Proceeding Series of the Brazilian Society of Applied and Computational Mathematics, v. 6, n. 1, 2018.

Trabalho apresentado no XXXVII CNMAC, S.J. dos Campos - SP, 2017.

**Proceeding Series of the Brazilian Society of Computational and Applied Mathematics**

# Alternative Integer linear and Quadratic Programming Formulations for HA-Assignment Problems

Hugo Lara Urdaneta[1]
Campus de Blumenau, UFSC, Blumenau, SC
Jinyun Yuan[2]
Departamento de Matemática, UFPR, Curitiba, PR
Abel Soares Siqueira[3]
Departamento de Matemática, UFPR, Curitiba, PR

**Abstract**. Home-Away Assignment problems are naturally cast as quadratic programming models in binary variables. In this work we compare alternative formulations. First, we propose another formulation by manipulating their special structure to obtain versions with 1/4 of the original size. By linearizing the quadratic objective function, we get two more alternative models to be compared with the quadratic ones. Numerical experiments exhibit the characteristics of each model.

**Palavras-chave**.Sport scheduling, HA-assignment, Integer linear programming, Integer quadratic programming.

## 1 Introduction

Several problems in sport scheduling have been focus of attention in the Operational Research community. One of them, the Home-Away assignment problem (HA-Assignment) assign the label home (H) or away (A) to each match of a double round robin tournament, allowing that some decision criteria is achieved (see [2], or [3]). Some of these criteria involves minimizing the total traveling distance of the teams, or minimizing the number of breaks [2, 9]. Models dealing with HA-Assignment problems have been proposed, most of them as linear integer programs [2, 3, 9], or as MIN-RES-CUT problems [6, 8], among others. We propose integer linear and quadratic formulations based on cuts over a graph, but in a different flavor if compared with MIN-RES-CUT in [8].

In the sequel we introduce, like in [8] the mathematical definition of the problem. Through this paper, 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 all the teams play. The number of slots is $2(2n-1)$, and each team has its home and each match is held at the home

[1]hugo.lara.urdaneta@ufsc.br
[2]yuanjy@gmail.com
[3]abel.s.siqueira@gmail.com

 

of one of the two playing teams. A *timetable* is a matrix $\mathcal{T}$ whose rows are indexed by a set of teams $T = \{1, 2, \ldots, 2n\}$, and columns 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 $\mathcal{T}$ should satisfy the following conditions: for each team $t \in T$, the $t$-th row of $\mathcal{T}$ contains each element of $T \backslash \{t\}$ exactly twice; and for any $(t,s) \in T \times S$, $\tau(\tau(t,s),s) = t$. Generating timetables has been focus of attention of some works in sport scheduling (see [3,5,7]). It is an easy task randomly generate timetables. If some matches are fixed in advance, the work becomes harder ( [7]).

A home-away assignment is a matrix whose rows are indexed by $T$, and the columns by $S$. Each entry of the HA-assignment, say $a_{ts}((t,s) \in T \times S)$, is either 'H' (home) or 'A' (away), according to the status of team $t$ at round $s$. Given a timetable $\mathcal{T}$, an HA assignment $\mathcal{A} = (a_{ts})((t,s) \in T \times S)$ is said to be *consistent* with $\mathcal{T}$ if it satisfy **(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 $\{a_{ts}, a_{ts'}\} = \{A, H\}$. A schedule of a round-robin tournament is defined as a pair $(\mathcal{T}, \mathcal{A})$ of a timetable and a HA-assignment consistent with the timetable.

Our decision making is as follows: given a fixed timetable $\mathcal{T}$, find a HA-assignment $\mathcal{A}$. consistent with $\mathcal{T}$, according to some criteria. In our case, based on a quadratic objective function in binary variables.

