Tema
Tendências em Matemática Aplicada e Computacional, **19**, N. 3 (2018), 471-488
© 2018 Sociedade Brasileira de Matemática Aplicada e Computacional
www.scielo.br/tema
doi: 10.5540/tema.2018.019.03.0471

A Reduced Semidefinite Programming Formulation for HA Assignment Problems in Sport Scheduling

H. J. LARA[1], A.S. SIQUEIRA[2] and J. YUAN[2]

Received on December 19, 2017 / Accepted on May 08, 2018

ABSTRACT. Home-Away Assignment problems are naturally considered as quadratic programming models in binary variables. For solving the problem, different formulations are studied here. First, the problem is rewritten as a quadratic programming formulation with linear constraints, and a quadratically constrained version respectively. For large scale problem, some reduced formulation are proposed by manipulating their special structure, with $1/4$ of the original size. Note that the quadratic programming formulations lead to semidefinite relaxations solved approximately by semidefinite programming method. Comparison between our SDP relaxation and the MIN-RES-CUT based formulation is given. Finally some numerical experiments are given to illustrate the characteristics of each model.

Keywords: Sports scheduling, Integer quadratic programming, Semidefinite programming.

1 INTRODUCTION

Several problems in sport scheduling have been the focus of attention in the Operational Research community, such as the Home-Away assignment problem (HA-Assignment) which assigns the label home (H) or away (A) to each match of a double round robin tournament to satisfy some decision criteria (see [6] or [7]). Some of these criteria involve minimizing the total traveling distance of teams, or minimizing the number of breaks [6, 16]. Models dealing with HA-Assignment problems have been proposed as linear integer programs [6, 7, 16], or as MIN-RES-CUT problems [15], and others. In the last cited article, a Semidefinite Programming relaxation (SDP) for HA-assignment was proposed. With the objective of deducing more efficient SDP models, we study alternative quadratic programming formulations for the problem, which in turn leads to other SDP relaxations with better properties than that those obtained from MIN-RES-CUT. We also provide numerical experiments illustrate the performance of the solver.

*Corresponding author: Hugo José Lara – E-mail: hugol@ucla.edu.ve
[1]Departamento de Engenharia, Universidade Federal de Santa Catarina-Blumenau, Rua João Pessoa, 2750, 89036-256, Blumenau, SC, Brazil. E-mail: hugol@ucla.edu.ve
[2]Departamento de Matemática, Universidade Federal do Parana, Curitiba, Brazil. E-mail: abel.s.siqueira@gmail.com; yuanjy@gmail.com

1.1 Modeling HA-assignments

In this subsection, we introduce the mathematical definition of the HA-Assignment problem, based on [15]. Here, we deal with a double round-robin tournament with a pair number ($2n$) of teams. In a round robin tournament each team plays every other team twice, once at home and the other away. A slot, or round, is a date where every team plays against other team. The number of slots is $2(2n-1)$, each team has its home and each match is held at the home of one of the two playing teams.

A *timetable* is a matrix $\mathscr{T}$ whose rows are indexed by a set of teams $T = \{1,2,\ldots,2n\}$, and columns are indexed by a set of slots (rounds) $S = \{1,2,\ldots,4n-2\}$. Each entry of a timetable, say $\tau(t,s)((t,s) \in T \times S)$, shows the opponent of team t in slot s. A timetable $\mathscr{T}$ should satisfy the following conditions:

- for each team $t \in T$, the t-th row of $\mathscr{T}$ contains each element of $T\backslash\{t\}$ exactly twice;
- for any $(t,s) \in T \times S$, $\tau(\tau(t,s),s) = t$.

The first condition means that each team plays exactly twice with every other team, while the second one establishes that the team playing with $\tau(t,s)$ in slot s should be t. In Figure 1 we show a timetable for $n = 2$ (four teams). Generating timetables has been focus on attention of some works in sport scheduling (see [7]). It is an easy task to randomly generate timetables. If some matches are fixed in advance, the work becomes harder ([14]).

$$\mathscr{T} = \begin{array}{|cccccc|} 2 & 3 & 2 & 4 & 3 & 4 \\ 1 & 4 & 1 & 3 & 4 & 3 \\ 4 & 1 & 4 & 2 & 1 & 2 \\ 3 & 2 & 3 & 1 & 2 & 1 \end{array}$$

Figure 1: A timetable matrix for $n = 2$

A home-away assignment $\mathscr{A}$ is a matrix whose rows are indexed by T, and the columns by S respectively. Each entry of the HA-assignment, $a_{ts}((t,s) \in T \times S)$, is either 'H' (home) or 'A' (away), according to the status of team t at round s.

Given a timetable $\mathscr{T}$, an HA assignment $\mathscr{A} = (a_{ts})((t,s) \in T \times S)$ is said to be *consistent* with $\mathscr{T}$ if it satisfies:

C1 $\forall (t,s) \in T \times S, \{a_{ts}, a_{\tau(t,s)s}\} = \{A,H\}$, and

C2 $\forall t \in T$, if $\tau(t,s) = \tau(t,s')$ and $s \neq s'$ then $\{a_{ts}, a_{ts'}\} = \{A,H\}$.

In Figure 2 we present an HA-assignment, consistent with the timetable shown in Figure 1. A schedule of a round-robin tournament is defined as a pair $(\mathscr{T},\mathscr{A})$ of a timetable and a HA-assignment consistent with the timetable.

$$\mathscr{A} = \begin{array}{|cccccc|} H & H & A & A & A & H \\ A & H & H & H & A & A \\ A & A & H & A & H & H \\ H & A & A & H & H & A \end{array}$$

Figure 2: An HA-assignment matrix consistent with $\mathscr{T}$

Our decision making is as follows: given a fixed timetable $\mathscr{T}$, find a HA-assignment $\mathscr{A}$, consistent with $\mathscr{T}$, according to some criteria. Break minimization and minimizing the total traveling distance, for instance, provides decision criteria based on a quadratic objective function in binary variables. Consequently, we propose an unified framework, grouping these decision criteria.

1.1.1 The total traveling distance minimization problem

We describe the HA-Assignment problem which minimize the total traveling distance. The distance matrix $\mathscr{D} \in \mathscr{R}^{2n\times 2n}$ contain the distances between each pair of venues in the tournament.

Instance: A timetable $\mathscr{T}$ and distance matrix $\mathscr{D}$.

Task: Find an HA-assignment $\mathscr{A}$ consistent with $\mathscr{T}$, minimizing the total traveling distance.

