A computational comparison of compact MILP formulations for the zero forcing number[*]

Agostinho Agra[1], Jorge Orestes Cerdeira[2], and Cristina Requejo[3]

[1] *Departamento de Matemática and Centro de Investigação e Desenvolvimento em Matemática e Aplicações (CIDMA), Universidade de Aveiro, 3810-193 Aveiro, Portugal.*
`aagra@ua.pt`

[2] *Departamento de Matemática and Centro de Matemática e Aplicações (CMA), Faculdade de Ciências e Tecnologia, Universidade NOVA de Lisboa, Quinta da Torre, 2829-516 Caparica - Portugal.*
`jo.cerdeira@fct.unl.pt`

[3] *Departamento de Matemática and Centro de Investigação e Desenvolvimento em Matemática e Aplicações, Universidade de Aveiro, 3810-193 Aveiro, Portugal.*
`crequejo@ua.pt`

January 2, 2020

Abstract

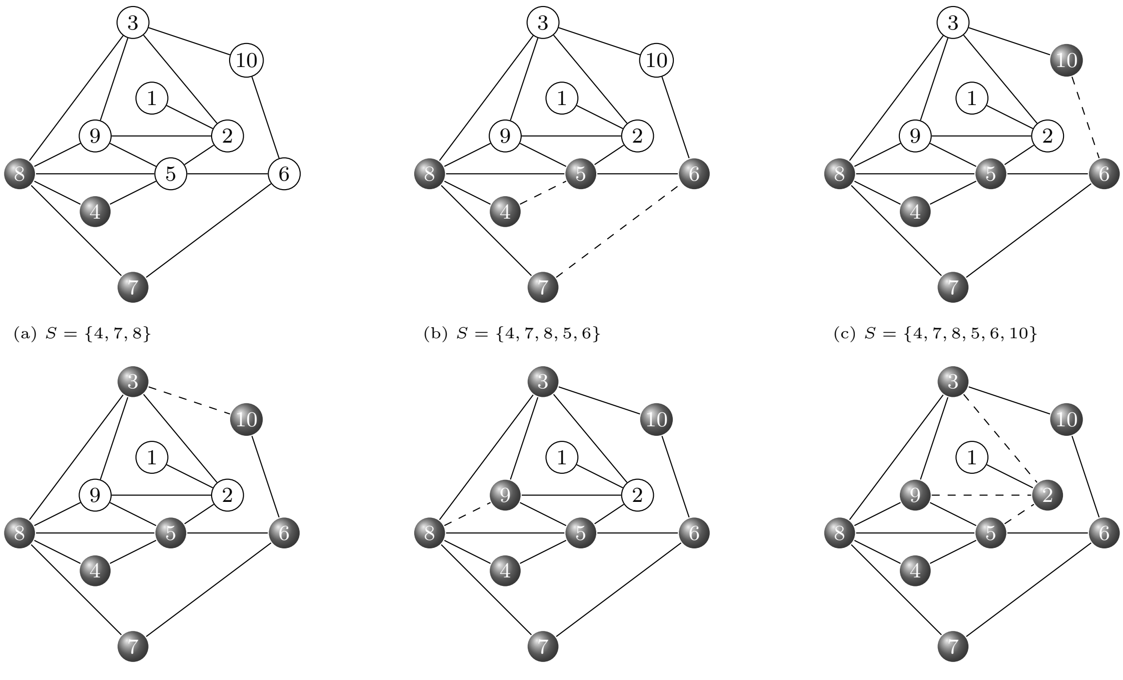
Consider a graph where some of its vertices are colored. A colored vertex with a single uncolored neighbor forces that neighbor to become colored. A zero forcing set is a set of colored vertices that forces all vertices to become colored. The zero forcing number is the size of a minimum forcing set. Finding the minimum forcing set of a graph is NP-hard. We give a new compact mixed integer linear programming formulation (MILP) for this problem, and analyse this formulation and establish relation to an existing compact formulation and to two variants. In order to solve large size instances we propose a sequential search algorithm which can also be used as a heuristic to derive upper bounds for the zero forcing number. A computational study using Xpress (a MILP solver) is conducted to test the performances of the discussed compact formulations and the sequential search algorithm. We report results on cubic, Watts-Strogatz and randomly generated graphs with 10, 20 and 30 vertices.

Keywords: Graphs; mixed integer linear programming; compact formulations; valid inequalities; zero forcing.

1 Introduction

Let $G = (V, E)$ be a simple graph with vertex set V and edge set E, and suppose that initially (at time-step 0) V contains a subset C of vertices which are colored. The *color change rule* is an operation in which a colored vertex u with a single uncolored neighbor v forces v to become colored. We say that u forces v. Set C is a *zero forcing set* if by applying the colored change rule until no more changes are possible all vertices are colored. The *zero forcing number* $Z(G)$ is the size of a minimum forcing set. See Figure 1 for an illustration of a sequence of applications of the color change rule on a graph with 10 vertices.

[*] Cite as: A. Agra, J. Orestes Cerdeira, C. Requejo. A computational comparison of compact MILP formulations for the zero forcing number. *Discrete Applied Mathematics* **269**, 169–183. 2019. [doi: 10.1016/j.dam.2019.03.027]



(a) $S = \{4, 7, 8\}$

(b) $S = \{4, 7, 8, 5, 6\}$

(c) $S = \{4, 7, 8, 5, 6, 10\}$

(d) $S = \{4, 7, 8, 5, 6, 10, 3\}$

(e) $S = \{4, 7, 8, 5, 6, 10, 3, 9\}$

(f) $S = \{4, 7, 8, 5, 6, 10, 3, 9, 2\}$

Figure 1: Example of applications of the color change rule on a graph with vertex set $V = \{1, 2, \ldots, 10\}$, starting from a zero forcing set $C = \{4, 7, 8\}$ in (a). Set S denotes the set of colored vertices at each time-step. In (b) 4 forces 5 and 7 forces 6, in (c) 6 forces 10, in (d) 10 forces 3, in (e) 8 forces 9, in (f) 3, 5 and 9 force 2. In the last step (not represented) 2 will force 1.

The zero forcing number was introduced in [2]. It has applications on different areas such as quantum physics and logic circuits (see [6, 9, 10, 12]). Determining the zero forcing number is NP-hard [1, 12]. The zero forcing number has been addressed in several recent papers [3, 4, 11, 14, 15, 16, 19] mostly focusing the determination of bounds. Integer programming approaches for the zero forcing number have been considered recently. Brimkov et al. [8] gave several integer programming formulations, including a compact model, for the minimum zero forcing set, and report computational experience on cubic graphs and Watts-Strogatz graphs.

In this paper we devise a new compact mixed integer linear program (MILP) model (i.e, a MILP formulation with a number of variables and constraints that is polynomial with respect to the size of the instance) based on time-step variables for the minimum zero forcing set. We show that this new model is stronger than the compact one introduced by Brimkov et al. [8] (i.e., the linear relaxation of the new model gives better bounds than those from [8]). Valid inequalities (i.e., inequalities that are satisfied by every feasible solution) and an extended formulation (i.e., a formulation including additional variables) are introduced to tighten the compact model. As the tighten compact model still has a weak linear relaxation, the model can only be used to solve to optimality small size instances. In order to solve larger instances we propose a sequential search algorithm that sequentially seeks for zero forcing sets of fixed cardinalities and stops when, for a given cardinality, no such zero forcing set exists. As the last step of proofing the infeasibility for a given cardinality is the most expensive from the computational point of view, by limiting the running time to find feasible solutions for large instances, the algorithm can be used as a heuristic. Computational results to compare the formulations and the algorithm are conducted on cubic, Watts-Strogatz and randomly generated graphs with 10, 20 and 30 vertices.

