{C_3} v_{C_4} \in E^*$, therefore we set $j = 2$ and $\alpha_4 = \max\{\alpha_3, \alpha_2 + |C_4|\} = \max\{5, 4 + 3\} = 7$. Thus, $\alpha_4 = 7$ is the optimal value of the stable set problem induced by vertices v_{C_1} and v_{C_4} that corresponds to the following 2-regular induced subgraph of G with maximum order:

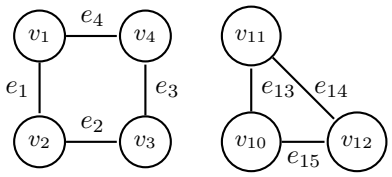


Figure 6: 2-regular induced subgraph of G with maximum order.

□

For these graphs it is also possible to obtain a polyhedral characterization of a polytope related to the 2-RIS problem for JC-graphs. Consider a JC-graph $G = J(C_1, \ldots, C_n)$. Choose k distinct vertices $v_1, v_2, \ldots, v_n$ such that $v_i \in C_i$ $(i \leq n)$. It is easy to see, from the construction of G that such a choice is possible.

We let $P_2(G)$ be the convex hull of all incidence vectors of vertex sets that induce 2-RIS feasible sets. Let K_i $(i \in I)$ denote the cliques in G^*.

Theorem 4.3 *Let G be a JC-graph. A complete linear description of $P_2(G)$ is*

$$\begin{array}{ll} \sum_{j \in K_i} y_{v_j} \leq 1 & (i \in I), \\ y_v = y_{v_i} & \text{for all } v \in C_i,\ i \leq n, \\ y_v \geq 0 & (v \in V). \end{array} \tag{16}$$

Proof. This follows by combining the two facts that (i) $P_2(G)$ is obtained from the stable set polytope of G^* by the linear map which maps $x \in \mathbb{R}^{V(G^*)}$ into $x' \in \mathbb{R}^V$ by letting $x'_v = x_{v_{C_i}}$ for each $v \in C_i$ $(i \leq n)$, and (ii) stable set polytopes for chordal graphs are completely described by $x_v \geq 0$ for each vertex v and the clique inequalities. □

Remark. One may extend the class of graphs beyond JC-graphs and still be able to solve 2-RIS essentially in the same way: If one connects JC-graphs by trees, no new cycles are introduced, and the same ideas apply.

Remark. The previous result is closely related to the theory of block-decomposition of 2-connected graphs; see [13], Chapter 3. Recall that a block is a maximal connected subgraph with no cut-vertex. For JC-graphs the blocks are the cycles, and they intersect in cut vertices. Our notion of cycle-graph is very similar, but not the same as block graphs (which also indicates how blocks are connected, and is a tree).

4.2 Valid inequalities for the set of induced cycles

Here we introduce two families of valid inequalities for the set of induced cycles, denoted by Y^{IC} (see (14)). The first one is a class of clique inequalities from

a conflict graph CG where the vertex set is E and there is an edge between e and f if two edges are incompatible, that is, if there is no induced cycle in G containing e and f simultaneously. These inequalities have the following form:

$$\sum_{e \in T} y_e \leq 1, \quad \text{whenever T is a clique in } CG \tag{17}$$

Identifying incompatible edges in general is not as simple as in the case $k = 1$. In order to simplify the separation problem a heuristic approach is to consider only pairs of edges e, f, that are adjacent (having a common end vertex), as e_1 and e_2 in Figure 7, or one end vertex of e and an end vertex of f that are the end vertices of a third edge, which is the case of e_2 and e_3 in Figure 7. If e and f have a common vertex v they are incompatible if and only if every chordless path between the other two end points contains at least one vertex which is adjacent to v. If e and f are connected by a third edge, h, they are incompatible if and only if every chordless path between the other two end points contains at least one vertex which is adjacent to an end point of h.

