Noname manuscript No.
(will be inserted by the editor)

Optimizing rest times and differences in games played: an iterative two-phase approach

David Van Bulck · Dries Goossens

Received: date / Accepted: date

Abstract A considerable number of sports competitions cope with limited availability of teams and sports infrastructure by organizing their timetable around a season that comprises many more time slots than games per team. However, in such timetables the rest period between teams' consecutive games can vary considerably and the difference in the number of games played at any point in the season can become large. In this paper, we propose an iterative two-phase approach to construct relaxed round-robin timetables that are less prone to these fairness issues. In particular, the first phase determines the game-off-day pattern (GOP) set which regulates when teams play (home or away) or have an off day (also called bye). Subsequently, the second phase constructs a compatible timetable which specifies the opponents and the home advantage of the games. If no compatible timetable exists, we generate one or more logic-based Benders cuts that rule out the infeasible GOP set in future iterations. We test the two-phase approach on a problem where feasibility of the timetable and the objective value is mainly determined by when teams play games, and show that our approach excels when the number of additional time slots is moderate.

Keywords Time-relaxed round-robin sports scheduling · First-break-then-schedule · Game-off-day pattern set feasibility · Logic-based Benders decomposition · Hybrid integer/constraint programming

1 Introduction

In a round-robin sports tournament each team meets every other team a fixed number of times. Due to their omnipresence in real-life, round-robin tournaments are intensively studied in the timetabling literature (for an overview, see e.g., Van Bulck et al. (2020) and Durán (2020)). Most contributions in the literature focus on so-called compact variants where the number of time slots to schedule all games is minimal. If the number of teams is even, this means that each team plays exactly one game per time slot. However, in several professional (e.g., the NBA (Bean and Birge, 1980) or NHL (Costa, 1994)) and in most non-professional round-robin tournaments (see e.g., Schönberger et al. (2004) and Knust (2010) for table tennis, or Van Bulck et al. (2019) for indoor football) the season comprises more time slots than strictly needed to schedule all games which makes that teams regularly have an off day (also called bye) during which they do not play any game. These so-called relaxed tournaments offer more flexibility to handle limited availability of teams and venues.

Two important downsides of relaxed timetables are that the rest period between teams' consecutive games can vary considerably and that the difference in the

D. Van Bulck
Ghent University, Faculty of Economics and Business Administration
E-mail: david.vanbulck@ugent.be

D. Goossens
Ghent University, Faculty of Economics and Business Administration
FlandersMake@UGent - core lab CVAMO, Ghent, Belgium
E-mail: dries.goossens@ugent.be

number of games played per team at any point in the season can become large. These issues have been studied before in the literature. Schönberger et al. (2004), Knust (2010), and Van Bulck et al. (2019) consider player and venue availability constraints and try to balance the rest time of teams over the season so as to avoid congested periods that potentially result in injuries. Van Bulck and Goossens (2020a) propose two heuristics (an adaptive large neighborhood search and a memetic algorithm) to deal with fairness issues related to rest time and differences in games played. For the former, they introduce a measure they coined the aggregated rest time penalty, which penalizes the occurrence of matches involving the same team, with few rest days in between. Goossens et al. (2020) discuss how to equitably split the unfair aspects over the teams. Instead of balancing the absolute rest time of teams over the season, Atan and Çavdaroğlu (2018) and Çavdaroğlu and Atan (2020) consider the difference in rest time of opposing teams. In summary, because the construction of relaxed sports timetables with availability constraints is a challenging and $\mathcal{NP}$-hard problem (see Van Bulck and Goossens (2020a)), the literature focuses on (meta)heuristics that often exploit the competition structure of the problem at hand. However, existing algorithms are hard to generalize and do not provide any bound on the optimal objective value. Moreover, previous contributions mostly focus on relaxed competitions with *many* more time slots than games per team, while the case with *slightly* more time slots has only sporadically been investigated (e.g., Durán et al. (2019)).

In compact scheduling, there are numerous examples that show the wide applicability of the so-called first-break-then-schedule approach which first determines when teams play home and away (i.e., home-away patterns or HAPs) after which it determines the opponents (see Nemhauser and Trick (1998)). In combination with backtracking, the first-break-then-schedule approach is exact (see Rasmussen and Trick (2007), Rasmussen (2008)). As far as we are aware, the only decomposition approach for relaxed scheduling is by Durán et al. (2019) who propose to first determine the sequence in which each team plays every other team, after which the specific time slot of each game is assigned. In this paper, we propose an alternative decomposition scheme to produce relaxed timetables that avoid short rest times and large differences between the number of games played by each team. Since rest times and differences in games played are fully determined once the game days of each team are fixed, this paper proceeds as follows. In a first phase, we use integer programming to determine the time slots on which teams play their games (i.e., game-off-day patterns or GOPs, see Section 3) thereby optimizing rest times and controlling for the maximal difference in games played. In a second phase, we use constraint programming to construct a compatible timetable which specifies the opponents and the home advantage of the games. We pay particular attention on how to use logic-based Benders decomposition to backtrack between the two phases and avoid infeasible GOP sets in subsequent iterations.

The remainder of this paper is as follows. In Section 2, we give a formal description of the problem followed by a state-of-the-art integrated integer programming formulation. Section 3 proposes the two-phase approach, and Section 4 presents computational results. Conclusions follow in Section 5.

2 Problem description and integer programming formulation

A competition has a set of teams T with $|T| = n$ and a set of time slots S containing all time slots of the season. In a double round-robin tournament (2RR) each team in T plays twice against every other team (once at home and once away). A 2RR is compact if $|S| = 2(n-1)$ and n even or $|S| = 2n$ and n odd, and is relaxed otherwise. In the remainder of this paper, we focus on relaxed 2RRs.

As most relaxed competitions studied in the literature consider venue and team availability constraints, we assume that each team $i \in T$ provides a team availability set $A_i \subseteq S$ during which team i can play and a venue availability set $H_i \subseteq A_i$ during which i can additionally host its game. Non-professional teams may for example use the team availability set to block inconvenient time slots such as Christmas and New Year while they may use the venue availability set to avoid playing home during time slots on which they do not have access to venues (e.g., Schönberger et al. (2004), Van Bulck et al. (2019)). On the other hand, professional teams may use the availability constraints to model constraints arising from police, international tournaments, or stadium availability (e.g., Costa (1994), Durán et al. (2019)). To avoid timetable congestion, it is usually also required that no team plays more than twice within σ consecutive time slots (e.g., Schönberger et al. (2004), Van Bulck et al. (2019)). Furthermore, we consider two measures that increase the quality and fairness of the timetables. First, to ensure more accurate tournament rankings and reduce the opportunities for match fixing, we limit the games played difference index of a timetable which is 'the minimum integer ρ such that at any point in the timetable, the difference between the number of games played by any two teams is at most ρ (Suksompong, 2016)'. Second, to avoid injuries and reduce the impact of not being fully rested, we minimize the aggregated rest time penalty (ARTP) that 'penalizes the timetable with a positive value of $p_r \geq p_{r+1}$ each time a team has only $r < \tau$ time slots between two consecutive games (Van Bulck and Goossens, 2020a)'. The parameter τ reflects the number of days after which we assume a team has fully recovered from its previous game.

