Relax-fix-optimize heuristics for time-relaxed sports timetabling

David Van Bulck[a] and Dries Goossens[a]

[a]Faculty of Economics and Business Administration, Ghent University, Ghent, Belgium

ABSTRACT

Time-relaxed sports timetables utilize (many) more time slots than there are games per team, and therefore offer the flexibility to take into account player and venue availability. However, time-relaxed tournaments also have the drawback that the difference in games played per team and the rest period between teams' consecutive games can vary considerably. In addition, organisers may want to avoid consecutive home and away games (i.e., breaks). To construct fair timetables, we propose relax-and-fix (R&F) and fix-and-optimize (F&O) heuristics that make use of team- and time-based variable partitioning schemes. While the team-based R&F constructs a timetable by gradually taking into account the integrality constraints related to all home games of a subset of teams, the time-based R&F maintains a rolling horizon of time intervals in which the integrality constraints of all games scheduled within the time interval are activated. The F&O heuristics use the same variable partitioning schemes, but they never relax the integrality constraints and allow to recover from mistakes by making a small number of changes with respect to the variables optimized in previous iterations. For numerous real-life instances, our heuristics generate high-quality timetables using only a fraction of the computational resources used by monolithic integer programming solvers.

KEYWORDS

Time-relaxed sports scheduling; Relax-and-fix; Fix-and-optimize; Round-robin tournaments; Availability constraints

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

1. Introduction

Sports are an important aspect of our daily life: millions of fans attend sporting events, follow their favourite players on social media, or participate in sports themselves. The success of these events heavily depends on the sports timetable that must be fair, profitable, thrilling, or simply convenient. Double round-robin tournaments, in which each team meets each other team exactly once at home and once away, are of particular interest due to their omnipresence in real life. Existing round-robin timetabling literature can be categorized either as time-constrained or as time-relaxed (for an overview, see e.g. Kendall et al. (2010) and Van Bulck et al. (2020)). Time-constrained timetables contain a minimal number of time slots to play all games: for a double round-robin tournament with n teams this number is $2(n-1)$ if n is even and $2n$ if n is odd. Time-relaxed timetables, on the other hand, contain (many) more time slots than games per

CONTACT D. Van Bulck. Email: david.vanbulck@ugent.be

team and hence a team regularly has a bye meaning that it does not play any game during a particular time slot.

In this paper, we will focus on the so-called time-relaxed availability-constrained double round-robin problem (`RAC-2RR`). The input of this problem consists of a set of arbitrarily many time slots S (e.g. one slot for each day in the season), a set of teams T, and for each team $i \in T$ a team and venue availability set. The team availability set $A_i \subseteq S$ contains the time slots during which team i can play a game. In non-professional competitions, players can use this set to exclude time slots on which they are not available due to work or family. In professional competitions, the team availability set is used to exclude time slots on which a team has a game in an international competition and hence cannot be scheduled for its national tournament. Similarly, the venue availability set $H_i \subseteq S$ contains the time slots during which team i can play home, and is particularly useful when i's venue is frequently used for other events. Since time slots in H_i can only be used if there are also in A_i, we assume that $H_i \subseteq A_i$ for each $i \in T$. If we denote with triple $(i, j, s) \in T \times T \times S$ a home game of team i against team j (i.e., game (i, j)) on time slot s, a feasible solution consists of an assignment of games to time slots such that:

- *(C1)* each team plays exactly one home game against each other team,
- *(C2)* the venue availability H_i $(i \in T)$ is respected (i.e., no triple (i, j, s) with $s \notin H_i, j \in T \setminus \{i\}$),
- *(C3)* the team availability A_i $(i \in T)$ is respected (i.e., no triple (i, j, s) or (j, i, s) with $s \notin A_i, j \in T \setminus \{i\}$), and
- *(C4)* each team plays at most one game per time slot $s \in S$.