A simple case can be identified with the following triangle rule: if one edge of a triangle is in the solution, then either the two remaining edges are in the solution or not (the corresponding variable values are equal). For instance, in Figure 8, if $y_{e_5} = 1$ then $y_{e_1} = y_{e_3}$ and $y_{e_4} = y_{e_{10}}$. If $y_{e_6} = 1$ then $y_{e_7} = y_{e_8}$. If $y_{e_9} = 1$ then $y_{e_7} = y_{e_{10}}$.

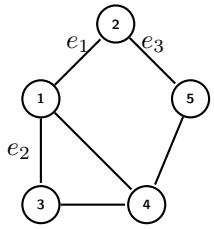


Figure 7: The edges e_1 and e_2 are incompatible, as are the edges e_2 and e_3.

Using these rules we can see that the pairs of edges e_6, e_9; e_5, e_6 and e_5, e_9 are pairwise incompatible. Consider the case of edges e_5, e_9 (the two other cases are similar). Assume both edges are in the solution. Thus, as the degree of vertex 6 is two, then $y_{e_3} = y_{e_{10}} = 0$. From the above triangle rule it follows that $y_{e_1} = y_{e_3} = y_{e_4} = y_{e_7} = y_{e_{10}} = 0$. Additionally, the removal of these edges implies that vertices 1 and 3 can be removed (and their incident edges) since the solution must be an induced graph. The resulting graph has no path between vertices 2 and 5. Thus, in the CG, there is an edge between e_5 and e_9. Similarly, edges between e_6, e_9 and between e_5, e_6 are in CG. Hence, the inequality $y_{e_5} + y_{e_6} + y_{e_9} \leq 1$ is valid for Y^{IC} and can be proven to define a facet of conv(Y^{IC}).

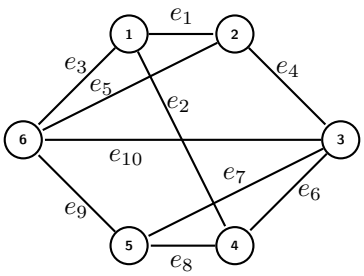


Figure 8: Induced cycles example.

Next we introduce a new class of inequalities that generalizes (14.ii) . Assume T is a clique (not necessarily maximal) in CG. Let $S \subseteq E$ be such that for each $e = uv \in T$ either $\delta(u) \subseteq S$ or $\delta(v) \subseteq S$. Then the following inequality is valid for Y^{IC}.

$$\sum_{e \in T} y_e \leq \sum_{e \in S} y_e \tag{18}$$

Note that if S is the union of disjoint sets $\delta(u), u \in U$ such that each edge $e \in T$ is incident to one different u in U, then (18) is the aggregation of (14.ii), otherwise (18) is not dominated by those inequalities.

In the example depicted in Figure 8 we have, using (14.ii), $y_{e_6} \leq y_{e_4} + y_{e_7} + y_{e_{10}}$ and $y_{e_9} \leq y_{e_7} + y_{e_8}$. Hence, as e_6 and e_9 are incompatible, then the following inequality of type (18) is valid for Y^{IC} :

$$y_{e_6} + y_{e_9} \leq y_{e_4} + y_{e_7} + y_{e_8} + y_{e_{10}} \tag{19}$$

Inequalities (18) can be strengthen using the triangle rule. In the previous example, if $y_{e_9} = 1$ then $y_{e_7} = y_{e_{10}}$, and if $y_{e_6} = 1$ then $y_{e_7} = y_{e_8}$. Hence, y_{e_7} can be removed from the right hand side of (19). The resulting inequality can be shown to define a facet of $\text{conv}(Y^{IC})$ for the example given in Figure 8.

In order to formalize the strengthened inequalities we need to introduce some definitions. For each edge $e = uv$ let

$$\partial_e(u) = \{uw \in E : w \in N(u) \setminus \{v\}\} \cup \{vw \in E : w \in N(u) \setminus \{v\}\}$$

be the edge cut set that separates the subset of vertices $\{u, v\}$ from the subset $N(u) \setminus \{v\}$, where $N(u)$ is the set of neighbors of vertex u in G. In the example depicted in Figure 8 we have $\partial_{e_6}(3) = \{e_4, e_7, e_8, e_{10}, \}$.