In summary, the problem is to minimize the ARTP while assigning each game (i, j) of the double round-robin tournament, with home team $i \in T$ and away team $j \in T \setminus \{i\}$, to a time slot $s \in S$ such that each team plays at most once per time slot and:

- *(C1)* the team availability A_i is respected for all teams (i.e., no game (i, j) or (j, i) is planned on a time slot $s \notin A_i$),
- *(C2)* the venue availability H_i is respected for the home teams (i.e., no game (i, j) is planned on a time slot $s \notin H_i$),
- *(C3)* the games played difference index is at most ρ, and
- *(C4)* each team plays at most two games per σ consecutive time slots.

The problem without the objective function and Constraints *(C3)* and *(C4)* (i.e., $\rho = 2(n-2)$ and $\sigma = 2$) is known in the literature as 'RAC-2RR', and is proven to be $\mathcal{NP}$-complete by Van Bulck and Goossens (2020a).

Equations (1)-(14) present an integer programming (IP) formulation for the problem just described. This formulation is based on Van Bulck and Goossens (2020a) and can be considered as current state-of-the-art. Our main decision variable is $x_{i,j,s}$, which is 1 if team $i \in T$ plays a home game against team $j \in T \setminus \{i\}$ on time slot $s \in S$, and 0 otherwise. Variable $q_{i,s}$ represents the number of games played by team i up to and including time slot $s \in S$, and variable $y_{i,s,t}$ is 1 if team i plays a game on time slot s, followed by its next game on time slot t, for each $s, t \in A_i$ such that $s < t \leqslant \tau + s$, and 0 otherwise.

$$\textbf{minimize} \sum_{i \in T} \sum_{\substack{s,t \in A_i: \\ s<t \leq s+\tau}} p_{t-s-1} y_{i,s,t} \tag{1}$$

$$\sum_{s \in H_i \cap A_j} x_{i,j,s} = 1 \qquad \forall i, j \in T : i \neq j \tag{2}$$

$$\sum_{j \in T \setminus \{i\}} (x_{i,j,s} + x_{j,i,s}) \leqslant 1 \qquad \forall i \in T, \forall s \in S \tag{3}$$

$$q_{i,1} = \sum_{j \in T \setminus \{i\}} (x_{i,j,1} + x_{j,i,1}) \qquad \forall i \in T \tag{4}$$

$$q_{i,s} = q_{i,s-1} + \sum_{j \in T \setminus \{i\}} (x_{i,j,s} + x_{j,i,s}) \qquad \forall i \in T, s \in S \setminus \{1\} \tag{5}$$

$$q_{i,s} - q_{j,s} \leqslant \rho \qquad \forall i, j \in T : i \neq j, \forall s \in A_i \tag{6}$$

$$\sum_{j \in T \setminus \{i\}} \sum_{p=s}^{s+\sigma-1} (x_{i,j,p} + x_{j,i,p}) \leqslant 2 \qquad \forall i \in T, \forall s \in S : s + \sigma - 1 \leq |S| \tag{7}$$

$$\sum_{j \in T \setminus \{i\}} \Big(x_{i,j,s} + x_{j,i,s} + x_{i,j,t} + x_{j,i,t} - \sum_{k=s+1}^{t-1} (x_{i,j,k} + x_{j,i,k}) \Big) - 1 \leqslant y_{i,s,t} \qquad \forall i \in T, s, t \in A_i : s < t \leqslant \tau + s \tag{8}$$

$$\sum_{j \in T \setminus \{i\}} (x_{i,j,s} + x_{j,i,s}) \geqslant y_{i,s,t} \qquad \forall i \in T, s, t \in A_i : s < t \leqslant \tau + s \tag{9}$$

$$\sum_{j \in T \setminus \{i\}} (x_{i,j,t} + x_{j,i,t}) \geqslant y_{i,s,t} \qquad \forall i \in T, s, t \in A_i : s < t \leqslant \tau + s \tag{10}$$

$$1 - \sum_{j \in T \setminus \{i\}} (x_{i,j,k} + x_{j,i,k}) \geqslant y_{i,s,t} \qquad \forall i \in T, s, k, t \in A_i : s < k < t \leqslant \tau + s \tag{11}$$

$$x_{i,j,s} = 0 \qquad \forall i, j \in T : i \neq j, s \notin H_i \cap A_j \tag{12}$$

$$x_{i,j,s} \in \{0, 1\} \qquad \forall i, j \in T : i \neq j, s \in H_i \cap A_j \tag{13}$$

$$y_{i,s,t} \in \{0, 1\} \qquad \forall i \in T, s, t \in A_i : s < t \leqslant \tau + s \tag{14}$$

$$q_{i,s} \in \mathbb{N} \qquad \forall i, j \in T : i \neq j, s \in H_i \cap A_j \tag{15}$$

Objective function (1) minimizes the sum of rest time penalties (ARTP). The first set of constraints ensures that each team plays exactly one home game against every other team, while respecting the team and venue availability constraints *(C1)* and *(C2)*. The next set of constraints enforces that a team plays at most one game per time slot. Constraints (4) and (5) recursively model the number of games a team played up to and including time slot $s \in S$, and Constraints (6) limit the maximal difference in games played *(C3)*. The next set of constraints ensures that a team plays at most two games per σ consecutive time slots *(C4)*. Constraints (8)-(11) regulate the value of the $y_{i,s,t}$ variables by considering the number of time slots between two consecutive games of the same team. It follows from *(C4)* that the games are consecutive if team i plays on time slot s and t and $|t - s| < \sigma$ in which case we can strengthen the formulation by dropping the negative summation term of (8). Moreover, under the assumption that all penalties p_r are positive, Constraints (9)-(11) are redundant. Constraints (12) reduce the number of variables in the system; when implementing this formulation, these variables need not be created. Constraints (13) to (15) are the binary constraints on the x and y-variables, and the integrality constraints on the $q_{i,s}$ variables. Note that the integrality of $q_{i,s}$ follows from (4), (5), and (13), and that the integrality of $y_{i,s,t}$ follows from the objective function and constraints (8) and (13). Nevertheless, we keep the binary and integrality constraints for these variables in our model since this may enable the generation of additional cuts (e.g., Gomory or 0/1 mixed-integer cuts), but, give the highest branching priority to the $x_{i,j,s}$ variables, and the lowest priority to all the remaining variables.