For each $t \in \{1,2,\dots,2n\}$ and $s \in \{0,1,\dots,4n-2\}$, we fix a_{t0} and a_{t4n-1} at H. The traveling distance $l(t,s)$ of team t between rounds s and $s+1$ is defined as:

$$l(t,s) = \begin{cases} d(t,t) = 0 & \text{if } (a_{ts}, a_{ts+1}) = (H,H) \\ d(\tau(t,s),\tau(t,s+1)) & \text{if } (a_{ts}, a_{ts+1}) = (A,A) \\ d(t,\tau(t,s+1)) & \text{if } (a_{ts}, a_{ts+1}) = (H,A) \\ d(\tau(t,s),t) & \text{if } (a_{ts}, a_{ts+1}) = (A,H). \end{cases} \tag{1.1}$$

The total traveling distance for all the teams is given by

$$l(y) = \sum_{t=1}^{2n} \sum_{s=0}^{4n-2} l(t,s).$$

1.1.2 Break minimization/maximization problems

Given an HA-assignment $\mathscr{A}$, it is said that the team $t \in T$ has a break at round $s \in S$ ($s \in S \backslash \{1\}$) if $a_{ts-1} = a_{ts} = A$ or $a_{ts-1} = a_{ts} = H$. The number of breaks $b(\mathscr{A})$ in a HA-assignment is defined as the number of breaks belonging to all teams. In practical sport scheduling, such as in [8], the number of breaks is required to be reduced.

Instance: A timetable $\mathscr{T}$

Task: Find an HA-assignment that is consistent with $\mathscr{T}$ and minimizes/maximizes the number of breaks.

1.2 SDP relaxations for HA-assignment problems

In [15] the HA-assignment problem is formulated as MIN-RES-CUT, and Goemans and Williamson approximation algorithm [9] was applied to the associated SDP relaxation. Our focus here is proposing an alternative scheme to get SDP relaxations, from a quadratic model which explore in more detail the special structure of the graph based problem.

1.3 Organization

The remainder of this article is organized as follows. In the next section we describe the optimization models to deal with our HA-assignment problem. First, we describe the MIN-RES-CUT formulation given in [15]. This combinatorial optimization problem induces a quadratic program in binary (1,-1) variables, which in turn leads to a SDP relaxation. In section 3 we describe an alternative integer quadratic program based on the same graph which conduce to MIN-RES-CUT, but with a simplified modeling strategy, which allows us to produce binary quadratic models with linear constraints, and with reduced size, when compared to MIN-RES-CUT. In section 4 we offer numerical results which compare the solver performance in each quadratic model, and also results comparing the SDP formulations. The last section is devoted to conclusion and final remarks.

2 OPTIMIZATION MODELS

In this section we describe some equivalent optimization models to treat HA-Assignment problems. We first describe the MIN-RES-CUT formulation, and SDP relaxation established in [15]; then our quadratic formulations which leads to alternative SDP relaxations are proposed.

2.1 MIN RES CUT and SDP Relaxation

Suzuka, et.al. proposes in [15] a MIN-RES-CUT formulation for HA-Assignment problems minimizing the total traveling distance, and in [12] for break minimization. Their motivation was to solve the problems via SDP relaxations. We first describe the general form for the MIN-RES-CUT problem and the associated SDP relaxation.

Consider an undirected graph $G = (V,E)$ with vertex set V and edge set E, and nonnegative weight function $w : E \to \mathscr{R}_+$. For any vertex subset $V' \subseteq V$ we define $\delta(V') = \{\{v_i, v_j\} : v_i, v_j \in V; v_i \notin V' \ni v_j\}$. The MIN-RES-CUT problem is defined as follows: Given a graph $G = (V,E)$, a specified vertex $r \in V$, a weight function $w : E \to \mathscr{R}_+$ and a set $E_{cut} \subseteq \{X \subset V : |X| = 2\}$, find a vertex subset V' as solution of the combinatorial optimization problem:

$$\begin{array}{rc} \text{minimize} & \sum_{e\in\delta(V')\cap E} w(e) \\ \text{subject to} & r \notin V' \\ & E_{cut} \subseteq \delta(V') \end{array} \tag{2.1}$$

It is known that MIN-RES-CUT is a NP-hard problem (see for example [11]). The base for an SDP relaxation associated to MIN-RES-CUT is the following binary quadratic programming formulation: For each $v \in V$, let x_v be a binary variable with value $x_v = -1$ if $v \in V'$, or $x_v = 1$ if $v \notin V'$. Then, the first constraint is imposed by $x_r = 1$, and the second by $x_v x_u = -1$ for $\{u,v\} \in E_{cut}$. The objective function add weight w_{uv} if $e = \{u,v\} \in \delta(V') \cap E$; that is if e is an edge and its ends are in opposite sides of the cut defined by V'. By using these binary variables, the objective function becomes $\frac{1}{4}\sum_{u,v\in V} w_{uv}(1 - x_u x_v)$. Finally, the constraint $x_v \in \{-1,1\}$ is imposed by $x_v^2 = 1$. The above considerations provide the binary quadratic model:

$$\begin{array}{lll} \text{minimize} & l(x) = \frac{1}{4}\sum_{u,v\in V} w_{uv}(1 - x_u x_v) & \\ \text{subject to} & x_v^2 = 1 & \forall v \in V \\ & x_u x_v = -1 & \forall \{u,v\} \in E_{cut} \\ & x_r = 1. & \end{array} \tag{2.2}$$

From this quadratic formulation, an SDP relaxation is built as follows: For $m1 = |E|$, let $W = diag(w)$ be the $m1 \times m1$ diagonal matrix with the weights w associated to the edges in the above problem as diagonal elements. For $C = \frac{1}{4}[diag(We - W)]$ and X, a symmetric $m1 \times m1$ matrix, the SDP relaxation is

$$\begin{array}{lll} \text{minimize} & l(X) = \langle C, X \rangle & \\ \text{subject to} & X_{vv} = 1 & \forall v \in V \\ & X_{uv} = -1 & \forall \{u,v\} \in E_{cut} \\ & X \succeq 0. & \end{array} \tag{2.3}$$

If $x \in \{-1,1\}^{m1}$ is a feasible solution for (2.2), then $X = xx^T$ is feasible for (2.3). The last problem is a relaxation because a solution for (2.3) does not necessarily provides a solution for (2.2). However, good approximated solutions for (2.2) can be extracted by Goemans and Williamson's procedure ([9]).