Let T be a set of pairwise incompatible edges. We say that S is an edge cover of T if for every edge $e = uv \in T$, S contains a cover by edges in $\partial_e(u)$ of the vertices in $N(u) \setminus \{v\}$ for an end vertex u of e. Recall that an edge cover of a graph is a subset of edges such that every vertex of the graph is incident to at least one edge in the cover.

Proposition 4.1 *For each clique T in CG, and each cover S of T, inequality* (18) *is valid for Y^{IC}.*

Proof. Consider a solution $y \in Y^{IC}$. As T is a clique in CG, then either $\sum_{e\in T} y_e = 0$ or $\sum_{e\in T} y_e = 1$. In the first case validity of (18) follows from the nonnegativity of $y_e, e \in S$. For the case $\sum_{e\in T} y_e = 1$, assume $y_f = 1$ for $f \in T$. As S is a cover of T then S contains a cover $C \subseteq \partial_e(u)$ of the vertices in $N(u) \setminus \{v\}$ for an end vertex, u, of edge f. By (14.ii), $1 = y_f \leq \sum_{\ell\in\delta(u)\setminus\{f\}} y_\ell$. Hence, $y_\ell = 1$ for some $\ell \in \delta(u) \setminus \{f\}$. If $\ell \in C$, then $1 = \sum_{e\in T} y_e \leq \sum_{e\in C} y_e \leq \sum_{e\in S} y_e$. Now assume $\ell \notin C$. Let $\ell = uw$. As $w \in N(u) \setminus \{v\}$ and C is a cover of $N(u)\setminus\{v\}$, then there must exist $\ell' = vw \in C$. As f, ℓ and ℓ' define a triangle, by the triangle rule $y_f = 1$ implies $y_{\ell'} = 1$. Hence, $\sum_{e\in T} y_e = 1 = y_{\ell'} \leq \sum_{e\in C} y_e \leq \sum_{e\in S} y_e$. □

The edge cover S is said minimal for T if for each $e \in S$, $S \setminus \{e\}$ is not an edge cover for T.

Proposition 4.2 *If* (18) *defines a facet of* $\mathrm{conv}(Y^{IC})$, *then S is a minimal cover of T.*

Proof. Suppose S is not a minimal cover of T. Then, there must exist $f \in S$ such that $S \setminus \{f\}$ is a cover of T. Hence, by Proposition 4.1 the inequality

$$\sum_{e\in T} y_e \leq \sum_{e\in S\setminus\{f\}} y_e \tag{20}$$

if valid for Y^{IC}. As (18) is the aggregation of (20) and $y_f \leq 1$, then (18) is not a facet, which is a contradiction. □

5 Computational tests

In this section we report some computational experiments conducted in order to accomplish the following:

(i) evaluate the quality of bound 2 for different values of k by comparing those bounds against the optimal value (obtained by solving the integer programming models) and the linear relaxation bound;

(ii) evaluate the quality of the bound obtained by solving the QIP and the $LRIP_{\lambda^*}$ and compare the corresponding running times.

(iii) evaluate the integer programming model and the inequalities derived for different values of k.

Three sets of instances were used: A set based on DIMACS instances [20], a set of low density graph instances called *Almost-trees*, and a third set based on randomly generated instances for graphs with 50 vertices and two different

graph densities; 25% and 50%. The *Almost-trees* include two sets of instances, the *10pctj* and the *doubletreej* instances, which are obtained by adding roughly 10 or 100 edges to trees on 100 vertices, respectively. The Random instances are denoted as *Rdj* where d denotes the density and is either 25 or 50.

All computations were performed on a computer with processor Intel Core i7, 2.4 GHz and with 16 GB RAM. The spectral bounds were computed using MATLAB and the integer programming models were optimized with Xpress-Optimizer Version 27.01.02 with Xpress Mosel Version 3.8.0.

In all the tests based on the integer program $k-$RIS described by (1), a preprocessing step was conducted. In this step we used an iterative elimination procedure consisting of removing all vertices with degree less than k.