3 A two-phase approach

A relaxed sports timetable can be seen as a combination of a game-off-day pattern set and a compatible timetable. First, the game-off-day pattern of a team (GOP) defines for each time slot whether a team plays a game ('G') or has an off day ('O'). Second, the compatible timetable specifies the opponent a team faces in every time slot in which it plays a game and whether this game is at home or away. An assignment of one GOP to each team is known as a GOP set, and is said to be feasible if there exists a compatible timetable (see Table 1). Clearly, not every GOP set is feasible: for instance, for every pair of teams there must be at least two time slots on which these teams have a 'G' in their pattern.

Motivated by the following two observations with regard to feasibility and optimality, an attractive alternative to the IP model from Section 2 is to first construct the game-off-day patterns and then find a compatible timetable.

Observation 1 *Once the GOP set is known, it is known whether its compatible timetables respect Constraints (C1), (C3), and (C4).*

Observation 2 *Once the GOP set is known, the ARTP of its compatible timetables is known.*

The remainder of this section explains how to construct a GOP set (Section 3.1), construct a compatible timetable (Section 3.2), and backtrack by generating logic-based Benders cuts when no compatible timetable exists (Section 3.3). Implementation details follow in Section 3.4. Figure 1 provides an overview of the structure of the two-phase approach we propose.

3.1 Generating GOP sets

In the first-break-then-schedule approach, typically all feasible home-away patterns are first enumerated, after which a pattern set is constructed by assigning exactly one different pattern to each team. However, the same approach cannot be used in relaxed scheduling as the same GOP can be assigned to more than one team and the large number of possible GOPs makes it impractical to enumerate all GOPs first. Indeed, since each team plays $2(n-1)$ games in a double round-robin tournament, there are $\binom{|S|}{2(n-1)}$ different GOPs. When there are 8 teams, this means there is only 1 GOP for a compact 2RR, 15 GOPs when there is one off day per team, and already 40.116.600 GOPs when there are 14 off days per team (i.e., one off day for each game). The construction of a GOP set may be simple enough, though, such that we can use IP model (16)-(27) to generate a GOP set that minimizes the ARTP of its compatible timetables. In this formulation, our main decision variable is $g_{i,s}$ which is 1 if team $i \in T$ has a 'G' on time slot $s \in S$, and 0 otherwise. Furthermore, the auxiliary variable $y_{i,s,t}$ is 1 if team $i \in T$ plays a game on time slot $s \in S$ followed by its next game on time slot $t \in S$, $s < t \leq s + \tau$, and 0 otherwise. Finally, the auxiliary variable u_s ensures that an even number of teams have a 'G' on time slot $s \in S$.

GOP SET GENERATION MODEL

$$\textbf{minimize} \sum_{i \in T} \sum_{\substack{s,t \in A_i: \\ s<t\leq s+\tau}} p_{t-s-1} y_{i,s,t} \tag{16}$$

$$\sum_{s \in A_i} g_{i,s} = 2(n-1) \qquad \forall i \in T \tag{17}$$

$$\sum_{s \in H_i} g_{i,s} \geqslant n-1 \qquad \forall i \in T \tag{18}$$

$$\sum_{\substack{p \in A_i: \\ s \leq p \leq s+\sigma-1}} g_{i,p} \leqslant 2 \qquad \forall i \in T, s \in A_i \tag{19}$$

$$\sum_{\substack{p \in A_i: \\ p \leqslant s}} g_{i,p} - \sum_{\substack{p \in A_j: \\ p \leqslant s}} g_{j,p} \leqslant \rho \qquad \forall i,j \in T: i \neq j, \forall s \in A_i \tag{20}$$

$$\sum_{i \in T: s \in A_i} g_{i,s} = 2u_s \qquad \forall s \in S \tag{21}$$

$$\sum_{i \in T: s \in H_i} g_{i,s} \geqslant \sum_{i \in T: s \in A_i \setminus H_i} g_{i,s} \qquad \forall s \in S \tag{22}$$

$$\left.\begin{array}{l} \sum_{s \in H_i \cap A_j} g_{i,s} \geq 1 \\ \sum_{s \in H_i \cap A_j} g_{j,s} \geq 1 \\ \sum_{s \in A_i \cap A_j} g_{i,s} \geq 2 \end{array}\right. \qquad \forall i,j \in T: i \neq j \tag{23}$$

$$g_{i,s} + g_{i,t} - \sum_{k=s+1}^{t-1} g_{i,k} - 1 \leq y_{i,s,t} \qquad \begin{array}{r}\forall i \in T, s,t \in A_i: \\ s < t \leqslant \tau + s\end{array} \tag{24}$$

$$g_{i,s} \in \{0,1\} \qquad \forall i \in T, s \in A_i \tag{25}$$

$$y_{i,s,t} \in \{0,1\} \qquad \begin{array}{r}\forall i \in T, s,t \in A_i: \\ s < t \leqslant \tau + s\end{array} \tag{26}$$

$$u_s \in \mathbb{N} \qquad \forall s \in S \tag{27}$$

Objective function (16) models the ARTP of the timetable by enforcing a penalty each time a team plays two games within τ time slots. By definition, each GOP contains $2(n-1)$ 'G's on time slots during which a team is available (Constraints (17)). Moreover, to be able to schedule all home games, a GOP must contain at least $(n-1)$ 'G's on time slots during which a team's venue is

This is the peer-reviewed author-version of https://doi.org/10.1007/s10951-021-00717-3, published in Journal of Scheduling.

Table 1 Illustration of a GOP set (left) and two compatible timetables (middle, right) for four teams playing a single round-robin during four time slots.

T/S	1	2	3	4
1	G	G	O	G
2	G	G	O	G
3	O	G	G	G
4	O	G	G	G

S	1	2	3	4
	(1,2)	(2,4)	(3,4)	(1,4)
		(3,1)		(2,3)

S	1	2	3	4
	(1,2)	(4,1)	(3,4)	(2,4)
		(2,3)		(1,3)

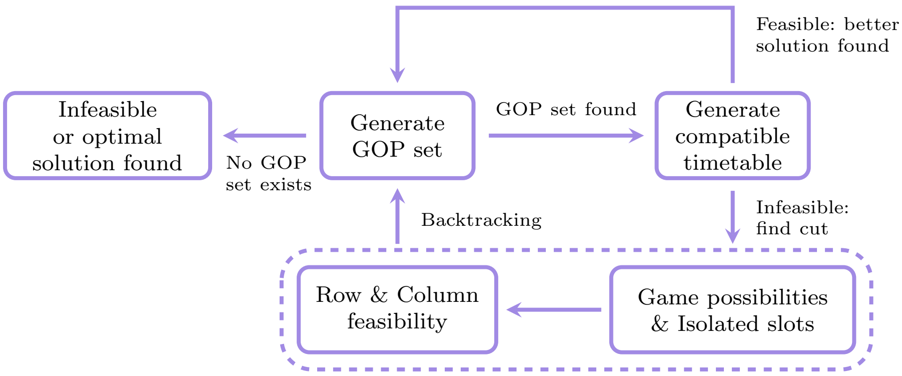

Fig. 1 General structure of the two-phase approach. The generation of GOP sets is discussed in Section 3.1, the construction of a compatible timetable in Section 3.2, and the generation of infeasibility cuts in Section 3.3.