In Section 2 we present the compact formulation from Brimkov et al. [8] and review related polyhedral studies. The new compact MILP formulation based on time-step variables is introduced and discussed in Section 3. Valid inequalities and an extended formulation to tighten the new proposed model are presented in Section 4. In Section 5 an exact sequential search algorithm is introduced. The computational experiments are reported in Section 6. Final conclusions are given in Section 7.

2 Related integer programming approaches

Here we review the integer programming approaches for the zero forcing number. In what follows $N(v)$ denotes the set of neighbors of vertex v, $n = |V|$ is the number of vertices of graph G, and we let $k = n - 1$ and $T = \{1, \dots, k\}$. Moreover, we will orient every edge $\{u, v\}$ of graph $G = (V, E)$ in two opposite directions obtaining arcs (u, v) and (v, u).

We first present the formulation from Brimkov et al. [8]. The model includes integer variables of three types:

s_v indicates if vertex v is in the forcing set ($s_v = 1$) or not ($s_v = 0$);

x_v indicates at which time-step vertex v is forced;

y_{uv} indicates if vertex u forces vertex v ($y_{uv} = 1$) or not ($y_{uv} = 0$).

$$
\begin{aligned}
Z(G) = \min \quad & \sum_{v \in V} s_v && && (2.1)\\
s.t. \quad & s_v + \sum_{u \in N(v)} y_{uv} = 1 && \forall v \in V && (2.2)\\
& x_u - x_v + (k+1) y_{uv} \le k && \forall (u,v) \in E && (2.3)\\
& x_w - x_v + (k+1) y_{uv} \le k && \forall (u,v) \in E, w \in N(u) \backslash \{v\} && (2.4)\\
& s_v \in \{0,1\} && \forall v \in V && (2.5)\\
& x_v \in \{0, \dots, k\} && \forall v \in V && (2.6)\\
& y_{uv} \in \{0,1\} && \forall (u,v) \in E && (2.7)
\end{aligned}
$$

The objective function (2.1) searches for a minimum size forcing set. Equations (2.2) state that either vertex v is in the initial forcing set, or else some neighbor of v will force v. Inequalities (2.3) and (2.4) ensure that vertices will be iteratively colored in accordance with the color change rule. Indeed, together they guarantee that if u forces v, then u and all of its neighbors, but v, should be forced before v (i.e. if $u' \in (N(u) \setminus \{v\}) \cup \{u\}, x'_u < x_v$). The ranges of the variables are given by (2.5)-(2.7).

Model (2.1)–(2.7) will be denoted by BFH (Brimkov, Fast and Hicks). As mentioned in [8] this formulation is compact and it not only finds a minimum zero forcing set, but it also gives a set of forcing chains associated with the forcing set. However, constraints (2.3) and (2.4) are of big-M form leading to weak linear relaxations.

Another approach proposed by Brimkov et al. [8] is based on the concept of forts. A *fort* is a nonempty set $F \subset V$ such that no vertex outside F is adjacent to exactly one vertex in F. Note that if F is a fort, then $V \setminus F$ is not a zero forcing set. They propose the following set-covering model based on forts.

$$
\begin{aligned}
Z(G) = \min \quad & \sum_{v \in V} s_v && && (2.8)\\
s.t. \quad & \sum_{v \in F} s_v \geq 1 && \forall F \in \mathcal{B} && (2.9)\\
& s_v \in \{0,1\} && \forall v \in V && (2.10)
\end{aligned}
$$

where $\mathcal{B}$ is the set all forts in the given graph. Constraints (2.9) state that at least one vertex from every fort must be in every forcing set.

This formulation avoids the big-M constraints but it includes an exponential number of constraints. To solve this model they propose a row generation algorithm where violated forts are identified by solving an auxiliary integer programming problem.

3 A compact time-step formulation

In this section we introduce a new compact model for determining a minimum zero forcing set of a graph and discuss relations of this model to other compact models. This new model uses two types of variables 0-1 depending on the time-step:

x_v^t indicates if vertex v is forced at some time-step less than or equal t ($x_v^t = 1$) or not ($x_v^t = 0$);

y_{uv}^t indicates if vertex u forces vertex v at time-step t ($y_{uv}^t = 1$) or not ($y_{uv}^t = 0$).

$$
\begin{aligned}
Z(G) = \min \quad & \sum_{v \in V} x_v^0 && && (3.1)\\
s.t. \quad & x_v^0 + \sum_{t \in T} \sum_{u \in N(v)} y_{uv}^t = 1, && \forall v \in V && (3.2)\\
& y_{uv}^t \leq x_u^{t-1} && \forall (u,v) \in E, t \in T && (3.3)\\
& y_{uv}^t \leq x_w^{t-1} && \forall (u,v) \in E, w \in N(u) \setminus \{v\}, t \in T && (3.4)\\
& x_v^t = x_v^{t-1} + \sum_{u \in N(v)} y_{uv}^t && \forall v \in V, t \in T && (3.5)\\
& x_v^t \in \{0,1\} && \forall v \in V, t \in T \cup \{0\} && (3.6)\\
& y_{uv}^t \in \{0,1\} && \forall (u,v) \in E, t \in T && (3.7)
\end{aligned}
$$

The objective function (3.1) seeks that the number of colored vertices at time-step 0 is minimum. Equations (3.2) ensure that at time-step k all vertices will be colored. Constraints (3.3)-(3.5) implement the color change rule. Inequalities (3.3) express that only colored vertices can force other vertices. Each constraint (3.4) states that, at each time-step, vertex u will not force a neighbor v (i.e. $y^t_{uv} = 0$), whenever some other uncolored neighbor of u exists (i.e. if $x^{t-1}_w = 0$, $v \neq w \in N(u)$). Each constraint (3.5) establishes, on the one hand, that vertex v will be uncolored at time-step t (i.e. $x^t_v = 0$), if at time-step $t-1$ v is uncolored (i.e. $x^{t-1}_v = 0$) and at time-step t none of its neighbors have forced v (i.e. $y^t_{uv} = 0, u \in N(v)$), and on the other hand, that v is a colored vertex at time-step t (i.e. $x^t_v = 1$) if either it is a colored vertex at time-step $t-1$ (i.e. $x^{t-1}_v = 1$), or it has been colored at time step t (i.e. $y^t_{u,v} = 1$, for some $u \in N(v)$) . Finally, (3.6) and (3.7) state that all variables are 0-1.

This compact time-step model will be denoted by TS.

Remark 3.1. *Equality (3.5) together with nonnegativity of variable y^t_{uv} implies the following inequality*

$$x^t_v \geq x^{t-1}_v \qquad \forall v \in V, t \in T, \tag{3.8}$$

ensuring that once a vertex is colored it will remain colored in every subsequent time-step.

Remark 3.2. *Using equation* (3.5) *we can write variable x^t_v, for $v \in V, t \in T$, as follows*

$$x^t_v = x^0_v + \sum_{\ell=1}^{t} \sum_{u \in N(v)} y^\ell_{uv} \qquad \forall v \in V, t \in T. \tag{3.9}$$

These equations establish that either vertex v is in the forcing set (i.e. $x^0_v = 1$) and it will remain colored (i.e. $x^t_v = 1$) at every time-step $t \geq 1$, or else v will be (un)colored at time-step t iff it has (not) been forced by some neighbor vertex $u \in N(v)$ at (any) some time-step less than or equal to t.

Next we state that the coefficients matrix associated with the variables y^t_{uv} is totally unimodular. This allows us to replace constraints (3.7) by the nonnegativity constraints $y^t_{uv} \geq 0, (u,v) \in E, t \in T$, since the RHS of each constraint is integer for each binary vector x.