5.1 Evaluation of bounds

Here we compare the optimum value of several instances with the bounds deduced in this paper and the bound (2).

In Tables 1–3 we report our results. The columns are labeled as follows. *Inst.* indicates the instance; n gives the number of vertices; *Op* is the value of the optimal solution; LR is the value of the linear relaxation; υ_k is the upper bound (2) and DL is the approximate value of the dual Langrangian problem obtained after 30 iterations of the subgradient method. The column *Bound* indicates the bound we obtained by solving the problems QIP and $LRIP_{\lambda^*}$ and the running times in seconds, given in columns T_Q and T_{LR} respectively. The instances omitted were not solved within 6 hours. An asterisk in the *Op* column means that the optimal solution is unknown and the value of the best known feasible solution is provided. For the *Almost-trees* with $k = 5$ the optimal value is zero for all tested instances, and so we have not reported on these.

We can see that for the DIMACS instances all the bounds are poor and distant from the optimum values. However, the spectral bounds provide better bounds than the linear relaxation for all instances. This is also true for *Random* graphs with density 50%, but the opposite behavior is observed when *Almost-trees* and *Random* graphs with density 25% are considered.

Recall Theorem 2.1: $V(IP) \leq V(DLIP) \leq V(LRIP_{\lambda^*}) \leq \upsilon_1(G)$. Consider the instance *R25b* with $k = 1$. Here we have the following values, as seen in Table 1:

$$\begin{aligned} V(IP) &= 16, \\ V(LRIP_{\lambda^*}) &= 19.1, \\ \upsilon_1(G) &= 21.9. \end{aligned}$$

Using a subgradient approach starting with λ^* we were able to find a better Lagrangian multiplier $\bar{\lambda}$, such that $V(LRIP_{\bar{\lambda}}) = 17.9$. Note that this bound is effectively a bound of 16, since the optimal value in the 1-RIS is a multiple of 2 (in fact, if k is an odd number, we can round the upper bound value to the immediately previous even number). In this case, therefore, we have $V(IP) \leq V(DLIP) < V(LRIP_{\lambda^*}) < \upsilon_1(G)$, and computational experience

does suggest most of the inequalities in Theorem 2.1 are typically strict. We note that in this case we also have $\upsilon_1(G) < V(LR) = 26.3$.

Usually the multiplier λ^* is suboptimal. Computationally, it seems to be relatively far from the optimal choice for the *Almost-trees*, so it is easy to obtain a better bound using a subgradient approach. For the *Random* instances, however, this requires a more careful choice of the subgradient step. In any case, since we do not know a polynomial-time algorithm for solving the $LRIP_\lambda$ problem the subgradient algorithm does not seem to offer a good alternative to solving the k-RIS problem directly.

Inst.	n	Op	LR	υ_1	DL	Bound	T_Q	T_{LR}
Kel4	171	12	85.9	32.9				
brok21	200	8	100.3	17.5				
brok22	200	12	100.5	28				
bock23	200	10	100.4	23				
brok24	200	10	100.4	21				
cfat21	200	36	103.4	71.1				
cfat22	200	18	101.6	59				
cfat25	200	6	100.6	46.2				
mann9	45	2	22.8	4.6	3.98	4	0.6	1.9
ham62	64	4	32.3	6.1	4.1	4.3	22	28
ham64	64	16	32.7	22	17.4	17.5	4518	2228
ham82	256	4	100.3	8				
ham84	256	16	101.4	46.5				
joh1624	120	6	60.3	16.1				
joh824	28	6	14.5	8.4	8.1	8.2	0.18	0.09
joh844	70	6	35.3	11.2	6.9	8	349	4412
10pct1	100	50	51	87.7	70.5	84.3	0.004	0.02
10pct2	100	52	52	87.1	70.7	83.5	0.005	0.03
10pct3	100	54	55	88.6	68.8	84.6	0.004	0.02
10pct4	100	52	52	88.1	70.4	84.7	0.004	0.02
10pct5	100	52	52	89.3	71.3	85.7	0.005	0.02
doubletree1	100	48	54.3	71.4	60.3	68	0.007	0.025
doubletree2	100	48	54.3	72.7	61.3	68.9	0.005	0.024
doubletree3	100	46	54.9	73.5	61.1	70.1	0.005	0.025
doubletree4	100	50	55.8	72.3	60.6	68.4	0.006	0.024
doubletree5	100	50	55.8	71.7	60.6	67.5	0.006	0.027
R25a	50	16	26.4	22.7	18.7	19.9	0.8	0.8
R25b	50	16	26.3	21.9	17.9	19.1	1.6	2.6
R25c	50	16	26.3	22.8	18.6	20.2	0.8	1.4
R25d	50	16	26.4	22.7	18.7	20	0.7	1.5
R25e	50	14	26.4	22.4	18.1	19.6	1.2	1.3
R50a	50	8	25.6	14.3	10.8	11.7	1236	1845
R50b	50	10	25.6	13.9	10.6	11.8	726	119
R50c	50	8	25.6	14.4	11.4	12.7	301	174
R50d	50	8	25.6	13.7	10.6	11.3	1176	884
R50e	50	10	25.6	14.3	10.7	11.8	2118	1570