available (Constraints (18)). Subsequently, Constraints (19) and (20) respectively model that no GOP contains more than two 'G's in any sequence of σ time slots, and that the maximal difference in games played is at most ρ. Constraints (21) enforce that the sum of 'G's is even on each time slot, a necessary condition since each game involves two teams. Since each game involves one home team and one away team, Constraints (22) enforce that the total number of teams that can play home and have a 'G' on time slot $s \in S$ must be larger than or equal to the number of teams that have a 'G' and can only play away on s. A necessary condition to be able to schedule game (i, j) is that team i and team j each have at least one 'G' during time slots on which i can play home and the players of j are available. In addition, to be able to schedule game (i, j) and (j, i) simultaneously, it is necessary that team i and team j each have at least two 'G's during time slots on which the players of team i and j are simultaneously available (Constraints (23)). Constraints (24) regulate the value of the $y_{i,s,t}$ variables. When $t - s$ is smaller than σ, we note that the negative summation term in Constraints (24) can be dropped. Finally, Constraints (25) to (27) are respectively the binary constraints on the $g_{i,s}$ variables, the binary constraints on the $y_{i,s,t}$ variables, and the integrality constraints on the u_s variables. Observe that it follows from Constraints (25) and the objective function that Constraints (26) could in principle be relaxed. For reasons discussed before, we do not relax these constraints but give the highest branching priority to the $x_{i,j,s}$ and u_s variables, and the lowest priority to all the remaining variables.

3.2 Generating a compatible timetable

Once a GOP set is found, we need to check whether it is feasible. This problem is known as the GOP-set feasibility problem and is $\mathcal{NP}$-complete (see Van Bulck and Goossens (2020b)). We use constraint programming formulation (28)-(29) to construct a compatible timetable or to prove that no such timetable exists. In this formulation, variable $x_{i,j}$ has domain $H_i \cap A_j$ and gives the time slot assigned to game (i, j), and parameter $g'_{i,s}$ is 1 if team $i \in T$ has a 'G' on time slot $s \in S$ in the GOP set found by model (16)-(27) and is 0 otherwise.

Timetable generation model

$$x_{i,j} \neq s \qquad \forall i, j \in T : i \neq j, \forall s \in H_i \cap A_j : g'_{i,s} \neq 1 \vee g'_{j,s} \neq 1 \tag{28}$$

$$\texttt{all-different}(x_{i,j}\ \forall j \in T \setminus \{i\} \cup x_{j,i}\ \forall j \in T \setminus \{i\}) \qquad \forall i \in T \tag{29}$$

Constraints (28) enforces the given GOP set by further reducing the domain of the $x_{i,j}$ variables to the time slots during which both of the teams have a 'G' in their pattern. Next, Constraints (29) make use of the global all-different predicate to enforce that no team plays more than one game per time slot.

This is the peer-reviewed author-version of https://doi.org/10.1007/s10951-021-00717-3, published in Journal of Scheduling.

3.3 Backtracking by adding logic-based Benders cuts

In case a GOP set turns out to be infeasible, we separate logic-based Benders cuts that rule out this and hopefully many other infeasible GOP sets in future iterations. In contrast to traditional Benders cuts that exploit information from the linear programming dual, logic-based Benders cuts are derived from an *inference dual*, i.e., a condition that when satisfied implies that the master problem is infeasible or suboptimal (Hooker and Ottosson, 2003). We consider four different families of cuts (Sections 3.3.1 to 3.3.4).

3.3.1 Game possibilities

Denote with $c_G(T', s)$ the number of 'G's in the GOPs of a subset of m teams $T' \subseteq T$ on time slot $s \in S$. The number of games between teams in T' on s is at most $\lfloor \frac{c_G(T',s)}{2} \rfloor$. Hence, Condition 1 is a necessary condition for the feasibility of a GOP set that involves $\mathcal{O}(2^n)$ constraints.

Condition 1 (Bao (2009)) $\sum_{s\in S} \left\lfloor \frac{c_G(T',s)}{2} \right\rfloor \geqslant m(m-1)$ *for each* $T' \subseteq T$.

Instead of explicitly checking Condition 1 for each subset of teams, we formulate an IP model to find a minimal subset of teams for which Condition 1 is violated or to prove that no such subset exists. The formulation of this IP is based on Rasmussen and Trick (2007), with the main difference that it checks the feasibility of a GOP set instead of an HAP set and that it requires to be solved only once instead of once for each cardinality of T'. Parameters $g'_{i,s}$ define the given GOP set, and parameter UB_{GP} gives an upper bound on the cardinality of the subsets to be checked and therefore controls for the expected computation time. Variable z_m, $2 \leqslant m \leqslant \text{UB}_{\text{GP}}$, determines whether the cardinality of the subset is m ($z_m = 1$) or not ($z_m = 0$), α_i determines whether team i is in the subset ($\alpha_i = 1$) or not ($\alpha_i = 0$), and variable β_s calculates an upper bound on the total number of games the teams in the subset can play on time slot $s \in S$.

$$\textbf{minimize} \quad \sum_{i\in T} \alpha_i \tag{30}$$

$$\sum_{i\in T} \alpha_i = \sum_{m=2}^{\text{UB}_{\text{GP}}} m z_m \tag{31}$$

$$\sum_{m=2}^{\text{UB}_{\text{GP}}} z_m = 1 \tag{32}$$

$$\beta_s \geqslant \Big(\sum_{i\in T} g'_{i,s}\alpha_i - 1\Big)/2 \qquad \forall s \in S \tag{33}$$

$$\sum_{s\in S} \beta_s \leqslant m(m-1) - 1 + \text{UB}_{\text{GP}}(\text{UB}_{\text{GP}} - 1)(1 - z_m) \qquad \forall m \in \mathbb{N} : 2 \leqslant m \leqslant \text{UB}_{\text{GP}} \tag{34}$$

$$\begin{aligned} &\alpha_i \in \{0,1\} && \forall i \in T \\ &z_m \in \{0,1\} && \forall m \in \mathbb{N} : 2 \leqslant m \leqslant \text{UB}_{\text{GP}} \\ &\beta_s \in \mathbb{N} && \forall s \in S \end{aligned} \tag{35}$$

The objective function minimizes the total number of teams chosen such that infeasibility of the GOP set can be traced back to as few patterns as possible, and Constraints (31)-(32) model the value of z_m. Constraints (33) calculate the upper bound on the total number of games between teams in the subset on time slot s, and Constraints (34) ensure that Condition 1 is violated. Finally, Constraints (35) are the binary and integrality constraints.