Proposition 3.3. *The coefficients matrix associated with the variables y^t_{uv} is totally unimodular.*

Proof. First notice that the coefficients matrix associated with each set of constraints (3.3) and (3.4) is an identity matrix. So, we focus only on the coefficients matrix resulting from (3.2), and (3.5). For each subset M of the matrix rows associated with (3.2), and (3.5) define the partition of M, $\{M_1, M_2\}$, such that M_1 are the rows in M resulting from constraints (3.2) and M_2 are the rows in M resulting from constraints (3.5). Since each variable y^t_{uv} occurs at most once in M_1, with coefficient 1, and occurs at most once in M_2, with coefficient 1, it follows that the partition satisfies $\mid \sum_{j \in M_1} a_{ij} - \sum_{j \in M_2} a_{ij} \mid \leq 1, i = 1, \ldots, \mid M \mid$, where a_{ij} represents the coefficient of column j in line i. Thus the coefficient matrix is totally unimodular (see [17]). □

3.1 Relation between models TS and BFH

Model TS can be related with BFH throughout the following set of equations:

$$s_v = x^0_v, \qquad v \in V \tag{3.10}$$

$$x_v = \sum_{t \in T} t(x^t_v - x^{t-1}_v), \qquad v \in V \tag{3.11}$$

$$y_{uv} = \sum_{t \in T} y^t_{uv}, \qquad (u,v) \in E \tag{3.12}$$

Model TS avoids the big-M constraints included in model BFH. Next we show that model TS is stronger than model BFH by showing that all the inequalities defining the linear relaxation of BFH are valid to the linear relaxation of model TS (see [17]).

Proposition 3.4. *Model TS is stronger than model BFH.*

Proof. In order to relate the two models we add the redundant constraints (3.10)-(3.12) to TS. The upper bounds on the variables $s_v \leq 1$ are implied by $x_v^0 \leq 1$. Constraints $x_v \leq k$ are implied by $x_v = \sum_{t\in T} t(x_v^t - x_v^{t-1}) \leq k\sum_{t\in T}(x_v^t - x_v^{t-1}) \leq k$, since from (3.8) $x_v^t - x_v^{t-1} \geq 0$ and $\sum_{t\in T}(x_v^t - x_v^{t-1}) = x_v^k - x_v^0 \leq x_v^k = 1$. Finally, from (3.5), for all $(u, v) \in E$, and $t \in T$, $x_v^t = x_v^{t-1} + \sum_{w\in N(v)} y_{wv}^t \geq x_v^{t-1} + y_{uv}^t$. Hence

$$y_{uv}^t \leq x_v^t - x_v^{t-1}. \tag{3.13}$$

Thus, $y_{uv} \leq 1$ is implied by $y_{uv} = \sum_{t\in T} y_{uv}^t \leq \sum_{t\in T}(x_v^t - x_v^{t-1}) \leq 1$.

Inequalities (2.2) follow directly from (3.2), (3.10) and (3.12). Next, we show that inequalities (2.3) are valid for the feasible set of TS. Using (3.13), it follows that

$$\sum_{t\in T} t y_{uv}^t \leq \sum_{t\in T} t(x_v^t - x_v^{t-1}) = x_v. \tag{3.14}$$

Now, observing that (i) $0 \leq x_u^0 \leq 1$, $0 \leq x_u^t - x_u^{t-1} \leq 1, \forall t \in T$; (ii) $0 \leq y_{uv}^t \leq 1, \forall t \in T$; (iii) $\sum_{t\in T}(x_u^t - x_u^{t-1}) + x_u^0 = 1$; (iv) $\sum_{t\in T} y_{uv}^t \leq 1$; (v) $\sum_{\ell=1}^{t} y_{uv}^\ell \leq x_u^{t-1} = \sum_{\ell=1}^{t-1}(x_u^\ell - x_u^{\ell-1}) + x_u^0$, $\forall t \in T$, and using Lemma 7.1 given in the Appendix II, with $a_0 = x_u^0, a_t = x_u^t - x_u^{t-1}, t \in T$ and $b_t = y_{uv}^t$, it follows that

$$\sum_{t\in T} t(x_u^t - x_u^{t-1}) + \sum_{t\in T}(k + 1 - t)y_{uv}^t \leq k \tag{3.15}$$

Adding (3.14) and (3.15) one obtains

$$\sum_{t\in T} t(x_u^t - x_u^{t-1}) + \sum_{t\in T}(k + 1 - t)y_{uv}^t + \sum_{t\in T} t y_{uv}^t \leq x_v + k$$

$$\Leftrightarrow x_u + (k + 1)y_{uv} \leq x_v + k$$

We omit the proof of validity of inequalities (2.4) since it is similar to the proof above for inequalities (2.3).

In order to show that the linear relaxation of model TS can provide better bounds than the linear relaxation of model BFH, it suffices to consider the example of a path graph with four vertices. The optimal value of the linear relaxation of model BFH is zero and the corresponding value for model TS is 1/3. □

To finalize the comparison between both models, we indicate below the number of variables and constraints of each of the two models.

Model	# int. var.	# binary var.	# constraints
BFH	$\mathcal{O}(n)$	$\mathcal{O}(n + \vert E\vert)$	$\mathcal{O}(n\vert E\vert)$
TS		$\mathcal{O}(k(n + \vert E\vert))$	$\mathcal{O}(kn\vert E\vert)$

3.2 Relation between model TS and other time-step based models

Here we relate model TS with two other models based on time-steps which we derive as a consequence of the following two observations.

- Equations (3.9) and (3.11) suggest that variables x_v^t, for $t \in T$, can be eliminated from the model TS leading to a new model with less variables.
- The color change rule establishes that at each time-step, a vertex v either maintains its color from the previous time-step or it is forced by a neighbor $u \in N(v)$. Thus, for each time-step $t \geq 1$ and each node v, such rule can be modeled as the union of the following polyhedron:
  $Q_v^t(v) = \left\{x_v^t = x_v^{t-1}\right\}$ if v maintains its color;
  $Q_v^t(u) = \left\{x_v^t = x_w^{t-1},\ w \in (N(u) \cup \{u\}) \backslash \{v\}\right\}$ if v is forced by u. Thus, a formulation can be derived from the technique introduced by Balas [5] to model the convex hull of the union of polyhedra.

Following the first observation, variables x_v^t for $t \in T$ are eliminated from model TS using the Fourrier-Motzkin elimination. The resulting projected formulation, denoted by PTS, is given below, where variables x_v^0 are replaced by s_v.

$$Z(G) = \min \quad \sum_{v \in V} s_v \tag{3.16}$$

$$s.t. \quad s_v + \sum_{t \in T} \sum_{u \in N(v)} y_{uv}^t = 1, \qquad \forall v \in V \tag{3.17}$$

$$y_{uv}^{t+1} \leq s_u + \sum_{\ell=1}^{t} \sum_{v' \in N(u) \backslash \{v\}} y_{v'u}^{\ell}, \qquad \forall (u,v) \in E, t \in T \setminus \{k\} \tag{3.18}$$

$$y_{uv}^{t+1} \leq s_w + \sum_{\ell=1}^{t} \sum_{v' \in N(w)} y_{v'w}^{\ell}, \qquad \forall (u,v) \in E, w \in N(u) \setminus \{v\}, \; t \in T \setminus \{k\} \tag{3.19}$$

$$s_v \in \{0,1\}, \qquad \forall v \in V \tag{3.20}$$

$$y_{uv}^t \in \{0,1\}, \qquad \forall (u,v) \in E, t \in T \tag{3.21}$$

Interestingly, model PTS has fewer constraint than the model TS.

Remark 3.5. *While for the TS model variables y_{uv}^t can be relaxed, in the PTS model the coefficients matrix is no longer totally unimodular, and it can be proved that variables y_{uv}^t cannot be relaxed.*

Now, we address the second observation. Following [5], in addition to variables x_v^t we add the following two sets of variables:

z_{uv}^t is the copy of variable x_u^t for polyhedron $Q_v^t(u)$, where $u \in N(v) \cup \{v\}$.