Table 1: $k = 1$

Inst.	n	Op	LR	v_2	DL	Bound	T_Q	T_{LR}
Kel4	171	12	86.3	34.2				
brok21	200	9	100.7	18.8				
brok22	200	14*	101.0	29.8				
bock23	200	10*	100.8	24.5				
brok24	200	11	100.8	22.4				
cfat21	200	54	107.1	80.1				
cfat22	200	27	103.2	63.5				
cfat25	200	9	101.2	48				
mann9	45	4	23.1	5.6	5	5	3.4	8.5
ham62	64	4	32.6	7.2	5.5	5.7	143	207
ham64	64	16	33.5	24	21.3	21.3	2330	1445
ham82	256	4	100.5	9				
ham84	256	16	100.8	47.8				
joh1624	120	6	60.6	17.3				
joh824	28	6	15	9.8	9	9	0.88	0.25
joh844	70	6	35.7	12.4	9.5	9.5	4637	
10pct1	100	18	18	26.5	21.3	25.4	0.004	0.02
10pct2	100	22	22	36.1	29.6	35.0	0.003	0.021
10pct3	100	29	29	40.2	35.1	39.0	0.003	0.021
10pct4	100	24	24	32.6	26.3	31.2	0.003	0.021
10pct5	100	22	22.5	30.3	25.2	28.4	0.004	0.024
doubletree1	100	47	53.8	77.5	58.2	73.1	0.006	0.026
doubletree2	100	47	53.9	78.1	59.4	74.3	0.005	0.025
doubletree3	100	44	51.3	80.3	60.0	76.3	0.006	0.026
doubletree4	100	48	52.6	80.0	59.6	76.3	0.005	0.025
doubletree5	100	51	57.1	80.8	62.6	77.4	0.01	0.041
R25a	50	18	28.2	25.3	21.9	23.7	0.7	1.0
R25b	50	17	27.9	25.1	22.4	22.7	1.8	2.5
R25c	50	17	27.9	25.9	21.8	24.1	0.4	0.8
R25d	50	18	28.2	25.9	22.0	23.8	0.5	1.2
R25e	50	17	28.2	24.9	22.1	23.7	0.8	1.2
R50a	50	11	26.2	16.1	13.8	13.8	8190	3515
R50b	50	11	26.2	15.8	13.2	14.1	3150	2539
R50c	50	11	26.2	16.1	13.7	14.7	1228	359
R50d	50	11	26.2	15.6	12.9	13.4	6653	5727
R50e	50	11	26.2	16.0	13.0	13.8	19394	14818