If a violating set of teams T' defined by the α_i's is found, $|T'| = m$, a logic-based Benders cut of type (36) is added to the GOP set generation model. Intuitively, this constraint forbids any GOP set in which the teams in T' play according to the GOPs currently assigned.

$$\sum_{i\in T'} \sum_{s\in S: g'_{i,s}=1} g_{i,s} \leq m(2(n-1)) - 1 \tag{36}$$

When $T' = T$, this constraint is known in the literature as a no-good constraint (Rahmaniani et al., 2017).

3.3.2 Isolated slots

Define with $S_{T'} \subseteq S$ the subset of time slots on which at least two teams in $T' \subseteq T$, $|T'| = m$, have a game and all teams not in T' have an off day, i.e., $\sum_{i\in T'} g'_{i,s} \geqslant 2$ and $\sum_{i\in T\setminus T'} g'_{i,s} = 0$ for all $s \in S_{T'}$. We refer to the subset of time slots $S_{T'}$ as isolated slots for the subset of teams T'.

Condition 2 *For each subset of teams $T' \subseteq T$, the sum of G's during isolated slots is smaller than twice the total number of mutual games in T', i.e., $\sum_{i\in T'} \sum_{s\in S_{T'}} g'_{i,s} \leqslant 2m(m-1)$.*

Condition 2 is a necessary condition since teams in T' can only play against other teams in T' during isolated slots, and the total number of games between teams in T' is limited by $m(m-1)$.

Instead of explicitly checking each subset, we formulate an IP model to find a minimal subset of teams for which Condition 2 is violated, or which proves that no such subset exists. In this formulation, parameter $\mathrm{UB_{IS}}$ denotes the maximal subset to be checked. Variable γ_s, $s \in S$, denotes the number of 'G's if s is an isolated slot for the subset of teams defined by α_i, and variable z_m determines whether the cardinality of the subset is m ($z_m = 1$) or not ($z_m = 0$).

$$\textbf{minimize} \quad \sum_{i \in T} \alpha_i \tag{37}$$

$$\sum_{i \in T} \alpha_i = \sum_{m=2}^{\mathrm{UB_{IS}}} m z_m \tag{38}$$

$$\sum_{m=2}^{\mathrm{UB_{IS}}} z_m = 1 \tag{39}$$

$$\gamma_s \leqslant \alpha_i \sum_{j \in T} g'_{j,s} \qquad \forall s \in S, i \in T : g'_{i,s} = 1 \tag{40}$$

$$\begin{aligned} &\textstyle\sum_{s \in S} \gamma_s \geqslant 2m(m-1) + 1 && \forall m \in \mathbb{N} : \\ &-\big(2\mathrm{UB_{IS}}(\mathrm{UB_{IS}} - 1) + 1\big)(1 - z_m) && 2 \leqslant m \leqslant \mathrm{UB_{IS}} \end{aligned} \tag{41}$$

$$\begin{aligned} &\alpha_i \in \{0,1\} && \forall i \in T \\ &z_m \in \{0,1\} && \forall m \in \mathbb{N} : 2 \leqslant m \leqslant \mathrm{UB_{IS}} \\ &\gamma_s \geq 0 && \forall s \in S \end{aligned} \tag{42}$$

$$\gamma_s \leq \sum_{j \in T} g'_{j,s} \qquad \forall s \in S \tag{43}$$

The objective function minimizes the total number of teams in the subset, while Constraints (38) and (39) model the z_m variables. If all teams that have a 'G' on time slot $s \in S$ are in the subset, s is an isolated slot and Constraints (40) counts the total number of 'G's. Constraints (41) ensure that Condition 2 is violated, and Constraints (42) are the binary and non-negativity constraints. Finally, Constraints (43) eliminate the γ_s variables related to time slots on which no team plays a game.

If a violating set of teams T' defined by the α_i's is found, the following logic-based Benders cut is added to the GOP set generation model.

$$\sum_{i \in T'} \sum_{s \in S_{T'}} g_{i,s} \leqslant 2m(m-1) + 2n(n-1) \sum_{i \in T \setminus T'} \sum_{s \in S_{T'}} g_{i,s} \tag{44}$$

Intuitively, Constraint (44) forbids the violation of Condition 2 by either limiting the total number of 'G's during time slots in $S_{T'}$ or by requiring that at least one team not in T' has a 'G' during one of the time slots in $S_{T'}$.

3.3.3 Row feasibility checks

For a given GOP set, the following condition essentially checks feasibility for a subset of rows (i.e., teams, see also Rasmussen and Trick (2007)).

Condition 3 *For each subset of teams $T' \subseteq T$, an assignment of the mutual games between teams in T' to time slots in S exists in which the opposing teams have a 'G' in their pattern and the venue of the home team is available.*

By the definition of a feasible GOP set, Condition 3 is a necessary condition for all $T' \subseteq T$ and a sufficient condition if $T' = T$. In order to check Condition 3 for each subset of teams with cardinality $\mathrm{UB_{ROW}}$ or lower, we use the constraint programming formulation (28)-(29) but define variables $x_{i,j}$ only for $i, j \in T', i \neq j$. In case a violating subset of teams T' is found, we add a cut of type (36) to the GOP set generation model.

3.3.4 Column feasibility checks

Instead of checking the rows of a GOP set, we may also check feasibility for a subset of columns (i.e., time slots).