δ_{uv}^t indicates whether polyhedron $Q_v^t(u)$ is selected ($\delta_{uv}^t = 1$) or not ($\delta_{uv}^t = 0$).

The resulting formulation (the details are given in Appendix I), which we will refer as UP model, is as follows.

$$Z(G) = \min \quad \sum_{v\in V} x_v^0 \tag{3.22}$$

$$s.t. \quad x_v^0 + \sum_{t\in T} \sum_{u\in N(v)} z_{uv}^t = 1, \qquad \forall v \in V \tag{3.23}$$

$$x_v^t \geq x_v^{t-1} \qquad \forall v \in V, t \in T \tag{3.24}$$

$$z_{vv}^t \leq x_v^{t-1} \qquad \forall (u,v) \in E, t \in T \tag{3.25}$$

$$z_{vv}^t \leq \delta_{vv}^t \qquad \forall (u,v) \in E, t \in T \tag{3.26}$$

$$z_{vv}^t + x_v^{t-1} \leq 1 + \delta_{vv}^{t-1} \qquad \forall (u,v) \in E, t \in T \tag{3.27}$$

$$z_{uv}^t \leq x_u^{t-1} \qquad \forall (u,v) \in E, t \in T \tag{3.28}$$

$$z_{uv}^t \leq x_w^{t-1} \qquad \forall (u,v) \in E, w \in N(u) \setminus \{v\}, t \in T \tag{3.29}$$

$$z_{uv}^t \leq \delta_{uv}^t \qquad \forall (u,v) \in E, t \in T \tag{3.30}$$

$$z_{uv}^t + x_w^{t-1} \leq 1 + \delta_{uv}^t \qquad \forall (u,v) \in E, t \in T \tag{3.31}$$

$$x_v^t = \sum_{u\in N(v)\cup\{v\}} z_{uv}^t \qquad \forall v \in V, t \in T \tag{3.32}$$

$$\sum_{u\in N(v)\cup\{v\}} \delta_{uv}^t = 1 \qquad \forall v \in V, t \in T \tag{3.33}$$

$$x_v^t \in \{0,1\} \qquad \forall v \in V, t \in T \cup \{0\} \tag{3.34}$$

$$z_{uv}^t \in \{0,1\} \qquad \forall (u,v) \in E, t \in T \tag{3.35}$$

$$\delta_{uv}^t \in \{0,1\} \qquad \forall (u,v) \in E, t \in T \tag{3.36}$$

Notice that variables z_{uv}^t are essentially the same as y_{uv}^t. In fact, it suffices to add variables y_{vv}^t and constraints $y_{vv}^t \leq x_t$ to model TS in order to obtain (3.1)–(3.7) from (3.22)–(3.25), (3.28), (3.29), (3.32), (3.34), (3.35). In addition, model UP contains (3.26), (3.27), (3.30), (3.31), (3.33) and the 0-1 variables δ_{uv}^t, that can actually be relaxed to $\delta_{uv}^t \geq 0$ (see [5]). Thus model UP has more variables and constraints than model TS. Moreover, preliminary computational results, show that some of the additional constraints, (3.27) and (3.31), are, in general, not tight for the linear relaxation solution of UP.

4 Strengthening model TS

Here we discuss two approaches to strengthen the model TS. The first approach consists in replacing inequalities (3.3) and (3.4) by the following stronger inequalities

$$\sum_{\ell=1}^{t} \sum_{w\in N(u)} y_{uw}^\ell \leq x_u^{t-1} \qquad u \in V, t \in T \tag{4.1}$$

$$\sum_{\ell=1}^{t} \sum_{u\in N(v)\cap N(w)} y_{uv}^\ell \leq x_w^{t-1} \qquad \forall v, w \in V, v \neq w, t \in T \tag{4.2}$$

Inequalities (4.1) imply (3.3) and state that if node u forced a neighbor w at a time-step ℓ between 1 to t, then u must be colored at time-step $t-1$ (since it must have been colored at time-step

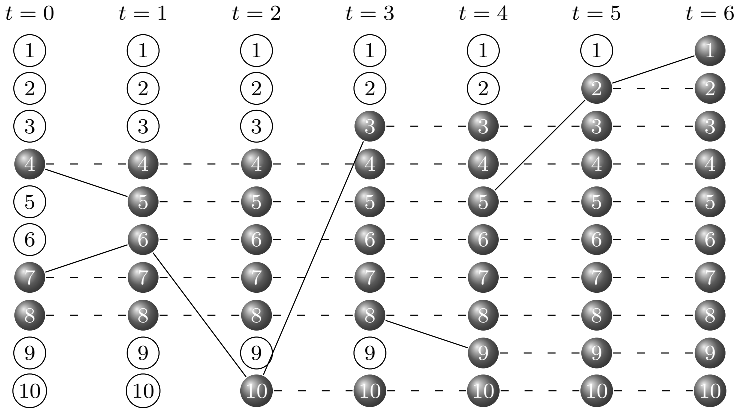


Figure 2: Example of the forced $v-$path for every vertex v of the graph of Figure 1 corresponding to the sequence of applications of the color changed rule illustrated in Figure 1.

$\ell - 1)$. Inequalities (4.2) imply (3.4) and establish that if node v is forced at a given time-step ℓ between 1 and t by a neighbor of node w, then node w must be colored at time-step $t - 1$.

Inequalities (4.1) and (4.2) are obviously valid for the feasible set of model TS.

Notice that by eliminating variables x_v^t, as we did before to derive model PTS, inequalities (4.1) and (4.2) can be written in the space of variables s_v and y_{uv}^t and added to the projected model (3.16)-(3.21).

Note that for a path graph with $n = 4$, the strengthened model TS has linear relaxation value equal to 1.

The second approach to reinforce model TS is obtained adding multi-commodity flow constraints. This model will be denoted by TS+MCF.

Let C be a forcing set and consider a sequence of applications of the color change rule that colored all vertices. The sequence can be described as a set of n $(k+1)$-paths $P_v = (v^0, v^1, \ldots, v^k = v)$ that describes the applications of the rule that forced each vertex v. In path P_v, $v^{t-1} = v^t$ if v^{t-1} did not forced v^t at time-step t; otherwise v^{t-1} is the vertex that forced v^t at time-step t. Thus, if $v \in C$, the $(k+1)$-path $P_v = (v, v, \ldots, v)$. Note that, for each sequence, path P_v is unique. We call P_v the *forced v-path.* Figure 2 shows, for every vertex v of the graph of Figure 1, the forced v-path corresponding to the sequence of applications of the color changed rule that is illustrated in Figure 1.

In this model we use the variables of model TS, and in addition the following (multi-commodity flow) 0-1 variables:

$f_{uw}^t(v)$ indicates if $u = v^{t-1}$ and $w = v^t$ in forced v-path ($f_{uw}^t(v) = 1$) or not ($f_{uw}^t(v) = 0$).

The TS+MCF model consists of (3.1)-(3.7), and

$$\sum_{u \in V} f_{uv}^{k}(v) = 1 \qquad \forall v \in V \tag{4.3}$$

$$\sum_{w \in V} f_{wu}^{t}(v) = \sum_{w \in V} f_{uw}^{t+1}(v) \qquad \forall u, v \in V, t \in T \setminus \{k\} \tag{4.4}$$

$$\sum_{u,w \in V} f_{uw}^{1}(v) = 1 \qquad \forall v \in V \tag{4.5}$$

$$f_{uw}^{t}(v) \leq y_{uw}^{t} \qquad \forall v \in V, (u, w) \in E, t \in T \tag{4.6}$$

$$f_{uu}^{t+1}(v) \leq x_{u}^{t} \qquad \forall u, v \in V, t \in T \setminus \{k\} \tag{4.7}$$

$$f_{uw}^{t}(v) \in \{0, 1\} \qquad \forall u, w, v \in V, t \in T \tag{4.8}$$

Constraints (4.3)-(4.5) define the set of forced v-paths, for every vertex v. More precisely, equations (4.3) and (4.5) state, for each vertex v, that, at time-step 0, there will be one unit of flow sent to v and, at time-step k, one unit of flow entering v. The flow conservation constraints (4.4) guarantee that the flow directed to vertex v that enters each vertex equals the flow leaving that vertex. Inequalities (4.6) and (4.7) relate forced v-paths with the color change rule, stating that if u and w are vertices in positions t and $t+1$ of the v-path, then (i) if $u \neq w$, u forced w at time-step t; and (ii) if $u = w$, u has been forced at some time-step less than or equal to t.