Given a timetable $\mathcal{T}$ 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 plays again. Thus we can partition the index set $T \times S$ into $K = n(2n-1)$ subsets. A procedure can be defined to assign to each index $(t,s) \in T \times S$ labels $\gamma_{ts} = [k,j]$ with $k = 1, \ldots, n(2n-1)$ and $j = 1,2,3,4$, in a such way that $\gamma_{t_k s_k} = [k,1], \gamma_{t_k s'_k} = [k,2], \gamma_{t'_k s_k} = [k,3]$ and $\gamma_{t'_k s'_k} = [k,4]$. Let us define the set $\mathcal{K}(k) = \{(t,s) \in T \times S : \gamma^1_{ts} = k\}$ Clearly $|\mathcal{K}(k)| = 4$, It is clear that for $k_1 \neq k_2$, $\mathcal{K}(k_1) \cap \mathcal{K}(k_2) = \emptyset$, since each component is labeled once. This last observation also explains that $T \times S = \cup_{k=1}^{n(2n-1)} \mathcal{K}(k)$, so we have our partition.

The remainder of this article is organized as follows. In the next section we describe the integer quadratic programming model; then we propose the procedure to reduce the problem size, and later, the linear versions are written. In section 3 we offer numerical results which compare the solver performance in each model. The last section is devoted to conclusion and final remarks.

## 2 Optimization models

In this part we model the consistency C1-C2 constraints of the HA-assignment problem as linear equations in binary variables. let us consider the undirected graph $G = (V, E)$ where the index set is $V = \{v_{ts} : (t,s) \in T \times S\}$, and the edges $E = \{(v_{ts}, v_{ts'}), (v_{ts}, v_{\tau(t)s}),$ $(v_{ts'}, v_{\tau(t)s'}), (v_{\tau(t)s}, v_{\tau(t)s'}) : (t,s) \in T \times S\}$. Consider also the partition $\{\mathcal{K}(k)\}_{k=1}^{n(2n-1)}$ defined in the section above. Associated to each entry in $\mathcal{K}(k) = \{(t_k, s_k), (t_k, s'_k), (t'_k, s_k), (t'_k, s'_k)\}$ we denote the binary $0-1$ variables $y_{t_k s_k}, y_{t_k s'_k}, y_{t'_k s_k}$, and $y_{t'_k s'_k}$. Each variable $y_{ts}$ represents an entry on the timetable, and it value will construct the HA-assignment $\mathcal{A}$ in such way that if $y_{ts} = 0$, then $a_{ts} = H$, and the case $y_{ts} = 1$ imply $a_{ts} = A$. The following linear constraints in binary variables model the conditions C1-C2:

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

For $k = 1, \ldots, n(2n-1)$, at each group $k$ the first equation means that only one of the teams plays at home in slot $s_k$. The second one 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. Now, for $k = 1, 2, \ldots, n(2n-1)$, the equations (1) provide a system of equations of the form $Ay = b$ where

$$A = \begin{bmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_{n(2n-1)} \end{bmatrix}, \; b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n(2n-1)} \end{bmatrix}, \; A_k = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix} \text{ and } b_k = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}.$$

Note that the imposed order in the "block diagonal" components of $A$, determined by the $k = 1, \ldots, n(2n-1)$ groups is different from the natural order provided by the set $T \times S$.

Some of the popular decision criteria that have been used to choose HA-assignments, like minimizing the total traveling distance (see [2,3]), or minimizing the number of breaks [4,6], leads to quadratic functions on $y$ to be minimized. Denote by $l(y) = c^T y + \frac{1}{2} y^T H y$ our objective function, where $c$ and $H$ are chosen according to the specific chosen criteria. Our integer program with linear constraints and quadratic objective function becomes

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

This formulation shares the same problem size with the one in [8], namely $O(4n(2n-1))$.

By noting that all the variables in the group $\mathcal{K}(k)$ are determined by fixing one of such variables, we can build a reduced model with $\frac{1}{4}$ of the original size. In fact, by equations (1) we have

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

Denoting by $z_k = y_{t_k s_k}$ we can write all the variables $y$ in function of $z \in \{0,1\}^{n(2n-1)}$. For each $k$ ($k = 1, \ldots, n(2n-1)$), we define the following subsets of the index set $T \times S$: $B(k) = \{(t_k, s'_k), (t'_k, s_k), (t'_k, s'_k)\}$ and $N(k) = \{(t_k, s_k)\}$.