Problem `RAC-2RR` and variants thereof have been studied by several researchers. Schönberger et al. (2004) study a non-professional table-tennis competition and propose a constraint programming formulation and a memetic algorithm. Knust (2010) models the problem as a multi-mode resource-constrained project scheduling problem for which a two-stage heuristic algorithm is proposed. Moreover, Knust (2010) tries to minimize the total number of breaks; a team has a break if it plays two consecutive games with the same home-away status. Van Bulck et al. (2019) study a non-professional indoor football league where the objective is to develop a schedule in which each team has a balanced spread of their games over the season. They propose a novel move operator that schedules or reschedules all home games of a team by solving a transportation problem. Van Bulck and Goossens (2020) prove that `RAC-2RR` is $\mathcal{NP}$-complete, even if venues are always available (i.e., $H_i = A_i \, \forall i \in T$). Moreover, they propose an integer programming (IP) formulation and two heuristics (an adaptive large neighbourhood search and a memetic algorithm) to deal with several fairness issues related to rest time, differences in games played, and breaks. Finally, Yi (2021) proposes proactive and reactive strategies to deal with postponed games in `RAC-2RR`.

Instead of solving a challenging IP formulation at once, a popular (heuristic) approach in the scheduling literature is to break down the monolithic IP formulation into a series of formulations that are simple enough to be solved one after another (see e.g. Dorneles et al. (2014), Oliveira et al. (2014)). A first approach, known as relax-and-fix (R&F), partitions the decision variables into different subsets and initially relaxes all integrality constraints. Subsequently, it selects one of the subsets for which it reactivates the integrality constraints, optimizes the updated model, and fixes the variables in the subset to their (near-)optimal value. A second approach is known as fix-and-optimize (F&O) and never relaxes the integrality constraints. Instead, it optimizes the variable subsets one by one by initially fixing all variables to an initial solution and

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

allowing only a small number of changes with regard to the variables optimized in previous iterations. The contribution of this paper is to show how to solve RAC-2RR with R&F and F&O by means of a team- and time-based variable partitioning scheme. Although time-based R&F and F&O techniques have been used before in a sports context (see Chandrasekharan et al. (2019); de Oliveira et al. (2014); Kim (2019)), as far as we are aware, this is the first application employing a team-based variable partitioning. In addition, we show how to avoid infeasible solutions in early iterations, use parameter tuning tools to analyse the impact of algorithm parameter choices, and investigate when each variable partitioning scheme and heuristic approach works best.

The remainder of this paper is as follows. First, Section 2 provides an integer programming model for RAC-2RR and shows how to extend it to cope with several fairness issues. Section 3 then proposes the R&F and F&O heuristics, and Section 4 presents computational results. Conclusions follow in Section 5.

2. IP formulation and fairness issues

Equations (1)-(4) provide the monolithic IP formulation proposed by Van Bulck and Goossens (2020) to solve RAC-2RR. In this model, the variable $x_{i,j,s}$ is 1 if team $i \in T$ and team $j \in T \setminus \{i\}$ meet at the venue of i on time slot s. The first set of constraints ensures that a team meets each other team exactly once in a home game; this makes that all games of the double round robin are scheduled *(C1)*. The next set of constraints enforces that a team plays at most once per time slot *(C4)*. Constraints (3) reduce the number of variables in the system by explicitly stating that two teams can meet only when they are both available *(C2,C3)*; when implementing this formulation these variables need not be created. Finally, constraints (4) are the binary constraints on the x variables.

Base model

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

$$\sum_{j \in T \setminus \{i\}} (x_{i,j,s} + x_{j,i,s}) \leqslant 1 \qquad \forall i \in T, s \in A_i \quad (2)$$

$$x_{i,j,s} = 0 \qquad \forall i, j \in T, i \neq j, s \in S \setminus \{H_i \cap A_j\} \quad (3)$$

$$x_{i,j,s} \in \{0,1\} \qquad \forall i, j \in T : i \neq j, s \in H_i \cap A_j \quad (4)$$

Clearly, RAC-2RR remains difficult when we additionally include an objective function. The remainder of this section explains how Van Bulck and Goossens (2020) extend the base model to handle several fairness issues.