5 A sequential search algorithm

Preliminary computational tests have shown that the linear relaxation value of the models presented in Section 3 give very poor lower bounds. To improve lower bounds we consider searching zero forcing sets of fixed cardinalities, tacking into account the following proposition that uses the fact that if C is a zero forcing set, then $C \cup \{j\}$ is a zero forcing set, for all $j \in V \setminus C$.

Proposition 5.1. *If there is a forcing set with cardinality κ, then there exist forcing sets with cardinality $\kappa + 1, \cdots, k$.*

Using Proposition 5.1 we propose the following procedure to find minimum zero forcing sets.

Algorithm 1 Sequential search algorithm:

```
initialize $f = n$;
repeat
    $f \leftarrow f - 1$;
    add the constraint $\sum_{v \in V} x_v^0 = f$ (or $\sum_{v \in V} s_v^0 = f$) to the model;
    solve the model using a solver until either a feasible solution is found or the problem if
      proven to be unfeasible;
until an unfeasible model is found.
the zero forcing number is $Z(G) = f + 1$.
```

Proposition 5.1 ensures that if there is no feasible set with cardinality κ, then there is no feasible set with cardinality $1, \ldots, \kappa - 1$.

A search for the minimum cardinality zero forcing set using bisection on the cardinality of the zero forcing set would require to solve ($\mathcal{O}(log_2(k))$ MILPs instead of the $\mathcal{O}(k)$ MILPs in the case of sequential search. However, computational tests (see Section 6) show that the running time to prove that there is no zero forcing with a given cardinality is greater than the running time to find a feasible solution for each cardinality such that a zero forcing set exists, that is, most of the running time of the sequential algorithm is spent on the last iteration. Thus, by following the sequential Algorithm 1, we ensure that the unfeasible problem is solved only once.

We found that Algorithm 1 did not improve models BFH and TS+MCF. However, for the largest graphs of our computational tests, model TS improved with Algorithm 1. To explore this further, we consider a refinement of Algorithm 1 to overcome some difficulties inherent to the minimum forcing set problem, that we now discuss.

The problem embodies some "symmetry" as can be seen from the example given in Figure 1. For instance, in Figure 1 (a) vertex 4 can force vertex 5 and vertex 7 can force vertex 6. The color change can occur simultaneously or in sequence. Additionally, as the zero forcing set has cardinality 3 (greater than 1), there is a slack in the number of time-steps necessary to color all the vertices. Hence, for each solution, it is possible to derive another solution by pushing forward or pushing back in time the color change. In the example, it is possible to derive solutions where in the first two time-steps there is no color change. In order to circumvent these difficulties we consider the following objective function that prioritizes solutions where the color change occurs as soon as possible:

$$\min \sum_{(u,v)\in E} \sum_{t\in T} t\, y_{uv}^{t}. \tag{5.1}$$

The size of the time-step formulation TS depends on the number of time-steps required to perform all the color changes. If the zero forcing set has cardinality κ, then at most $k = n-\kappa$ time-steps are required to color all the graph vertices. In the original formulation we set the maximum number of time-steps to $k = n-1$, since all zero forcing sets have cardinality of at least one. The knowledge of the cardinality of the zero forcing set, as it is the case in each step of Algorithm 1, permits to eliminate many variables and constraints, thus reducing the size of the problem substantially. In particular, by considering a fixed cardinality of κ vertices, the following variables are eliminated:

$$y_{uv}^{t}, \quad \text{with } (u,v) \in E, t \in T | t > n-\kappa \quad \text{(which would be equal to 0)} \tag{5.2}$$

$$x_{u}^{t}, \quad \text{with } u \in V, t \in T | t > n-\kappa \quad \text{(which would be equal to 1)} \tag{5.3}$$

$$f_{uw}^{t}(v), \quad \text{with } v \in V, (u,w) \in E, t \in T | t > n-\kappa \quad \text{(which would be equal to 0)} \tag{5.4}$$

To take advantage of this we propose to upgrade Algorithm 1 adding the two following steps:

Algorithm 2 Refined sequential search algorithm (Algorithm 1 with these two additional steps):

```
0: replace the objective function by (5.1);
4': eliminate variables (5.2)-(5.4).
```

We call this refined algorithm the RefinedAlgorithm 1.

It is worthy of notice that Algorithm 1 can act as a heuristic if we restrict the running time in Step 5. In this case the algorithm may be applied to large size instances and returns an upper bound and the corresponding forcing set.

6 Computational results

This section reports the computational experiments performed to compare the performance of the models BFH, TS and TS+MCF and of the two-level exact algorithm.

All tests were run on a computer with an Intel(R) Core(TM) i7-4750HQ processor, having a 2.00GHz CPU and 8GB of RAM, using the solver Xpress-Optimizer 28.01.04 with the default options.

Three sets of instances were used with a number of vertices n taking the values $10, 20, 30$: the set of the "*cubic*" graphs, the set of the "*Watts-Strogatz*" graphs and the set of the "*generated*" graphs. The two sets "*cubic*" graphs and "*Watts-Strogatz*" graphs are borrowed from [8] and were obtained from $https : //github.com/calebfast/zero_forcing$. The "*cubic*" graphs have a number of edges m equal to $3n/2$. The "*Watts-Strogatz*" graphs are grouped in two subsets: WS-03, having a number of edges m equal to $2n$ and WS-05, having a number of edges m equal to $5n$. The third set of graphs, the set of the "*generated*" graphs, were randomly generated having density of 0.25 (we obtained instances with an average of $m = 11$ edges for $n = 10$, average of $m = 44.6$ edges for $n = 20$ and an average of $m = 107.6$ for $n = 30$), having density of 0.5 (we obtained instances with an average of $m = 20$ edges for $n = 10$, an average of $m = 92.8$ edges for $n = 20$ and an average of $m = 218.4$ for $n = 30$) and having density of 0.75 (we obtained instances with an average of $m = 32.6$ edges for $n = 10$, an average of $m = 141.8$ edges for $n = 20$ and an average of $m = 328.2$ for $n = 30$).

In this paper several variants of model TS were discussed and two approaches to tighten the TS model were introduced. A full detailed report of the comparison between all the possible combinations would be too extensive. So, we opt to report only the comparison between models BFH, TS and TS+MCF. Preliminary tests have allowed us to draw a few conclusions: (i) solving the model TS with inequalities (3.3) and (3.4) replaced by (4.1) and (4.2) do not bring apparent improvement; (ii) adding (4.1) and (4.2) is in general worse than solving the model TS; (iii) further including valid inequalities such as a generated set of forts did not prove evidence of improvement, and therefore we do not include forts; (iv) solving model UP is slower than solving model TS, and the running times become comparable when inequalities (3.27) and (3.31) are removed from model UP; (v) although two variants of TS have theoretical advantages in relation to TS, namely, the one where variables y^t_{uv} are relaxed, and the TS+MCF model, the tests have not been conclusive, that is, for some instances solving such variants was faster and for others was slower. Thus, in the computational tests reported here model TS refers to formulation (3.1)-(3.7).

Tables 1, 2, 3 and 4 display the obtained computational results. The Table 1 shows computational execution elapsed times for each model. In Table 2 are displayed the computational results of Algorithm 1 when applied to the models BFH, TS and TS+MCF. The computational results of the RefinedAlgorithm 1 using model TS are displayed in Tables 3 and 4.