**Proposição 2.1.** *Consider the sets $B = \cup_{k=1}^{n(2n-1)} B(k)$ and $N = \cup_{k=1}^{n(2n-1)} N(k)$. Then $\{B, N\}$ is a partition of the index set $T \times S$.*

*Proof.* : In fact,

$$\begin{aligned} B \cup N &= (\cup_{k=1}^{n(2n-1)} B(k)) \cup (\cup_{k=1}^{n(2n-1)} N(k)) \\ &= \cup_{k=1}^{n(2n-1)} \mathcal{K}(k) = T \times S. \end{aligned}$$

 

On the other hand, the fact that for $k \neq k'$, $\mathcal{K}(k) \cap \mathcal{K}(k) = \emptyset$, leads to $B(k) \cap N(k') = \emptyset$ for all $k \neq k'$. So,

$$\begin{aligned} B \cap N &= (\cup_{k=1}^{n(2n-1)} B(k)) \cap (\cup_{k=1}^{n(2n-1)} N(k)) \\ &= \cup_{k=1}^{n(2n-1)} (B(k) \cap N(k)) \end{aligned}$$

for any $k$, $B(k) \cap N(k) = \emptyset$ by definition, leading $B \cap N = \emptyset$. □

By using the partition above, our variables can be rewritten as follows: $y_B = b - A_N z$, and $y_N = z$, which in turn define the linear transformation $\mathcal{Y} : \mathcal{R}^{n(2n-1)} \to \mathcal{R}^{4n(2n-1)}$, by $\mathcal{Y}(z) = y$ . The objective function $l(y)$ becomes

$$\bar{l}(z) = l(\mathcal{Y}(z)) = \bar{a} + \bar{c}^T z + \frac{1}{2} z^T \bar{H} z,$$

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

$$\begin{array}{ll} \text{Minimize} & \bar{l}(z) = \bar{a} + \bar{c}^T z + \frac{1}{2} z^T \bar{H} z \\ \text{subject to} & z \in \{0,1\}^{n(2n-1)}, \end{array} \tag{3}$$

which is a 0-1 quadratic programming with $\frac{1}{4}$ of the number of variables, compared with problem (2). The fact that this formulation avoid constraints other than the binary´s, makes it suitable for using some metaheuristics, like genetic algorithms, without the care of generating feasible solution populations. The price we pay for using the simplification is the fill-in effect on the matrix $\bar{H}$.

Motivated by the existence of robust solvers and the simplicity of linear programming, ti is desirable whenever it is possible, to reformulate combinatorial programs into integer linear programs. When there are quadratic relationships between binary variables, this is always possible. One suitable linear formulation for a quadratic model with binary variables replaces each quadratic term say $h_{ij} x_i x_j$, by a linear term $h_{ij} x_{ij}$, incorporating the following linear constraints into the original model

$$\begin{array}{l} -x_{ij} + x_i + x_j \leq 1 \\ x_{ij} - x_i \qquad \leq 0 \\ x_{ij} \qquad - x_j \leq 0 \\ x_{ij}, x_i, x_j \in \{0,1\}. \end{array}$$

The effect is to enforce $x_{ij}$ to be 1 only if both $x_i$ and $x_j$ are one, otherwise $x_{ij}$ should be 0. This kind of transformation add to the model one extra binary variable for each quadratic term, that is if the problem has $m$ variables, then potentially $m^2$ new binary variables should be incorporated. Fortunately, the quadratic relationships in our model only appear in consecutive slots, so, the number of new variables/constraints is $O(m)$. By using this kind of transformation, the problem 2 is equivalently written as

$$\begin{array}{ll} \text{Minimize} & l(y,w) = c^T y + r^T w \\ \text{subject to} & Ay = b \\ & Cy + Dw \leq d \\ & (y,w) \in \{0,1\}^{4n(2n-1)} \times \{0,1\}^{2n(4n-3)}. \end{array} \tag{4}$$