Aggregated rest time penalty A first fairness measure is related to rest time, i.e., the number of byes between two consecutive games of a team. In particular, to avoid injuries and reduce the impact of a team not being fully rested from the previous game, we minimize the aggregated rest time penalty (ARTP), that 'penalizes a timetable with a value p_r each time a team has only $r < \tau$ time slots between two consecutive games (Van Bulck and Goossens 2020)'. The parameter τ reflects a period which allows any team to fully recover from its previous games. We assume that the penalties p_r are positive and non-increasing in r, that is $p_r \geq p_{r+1} \geq 0$. Ideally, a team thus has at least

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

τ time slots of rest between any two consecutive games in which case the ARTP is zero. Unfortunately, the availability constraints make it unlikely that such timetable exists. To avoid injuries, Van Bulck and Goossens (2020) propose to minimize the ARTP while additionally requiring that a team plays at most twice per $\tau+1$ time slots (refer to this constraint with the symbol *(A1)*). To minimize the ARTP of a timetable, they use an auxiliary variable $y_{i,s,t}$ which is 1 if team i plays a game on time slot s followed by its next game on time slot t, and 0 otherwise. Constraints (5) regulate the value of these auxiliary variables. Additionally, constraints (6) enforce *(A1)*. We note that it follows from *(A1)* that the games are consecutive if team i plays on time slot s and t and $|t-s| \leq \tau$. In the presence of constraint *(A1)*, we can therefore strengthen the formulation by dropping the negative summation term of Equation (5). Finally, constraints (7) state that the y-variables are non-negative; integrality follows from the objective function and the integrality of the x variables.

ARTP model

$$\textbf{minimize} \sum_{i \in T} \sum_{s \in A_i} \sum_{t=s+1}^{s+\tau} p_{(t-s-1)} y_{ist}$$

subject to

$$(1) - (4)$$

$$\sum_{j \in T \setminus \{i\}} (x_{i,j,s} + x_{j,i,s} + x_{i,j,t} + x_{j,i,t} - \sum_{k=s+1}^{t-1} (x_{i,j,k} + x_{j,i,k})) - 1 \leqslant y_{i,s,t} \qquad \forall i \in T, s, t \in A_i : s < t, t - s \leqslant \tau \quad (5)$$

$$\sum_{j \in T \setminus \{i\}} \sum_{k=s}^{s+\tau} (x_{i,j,k} + x_{j,i,k}) \leq 2 \qquad \forall i \in T, s \in A_i \quad (6)$$

$$y_{i,s,t} \geq 0 \qquad \forall i \in T, s, t \in A_i : s < t, t - s \leqslant \tau \quad (7)$$

Games played difference index In a time-constrained timetable with an even number of teams, each team has played the same number of games at the end of each time slot. This is different in time-relaxed timetables in which one team may have played considerably more games than other teams at a given moment in time. Suksompong (2016) defines the games played difference index (GPDI) as 'the minimum integer p such that at any point in the timetable, the difference between the number of games played by any two teams is at most p'. Van Bulck and Goossens (2020) propose to use variables $g_{i,s}$ to represent the number of games played by team $i \in T$ up to and including time slot $s \in S$. Equations (8), (9) and (10) recursively model these auxiliary variables. Next, equations (11) calculate the GPDI value which is minimized by the objective function.

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

GPDI model
minimize GPDI
subject to
$(1) - (4)$

$$g_{i,1} = \sum_{j \in T \setminus \{i\}} (x_{i,j,1} + x_{j,i,1}) \qquad \forall i \in T \quad (8)$$

$$g_{i,s} = g_{i,s-1} + \sum_{j \in T \setminus \{i\}} (x_{i,j,s} + x_{j,i,s}) \qquad \forall i \in T, s \in A_i : s > 1 \quad (9)$$

$$g_{i,s} = g_{i,s-1} \qquad \forall i \in T, s \in S \setminus A_i : s > 1 \quad (10)$$

$$g_{i,s} - g_{j,s} \leqslant \text{GPDI} \qquad \forall i, j \in T : i \neq j, s \in S \quad (11)$$

Total number of breaks A team has a break whenever it plays two consecutive games with the same home-away status. Even though in most competitions, teams return home after each away game, there are (professional) competitions like the NBA or NHL where travel minimization plays an important role. Minimizing a team's total travel distance is typically done by scheduling two (or more) consecutive away games against opponents that are close to each other, such that the team can travel from one opponent to the next without returning home in between. In such a setting, (away) breaks can be favourable. In general though, it is common practice to let teams alternate as much as possible between home and away games, since breaks negatively impact game attendance (Forrest and Simmons 2006) and may be considered unfair as breaks imply (not) having the advantage related to home games for two consecutive games (Durán et al. 2017).