Table 2: $k = 2$

Inst.	n	Op	LR	v_5	DL	Bound	T_Q	T_{LR}
Kel4	171	24	87.5	38.2				
brok21	200	12*	101.7	22.6				
brok22	200	14*	102.6	35.4				
bock23	200	10*	102.1	29.1				
brok24	200	12*	101.9	26.7				
cfat21	200	90*	119.7	107.3				
cfat22	200	54	108.4	77				
cfat25	200	18	103.1	53.5				
mann9	45	6	23.9	8.7	8.2	8.2	261	859
ham62	64	8	33.5	10.3				
ham64	64	24	36.1	30	29.8	29.9	3749	996
ham82	256	8	101.3	12				
ham84	256	24	102.1	51.7				
joh1624	120	12	61.7	20.7				
joh824	28	12	16.8	14	13.5	13.5	2.9	2.8
joh844	70	12	36.7	15.8				
R25a	50	18	31.3	34.8	28.3	33.2	0.2	0.4
R25b	50	20	32.3	34.6	28.5	32.9	0.3	0.5
R25c	50	20	31.9	35.4	31.2	33.6	0.1	0.4
R25d	50	16	31.6	35.6	26.7	34.2	0.1	0.3
R25e	50	18	31.6	34.7	30.9	32.5	0.2	0.6
R50a	50	14	28.2	21.5	19.9	19.9	636	934
R50b	50	16	28.2	21.4	20.1	20.1	384	571
R50c	50	14	28.2	21.5	20.1	20.1	321	824
R50d	50	16	28.2	21.2	19.9	19.9	500	853
R50e	50	14	28.1	21.2	19.7	19.7	750	1360

Table 3: $k = 5$

5.2 Integer programming approach

In this section we test the inclusion of valid inequalities and branching priorities to solve the integer programming models by the commercial solver. The *Almost-trees* are, as seen above, very particular instances, while solving the DIMACS instances to optimality takes a very long time. Therefore, we only report on the *Random* instances in this section.

In Table 4 we compare the running times in seconds (labelled T) and the number of branch and bound vertices (labelled N) to solve each instance to optimality with (**B**) and without (**NB**) branching priorities on the vertices. We can observe that, in general, giving priority to branching on vertices is slightly better than the default option. This difference is more clear for the case $k = 2$.

	k=1				**k=2**				**k=5**			
	NB		**B**		**NB**		**B**		**NB**		**B**	
	T	N	T	N	T	N	T	N	T	N	T	N
R25a	4	263	3	141	10	2195	4	887	98	22099	82	26619
R25b	5	822	5	548	26	6301	14	4383	68	14543	131	26277
R25c	4	171	4	337	16	3723	7	2317	77	17689	69	18815
R25d	4	351	3	201	9	1449	8	1975	78	21245	93	30309
R25e	5	358	4	436	22	6563	8	2561	195	38031	212	48567
R50a	7	468	5	453	12	1487	9	1321	1440	158263	1742	156111
R50b	5	333	4	273	14	2005	10	2575	229	43053	452	53007
R50c	8	490	7	489	12	2099	8	1555	1304	150949	1008	116883
R50d	7	541	5	465	14	2093	10	1931	920	120655	387	65233
R50e	6	230	5	191	12	1599	7	897	2901	233867	3091	265165
Average	6	403	5	353	15	2951	9	2040	731	82039	727	80697

Table 4: Branching on vertices.

Next we test the inclusion of valid inequalities. These tests are preliminary and performed just to evaluate a possible use of the proposed inequalities. In order to use them efficiently a deeper study should be conducted to test the inclusion of the most violated ones at several nodes of the branching and bound scheme, and to find a balance between the number of cuts added and the size of the model. Here we run the linear relaxation and add all the violated cuts found using a separation heuristic. Then the model is solved to optimality with all the cuts added.

For $k = 1$ we include clique inequalities at the root node (see Section 3.3). Given the linear relaxation solution y^* the graph $L(G)^2$ is constructed with weights y^*_e, associated to each edge e. Edges are labeled from 1 to m. Starting from the first uncovered edge we apply a greedy heuristic to find a maximal clique. If the weight of that clique is greater than 1 the inequality is added. Then the edges covered by the clique (whether the inequality is added or not) are removed, and the procedure is repeated until the set of edges is covered by a family of cliques generated in the described way.