 

The price we pay for linearization is to add extra variables and constraints to the model. A similar linearization is made in the simplified version:

$$\begin{array}{lc} \text{Minimize} & \bar{l}(z,w) = \bar{a} + \bar{c}^T z + \bar{r}^T w \\ \text{subject to} & \bar{C}z + \bar{D}w \leq d \\ & (z,w) \in \{0,1\}^{n(2n-1)} \times \{0,1\}^{n^2(2n-1)^2}, \end{array} \tag{5}$$

which is a linear integer program with potentially more variables (paying the price of fill-in the quadratic form). Other choices for linearizing are possible. For instance, in [9] a linear relaxation for the problem 2 was proposed, in the case of minimizing the total traveling distance in a single round robin tournament. They propose adding continuous variables instead, obtaining a relaxation which approximate the quadratic model.

# 3 Numerical Results

Four different integer programming formulations for the same problem are solved with simulated data: two quadratic programs with linear objective constraints (2) and (3), and two integer linear programs (4) and (5). All computations were performed on a PC Intel(R) core(TM) i7-3632 QM, 2.20 GHZ, 64 bits. At the first experiment we fix the

Table 1: Runtime for the linear and quadratic models fixing the timetable

| | Quadratic fixing timetable | | | | Linear fixing timetable | | | |
|---|---|---|---|---|---|---|---|---|
| | full | | reduced | | full | | reduced | |
| $2n$ | mean | SD | mean | SD | mean | SD | mean | SD |
| 4 | 0.02768 | 0.01198 | 0.02247 | 0.01144 | 0.07063 | 0.00445 | 0.26672 | 0.62680 |
| 6 | 0.09818 | 0.01719 | 0.10523 | 0.03064 | 0.18222 | 0.09887 | 0.20801 | 0.09900 |
| 8 | 0.29231 | 0.07935 | 0.32632 | 0.12838 | 0.54525 | 0.28502 | 0.68075 | 0.42887 |
| 10 | 1.14851 | 0.29057 | 1.17092 | 0.30648 | 2.75070 | 0.82891 | 3.52782 | 1.07226 |
| 12 | 6.92692 | 0.74739 | 6.68803 | 1.08223 | 8.08235 | 2.47076 | 12.3484 | 1.84931 |
| 14 | 10.4368 | 0.99115 | 9.95315 | 0.89028 | 17.2353 | 3.84266 | 26.0670 | 9.45421 |
| 16 | 15.1239 | 1.65025 | 13.8831 | 1.45535 | 141.171 | 301.716 | 1890.36 | 2953.40 |
| 18 | 20.6103 | 1.86102 | 21.6154 | 7.18129 | | | | |
| 20 | 50.1043 | 21.6128 | 60.6731 | 22.7746 | | | | |

objective function and solve problems 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 problems, because different timetables leads to different instances.For the second experiment we fix a timetable, and then we solve the problem for 10 different objective functions, for each problem size (single configuration).We solve HA-assignment problems for a par 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 SICP [1]. In table 1 we

DOI: 10.5540/03.2018.006.01.0311  

Table 2: Runtime for the linear and quadratic models fixing the objective function