Condition 4 *For each subset of time slots $S' \subseteq S$, an assignment of games to time slots in S' exists such that for each $s \in S'$ team $i \in T$ plays exactly one game (i,j) or (j,i) if $g'_{i,s} = 1$ and $s \in H_i$, exactly one game (j,i) if $g'_{i,s} = 1$ and $s \in A_i \setminus H_i$, and no game if $g'_{i,s} = 0$.*

By the definition of a feasible GOP set, Condition 4 is a necessary condition for all $S' \subseteq S$ and a sufficient condition if $S' = S$. Given a set of time slots S', we use CP formulation (45)-(51) to check Condition 4. In this formulation variable $o_{i,s}$ has domain $\{-n, -n+1, \ldots, -2, -1, 1, 2, \ldots, n-1, n\}$ and its absolute value gives the opponent of team $i \in T$ on time slot $s \in S$ whenever i's pattern has a 'G' on time slot s. If the sign of $o_{i,s}$ is positive, then i plays a home game against team $o_{i,s}$ whereas it plays away against team $-o_{i,s}$ if the sign is negative.

$$o_{i,s} \neq i \wedge o_{i,s} \neq -i \qquad \forall i \in T, s \in S' : g'_{i,s} = 1 \tag{45}$$

$$o_{i,s} < 0 \qquad \forall i \in T, s \in A_i \setminus H_i : g'_{i,s} = 1 \tag{46}$$

$$o_{i,s} \neq j \wedge o_{i,s} \neq -j \qquad \begin{aligned} \forall i \in T, s \in S' : g'_{i,s} = 1, \\ \forall j \in T \setminus \{i\} : g'_{j,s} = 0 \end{aligned} \tag{47}$$

$$o_{i,s} \neq -j \qquad \begin{aligned} \forall i \in T, s \in S' : g'_{i,s} = 1, \\ \forall j \in T \setminus \{i\} : s \notin H_j \end{aligned} \tag{48}$$

$$\texttt{all-different}(o_{i,s} \; \forall s \in S' : g'_{i,s} = 1) \qquad \forall i \in T \tag{49}$$

$$\texttt{all-different}(o_{i,s} \; \forall i \in T : g'_{i,s} = 1) \qquad \forall s \in S' \tag{50}$$

$$o_{i,s} = j \Rightarrow o_{j,s} = -i \qquad \begin{aligned} \forall i \in T, s \in H_i : g'_{i,s} = 1, \\ \forall j \in T \setminus \{i\} : g'_{j,s} = 1 \end{aligned} \tag{51}$$

Constraints (45) to (48) reduce the domain of the decision variables by respectively stating that team $i \in T$ cannot play against itself, i cannot play home when its venue is not available, i cannot play against j when

j does not have a 'G' in its pattern, and i cannot play away against j when j's venue is not available. Constraints (49) state that a team plays at most once against every other opponent with the same home-away status. Next, Constraints (50) enforce a one factor between all teams that play on s; these constraints are in principle redundant but can strengthen the formulation considerably (see Trick (2003)). Finally, Constraints (51) link the opponent variables.

When drawing a (relatively small) number of time slots at random, initial experiments revealed that Condition 4 is only sporadically violated. Moreover, in a relaxed timetable the total number of slot subsets may quickly become intractable. Instead, we therefore enumerate all subsets of teams T' with cardinality $\mathrm{UB}_{\mathrm{COL}}$ or lower and set S' equal to the subset of all time slots during which the majority of the teams that have a 'G' are in T'. This way, there are fewer subsets to be enumerated and the subsets are more likely to trigger a violation of Constraints (49).

Whenever an infeasible subset of columns S' is found, we add a constraint of type (52) to the GOP set generation model.

$$\sum_{s \in S'} \sum_{i \in T : g'_{i,s}=1} g_{i,s} + \sum_{s \in S'} \sum_{i \in T : g'_{i,s}=0} (1 - g_{i,s}) \leq n|S'| - 1 \quad (52)$$

3.4 Implementation

The outline of the algorithm is as follows (see also Figure 1). In the first iteration, we use the model from Section 3.1 to generate a GOP set or prove that none exists in which case the problem instance is obviously infeasible. For a given GOP set, we then use the model from Section 3.2 to construct a compatible timetable or prove that the GOP set is infeasible. In the latter case, we try to separate one or more logic-based Benders cuts that forbid this and hopefully many other infeasible GOPs as well. To this purpose, we start by separating both the game possibility and the isolated slots cuts; we set $\mathrm{UB}_{\mathrm{GP}} = \mathrm{UB}_{\mathrm{IS}} = 6$ as we believe that for larger values the cut would be weak as too many teams or time slots would be involved. The outcome is either one or two violated cuts in which case we continue the search for a new GOP set, or no violated cut in which case we try to separate both the row and column cuts. We thereby set $\mathrm{UB}_{\mathrm{ROW}} = \mathrm{UB}_{\mathrm{COL}} = 3$ since the enumerative nature of these constraints makes checking larger values impractical. Again, the result is either one or two violated cuts in which case we continue the search for a new GOP set, or no violated cut in which case we add a cut of type (36) for $T' = T$.

In the classical (logic-based) Benders decomposition framework, the master problem is solved to optimality and a new branch-and-bound tree is built at each iteration. This approach, however, likely revisits candidate solutions that have been visited in previous trees and does not fully exploit the re-optimization tools implemented in modern IP solvers (Rahmaniani et al., 2017). An alternative approach, known as branch-and-Benders-cut and followed in this paper, is therefore to build a single search tree and use (lazy-constraint) callbacks to generate logic-based Benders cuts for all integer solutions encountered inside the tree (Rahmaniani et al., 2017). If the GOP set generation model extended with all generated cuts is infeasible, then the last found solution is thus optimal or in case no solution was found the problem instance is proven infeasible.

4 Computational results

This section describes a benchmark of problem instances (Section 4.1), experimentally evaluates the performance of the iterative two-phase approach (Section 4.2), and provides more insights into the cut generation process of the two-phase approach (Section 4.3).

4.1 Problem instances

We use the generator from Van Bulck and Goossens (2020a) to create a benchmark of 32 double round-robin problem instances. A problem instance in this set is of type (n, o, h, a) if it contains n teams and $2(n-1)+o$ time slots (i.e., each team has o off days), and for each team $i \in T$ it holds that $|H_i| = (n-1)+h$ and $|S \setminus A_i| = a$. We consider n in the set $\{8, 12, 16, 20\}$, o in $\{(n-1), 2(n-1), 3(n-1), 4(n-1), 5(n-1), 6(n-1), 7(n-1), 8(n-1)\}$, set $h = \lfloor o/2 \rfloor$ and $a = \lfloor o/4 \rfloor$, and assume that a team is fully rested after five time slots (i.e., $\tau = 5$, see e.g., Scoppa (2015) for football). Furthermore, we set $p_r = 2^{\tau - r - 1}$ for all $r < \tau$, and require that the games-played difference index is not larger than 2 (i.e., $\rho = 2$). If the total number of time slots is at least twice the minimal number needed we require that no team plays more than 2 games within 3 consecutive time slots (i.e., $\sigma = 3$); we set $\sigma = 2$ otherwise. While these parameters are chosen somewhat arbitrarily (based on a real-life application in indoor football, see Van Bulck et al. (2019)), no significant differences were observed when slightly deviating from these parameters during preliminary testing.

4.2 Performance analysis

All IP formulations are solved with ILOG CPLEX version 12.10, and the CP formulations are solved with ILOG CPLEX CP OPTIMIZER 12.10. The monolithic IP formulation (1)-(15) and the GOP set generation model were both granted 5 cores and 3600 seconds of computation time. To allow for parallel optimization, the separation models from Section 3.3 were granted one core with a time limit of 600 seconds. No time limit was imposed on the timetable generation model (also granted one core). All models were run on a CentOS 7.9 GNU/Linux based system with an Intel Gold 6240 processor, running at 2.6 GHz and provided with 16 GB of RAM.