2.1.1 MIN-RES-CUT formulation for HA-assignment problems

The MIN-RES-CUT formulation in [15] for the total traveling distance minimization HA-Assignment problem is described as follows: Given a timetable $\mathscr{T} = (\tau(t,s))((t,s) \in T \times S)$, consider an artificial vertex r and construct the graph $G = (V,E)$ as: $V = \{v_{ts} : (t,s) \in T \times S\} \cup \{r\}$ is the vertex set, $E = \{\{v_{t(s-1)}, v_{ts}\} : t \in T, s \in S \backslash \{1\}\} \cup \{\{r, v_{ts}\} : (t,s) \in T \times S\}$, the edges, and $E_{cut} = \{\{v_{ts}, v_{\tau(t,s)s}\} : (t,s) \in T \times S\} \cup \{\{v_{ts}, v_{ts'}\} : t \in T, s, s' \in S, \tau(t,s) = \tau(t,s'), s \neq s'\}$ the restriction set.

A subset of the vertices $V' \subset V$ is feasible for the MIN-RES-CUT formulation if $r \notin V'$ and $E_{cut} \subset \delta(V')$. Note that feasibility here is associated onl to the partition of the vertex set, that is the edges of the graph only participates at the objective function.

For a feasible solution V' we construct an HA-Assignment $\mathscr{A} = (a_{ts})((t,s) \in T \times S)$ as follows: if $v_{ts} \in V'$ then $a_{ts} = A$, else $a_{ts} = H$. This HA-assignment is consistent with timetable $\mathscr{T}$ because

(C1) each pair of vertices corresponding to a match is in E_{cut}, and (C2) for each team, every pair of vertices corresponding to matches with a common opponent is in E_{cut}.

2.1.2 Minimizing the total traveling distance

Now, we describe the minimization of the total traveling distance: Consider the traveling distance $l(t,s)$, between the venues where matches in rounds s and s+1 are held, as in (1.1), and denote by $t' = \tau(t,s)$ and $t'' = \tau(t,s+1)$ the teams which plays in these rounds. Then

$$\begin{aligned} l(t,s) = d(t',t'')|v_{ts} \cap V'||v_{ts+1} \cap V'| + d(t,t'')(1 - |v_{ts} \cap V'|)|v_{ts+1} \cap V'| \\ + d(t',t)|v_{ts} \cap V'|(1 - |v_{ts+1} \cap V'|) \end{aligned} \tag{2.4}$$

where $|v_{ts} \cap V'| \in \{0,1\}$ according to $v_{ts} \in V'$ or $v_{ts} \notin V'$. The function $l(t,s)$ is quadratic on the binary variables ($\{0,1\}$). These binary variables are associated with the vertices $v_{ts} \in V$ (instead of the edges). The idea in [15] is to model HA-assignment problems as MIN-RES-CUT. To do it, a (linear) weight function on the edges should be defined. In (2.4) there is a quadratic relation on binary variables associated to the vertices, not linear weights on the edges. To fix it, they performed the following linearization: Since $r \notin V'$, we have

$$|v_{ts} \cap V'| = |\{r, v_{ts}\} \cap \delta(V')| \text{ for all } v_{ts} \in V.$$

On the other hand, taking $|\{v_{ts}, v_{ts+1}\} \cap \delta(V')| \in \{0,1\}$ the product in equation (2.4) is modeled as

$$\begin{aligned} |v_{ts} \cap V'||v_{ts+1} \cap V'| &= \tfrac{1}{2}(|\{r, v_{ts}\} \cap \delta(V')| + |\{r, v_{ts+1}\} \cap \delta(V')| \\ &\quad - |\{v_{ts}, v_{ts+1}\} \cap \delta(V')|). \end{aligned} \tag{2.5}$$

Merging (2.5) into (2.4) we obtain a linear function on binary variables centered at the edges, as needed. The weights of the objective function are the coefficients for $|\{r, v_{ts}\} \cap \delta(V')|$, $|\{r, v_{ts+1}\} \cap \delta(V')|$ and $|\{v_{ts}, v_{ts+1}\} \cap \delta(V')|$; and the constraints $r \notin V'$ and $E_{cut} \subset \delta(V')$. We denote by $W1$ this set of weights.

2.1.3 Minimizing the number of breaks

There is a break between round s and $s+1$ if $(a_{ts}, a_{ts+1}) \in \{(H,H),(A,A)\}$. The number of breaks $b(t,s)$ between s and $s+1$ can be modeled by

$$b(t,s) = |v_{ts} \cap V'||v_{ts+1} \cap V'| + (1 - |v_{ts} \cap V'|)(1 - |v_{ts+1} \cap V'|)$$

which share characteristics with (2.5), being a quadratic relation in the same binary variables. This relation leads to another coefficient matrix $W2$ of weights similar to $W1$.

2.1.4 The MIN-RES-CUT SDP relaxation

The combinatorial problem in (2.1) with the specific data provided by the graph $G = (V,E)$, the set E_{cut} and weight matrix $W1$ (or $W2$) specified in this section provides a combinatorial formulation for HA assignment problems. A quadratic formulation, and respective SDP relaxation are also specified from (2.2) and (2.3) for the abovementioned data, leading to the following MRC SDP relaxation:

$$\begin{array}{lrl} \text{Minimize} & l(X) = \langle C, X \rangle & \\ \text{subject to} & X_{vv} = 1 & v \in V1 \\ & X_{vw} = -1 & \{v,w\} \in E1_{cut} \\ & X \succeq 0 & \end{array} \tag{2.6}$$

with $C = \frac{1}{4} diag(We - W)$, which is the formulation given in [15].

3 ALTERNATIVE FORMULATIONS FOR HA ASSIGNMENT PROBLEMS

In this section we construct an alternative SDP relaxation for HA-assignment problems, based on quadratic programming in binary variables, which consider in more details the special structure of the assignment problem. The MIN-RES-CUT formulation described in section above also leads to a quadratic programming problem in binary variables, and consequently to a SDP relaxation. The structure of the constraints in this formulation is a consequence of the graph structure which consider each node independent of the other. Our formulation group the nodes in classes, leading to a simplification of the resulting model. The first tool that we use is described in next subsection:

3.1 Dealing with timetables

In [13] is proposed a procedure to build a partition in a timetable, in order to extract crucial information from it, and simplify the optimization models. We here formalize such procedure. Given a timetable $\mathscr{T}$ and the index set $T \times S$, we construct a partition of the indices according to the following observation: For each $(t,s) \in T \times S$, there exist unique indices $(t',s') \in T \times S$, such that the indices in the set $\{(t,s),(t,s'),(t',s),(t',s')\}$ are related each other, and isolated from the rest of the indices. $t' = \tau(t,s)$ is the team which plays with t in slot s, and s' is the slot where the two teams play again. Thus we can partition the index set $T \times S$ into $K = n(2n-1)$ subsets. Furthermore, we can order the elements in each group by appearance order in the table from left to right and from up to down. The following procedure assigns the label $\gamma_{ts} = [\gamma_{ts}^1, \gamma_s^2]$ to each $(t,s) \in T \times S$:

First let us define the set $\mathscr{K}(k) = \{(t,s) \in T \times S : \gamma_{ts}^1 = k\}$ Clearly $|\mathscr{K}(k)| = 4$, because four labels $\gamma_{ts}^1 = k$ where assigned in each step. The procedure pass though the timetable sequentially, by visiting each of the (t,s) components from left to right, and form up to down. Each component of $T \times S$ is labeled once, placing it into the $\mathscr{K}(k)$ group, for some $k = 1, \ldots, n(2n-1)$. It is clear that for $k_1 \neq k_2$, $\mathscr{K}(k_1) \cap \mathscr{K}(k_2) = \emptyset$, since each component is labeled once. This last observation also explains that $T \times S = \cup_{k=1}^{n(2n-1)} \mathscr{K}(k)$, so we have our partition. In Figure 3 we show labels assigned to each component of the timetable in Figure 1. The partition will be useful to construct the models later.

Algorithm 1: Procedure:

```
1  begin
       Input: A Timetable $\mathscr{T}$
       Output: labels $\gamma_{ts} = [\gamma^1_{ts}, \gamma^2_s]$ to each $(t,s) \in T \times S$:
2      Label $\gamma_{ts} := [0,0]$; $\forall (t,s) \in T \times S$;
3      $k := 1$;
4      for $t = 1$ to $2n$ do
5          for $s = 1$ to $4n-2$ do
6              if $\gamma_{ts} = [0,0]$ then
7                  identify $t' > t$, $s' > s$ such that $\tau(t,s) = t'$, and $\tau(t,s) = \tau(t,s')$;
8                  Label $\gamma_{ts} := [k,1]$; $\gamma_{ts'} := [k,2]$; $\gamma_{t's} := [k,3]$; $\gamma_{t's'} = [k,4]$; $k := k+1$
9              end
10         end
11     end
12 end
```

$$\gamma = \begin{array}{|cccccc|}
\hline
[1,1] & [2,1] & [1,2] & [3,1] & [2,2] & [3,2] \\
[1,3] & [4,1] & [1,4] & [5,1] & [4,2] & [5,2] \\
[6,1] & [2,3] & [6,2] & [5,3] & [2,4] & [5,4] \\
[6,3] & [4,3] & [6,4] & [3,3] & [4,4] & [3,4] \\
\hline
\end{array}$$

Figure 3: Labels assigned to each component on the timetable

3.1.1 Quadratic constraints by using the partition

Let us give a closer look at the constraints in (2.6). By using the partition provided by $\mathscr{K}(k), k = 1, \ldots, n(2n-1)$, we can characterize the restriction set $E1_{cut}$ as follows: We denote by $E1_{cut}(k) = \{\{v_{t_k s_k}, v_{t_k s'_k}\}, \{v_{t_k s_k}, v_{t'_k s_k}\}, \{v_{t_k s'_k}, v_{t'_k s'_k}\}, \{v_{t'_k s_k}, v_{t'_k s'_k}\}\}$, and observe that $E1_{cut} = \cup_{k=1}^{n(2n-1)} E1_{cut}(k)$. So we have an explicit formulation for the second group of constraints in the above model, namely

$$\begin{aligned}
X_{(t_k s_k)(t_k s'_k)} &= -1 \\
X_{(t_k s_k)(t'_k s_k)} &= -1 \\
X_{(t_k s'_k)(t'_k s'_k)} &= -1 \\
X_{(t'_k s_k)(t'_k s'_k)} &= -1,
\end{aligned}$$

for $k = 1, \ldots, n(2n-1)$ (denoting $X_{(ts)(\hat{t}\hat{s})} = X_{v\hat{v}}$ for $v = v_{ts}$ and $\hat{v} = v_{\hat{t}\hat{s}}$). We shall use this explicit characterization for the constraints to build an alternative formulation.

3.2 Alternative quadratic programming formulations for HA-assignment problems

We here propose an alternative quadratic programming formulation which in turn leads to another SDP relaxation. It will be shown that the size of this new formulation, in terms of the number of variables is $1/4$ of the size of (2.2). Numerical advantages will be exhibited too.

Let us consider the graph $G = (V2, E2)$ whose vertex set is $V2 = \{v_{ts} : (t,s) \in T \times S\}$, and edges $E2 = \{\{v_{ts-1}, v_{ts}\} : t \in T, s \in S \backslash \{1\}\}$, and the partition $\mathscr{K}$ defined earlier. We here shall describe the formulation for the total traveling distance minimization problem. For minimization of breaks, the development is analogous. Recall that the traveling distance $l(t,s)$ of team t from slot s to slot $s+1$ is equivalent to (2.4) which is a quadratic function in binary $\{0,1\}$ variables. The MIN-RES-CUT formulation given in [15] replaces $l(t,s)$ by a linear function in binary variables centered at the edges, and then establishes a quadratic $\{1,-1\}$ model to deal with it, through an SDP-relaxation. We here propose to employ the original quadratic objective function (2.4), which is naturally defined in binary $\{0,1\}$ variables, and derive an SDP-relaxation from it. Let us denote by $y_{ts} \in \{0,1\}$ with value 0 if $v_{ts} \notin V'$ and 1 if $v_{ts} \in V'$. The equation (2.4) becomes:

$$\begin{aligned} l(t,s) &= d(t',t'')y_{ts}y_{ts+1} + d(t,t'')(1-y_{ts})y_{ts+1} + d(t',t)y_{ts}(1-y_{ts+1}) \\ &= d(t,t'')y_{ts+1} + d(t',t)y_{ts} + (d(t',t'') - d(t,t'') - d(t',t))y_{ts}y_{ts+1} \end{aligned} \tag{3.1}$$

for $t = 1,\ldots,2n$ and $s = 1,\ldots,4n-3$. Initial and final coefficients are $l(t,0) = d(t,\tau(t,1))y_{t1}$ and $l(t,4n-2) = d(\tau(t,4n-2),t)y_{t4n-2}$ respectively. The entire objective function have the form

$$l(y) = \sum_{t=1}^{2n} \sum_{s=0}^{4n-2} l(t,s) = c^T y + \frac{1}{2} y^T Q y.$$

The quadratic form Q contains nonzero elements only on the entries above and below the diagonal, since only two consecutive variables s and $s+1$ are involved in the quadratic terms. Note that this formulation does not increases with the artificial vertex r.