In Tables 1 and 2, the first column named "Set" shows the graph set for which the results are shown in the corresponding lines. The second column named "n" gives the number of vertices and column named "m" indicates the number of edges of the graph. Columns named "BFH", "TS" and "TS+MCF" in Table 1 indicate the computational times to solve the instance using models BFH, TS and TS+MCF, respectively. Columns named "BFH", "TS" and "TS+MCF" in Table 2 indicate the computational times to solve the instance with Algorithm 1 using models BFH, TS and TS+MCF, respectively. The values in bold In both tables show the best performance for each instance.

A maximum time limit of 3600 seconds is imposed. The symbol “T” indicates that the maximum time limit is reached without completing the search for a solution.

In Tables 3 and 4 computational execution times of the RefinedAlgorithm 1 using model TS are discriminated as follows. The column named “time 1” shows the computational time to obtain zero forcing sets for all cardinalities for which feasible solutions exist. The column named “time 2” reports the computational time spent in the last problem to prove infeasibility. The column named “total time” gives the total computational time used by the RefinedAlgorithm 1 it is the sum of the values in columns “time 1” and “time 2”. In Table 1 observe that it is possible to solve, with each of the three models, the problem for instances with a few number of edges, that is a optimal solution is found for all the instances with $n = 10$, and $n = 20$ and with a number of edges lower than 50. For the instances with a higher number of edges (in our instances, when the number of edges is higher than 100) all the models get difficulties to find the optimal solution.

In Table 2 observe that using the Algorithm 1 some of the instances with a number of edges greater than 100 are solved, however there are many of these instances for which this algorithm is unable to obtain a solution within a reasonable amount of time. The computational results suggest that better computational times are obtained when using model TS. The computational times for Algorithm 1 with the three models were statistically compared using the non-parametric ANOVA Friedman test with the post-hoc Nemenyi’s all-pairs comparisons tests, implemented in the functions *friedmanTest* and *frdAllPairsNemenyiTest*, respectively, from package PMCMRplus (T. Pohlert, 2018) of R Statistical Software (R Core Team, 2018) [18]. The Friedman test concluded that the methods performed significantly differently (p-value = 1.046e-07). The post-hoc Nemenyi’s test showed that TS was the method with lower computational times (p-values =1.4e-05 and 7.9e-07, when TS is compared with BFH and TS+MCF, respectively).

Tables 3 and 4 report the discriminated computational execution times for the RefinedAlgorithm 1 using model TS. Table 3 refers to the same instances of Tables 1 and 2. The Friedman test applied to the computational times of Tables 1, 2 and 3 clearly indicates that the seven approaches performed distinctly (p-value $<$ 2.2e-16), and the post-hoc Nemenyi’s test showed that the RefinedAlgorithm 1 using model TS had the best performance (p-values ≤ 0.0006 when compared with each of the other six approaches). Table 4 indicates the computational times of the RefinedAlgorithm 1 using model TS on larger graphs.

Regarding Tables 3 and 4, observe that for the instances with a number of edges lower than 50 when comparing the time values “time 1” and “time 2” the lowest value is “time 2” that corresponds to the time spent in the last problem that is unfeasible. For the instances with a number of edges greater than 100 when comparing the time values “time 1” and “time 2” the lowest value is “time 1” that corresponds to the computational time to solve all the feasible problems. For these instances the huge amount of computational execution time is due to the time that the solver uses to prove the infeasibility of the last problem. The major part of the total computational time reported in the last column is spent proving the infeasibility of the last problem. Computational execution times reported in column “time 1” correspond to solving an average of $2n/3$ models, each until a feasible solution is found, using an average of 28.2 seconds and ranging from 0.1 seconds to 631.825 seconds. A value for “time 1” is obtained for all the instances. A value for "time 2" could not be obtained for all the instances within a time limit of 3600 seconds. For the values displayed in column "time 2", the computational time spent in the last model uses an average of 150.6 seconds and ranges from 0.003 to 1939.482 seconds. However, for half of the instances with number of vertices $n = 30$ the time spent in the last model reaches the time limit and the symbol "T" is shown in column "time 2".

Computational results not reported here for instances with $n = 100$ and $m = 150$ used a computational time of 5000 seconds to solve 81 models indicating that this algorithm can be used as a heuristic to obtain very good upper bounds to the zero forcing number.

Table 1: Computational execution times (in seconds) for the models.

Set	n	m	BFH	TS	TS+MCF
cubic graphs	10	15	**0.116**	0.401	1.967
			0.163	0.384	3.021
			1.010	**0.448**	2.284
			0.200	0.363	4.048
			0.232	0.369	2.075
	20	30	**70.274**	344.178	758.987
			17.477	266.576	1697.307
			944.579	1004.301	T
			98.090	594.904	**42.816**
			96.586	364.227	2476.605
Watts-Strogatz graphs	10	20	**0.332**	0.555	2.990
			0.402	0.447	2.181
			0.532	0.549	2.665
			5.691	**0.731**	3.691
			0.940	**0.724**	3.168
	20	40	2318.259	521.564	**65.707**
			702.867	567.773	**87.335**
			1783.857	488.780	**73.368**
			1557.586	1819.668	2709.792
			1269.475	708.378	**78.058**
	20	100	T	T	T
			T	T	T
			T	T	T
			T	T	T
			T	T	T
generated graphs density=0.25	10	8	0.064	**0.032**	0.815
		11	**0.069**	0.216	1.499
		14	**0.084**	0.332	1.592
		14	**0.115**	0.238	1.632
		8	**0.084**	0.115	0.782
	20	44	810.418	**33.549**	107.745
		43	542.256	561.824	**83.026**
		46	810.076	1961.876	**105.096**
		47	749.080	**20.308**	97.363
		43	**159.710**	674.272	2591.311
generated graphs density=0.5	10	20	**0.635**	0.664	2.544
		20	**0.449**	0.504	2.908
		20	1.131	**0.586**	3.026
		19	**0.121**	0.337	2.100
		21	1.736	**0.517**	2.489
	20	94	T	T	T
		93	T	**84.404**	T
		94	T	**528.662**	T
		83	T	T	T
		100	T	T	T
generated graphs density=0.75	10	30	18.914	**0.961**	5.204
		31	8.547	**0.602**	3.063
		31	49.379	**1.186**	3.662
		37	5.893	**1.013**	3.461
		34	4.613	**0.958**	3.740
	20	142	T	**2044.048**	T
		134	T	T	**3210.809**
		141	T	1885.940	**397.036**
		143	T	2308.450	**668.422**
		149	T	T	**2045.434**

Table 2: Computational execution times (in seconds) for Algorithm 1, using the different models.