As proposed by Knust (2010), the model below uses a binary variable $b_{i,s}$ that is one if team i has a break on time slot s, and 0 otherwise. Constraints (12) model the home breaks of a team, whereas constraints (13) model the away breaks of a team. Constraints (14) reduce the number of break variables by stating that a team cannot have a break on time slots during which it is unavailable. Finally, constraints (15) are the non-negativity constraints; integrality follows from the objective function and the integrality of the x variables.

Break model
minimize $\sum_{i \in T} \sum_{s \in A_i} b_{i,s}$
subject to
$(1) - (4)$

$$\sum_{j \in T \setminus \{i\}} (x_{i,j,s} + x_{i,j,t} - \sum_{\substack{u \in S: \\ s<u<t}} (x_{i,j,u} + x_{j,i,u})) - b_{i,t} \leqslant 1 \qquad \forall i \in T, s, t \in H_i : s < t \quad (12)$$

$$\sum_{j \in T \setminus \{i\}} (x_{j,i,s} + x_{j,i,t} - \sum_{\substack{u \in S: \\ s<u<t}} (x_{i,j,u} + x_{j,i,u})) - b_{i,t} \leqslant 1 \qquad \forall i \in T, s, t \in A_i : s < t \quad (13)$$

$$b_{i,s} = 0 \qquad \forall i \in T, s \in S \setminus A_i \quad (14)$$

$$b_{i,s} \geq 0 \qquad \forall i \in T, s \in A_i \quad (15)$$

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

3. Relax-fix-optimize heuristics

Since the IP formulations in Section 2 require a considerable amount of computational resources using a state-of-the-art IP solver (see Section 4), this section proposes time- and team-based relax-and-fix and fix-and-optimize approaches. For an introduction to these heuristic approaches, we refer to Pochet and Wolsey (2006) where fix-and-optimize is called 'exchange'.

3.1. Relax-and-fix

The proposed R&F approach works on the level of the x variables: once the x variables are fixed, the value of all other variables can be easily inferred. To partition the x variables into a sequence of subsets, we consider a time- and team-based approach. The 'time-based' approach (time R&F) partitions the time horizon into K equal-sized intervals in which a variable subset consists of all x variables belonging to the same interval. These intervals are constructed randomly, chronologically, or in function of the cost induced by all games scheduled in an interval in the initial solution (parameter O). The motivation for the time-based groupings follows from their frequent use in the related timetabling literature (e.g., Oliveira et al. (2014); de Oliveira et al. (2014); Chandrasekharan et al. (2019)). Similarly, the 'team-based' approach (team R&F) partitions the teams into K groups: a subset of variables now constitutes all x variables for which the home teams are in the same group. These subsets are either created randomly or in function of the total cost induced by a team in the initial solution. The motivation for the team-based grouping follows from the structure of the ARTP and number of breaks objective function which is mainly determined by the interplay of games of the same team.

In the relax-and-fix heuristic, decision variables are always in one of the following three states: integer, relaxed, or fixed. In contrast to integer variables for which the integrality constraints are activated, relaxed variables can take on fractional values. Similarly, it is not allowed to modify the value of fixed variables. Initially, the relax-and-fix framework turns on the integrality constraints for the first $L < K$ variable subsets in the sequence, relaxes all other variables, and solves the resulting model. In the next iteration, the integer subset with the lowest index becomes fixed and the integrality constraints for the relaxed subset with the lowest index are activated. This method is repeated until all variables are either fixed or integer (see Figure 1).