In order to benchmark our two-phase approach against state-of-the-art, we also implemented the Adaptive Large Neighborhood Search (ALNS) heuristic as proposed by Van Bulck and Goossens (2020a). This heuristic was originally designed for relaxed competitions with many more time slots than games per team, and essentially performs a sequence of optimize-and-fix operations. In particular, the ALNS heuristic repeatedly selects a subset of free $x_{i,j,s}$ variables related to a subset of teams (team destructor) or time slots (time destructor), fixes all other $x_{i,j,s}$ variables at their value in the current solution, and re-optimizes all free $x_{i,j,s}$ variables using an IP solver (Gurobi version 9.1). An initial solution is generated by disregarding the optimization criteria and solving the corresponding feasibility problem with the IP solver. The ALNS method was granted the same computation time and was run on the same hardware as all other methods.

Table 2 presents the computational results for each of the algorithms. The first four columns provide the aforementioned features of the problem instances. Next, the table gives the best found solution ('Best') and lower bound ('LB'), as well as the total time used by the monolithic IP formulation from Section 2 and the iterative two-phase approach from Section 3. If the two-phase approach ran for less than an hour, either the best solution found is optimal (i.e., the lower bound equals the best found solution) or the algorithm ran out-of-memory (i.e., the lower bound is less than the best found solution). Finally, the table shows the best solution found by the ALNS approach. The results show that the two-phase approach is clearly the best choice when the number of off days is no more than twice the number of games per team: not only does the two-phase approach find all best found solutions for the instances with $o \leq 4(n-1)$, it is also able to prove optimality for eight out of these sixteen problem instances. Compared to the monolithic IP formulation which finds an optimal solution for only five problem instances, optimality is also proven considerably faster (e.g., a factor of 60 for the instance with $n = 12$ and $o = 11$). However, when the number of off days per team becomes larger, the two-phase approach struggles to find high-quality feasible GOP sets and the ALNS method may become a better choice (although unable to provide lower bounds).

4.3 Insights into cut generation

Table 3 provides more insights into the backtracking and cut generation of the two-phase approach for the problem instances with 12 and 16 teams. Columns 5 and 6 respectively show the total number of feasible and infeasible GOP sets found. As can be expected, more GOPs turn out to be be infeasible when the number of off days increases: the 'G's become more sparsely distributed over the time slots and hence there is less flexibility to schedule games between opponents as more and more games become implicitly fixed.

The next column of Table 3 gives (*i*) the total number of game possibility and (*ii*) isolated slot cuts added to the first phase, (*iii*) the total number of times we tried to separate these cuts, and (*iv*) the total time in seconds the separation process took over all trials. When the number of off days is small, most GOPs are feasible and consequently not many violated cuts are found. However, when the number of off days increases more violated cuts are separated and this within a modest amount of time: less than half a minute to check more than 200 GOPs in the problem instance with $n = 12$ and $o = 88$. The observation that fewer cuts were found for the instances with 16 teams could perhaps be explained by that fact that a team's pattern contains more 'G's when the number of teams increases. Hence, there are more time slots on which teams can possibly play against each other and on each time slot there are also more teams against which a team can play.

When infeasibility of the problem instance could not be traced back to game possibility and isolated slot cuts, the two-phase approach separates row and column infeasibility cuts. The penultimate column of Table 3 gives (*i*) the total number of row and (*ii*) column infeasibility cuts added to the first phase, (*iii*) the total number of times we tried to separate these cuts, and (*iv*) the total time in seconds the separation process took over all trials. For a considerable number of infeasible GOPs, infeasibility could be traced back to no more than three rows (i.e., teams) or columns (i.e., time slots). Although row and column infeasibility cuts are more general than game possibility and isolated slot cuts, they are more expensive to calculate.

				IP			Two-phase			ALNS
n	o	h	a	Best	LB	Time	Best	LB	Time	Best
8	7	3	1	1216	1216	5	1216	1216	3	1216
	14	7	3	824	824	22	824	824	5	828
	21	10	5	596	596	613	596	596	471	610
	28	14	7	405	405	1140	405	405	1028	411
	35	17	8	305	281	3600	361	277	2809	294
	42	21	10	246	0	3600	535	172	2185	200
	49	24	12	187	1	3600	/	0	1360	154
	56	28	14	116	0	3600	789	0	940	88
12	11	5	2	2996	2996	362	2996	2996	6	3036
	22	11	5	2036	2030	3600	2032	2032	42	2184
	33	16	8	1522	1458	3600	1462	1462	1461	1610
	44	22	11	1121	993	3600	1119	935	3600	1190
	55	27	13	956	664	3600	1177	521	3600	884
	66	33	16	759	0	3600	1231	0	3469	654
	77	38	19	612	0	3600	1448	0	3600	438
	88	44	22	380	0	3600	1179	0	3600	297
16	15	7	3	5652	5620	3600	5620	5620	42	5842
	30	15	7	4522	3784	3600	3800	3728	3600	4163
	45	22	11	2909	2714	3600	2780	2525	2520	3279
	60	30	15	4727	1814	3600	2127	1007	3600	2575
	75	37	18	4682	0	3600	2517	146	3600	2006
	90	45	22	1535	0	3600	2108	0	3600	1556
	105	52	26	4735	0	3600	3353	0	3600	1436
	120	60	30	4547	0	3600	4361	0	3600	908
20	19	9	4	9228	8844	3600	8856	8773	3600	9579
	38	19	9	7308	6015	3600	6216	5826	3600	6764
	57	28	14	7382	4100	3600	4627	2727	3600	5476
	76	38	19	7333	1100	3600	3780	342	3600	4710
	95	47	23	/	0	3600	7610	0	3600	3974
	114	57	28	/	0	3600	7340	0	3600	3259
	133	66	33	/	0	3600	5314	0	3600	3215
	152	76	38	/	0	3600	7271	0	3600	3587

Table 2 Computational results for the three algorithms. The first four columns respectively refer to the number of teams (n), the number of off days per team (o), the average venue availability (h), and the average team unavailability (a). Symbol '/' means that no solution was found within the given computation time.

Finally, the last column of Table 3 gives (i) the total number of infeasible GOPs for which we could not trace back the source of infeasibility in which case we add a cut of type (36) for $T' = T$, (ii) the total number of GOPs found, and (iii) the total time taken by the CP model over all trials to construct a compatible timetable or prove that none exists. The results show that only for a few GOPs we could not find a violated cut. Moreover, the low overall computation time for the CP formulation shows that CP is an excellent method to construct a compatible timetable or to identify an infeasible GOP set as such.