To define the constraints we shall denote the variables y_{ts} according to the partition $\mathscr{K}$ of the index set. Each entry $(t,s) \in T \times S$ is labeled with $\gamma_{ts} = [k,j]$, $k \in \{1,\ldots,n(2n-1)\}$, $j \in \{1,2,3,4\}$. Let us denote y_{ts} by $y_{t_k s_k}$ if $\gamma_{ts} = [k,1]$; $y_{ts} = y_{t_k s'_k}$ if $\gamma_{ts} = [k,2]$, $y_{ts} = y_{t'_k s_k}$ if $\gamma_{ts} = [k,3]$ and finally $y_{ts} = y_{t'_k s'_k}$ if $\gamma_{ts} = [k,4]$. By using the $\{0,1\}$ to $\{-1.1\}$ transformation, namely $x(y) = 2y-1$ we write the cut constraints as

$$\begin{aligned} (2y_{t_k s_k} - 1)(2y_{t_k s'_k} - 1) &= -1 \\ (2y_{t_k s_k} - 1)(2y_{t'_k s_k} - 1) &= -1 \\ (2y_{t_k s'_k} - 1)(2y_{t'_k s'_k} - 1) &= -1 \\ (2y_{t'_k s_k} - 1)(2y_{t'_k s'_k} - 1) &= -1, \end{aligned}$$

for $k = 1,\ldots,n(2n-1)$, which provide the quadratic constraints

$$\begin{aligned} y_{t_k s_k} + y_{t_k s'_k} - 2y_{t_k s_k} y_{t_k s'_k} &= 1 \\ y_{t_k s_k} + y_{t'_k s_k} - 2y_{t_k s_k} y_{t'_k s_k} &= 1 \\ y_{t_k s'_k} + y_{t'_k s'_k} - 2y_{t_k s'_k} y_{t'_k s'_k} &= 1 \\ y_{t'_k s_k} + y_{t'_k s'_k} - 2y_{t'_k s_k} y_{t'_k s'_k} &= 1 \end{aligned}$$

for $k = 1,\dots,n(2n-1)$. The result is a group of $4n(2n-1)$ quadratic constraints on y, which we denote for each k and j by

$$1 + c_{kj}^T y + \frac{1}{2} y^T H_{kj} y = 0, \text{ for } j = 1,2,3,4.$$

Now we cast the problem as

$$\begin{array}{lll} \text{minimize} & l(y) = a + c^T y + \frac{1}{2} y^T Q y & \\ \text{subject to} & 1 + c_{kj}^T y + \frac{1}{2} y^T H_{kj} y = 0, & \begin{array}{c} k = 1,\dots,n(2n-1); \\ j = 1,2,3,4. \end{array} \end{array}$$

The matrices H_{kj} are highly sparse, because they involve only two consecutive variables, and Q is a tridiagonal matrix with zero diagonal. By using the relation $a + c^T y + \frac{1}{2} y^T Q y = \langle \begin{pmatrix} a & \frac{1}{2}c^T \\ \frac{1}{2}c & \frac{1}{2}Q \end{pmatrix}, \begin{pmatrix} 1 & y^T \\ y & yy^T \end{pmatrix} \rangle$, and similarly for the constraints, we write equivalently the quadratic program as

$$\begin{array}{llll} \text{minimize} & l(y) = \langle \begin{pmatrix} a & \frac{1}{2}c^T \\ \frac{1}{2}c & \frac{1}{2}Q \end{pmatrix}, \begin{pmatrix} 1 & y^T \\ y & yy^T \end{pmatrix} \rangle & & \\ \text{subject to} & \langle \begin{pmatrix} 1 & \frac{1}{2}c_{kj}^T \\ \frac{1}{2}c_{kj} & \frac{1}{2}H_{kj} \end{pmatrix}, \begin{pmatrix} 1 & y^T \\ y & yy^T \end{pmatrix} \rangle = 0, & \begin{array}{c} k = 1,\dots,n(2n-1); \\ j = 1,2,3,4. \end{array} & (3.2) \\ & y \in \{0,1\}^{2n(4n-2)}. & & \end{array}$$

To build a SDP relaxation, we replace the rank one matrix variable $\begin{pmatrix} 1 & y^T \\ y & yy^T \end{pmatrix}$ by $Y \in \mathscr{R}^{2n(2n-1)\times 2n(2n-1)}$, satisfying $Y \succeq 0$, and $Y_{(ts)(ts)} - Y_{(11)(ts)} = 0$ for all $(t,s) \in T \times S$.

The quadratic programming formulation provided by the MIN-RES-CUT model in [15] associates to each vertice of the underlying graph a binary $\{1,-1\}$ variable; and the objective function is defined as the sum of the linear weights on the edges (see (2.1)). In order to offer these linear weights, a linearization given by (2.5) was performed. Our proposal recognizes the original quadratic structure of the objective function (prior to be linearized) to define directly an equivalent quadratic model suitable to be relaxed through the following SDP model:

$$\begin{array}{llll} \text{minimize} & l(y) = \langle \begin{pmatrix} a & \frac{1}{2}c^T \\ \frac{1}{2}c & \frac{1}{2}Q \end{pmatrix}, Y \rangle & & \\ \text{subject to} & \langle \begin{pmatrix} 1 & \frac{1}{2}c_{kj}^T \\ \frac{1}{2}c_{kj} & \frac{1}{2}H_{kj} \end{pmatrix}, Y \rangle = 0, & \begin{array}{c} k = 1,\dots,n(2n-1); \\ j = 1,2,3,4. \end{array} & (3.3) \\ & Y_{(ts)(ts)} - Y_{(11)(ts)} = 0, & \forall (t,s) \in T \times S & \\ & Y \succeq 0. & & \end{array}$$

3.2.1 A linearly constrained quadratic model

We can use linear constraints to model the $C1-C2$ conditions:

$$\begin{aligned} y_{t_k s_k} + y_{t'_k s_k} &= 1 \\ y_{t_k s_k} + y_{t_k s'_k} &= 1 \\ -y_{t_k s_k} + y_{t'_k s_k} &= 0 \end{aligned} \tag{3.4}$$

At each group k, the first equation means that only one of the teams plays at home in slot s_k. The second equation establishes that team t_k should play alternatively at home and away in slots s_k and s'_k; and the third one ensures that if the first team plays at home (away) at the first slot, then the second team should be home (away) in the second match between them. Putting all the constraints together for $k = 1,\dots,n(2n-1)$, we can write the above constraints as a linear system of equations, say $Ay = b$. Our quadratic binary program with linear constraints becomes

$$\begin{array}{ll} \text{Minimize} & l(y) = c^T y + \frac{1}{2} y^T Q y \\ \text{subject to} & Ay = b \\ & y \in \{0,1\}^{4n(2n-1)} \end{array} \tag{3.5}$$