Set	n	m	BFH	TS	TS+MCF
cubic graphs	10	15	**0.220**	0.648	6.359
			0.210	0.430	6.698
			0.430	0.503	6.428
			0.274	0.793	7.791
			0.313	**0.237**	7.948
	20	30	386.692	**21.445**	343.518
			92.450	**17.554**	459.222
			1727.584	**37.000**	597.148
			378.031	**27.599**	456.772
			761.453	**28.498**	567.149
Watts-Strogatz graphs	10	20	**0.635**	0.716	3627.761
			1.254	**0.835**	3604.870
			1.264	**0.671**	10.014
			1.835	**0.750**	7.268
			1.621	**0.566**	6.504
	20	40	1375.116	**86.911**	699.838
			1953.090	**86.949**	764.752
			2476.575	**73.808**	666.853
			T	**106.543**	611.255
			1398.838	**116.693**	702.626
	20	100	T	2980.339	**71.27**
			T	3231.598	**75.163**
			T	2774.608	**220.111**
			T	1814.136	**71.688**
			T	3310.442	**399.616**
generated graphs density=0.25	10	8	**0.016**	0.329	3.669
		11	**0.084**	3600.731	3603.218
		14	**0.116**	0.514	7.025
		14	**0.153**	0.414	4.685
		8	**0.079**	0.411	3.529
	20	44	1119.450	**98.555**	745.827
		43	2424.166	**132.240**	707.126
		46	1806.567	**119.347**	1315.449
		47	396.274	**109.063**	816.265
		43	960.793	**115.363**	916.108
generated graphs density=0.5	10	20	**0.385**	0.609	3604.679
		20	**0.361**	0.739	9.905
		20	**0.590**	0.679	6.743
		19	**0.200**	0.726	7.424
		21	**1.039**	1.185	8.536
	20	94	T	**248.068**	2137.264
		93	T	**815.919**	2750.737
		94	T	**378.635**	1516.657
		83	T	**501.256**	T
		100	T	**458.897**	T
generated graphs density=0.75	10	30	16.712	**1.244**	7.586
		31	8.009	**0.831**	5.632
		31	44.532	**1.110**	7.665
		37	5.289	**1.933**	8.909
		34	4.338	**1.918**	11.155
	20	142	T	**53.237**	751.393
		134	T	**50.742**	1811.061
		141	T	**77.812**	1392.712
		143	T	**86.079**	1142.248
		149	T	**60.608**	549.997

Table 3: Discrimination of the computational execution times (in seconds) for RefinedAlgorithm 1 using model TS, with the instances of Tables 1 and 2.

Set	n	m	time 1	time 2	total time
cubic graphs	10	15	0.254	0.004	0.258
			0.33	0.004	0.334
			0.348	0.006	0.354
			0.377	0.004	0.381
			0.291	0.006	0.297
	20	30	3.107	1.466	4.573
			3.046	1.011	4.057
			3.163	1.464	4.627
			3.192	0.301	3.493
			3.104	1.322	4.426
Watts-Strogatz graphs	10	20	0.387	0.078	0.465
			0.298	0.104	0.402
			0.269	0.147	0.416
			0.266	0.102	0.368
			0.191	0.105	0.296
	20	40	4.649	3.713	8.362
			4.153	4.048	8.201
			4.835	3.626	8.461
			4.27	3.307	7.577
			4.869	4.183	9.052
	20	100	6.334	21.315	27.649
			11.717	21.352	33.069
			11.145	23.347	34.492
			11.37	20.233	31.603
			5.939	21.256	27.195
generated graphs density=0.25	10	8	0.1	0.025	0.125
		11	0.187	0.003	0.19
		14	0.237	0.009	0.246
		14	0.203	0.04	0.243
		8	0.178	0.049	0.227
	20	44	4.968	3.491	8.459
		43	4.359	4.292	8.651
		46	7.227	6.26	13.487
		47	5.005	3.416	8.421
		43	4.451	3.714	8.165
generated graphs density=0.5	10	20	0.308	0.005	0.313
		20	0.332	0.005	0.337
		20	0.231	0.12	0.351
		19	0.269	0.043	0.312
		21	0.294	0.027	0.321
	20	94	9.95	17.399	27.349
		93	5.439	27.647	33.086
		94	10.956	17.216	28.172
		83	6.014	26.194	32.208
		100	9.622	16.371	25.993
generated graphs density=0.75	10	30	0.345	0.162	0.507
		31	0.425	0.179	0.604
		31	0.437	0.232	0.669
		37	0.292	0.07	0.362
		34	0.392	0.03	0.422
	20	142	11.651	35.453	47.104
		134	5.555	24.953	30.508
		141	6.049	20.027	26.076
		143	7.146	15.975	23.121
		149	7.273	9.122	16.395

Table 4: Discrimination of the computational execution times (in seconds) for RefinedAlgorithm 1 using model TS, for instances with 30 vertices.

Set	n	m	time 1	time 2	total time
cubic graphs	30	45	13.325	83.603	96.928
			11.739	94.063	105.802
			11.431	150.327	161.758
			12.549	103.117	115.666
			12.424	115.18	127.604
Watts-Strogatz graphs	30	60	15.731	648.532	664.263
			16.853	472.763	489.616
			36.588	146.56	183.148
			244.787	148.771	393.558
			21.669	453.151	474.82
	30	150	56.536	T	T
			55.591	T	T
			61.029	T	T
			81.832	T	T
			43.004	T	T
generated graphs density=0.25	30	105	26.593	T	T
		101	631.823	T	T
		118	70.867	T	T
		111	21.929	T	T
		103	67.483	T	T
generated graphs density=0.5	30	214	79.537	T	T
		223	108.999	T	T
		230	56.931	T	T
		200	32.736	T	T
		225	56.08	T	T
generated graphs density=0.75	30	335	72.642	1220.779	1293.421
		323	64.78	1936.324	2001.104
		333	72.072	1293.738	1365.81
		318	67.579	1939.482	2007.061
		332	66.84	1367.482	1434.322

7 Conclusions

A new compact model is introduced to find a minimum forcing set of a graph. In relation to the previous known compact model, this new model has the advantage of avoiding the use of big-M constraints. An analysis of this model is conducted and its relation to related models and variants are discussed. The formulation is tightened with valid inequalities and an extended formulation. Despite the improvements on the model, computational tests revealed that it can only be used to solve small size instances to optimality. In order to consider large size instances, a sequential search algorithm is proposed. This algorithm allows to solve to optimality larger size instances than using a commercial solver based on the compact model. The sequential search algorithm can be used as a heuristic to provide upper bounds for the zero forcing number.

As future research it would be interesting to compare the upper bounds provided by the search algorithm used as a heuristic, with the known theoretical upper and lower bounds described in the literature.

Appendix I

For each $v \in V$ and $t \in T$ the union of $Q^t_v(u)$, $u \in N(v) \cup \{v\}$ can be described by the following set of constraints.

$$z^t_{vv} = x^{t-1}_v \delta^t_{vv} \qquad \forall (u,v) \in E, t \in T \tag{7.1}$$

$$z^t_{uv} = x^{t-1}_w \delta^t_{uv} \qquad \forall (u,v) \in E, w \in N(u) \cup \{u\} \setminus \{v\}, t \in T \tag{7.2}$$

$$\sum_{u \in N(v) \cup \{v\}} \delta^t_{uv} = 1 \qquad \forall v \in V, t \in T \tag{7.3}$$

Constraints (7.1) and (7.2) are nonlinear and can be linearized using additional variables $\zeta^t_{uvw} = x^{t-1}_w \delta^t_{uv}$. Constrains (7.1) are linearized as follows.

$$\begin{aligned}
& z^t_{vv} = \zeta^t_{vvv} && \forall (u,v) \in E, t \in T \\
& \zeta^t_{vvv} \le x^{t-1}_v && \forall (u,v) \in E, t \in T \\
& \zeta^t_{vvv} \le \delta^t_{vv} && \forall (u,v) \in E, t \in T \\
& x^{t-1}_v + \delta^t_{vv} \le 1 + \zeta^t_{vvv} && \forall (u,v) \in E, t \in T
\end{aligned}$$

Similarly, constrains (7.2) are linearized as follows.

$$\begin{aligned}
& z^t_{uv} = \zeta^t_{uvw} && \forall (u,v) \in E, w \in N(u) \cup \{u\} \setminus \{v\}, t \in T \\
& \zeta^t_{uvw} \le x^{t-1}_w && \forall (u,v) \in E, w \in N(u) \cup \{u\} \setminus \{v\}, t \in T \\
& \zeta^t_{uvw} \le \delta^t_{uv} && \forall (u,v) \in E, w \in N(u) \cup \{u\} \setminus \{v\}, t \in T \\
& x^{t-1}_w + \delta^t_{uv} \le 1 + \zeta^t_{uvw} && \forall (u,v) \in E, w \in N(u) \cup \{u\} \setminus \{v\}, t \in T
\end{aligned}$$