In Table 5 we report the time (T) and number of nodes (N) without addition of cuts, and the time (T), number of nodes (N), and number of cuts (Ncuts) when cuts are added to solve the instance. The table shows that the addition of clique cuts is effective for graphs of 25% density. In particular, the number of branch and bound nodes is reduced significantly. At a density of 50% too many cuts are added. In this case a deeper study would be necessary to find rules for deciding when a violated inequality should be added or not.

	Without cuts		With cuts		
Inst.	**T**	**N**	**T**	**N**	**Ncuts**
R25a	3	141	4	41	37
R25b	5	548	4	99	53
R25c	4	337	3	77	54
R25d	3	201	4	44	59
R25e	4	436	4	346	41
R50a	5	453	12	476	141
R50b	4	273	9	295	136
R50c	7	489	12	444	157
R50d	5	465	12	406	117
R50e	5	191	16	765	207

Table 5: Addition of clique inequalities for $k = 1$.

For $k = 2$, the inequalities (18) are tested for sets T including only two non-adjacent edges belonging to two triangles sharing one edge. For such inequalities, with $|T| = 2$, the separation problem can be done by inspection.

In this case ($k = 2$) no violated cut was found for the tested instances. This is due to the fact that the objective function is to maximize the number of edges, leading to fractional solutions where many y_e are positive (for most of the

instances, all variables are strictly positive) but with a very small value. Thus we tested the valid inequalities for the *Random* instances where the objective function was changed to max $\sum_{e \in E} c_e y_e$, each c_e being a random number in the interval $[1, 20]$. The results are given in Table 6.

	Without cuts		With cuts		
Inst.	**T**	**N**	**T**	**N**	**Ncuts**
R25a	3	1635	3	1083	11
R25b	2	1397	2	695	7
R25c	3	1563	3	986	7
R25d	3	1211	2	481	9
R25e	3	1917	3	1803	14
R50a	4	1671	8	3111	105
R50b	5	2546	8	3046	97
R50c	4	2365	5	1791	141
R50d	6	2868	7	3247	93
R50e	4	1715	6	2319	146

Table 6: Valid inequalities for $k = 2$ with random objective function coefficients.

Again such inequalities are effective for the lower density random graphs, but not for the 50% density graphs. These preliminary results indicate that a deeper study of separation of the inequality class (18) can be of interest for solving 2-RIS problems on low density graphs.

The polyhedral results given in the previous sections and the computational tests indicate that the polyhedral structure of the k-RIS changes significantly with k, and no general class of inequalities that could be effective for several different values of k was found. This seems to indicate that for each k a deeper polyhedral study should be done separately.

Acknowledgment

The work of the first and fourth authors has been conducted within project EXPL/MAT-NAN/1761/2013 (funded by FCT through program COMPETE: FCOMP-01-0124-FEDER-041898) and within project UID/MAT/04106/2013 (funded by FCT).

References

[1] F. C. Bussemaker, D. Cvetković, J. Seidel, Graphs related to exceptional root systems, T. H. - Report 76-WSK-05, Technical University Eindhoven, 1976

[2] K. Cameron, Induced matchings, *Discrete Applied Mathematics*, 24 (1989) 97-102.

[3] K. Cameron, Induced matchings in intersection graphs, *Discrete Mathematics*, 278 (2004) 1-9.

[4] D. M. Cardoso, M. Kaminski, V. Lozin, *Maximum k-regular induced subgraphs*, J. Comb. Optim. 14 (2007), 455-463.

[5] D. M. Cardoso, S. J. Pinheiro, *Spectral upper bounds on the size of k-regular induced subgraphs*, Electron. Notes Discrete Math. 32 (2009), 3-10.

[6] D. M. Cardoso, P. Rowlinson, *Spectral upper bounds for the order of a k-regular induced subgraph*, Linear Algebra Appl. 433 (2010), 1031–1037.

[7] G. H. Chen, M.T. Kuo, J. P. Sheu, *An optimal time algorithm for finding the maximum weight independent set in a tree*, BIT 27 (1987), 170–180.