A potential drawback of R&F is to end without any solution because integrality constraints are only gradually taken into account and subproblems may thus become infeasible after fixing the value of variables optimized in earlier iterations. For this reason, we use a state-of-the-art IP solver to construct an initial solution by neglecting the optimization criteria and solving the associated feasibility problem with all integrality constraints. If we do not find a solution for an iteration, we reset the value of all fixed and integer values to their value in the initial solution and proceed to the next

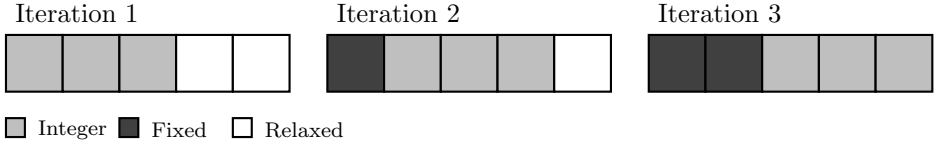

Figure 1. Illustration of the relax-and-fix method for $K = 5$ and $L = 3$.

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

iteration. In practice, resetting is extremely rare: in the computational experiments of Section 4 it was never observed for ARTP and break optimization and only 6 times for GPDI optimization.

Initial experiments revealed that the relax-and-fix framework made many expensive assignments in the final iterations. This may be a consequence of the IP solver exploiting the fractional structure of the relaxed variables to improve the objective value in early iterations. Inspired by a technique used in Oliveira et al. (2014), we mitigate this problem by adding constraint (16) at the beginning of each iteration.

$$|B| - \sum_{\{i,j,s\} \in B} x_{i,j,s} \leqslant \alpha \tag{16}$$

In this constraint, parameter α controls for the size of the search space. Moreover B represents the set of triples $\{i, j, s\}$ for which $x_{i,j,s}$ belongs to the relaxed variable subsets and $x_{i,j,s} = 1$ in the solution of the previous iteration, or in the initial solution for the first iteration. Intuitively, this constraint tries to preserve the integrality of some variables in early iterations, which is however not guaranteed since the x variables can be fractional, and thereby also limits the search space.

3.2. Fix-and-optimize

Section 3.1 argued that the R&F method without constraint (16) makes many expensive assignments towards the last iterations because it seems to exploit the fractional variable structure in early iterations. This section therefore proposes an F&O heuristic which considers all integrality constraints at every iteration and which is able to (partially) recover from mistakes made in previous iterations.

Just like the R&F approach, the F&O approach starts with the construction of an initial solution by solving the associated feasibility problem with all integrality constraints enabled, after which it partitions the x variables in time- (time F&O) or team-based (team F&O) groups. Next, it sequentially optimizes the IP model with all integrality constraints (but without constraint (16)), modifying an arbitrary number of values of the first $L < K$ different variable subsets but at most β values of variables in the other subsets. In the next iteration, the approach optimizes variable subset 2 to $L + 1$; this approach is repeated until all variable subsets have been optimized. Variables in the F&O heuristic can thus only be in two states: free or β-fixed (see Figure 2). We implement this technique by adding 'local-branching' constraint (17) at the beginning of each iteration (see also Fischetti and Lodi (2003)).

$$|C| - \sum_{\{i,j,s\} \in C} x_{i,j,s} \leqslant \beta \tag{17}$$

In this constraint, C represents the set of triples $\{i, j, s\}$ for which $x_{i,j,s}$ does not belong to a variable subset currently being optimized and $x_{i,j,s} = 1$ in the solution of

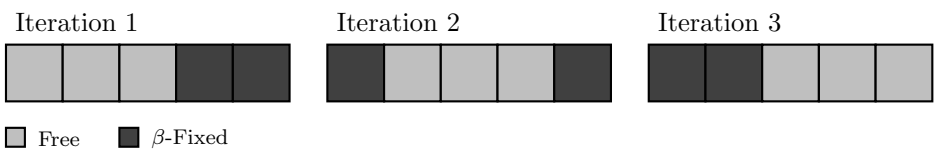

Figure 2. Illustration of the fix-and-optimize method for $K = 5$ and $L = 3$.

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

the previous iteration, or in the initial solution for the first iteration.