| | Quadratic fixing OF | | | | Linear fixing OF | | | |
|---|---|---|---|---|---|---|---|---|
| | full | | reduced | | full | | reduced | |
| $2n$ | mean | SD | mean | SD | mean | SD | mean | SD |
| 4 | 0.01881 | 0.00349 | 0.02049 | 0.00817 | 0.07151 | 0.00443 | 0.06999 | 0.00469 |
| 6 | 0.10556 | 0.03898 | 0.09833 | 0.02711 | 0.24210 | 0.10551 | 0.21468 | 0.08341 |
| 8 | 0.22068 | 0.06098 | 0.24719 | 0.07874 | 0.72274 | 0.30959 | 0.58631 | 0.40193 |
| 10 | 1.02054 | 0.30153 | 1.15223 | 0.54655 | 2.55786 | 1.35314 | 3.15127 | 1.08560 |
| 12 | 6.27446 | 1.73542 | 6.07395 | 1.71097 | 6.31574 | 1.45673 | 9.04688 | 2.50039 |
| 14 | 10.9502 | 1.57499 | 9.91357 | 0.59958 | 18.4189 | 4.30626 | 125.277 | 286.680 |
| 16 | 14.5375 | 1.38106 | 13.5211 | 1.30986 | 131.309 | 306.120 | 903.701 | 306.196 |
| 18 | 21.3364 | 1.86914 | 21.9286 | 6.80267 | | | | |
| 20 | 60.8347 | 19.6424 | 41.4042 | 22.7594 | | | | |

present mean and standard deviation for the runtime, in the instances where the objective function is fixed. The first four columns show results for the full and reduced size quadratic models (2) and (3), while the last four columns are about the linear models (4) and (5). As can be expected, the running times for the reduced version in the quadratic models are consistently smaller. Studying the linear versions, we show results for $n$ from 2 to 8 due to the time consuming of larger problems. Observe that while in small problems ($n = 2$ to 6) the reduced version behaves better than the full size, for larger problems this advantage is lost. This fact is explained by the multiplication effect of the binary variables. Another comparison is between the quadratic models vs linearized models. We use the same solver for quadratic and linear binary programs with the intent of reducing the effect of using particular solvers when comparing the models. Now we shall study table 2, 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 functions, 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 comparison between the quadratic and linear models also suggest that quadratic models are stronger.

## 4 Conclusion

In this work we study the effect that different integer linear and quadratic equivalent formulations for the HA-assignment problem have over the solver's performance. By manipulating the special structure of the constraints, we propose a quadratic formulation with 1/4 of the original size. Then we linearize both of the above formulations, obtaining two more versions.

We compare the formulations by showing the runtime of the solver SCIP, in each version. For the set of problems with the same structure (timetable) we observe, as can be

DOI: 10.5540/03.2018.006.01.0311  

expected, that the reduced versions behave better for the quadratic formulations. We also note that the quadratic version behaves much faster than the linear ones. This shows that even the linear models are equivalent to the quadratic ones, they make the solver behave worst, due to the multiplication of the number of variables. This fact provides evidence that using the model in its natural form sometimes brings advantages.

In [8] a SDP relaxation was proposed for HA-assignment problems, based on a MIN-RES-CUT modeling. The problem (2) provides a quadratic programming model in binary variables which lead to a different SDP formulation, which we are addressing in in an ongoing work.

## References

[1] Achterberg, T. "Solving constraints integer programs" *Mathematical Programming Computation*, 1 (1) pp. 1–41, 2009.

[2] Easton, K., Nemhauser, G. and Trick, M. 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*, pages 580–585. Springer, Berlin, 2001.

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

[4] Elf, M., Jünger, M and Rinaldi, G.: "Minimizing breaks by maximizing cuts"; *Operations Research Letters*, 31 (2003) 343–349.

[5] Kendall, G., Knust, S., Ribeiro, C. and Urrutia, S.: "Scheduling in sports: An annotated bibliography". *to appear*

[6] Miyashiro, R. and Matsui, T.: "Semidefinite programming based approaches to the break minimization problem". *Computers and Operations Research* (to appear).

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

[8] Suzuka, A., Miyashiro, R., Yoshise, A. and Matsui, T. 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.
http://www.i.u-tokyo.ac.jp/mi/mi-e.htm

[9] Trick, M. A Schedule-then-Break Approach to Sport Timetabling *GSIA, Carnegie Mellon,* Pittsburgh
http://mat.gsia.cmu.edu

DOI: 10.5540/03.2018.006.01.0311