A is an $3n(2n-1) \times 4n(2n-1)$ matrix with entries in $\{0,1\}$, while H is tridiagonal (with zero diagonal) $4n(2n-1) \times 4n(2n-1)$. We can directly construct an SDP relaxation from (3.5), but the special structure presented by these linear constraints allows us another simplification: By equations (3.4) we have that

$$\begin{bmatrix} y_{t'_k s_k} \\ y_{t_k s'_k} \\ y_{t'_k s'_k} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} y_{t_k s_k} = b_k - A_{t_k s_k} y_{t_k s_k}.$$

Denoting by $z_k = y_{t_k s_k}$ we write all the variables y in function of $z \in \{0,1\}^{n(2n-1)}$. For each k, consider the subsets $B_k = \{(t_k, s'_k), (t'_k, s_k), (t'_k, s'_k)\}$ and $N_k = \{(t_k, s_k)\}$ of the indices $T \times S$; and construct $B = \cup_{k=1}^{n(2n-1)} B_k$ and $N = \cup_{k=1}^{n(2n-1)} N_k$. This leads us to $y_B = b - A_N z$, and $y_N = z$ which in turn defines the linear transformation $\mathscr{Y}$ by $\mathscr{Y}(z) = y$. The objective function $l(y)$ becomes

$$\bar{l}(z) = l(\mathscr{Y}(z)) = \bar{a} + \bar{c}^T z + z^T \bar{Q} z,$$

where $\bar{a} = a + c_B^T b + b^T H_{BB} b$; $\bar{c} = (c - A_N^T c_B - 2A_N^T Q_{BB} b - 2Q_{NB} b)^T$ and $\bar{H} = [A_N^T Q_{BB} A_N + A_N^T Q_{BN} + Q_{NB} A_N]$. Our equivalent reduced model is:

$$\begin{array}{ll} \text{Minimize} & \bar{l}(z) = \bar{a} + \bar{c}^T z + \frac{1}{2} z^T \bar{Q} z \\ \text{subject to} & z \in \{0,1\}^{n(2n-1)}, \end{array} \tag{3.6}$$

which is a 0-1 quadratic programming with $\frac{1}{4}$ of the number of variables, compared with problem (3.5). This formulation only has the binary variables constraint, making it suitable for using some metaheuristics, like genetic algorithms, without the care of generating feasible solution populations. The resulting SDP relaxation is

$$\begin{array}{rl} \text{minimize} & l(Z) = \langle \bar{C}, Z \rangle \\ \text{subject to} & diag(Z) = e \\ & Z \succeq 0 \end{array} \tag{3.7}$$

where $\bar{C} = \begin{pmatrix} \bar{a} & \frac{1}{2}\bar{c}^T \\ \frac{1}{2}\bar{c} & \frac{1}{2}\bar{Q} \end{pmatrix}$ which we call reduced SDP.

3.2.2 Comparison between the SDP formulations

Notice that the combinatorial problem (2.1) is centered at the edges of the underlying graph, in the sense that the weights in the objective function act over the edges in the feasible cut, and the constraints determine where the cut edges are chosen. When specialized to HA-assignment problems, the original formulation of the objective function, based on (1.1), in the case of total traveling distance minimization, shows a quadratic relation in binary variables defined on the vertices (not on the edges). The authors in [15] adapt the model by adding an artificial vertex r, and $2n(4n-2)$ additional edges (linking r to every other vertex). The relation (3.1) provides a linearization of the objective function. Now the variables are centered at the edges, instead of the vertices, and a linear relation is obtained, as required by the MIN-RES-CUT format.

On the other hand, when the quadratic program (2.2) is formulated, binary variables (centered at the vertices) model the combinatorial problem (2.1). Simple quadratic constraints are natural to model the cut constraints, and the objective function becomes a quadratic function on these binary variables. This quadratic function is then linearized, with variables centered at the edges, to be in the sequel modeled as a quadratic program in binary variables. The number of variables is equal to the number of vertices, namely $2n(4n-2)+1$.

Also, our quadratic model (2.2) is meant to keep the original binary variables centered at the vertices, avoiding the use of any artificial vertex, and the linearization of any variable. The result is a quadratic program with $2n(4n-2)$ variables and $4n(4n-2)$ constraints, the same size that (2.2). The associated SDP relaxation (2.6) is also the same size. Now, the reduced problem (3.6) is an equivalent quadratic program with $n(2n-1)$ binary variables. The constraints were reduced to the binary condition. Such reduction on the number of variables achieve $\frac{1}{4}$ of the original size (given by the size of (2.2). This reduction leads to improving the performance of the solvers, when dealing with the respective relaxations. Even though the reduction on the problem size seems to be very good, the obtained objective function suffer an undesirable transformation: In problem (2.2), the matrix of coefficients associated to the quadratic constraints is highly sparse (actually, it only has nonzero components above and below of the diagonal entries), because quadratic terms only occur in consecutive slots. When the transformation $\mathscr{Y}$ is performed, the quadratic term's matrix lose it sparsity condition, and thus, the objective function becomes more difficult to be optimally solved.

4 NUMERICAL RESULTS

In this section we offer some computational experiments to compare the different studied models. At first, we deal with integer quadratic models, comparing the full sized and reduced versions, in a preliminary experiment, in part published at [10]. A more robust experiment is reported at the second part, comparing SDP relaxations for the HA-Assignment problem.

4.1 Integer quadratic programming models

Table 1: Objective function values for quadratic and linear models with linear constraints, with full and reduced size, vs quadratically constrained quadratic models.

n	Fixing FO			Fixing timetable		
	QLC	QQC	Deviation	QLC	QQC	Deviation
2	835.5	957.7	0.12760	840.2	886,4	0.05212
3	2148.5	2395	0.10292	1845.3	2231	0.17288
4	3244.9	3727.9	0.12956	3332.5	3925,8	0.15113
5	5479.9	6205.8	0.11697	5364.0	6110,8	0.12221
6	7334.4	8730.1	0.15987	7672.3	9263,6	0.17178
7	10380.1	12089.2	0.14137	10757.9	12433,7	0.13478
8	13779.5	15912.5	0.13405	13799.5	15893,6	0.13176
9	16792.9	19349.2	0.13211	17531.8	20705,9	0.15329
10	21204.6	24875.6	0.14757	21869.0	25935,2	0.15678

Three different integer programming formulations for the same problem are solved with simulated data: A quadratic program with linear constraints QLC (3.5), a reduced quadratic program QR (3.6), and a quadratically constrained quadratic program QQC (3.2). All computations were performed on a PC Intel(R) core(TM) i7-3632 QM, 2.20 GHZ, 64 bits.