4. Computational experiments

This section experimentally evaluates the IP models from Section 2 and the relax-and-fix and fix-and-optimize heuristics from Section 3. The goal of our experiments is to answer the following four research questions.

(i) What type of heuristic (R&F or F&O) is most promising for `RAC-2RR`?
(ii) What type of variable partitioning (time- or team-based) is most promising for `RAC-2RR`?
(iii) Is the quality of the solutions generated with R&F and F&O comparable to those generated with an off-the-shelf IP solver?
(iv) Is the quality of the solutions generated with R&F and F&O comparable to those generated with other state-of-the-art metaheuristics?

The R&F heuristics were implemented in `C++`, compiled with `g++` 4.8.5. All integer formulations and the subproblems of the R&F and F&O heuristics were solved using Gurobi Optimizer 7.5.2 on a CentOS 7.4 GNU/Linux based system with an Intel E5-2680 processor, running at 2.5 GHz and provided with one thread and 8 GB of RAM.

The remainder of this section is as follows. First, Section 4.1 describes the problem instance benchmark. Section 4.2 then explains how we tuned the different parameters of the heuristics and derives some insights from the parameter space. Finally, Section 4.3 compares the performance of the R&F and F&O heuristics with a state-of-the-art IP solver and the adaptive large neighbourhood (ALNS) and memetic algorithm proposed by Van Bulck and Goossens (2020).

4.1. Real-life problem instances

The problem instance set consists of 53 problem instances originating from a real-life non-professional indoor football competition in Belgium (see Van Bulck et al. (2019)). These instances have between 13 and 15 teams and contain 273 or 274 time slots. On average teams can play home during 4.5 time slots more than the number of opponents in the tournament and cannot play any game during 14.8 time slots. For the ARTP objective, we set $\tau = 9$ for the instances with 13 or 14 teams, and $\tau = 8$ for the instances with 15 teams. The penalty values for playing two consecutive games within $r < \tau$ time slots were set to $p_r = 2^{\tau - r - 1}$.

Van Bulck and Goossens (2020) show that it is $\mathcal{NP}$-complete to decide whether a feasible solution respecting all availability constraints exists. Nevertheless, for all real-life instances, a state-of-the-art IP solver was able to solve the base model from (1)-(4) within less than a second. In total, there were 9 infeasible instances. These instances were discarded, leaving us with 44 feasible real-life double round-robin problem instances.

4.2. Parameter tuning and analysis

The heuristics from Section 3 together feature six parameters that need to be set to define a search strategy. The parameters K and L respectively control the total

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

Parameter	Explanation	Range	ARTP	GPDI	Breaks
Team based					
K	Total number of variable subsets	$1, \ldots, 13$	10	3	4
L	Initial number of variable subsets	$1, \ldots, 7$	4	1	2
D	Conditional variable α-constraint	$\{0, 1\}$	1	1	1
α	Upper bound relaxed modifications	$0, \ldots, 20$	0	11	0
O	Grouping order teams	{rand, weight}	w	w	w
Time based					
K	Total number of variable subsets	$1, \ldots, 40$	30	19	6
L	Initial number of variable subsets	$1, \ldots, 20$	13	17	3
D	Conditional variable α-constraint	$\{0, 1\}$	1	1	1
α	Upper bound relaxed modifications	$0, \ldots, 20$	0	0	0
O	Grouping order time slots	{rand, weight, chr.}	r	w	r

Table 1. Overview of the parameters for the relax-and-fix heuristic.

number of variable subsets and the total number of subsets that are simultaneously integer (R&F) or free (F&O). Since the total number of subproblems equals $K - L + 1$, increasing L for fixed K results in fewer but larger subproblems whereas increasing K for fixed L results in more but smaller subproblems. For the objectives that are hard to optimize, we therefore expect a high ratio of K over L. The parameter α and β influence the size of the search space and thereby influences the trade-off between exploitation and exploration. For the relax-and-fix heuristic, α is subordinate to variable D which is activated when Equation (16) is included. Finally, parameter O determines in which order we solve the different subproblems. Tables 1-2 summarize the different parameters and the range of values that were considered.