[8] M. Constantino, X. Klimentova, A. Viana, A. Rais, *New insights on integer-programming models for the kidney exchange problem*, European Journal of Operational Research, 231 (2013), 57–68.

[9] D. M. Cvetković, *Graphs and their spectra*, Publ. Elektrotehn. Fack. Ser. Mat. Fiz. 354-356 (1971), 1-50.

[10] D. M. Cvetković, P. Rowlinson, S.K. Simić, *An Introduction to the Theory of Graph Spectra*, London Mathematical Society Student Texts, 75. Cambridge University Press, Cambridge (2010).

[11] G. Dahl, K. Jörnsten, G. Løvnes, S. Svaet, Graph optimization problems in connection with the management of mobile communications systems, *Telecommunication Systems*, 3 (1995), 319-339.

[12] N. Deo, *Graph Theory with Applications to Engineering and Computer Science*, Prentice-Hall, Englewood Cliffs, N.J. (1974).

[13] R. Diestel, *Graph Theory*, Springer, New York, 2000.

[14] C. D. Godsil, M. W. Newman, *Eigenvalue bounds for independent sets*, J. Comb. Theory, Ser. B, 98(4) (2008) 721-734.

[15] M. Golombic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press (1980).

[16] J. R. Griggs *Lower Bounds on the Independence Number in Terms of the Degrees*, J. Comb. Theory, Ser. B, 34 (1983), 22-39.

[17] W. Haemers, *Eigenvalue techniques in design and graph theory* (thesis Technical University Eindhoven 1979), Math. Centre Tract 121, Mathematical Centre, Amsterdam (1980).

[18] P. Hansen, M. Zheng, *Sharp bounds on the order, size, and stability number of graphs*, Networks 23(2) (1993), 99-102.

[19] J. Harant, *A lower bound on the independence number of a graph*, Discrete Math. 188 (1998), 239-243.

[20] D.S. Johnson, M.A. Trick (Eds.), *Cliques, coloring, and satisfiability: second DIMACS challenge*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, 1996

[21] M. Lauren, *A generalization of antiwebs to independence systems and their canonical facets*, Mathematical Programming, 45 (1989) 97-108.

[22] L. Lovász, *On the Shannon capacity of a graph*, IEEE Trans. Inform. Theory, 25(2) (1979), 1-7.

[23] M. Lu, H. Liu, F. Tian, *New Laplacian spectral bounds for clique and independence numbers of graphs*, J. Combin. Theory Ser. B 97 (2007), 726-732.

[24] C. J. Luz, *An upper bound on the independence number of a graph computable in polynomial time*, Operations Research Letters, 18 (1995), 139-145.

[25] C. J. Luz, *Improving an upper bound on the size of k-regular induced subgraphs*, J. Comb. Optim, 22 (2011), 882-894.

[26] Mathieson, Luke *The Parameterized Complexity of Degree Constrained Editing Problems*, Doctoral thesis, Durham University (2009).

[27] Y. Matsumoto, N. Kamiyama, K. Imai, *On Total Unimodularity of Edge–Edge Adjacency Matrices*, Algorithmica, 67 (2013), 277–292.

[28] Hannes Moser, Dimitrios M. Thilikos, *Parameterized complexity of finding regular induced subgraphs*, J. Discrete Algorithms 7(2) (2009), 181-190.

[29] T. Motzkin, E. G. Straus, *Maxima for graphs and new proof of a theorem of turán* Can. J. Math. 17 (1965), 533-540.

[30] G. L. Nemhauser, L. Wolsey, *Integer and combinatorial optimization*, Wiley-Interscience, New York (1988).

[31] M. Padberg, *On the facial structure of set packing polyhedra*, Mathematical Programming, 5 (1973), 119-215.

[32] A. Schrijver, *Theory of linear and integer programming*, Wiley, New York (1986).

[33] H. S. Wilf, *The eigenvalues of a graph and its chromatic number*, J. London Math. Soc. 42 (1967), 330-332.