5 Conclusion

It is common in sports timetabling to break down the timetabling process into different subproblems. Perhaps the most popular approach is first-break-then-schedule where the first subproblem is to determine the home-away pattern (HAP) set which defines when teams must play at home or play away. Observing that existing two-phase approaches focus on compact timetables where the number of time slots is the minimally needed, this paper investigates how to use a two-phase approach when there are more time slots than games per team. In particular, this paper proposes to first use integer programming to determine the game-off-day pattern (GOP) set defining when teams play (home or away) or have an off day, after which it constructs a compat-

				GOPs		Game + Isolated				Row + Column				CP		
n	o	h	a	Feas.	Inf.	Ga.	Isol.	No.	s.	Row	Col.	No.	s.	Ga.	No.	s.
12	11	5	2	9	0	0	0	0	0	0	0	0	0	0	9	0
	22	11	5	18	15	0	1	15	1	0	0	14	23	14	33	1
	33	16	8	24	361	0	2	361	37	19	106	359	823	238	385	12
	44	22	11	56	480	23	73	480	62	76	281	389	652	104	536	14
	55	27	13	15	279	17	54	279	34	41	186	213	310	26	294	9
	66	33	16	15	234	4	104	234	31	13	122	129	179	7	249	8
	77	38	19	7	230	22	84	230	26	17	117	136	199	19	237	8
	88	44	22	5	216	8	128	216	25	14	80	87	124	7	221	9
16	15	7	3	19	0	0	0	0	0	0	0	0	0	0	19	15
	30	15	7	44	2	0	0	2	0	1	0	2	4	1	46	89
	45	22	11	30	3	0	0	3	0	0	0	3	13	3	33	80
	60	30	15	32	44	0	0	44	9	1	3	44	263	40	76	232
	75	37	18	18	60	6	0	60	15	12	19	54	314	24	78	25
	90	45	22	14	37	0	10	37	7	1	24	27	100	3	51	10
	105	52	26	15	24	8	5	24	4	3	8	13	56	4	39	10
	120	60	30	4	9	7	3	9	2	0	0	0	0	0	13	4

Table 3 Detailed information on backtracking.

ible timetable with constraint programming. To avoid infeasible GOPs in future iterations, we use logic-based Benders decomposition. This paper is therefore also an interesting application of how to combine integer programming techniques that excel in solving optimization problems with constraint programming techniques that excel in solving feasibility problems. It turns out that the construction of GOP sets is simple enough so that there is no need to explicitly enumerate all patterns first. This avoids a combinatorial explosion by only implicitly enumerating all patterns, and allows to check feasibility of GOP sets both on the level of rows (representing patterns of teams) and columns (representing the teams that play on a particular time slot).

We use our approach to generate a number of relaxed double round-robin timetables with availability constraints, and where fairness issues mainly determined by whether or not a team plays are prominent. In particular, we minimize the sum of rest time penalties while controlling for the maximal difference in games played. Computational results show that the approach outperforms existing approaches when the total number of off days is no more than twice the number of games per team. Our approach also has the advantage of being easily extensible towards several other real-life constraints. For instance, it would be trivial to adapt the models to impose a limit on the maximal number of off days in a row. However, in case of fairness issues that mainly depend on the opponent sequence (e.g., travel distance), we expect other decomposition approaches to be more effective.

Acknowledgements David Van Bulck is a postdoctoral research fellow funded by the Research Foundation Flanders (FWO) [1258021N].

References

Atan T, Çavdaroğlu B (2018) Minimization of rest mismatches in round robin tournaments. Comput Oper Res 99:78 – 89

Bao R (2009) Time relaxed round robin tournament and the NBA scheduling problem. PhD thesis, Cleveland State University

Bean JC, Birge JR (1980) Reducing travelling costs and player fatigue in the National Basketball Association. Interfaces 10:98–102

Çavdaroğlu B, Atan T (2020) Determining matchdays in sports league schedules to minimize rest differences. Oper Res Lett 48:209 – 216

Costa D (1994) An evolutionary tabu search algorithm and the NHL scheduling problem. Infor 33:161–178

Durán G (2020) Sports scheduling and other topics in sports analytics: a survey with special reference to Latin America. TOP pp 1–31

Durán G, Durán S, Marenco J, Mascialino F, Rey PA (2019) Scheduling Argentina's professional basketball leagues: A variation on the travelling tournament problem. Eur J Oper Res 275:1126 – 1138

Goossens D, Yi X, Van Bulck D (2020) Fairness trade-offs in sports timetabling. In: Ley C, Dominicy Y (eds) Science meets Sports: when statistics are more than numbers, Cambridge Publishing Scholars

Hooker JN, Ottosson G (2003) Logic-based benders decomposition. Math Program 96:33–60

Knust S (2010) Scheduling non-professional table-tennis leagues. Eur J Oper Res 200:358–367

Nemhauser GL, Trick MA (1998) Scheduling a major college basketball conference. Oper Res 46:1–8

Rahmaniani R, Crainic TG, Gendreau M, Rei W (2017) The benders decomposition algorithm: A literature review. European Journal of Operational Research 259:801 – 817

Rasmussen RV (2008) Scheduling a triple round robin tournament for the best Danish soccer league. Eur J Oper Res 185:795 – 810

Rasmussen RV, Trick MA (2007) A benders approach for the constrained minimum break problem. Eur J Oper Res 177:198 – 213

Schönberger J, Mattfeld DC, Kopfer H (2004) Memetic algorithm timetabling for non-commercial sport leagues. Eur J Oper Res 153:102–116

Scoppa V (2015) Fatigue and team performance in soccer: Evidence from the FIFA World Cup and the UEFA European Championship. J Sport Econ 16:482–507

Suksompong W (2016) Scheduling asynchronous round-robin tournaments. Oper Res Lett 44:96–100

Trick MA (2003) Integer and constraint programming approaches for round-robin tournament scheduling. In: Burke E, De Causmaecker P (eds) Practice and Theory of Automated Timetabling IV, Springer, Berlin, pp 63–77

Van Bulck D, Goossens D (2020a) Handling fairness issues in time-relaxed tournaments with availability constraints. Comput Oper Res 115:104856

Van Bulck D, Goossens D (2020b) On the complexity of pattern feasibility problems in time-relaxed sports timetabling. Operations Research Letters 48:452 – 459

Van Bulck D, Goossens DR, Spieksma FCR (2019) Scheduling a non-professional indoor football league: a tabu search based approach. Ann Oper Res 275:715–730

Van Bulck D, Goossens D, Schönberger J, Guajardo M (2020) RobinX: A three-field classification and unified data format for round-robin sports timetabling. Eur J Oper Res 280:568 – 580