Projection out variables ζ^t_{uvw} one obtains the following set of constraints on variables.

$$\begin{aligned}
& z^t_{vv} \le x^{t-1}_v && \forall (u,v) \in E, t \in T \\
& z^t_{vv} \le \delta^t_{vv} && \forall (u,v) \in E, t \in T \\
& z^t_{vv} + x^{t-1}_v \le 1 + \delta^{t-1}_{vv} && \forall (u,v) \in E, t \in T \\
& z^t_{uv} \le x^{t-1}_u && \forall (u,v) \in E, t \in T \\
& z^t_{uv} \le x^{t-1}_w && \forall (u,v) \in E, w \in N(u) \setminus \{v\}, t \in T \\
& z^t_{uv} \le \delta^t_{uv} && \forall (u,v) \in E, t \in T \\
& z^t_{uv} + x^{t-1}_w \le 1 + \delta^t_{uv} && \forall (u,v) \in E, t \in T \\
& x^t_v = \sum_{u \in N(v) \cup \{v\}} z^t_{uv} && \forall v \in V, t \in T \\
& \sum_{u \in N(v) \cup \{v\}} \delta^t_{uv} = 1 && \forall v \in V, t \in T \\
& z^t_{uv} \in \{0,1\} && \forall (u,v) \in E, t \in T \\
& \delta^t_{uv} \ge 0 && \forall (u,v) \in E, t \in T
\end{aligned}$$

Appendix II

Lemma 7.1. *Given* $0 \leq a_i \leq 1, i = 0, \ldots, k$ *and* $0 \leq b_i \leq 1, i = 1, \ldots, k$ *such that* $\sum_{j=1}^{i} b_j \leq \sum_{j=0}^{i-1} a_j, i = 1, \ldots, k$, $\sum_{j=0}^{k} a_i = 1$, *and* $\sum_{j=1}^{k} b_j \leq 1$, *then*

$$\sum_{i=1}^{k} ia_i + \sum_{i=1}^{k}(k+1-i)b_i \leq k.$$

Proof. First we show by induction that

$$\sum_{i=1}^{k}(i-1)(a_{i-1}-b_i) \leq k\sum_{i=1}^{k}(a_{i-1}-b_i) \tag{7.4}$$

The inequality holds trivially for $k = 1$. Suppose it hold for $k - 1$, thus

$$\sum_{i=1}^{k}(i-1)(a_{i-1}-b_i) = \sum_{i=1}^{k-1}(i-1)(a_{i-1}-b_i) + (k-1)(a_{k-1}-b_k)$$

$$\leq (k-1)\sum_{i=1}^{k-1}(a_{i-1}-b_i) + (k-1)(a_{k-1}-b_k) = (k-1)\sum_{i=1}^{k}(a_{i-1}-b_i) \leq k\sum_{i=1}^{k}(a_{i-1}-b_i)$$

where the first inequality follows from the induction hypothesis. Then,

$$\sum_{i=0}^{k} ia_i + \sum_{i=1}^{k}(k+1-i)b_i$$

$$= \sum_{i=1}^{k}(i-1)b_i + \sum_{i=1}^{k}(i-1)(a_{i-1}-b_i) + ka_k + \sum_{i=1}^{n}(k+1-i)b_i$$

$$\leq \sum_{i=1}^{k}(i-1)b_i + k\sum_{i=1}^{k}(a_{i-1}-b_i) + ka_k + \sum_{i=1}^{k}(k+1-i)b_i$$

$$\leq \sum_{i=1}^{k}(i-1)b_i + k(1-\sum_{i=1}^{k}b_i) + \sum_{i=1}^{k}(k+1-i)b_i = k$$

where the inequality follows from (7.4) and the second inequality follows from the assumption $\sum_{j=1}^{n} a_i = 1$. □

Acknowledgements

We are grateful to Prof. Domingos Cardoso, from University of Aveiro, for discussions, and to Prof. Maria Joao Martins, from ISA-CEF University of Lisbon, for support on the statistical analysis of the computational results. This research was partially supported by the Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) through projects UID/MAT/04106/2019 (A. Agra and C. Requejo) and UID/MAT/00297/2019 (J.O. Cerdeira).

References

[1] A. Aazami. *Hardness Results and Approximation Algorithms for Some Problems on Graphs.* Ph.D. thesis, University of Waterloo, 2008.

[2] AIM Minimum Rank – Special Graphs Work Group (F. Barioli, W. Barrett, S. Butler, S. M. Cioabă, D. Cvetković, S. M. Fallat, C. Godsil, W. Haemers, L. Hogben, R. Mikkelson, S. Narayan, O. Pryporova, I. Sciriha, W. So, D. Stevanović, H. van der Holst, K. Vander Meulen, A. Wangsness). *Zero forcing sets and the minimum rank of graphs.* Linear Algebra and its Applications, 428: 1628–1648, 2008.

[3] D. Amos, Y. Caro, R. Davila, R. Pepper. *Upper bounds on the k-forcing number of a graph.* Discrete Appl. Math. 181, 1–10, 2015.

[4] D. Bal, P. Bennett, S. English, C. MacRury, P. Prałat. *Zero Forcing Number of Random Regular Graphs.* arXiv:1812.06477v1, 16 Dec 2018.

[5] E. Balas. *Disjunctive Programming: properties of the convex hull of feasible points.* Discrete Appl. Math., 89, 3–44, 1998.

[6] F. Barioli, W. Barrett, S. Fallat, H. T. Hall, L. Hogben, B. Shader, P. van den Driessche, H. van der Holst. *Parameters related to tree-width, zero forcing, and maximum nullity of a graph.* J. Graph Theory, 72, 146–177, 2013.

[7] F. Barioli, W. Barrett, S. Fallat, H. T. Hall, L. Hogben, B. Shader, P. van den Driessche, H. van der Holst. *Zero forcing parameters and minimum rank problems.* Linear Algebra and its Applications, 433: 401–411, 2010.

[8] B. Brimkov, C. C. Fast, I. V. Hicks. *Computational Approaches for Zero Forcing and Related Problems.* European Journal of Operational Research, 273: 889–903, 2019.

[9] D. Burgarth and V. Giovannetti. *Full control by locally induced relaxation.* Physical Review Letters, 99, 100501, 2007.

[10] D. Burgarth, V. Giovanetti, L. Hogben, S. Severini, M. Young. *Logic circuits from zero forcing.* Natural Computing, 14, 485–490, 2015.

[11] C.J. Edholm, L. Hogben, M. Huynh, J. LaGrange, D.D. Row. *Vertex and edge spread of the zero forcing number, maximum nullity, and minimum rank of a graph.* Linear Algebra Appl. 436, 4352–4372, 2012.

[12] S. Fallat, K. Meagher, and B. Yang. *On the complexity of the positive semidefinite zero forcing number.* Linear Algbera Appl., 491, 101–122, 2016.

[13] FICO. *Xpress Optimization Suite*, July 2015.

[14] M. Gentner, D. Rautenbach. *Some bounds on the zero forcing number of a graph.* Discrete Appl. Math. 236, 203–213, 2018.

[15] L. Hogben, M. Huynh, N. Kingsley, S. Meyer, S. Walker, M. Young. *Propagation time for zero forcing on a graph.* Discrete Appl. Math. 160, 1994–2005, 2012.

[16] T. Kalinowski, N. Kamcev, B. Sudakov. *Zero forcing number of graphs.* SIAM J. of Discrete Math. 33, 95–115, 2019.

[17] G.L. Nemhauser, and L.A. Wolsey. *Integer and Combinatorial Optimization.* John Wiley & Sons, 1988.

[18] R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, 2018 https://www.R-project.org.

[19] D.D. Row. *A technique for computing the zero forcing number of a graph with a cut-vertex.* Linear Algebra Appl. 436, 4423–4432, 2012.