In the first experiment, we fix the objective function and solve instances with 10 different randomly generated timetables, for each problem size n, with the objective of exploring the performance of the solver in a variety of configurations (timetables). Even sharing the same objective function for each n, we solve 10 distinct instances, because different timetables lead to different constraints. For the second experiment we fix a timetable, and then we solve the problem for 10 different objective functions data, for each problem size (single configuration). We solve HA-assignment problems for a even number of teams, between 4 and 20. Since our objective is to compare the formulations for the problem, we use a non commercial solver which deal with both, linear and quadratic integer programs, namely SCIP [3].

In Table 1 we present the objective function values for both experiments. The second and fifth column show values for the quadratic model with linear constraints (QLC), while the third and

Table 2: Runtime for the quadratic models, with linear constraints, fixing the timetable.

n	QLC fixing timetable			
	full		reduced	
	mean	SD	mean	SD
2	0.02768	0.01198	0.02247	0.01144
3	0.09818	0.01719	0.10523	0.03064
4	0.29231	0.07935	0.32632	0.12838
5	1.14851	0.29057	1.17092	0.30648
6	6.92692	0.74739	6.68803	1.08223
7	10.4368	0.99115	9.95315	0.89028
8	15.1239	1.65025	13.8831	1.45535
9	20.6103	1.86102	21.6154	7.18129
10	50.1043	21.6128	60.6731	22.7746

Table 3: Runtime for the quadratic models with linear constraints, fixing the objective function.

n	QLC fixing OF			
	full		reduced	
	mean	SD	mean	SD
2	0.01881	0.00349	0.02049	0.00817
3	0.10556	0.03898	0.09833	0.02711
4	0.22068	0.06098	0.24719	0.07874
5	1.02054	0.30153	1.15223	0.54655
6	6.27446	1.73542	6.07395	1.71097
7	10.9502	1.57499	9.91357	0.59958
8	14.5375	1.38106	13.5211	1.30986
9	21.3364	1.86914	21.9286	6.80267
10	60.8347	19.6424	41.4042	22.7594

sixth columns are for the model with quadratic constraints (QQC). The remaining columns contain relative deviations of the values. The difference between linear constrained and quadratic constrained models is possibly because the solver only guarantee locally optimal solutions in problems with quadratic constraints.

In Table 2 we present mean and standard deviation for the runtime, in the instances where the objective function is fixed for both cases, full and reduced size quadratic models (3.5) and (3.6).

Table 4: Runtime for the quadratically constrained model (QQC), fixing the objective function and the timetable.

n	Fixing OF		Fixing Timetable	
	mean	sd	mean	sd
2	0.02107	0.00155	0.02079	0.00139
3	0.03974	0.00286	0.03858	0.00298
4	0.06538	0.00474	0.06054	0.00273
5	0.10066	0.00444	0.10122	0.00608
6	0.15408	0.00958	0.14684	0.00681
7	0.22715	0.00974	0.21647	0.00427
8	0.30999	0.00893	0.30643	0.00908
9	0.41958	0.01458	0.42272	0.01283
10	0.56785	0.00973	0.57336	0.00966

It was expected that the running times for the reduced version in the quadratic models to be consistently smaller, but this does not occur.

Now we shall study Table 3, which provides runtime for our integer quadratic and linear programs, with full and reduced size configurations, but fixing the structure (fixing a timetable). In each case we solve for 10 different objective function's data, for each problem size. There is no clear evidence of advantage in choosing the reduced version. Sometimes is faster and in other times slower.

The Table 4 deal with runtime for both experiments, but for the quadratically constrained model (QQC). When compared with Table 2 and Table 3, we observe a huge difference in the performance. The price we pay is that global optimization is not guaranteed.

4.2 SDP models

In this subsection we report more robust computational experiments that show the differences between the MIN-RES-CUT formulation in ([15]), and our reduced SDP version. Ten instances for each problem size of $n = 8$ to 20 (that is, tournaments with 16 to 40 teams) where generated. We construct a timetable of "double" round robin by concatenating two different "single" randomly generated round round robin table. Distance matrices where also randomly generated. All the problems where optimally solved as linear integer programs, as in ([15]), and solutions for SDP relaxations where also reported. All computations where implemented in the Julia language [4], using the JuMP [5] mathematical modeling package, with the solvers CPLEX [2] for Linear Integer Programming, and MOSEK [1] for SDP. Examining the running times for MRC and the reduced SDP, the advantage of the reduced version is overwhelming.

Table 5: Running times for S.

n	Linear		SDP MRC		Reduced SDP	
	$\overline{f}$	$\bar{t}$	$\overline{E}_f$	$\bar{t}$	$\overline{E}_f$	$\bar{t}$
2	7.3300e+02	0.02	-1.2258e-01	1.10	0.0000e+00	0.44
3	1.7891e+03	0.05	-2.8128e-01	3.95	0.0000e+00	0.27
4	3.2777e+03	0.15	-3.0212e-02	13.81	8.7450e-03	0.64
5	5.1840e+03	0.29	7.8695e-03	38.74	9.7976e-03	1.47
6	7.7355e+03	0.48	1.1984e-02	113.39	1.6149e-02	3.13
7	1.0212e+04	0.87	3.3375e-02	253.88	3.2369e-02	5.33
8	1.3432e+04	3.16	3.5014e-02	516.90	5.3592e-02	9.33
9	1.6894e+04	5.11	2.8511e-02	1070.48	4.4611e-02	16.47
10	2.0920e+04	7.46	8.2483e-02	2065.31	8.0239e-02	29.03
11	2.5079e+04	13.24	6.5871e-02	3600.69	6.0877e-02	49.06
12	2.8963e+04	17.38	8.8370e-02	5487.36	7.5394e-02	75.73
13	3.4313e+04	25.94	9.5084e-02	8925.42	9.8324e-02	114.65
14	4.0464e+04	91.94	1.3480e-01	13786.32	1.3321e-01	180.81
15	4.5725e+04	97.57	1.1949e-01	21204.91	1.1993e-01	229.93
16	5.3133e+04	172.20	1.3060e-01	34686.28	1.2608e-01	363.37
17	6.0957e+04	674.33	1.1399e-01	46958.26	1.1160e-01	731.70
18	6.6762e+04	1013.89	1.3102e-01	67643.59	1.2555e-01	819.57
19	7.5247e+04	1666.63	1.3078e-01	104376.27	1.3800e-01	1039.97
20	8.0043e+04	1104.60	1.4696e-01	123204.42	1.4079e-01	1459.20

5 CONCLUSIONS

In this article we study integer quadratic programming formulations for HA-Assignment problems which appear in sport scheduling, and their associated semidefinite positive (SDP) relaxations. We start from the combinatorial optimization model (2.1) given by the MIN-RES-CUT model, and previously studied in ([15]), obtaining a quadratic programming formulation based on (2.2) and it respective SDP relaxation (2.3); with de data associated to HA-assignment problems.

By exploring the special structure of the HA-assignment problem, through the modeling tools shown in the subsection 3.1, we write the integer quadratic model (3.2) which lead to the SDP relaxation (3.3), sharing the same problem size that MRC.

The first advantage of our modeling stand on the linear constraints (3.4) which in turn leads to a reduction on the problem size to $1/4$ of the original size (3.6). Despite the reduction, improvement is not accomplished in practice for the quadratic models, as shown in Tables 2 and 3, in which the reduced problems are not solved in a faster way. An explanation for this phenomena is that when the transformation $\mathscr{Y}$ is applied in (3.6), the matrix associated to the quadratic objective function lose the sparsity condition.

The quadratic model with quadratic constraints given by (3.2) behaves much better than the linear constrained one (3.5), as exhibited by Table 4. Nevertheless, the solutions are not globally optimal (see Table 1). This observation suggests that quadratic relations provide better characteristics to the solver´s performance than the linear counterparts

The main contribution of this paper are the SDP relaxations given by (3.3) and the reduced version (3.7), which is compared to the MRC relaxation introduced in [15]. In the Table 5 are shown results for instances from 4 to 40 teams. The quality of these solutions, measured by the relative errors are similar for both of the models, but for our reduced SDP relaxation the running times are much better.

ACKNOWLEDGMENTS

This work is supported in part by the Brazilian Government, through the Excellence Fellowship Program CAPES/IMPA of the first author while visiting the department of Mathematics at UFPR, Curitiba, Brazil.

RESUMO. Os problemas de alocação local visitante são considerados de forma natural como modelos de programação quadrática em variáveis binárias. Para resolver ditos problemas, estudamos aqui diferentes formulações. Primeiro, o problema é reformulado como um programa quadrático com restrições lineares, e restrições quadráticas, respectivamente. Para problemas de grande porte, propomos uma formulação de tamanho reduzido, obtida manipulando sua estrutura especial a 1/4 do problema original. Notemos que a formulação de programação quadrática nos leva a uma relaxação semidefinida, que é resolvida de forma aproximada por métodos de programação semidefinida. Comparamos nossa formulação reduzida de programação semidefinida, com a conhecida formulação MIN-RES-CUT. Finalmente, oferecemos experimentação numérica para ilustrar as características de cada modelo.

Palavras-chave: Calendários esportivos, programação quadrática inteira, programação semidefinida.

REFERENCES

[1] "Introducing the MOSEK Optimization Suite 8.1.0.47" (2018 (accessed February 10, 2018)). URL `https://www.mosek.com/documentation/`.

[2] "CPLEX Optimization" (2018 (accessed February 3, 2018)). URL `https://www.ibm.com/analytics/data-science/prescriptive-analytics/cplex-optimizer`.

[3] T. Achterberg. "Solving constraints integer programs", volume 1 (1) (2009), pp. 1–41.

[4] J. Bezanson, A. Edelman, S. Karpinski & S. Viral B. Julia: A Fresh Approach to Numerical Computing. *SIAM Review*, **59** (2017), 65–98. doi:10.1137/141000671. URL `http://julialang.org/publications/julia-fresh-approach-BEKS.pdf`.

[5] I. Dunning, J. Huchette & M. Lubin. JuMP: A Modeling Language for Mathematical Optimization. *SIAM Review*, **59**(2) (2017), 295–320. doi:10.1137/15M1020575.

[6] K. Easton, G. Nemhauser & M. Trick. The traveling tournament problem: description and benchmarks. In T. Walsh (editor), "Principles and Practice of Constraint Programming, Volume 2239 of Lecture Notes in Computer Science". Springer, Berlin (2001).

[7] K. Easton, G. Nemhauser & M. Trick. Solving the travelling tournament problem: a combined integer programming and constraint programming approach. In J.Y.T. Leung & J.H. Anderson (editors), "In Handbook of Scheduling: Algorithms, Models and Performance Analysis". Chapman and Hall (2004).

[8] M. Elf, M. Junger & G. Rinaldi. Minimizing breaks by maximizing cuts. *Operations Research Letters*, **31** (2003), 343–349.

[9] M.X. Goemans & D.P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, **41** (1995), 1115–1145.

[10] H. Lara Urdaneta, J. Yuan & A. Siqueira. Alternative linear and quadratic programming formulations for HA-Assignment problems. In "Proceeding Series of the Brazilian Society of Applied and Computational Mathematics". SBMAC, São Jose dos Campos, SP. (2018), pp. 1–8.

[11] M. Laurent & F. Rendl. Semidefinite Programming and Integer Programming. Austria, 2002., KUniversität Klagenfurt, Institut für Mathematik (2002).

[12] R. Miyashiro & T. Matsui. Semidefinite programming based approaches to the break minimization problem. *Computers and Operations Research*, **33** (2006), 1975 – 1982.

[13] J. Perdomo & H. Lara. A combinatorial optimization formulation for the HA-Assignment problem. *Publicaciones en Ciencias y Tecnología*, **7**(2) (2013), 127–141.

[14] C. Ribeiro. Sport Scheduling: a tutorial on fundamental problems and applications. Technical report, Department of Computer Science, Universidade Federal Fluminense, Brazil (2010). URL `http://www.ic.uff.br/~celso/artigas/sport-scheduling`.

[15] A. Suzuka, R. Miyashiro, A. Yoshise & T. Matsui. Semidefinite Programming Based Approaches to Home-Away Assignment Problems in Sports Scheduling. Mathematical engineering technical reports, Department of Mathematical Informatics, The University of Tokyo. Japan. (2005). URL `http://www.i.u-tokyo.ac.jp/mi/mi-e.htm`.

[16] M. Trick. A Schedule-then-Break Approach to Sport Timetabling. Technical report, GSIA, Carnegie Mellon, Pittsburgh (2006). URL `http://mat.gsia.cmu.edu`.