Algorithmic parameters were tuned for best performance using the `irace` package proposed by López-Ibáñez et al. (2016). The `irace` package implements an advanced iterated racing procedure that mainly consists of the following three steps. First, a number of parameter configurations are sampled from a particular distribution. Second, the best configurations are determined by means of racing: at each step of the race the candidate solutions are tested on a single instance, after which the candidate configurations that perform statistically worse are discarded. Third, the parameter configurations that survived after the last step of the race are used to update the sampling distributions.

We independently tuned the parameters for each heuristic and each objective on the first 10 problem instances using a training budget of 5,000 experiments. The last three columns of Tables 1-2 display the best-found values. To get a better understanding of the parameter space of the different heuristics, Figure 3 plots the frequency of the algorithmic parameters as sampled by `irace` during tuning for the ARTP objective. Due to space limitations, the plots for the variable D were omitted: for both the team and time R&F the frequency of enabling constraint (16) was three times higher than the frequency of not activating this constraint. This constraint was also enabled in all best-found configurations of Table 1, hinting that constraint (16) helps to improve the performance of the relax-and-fix method. When comparing the two different variable partitioning schemes, it is interesting to note that the values of K and L are generally higher in the time-based partitioning. We also observe that the ratio of K over L in the F&O heuristic is higher than that in the F&R heuristic for ARTP optimiza-

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

Parameter	Explanation	Range	ARTP	GPDI	Breaks
Team based					
K	Total number of variable subsets	$1, \ldots, 13$	11	4	2
L	Initial number of variable subsets	$1, \ldots, 7$	1	1	1
β	Upper bound modifications	$0, \ldots, 40$	21	28	0
O	Grouping order teams	{rand, weight}	w	w	r
Time based					
K	Total number of variable subsets	$1, \ldots, 40$	21	12	14
L	Initial number of variable subsets	$1, \ldots, 20$	1	11	13
β	Upper bound modifications	$0, \ldots, 40$	17	36	28
O	Grouping order time slots	{rand, weight, chr.}	w	c	r

Table 2. Overview of the parameters for the fix-and-optimize heuristic.

tion. Intuitively, this can be explained by the fact that the subproblems in the F&O heuristic tend to be more challenging as all integrality constraints are considered at every iteration. Finally, Figure 3 hints that the order given by parameter O is rather unimportant when using a team-based variable partitioning.

4.3. Performance analysis

For each heuristic, the best configuration found by `irace` was run 10 times on each of the 44 feasible problem instances, each time using a different random seed and granted 2 minutes of computation time. To compare the performance of the R&F and F&O heuristics, we also solved the IP formulations (strongest versions) of Section 2 using Gurobi with a time limit of 60 and 180 minutes. In addition, we implemented the Adaptive Large Neighbourhood Search (ALNS) and memetic algorithm proposed by Van Bulck and Goossens (2020). The ALNS heuristic essentially tries to improve an incumbent solution by repeatedly selecting one of multiple destroy operator and repairing this solution with Gurobi. To escape local optima, the ALNS heuristic additionally requires that the returned solution is different from the incumbent solution. The memetic algorithm is a genetic algorithm backed by a local improvement heuristic that schedules or reschedules all home games of a team. These two algorithms were granted 2 minutes of computation time and were run on the same machine as the R&F and F&O heuristics.

Figure 4 displays the absolute gaps for the various algorithms defined as the best solution found by the algorithm minus the best lower bound found by Gurobi run with a time limit of 180 minutes. When comparing the performance of the algorithms on the ARTP objective, it is remarkable to see that the proposed heuristics perform similar or even better than Gurobi, despite being given considerably less computational resources. When comparing the absolute GPDI gap, Figure 4 hints that the time-based F&O heuristic performs slightly worse than the other algorithms. The figure also reveals that Gurobi typically cannot improve its best found solution when run with a time limit of three hours instead of one hour. Finally, Figure 4 shows that Gurobi, the team F&O, and the memetic algorithm regularly find the optimal solution value when minimizing the total number of breaks in the timetable. The other heuristics, in contrast, seem less suitable to minimize breaks as the absolute gap values and the variance thereof are higher.

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

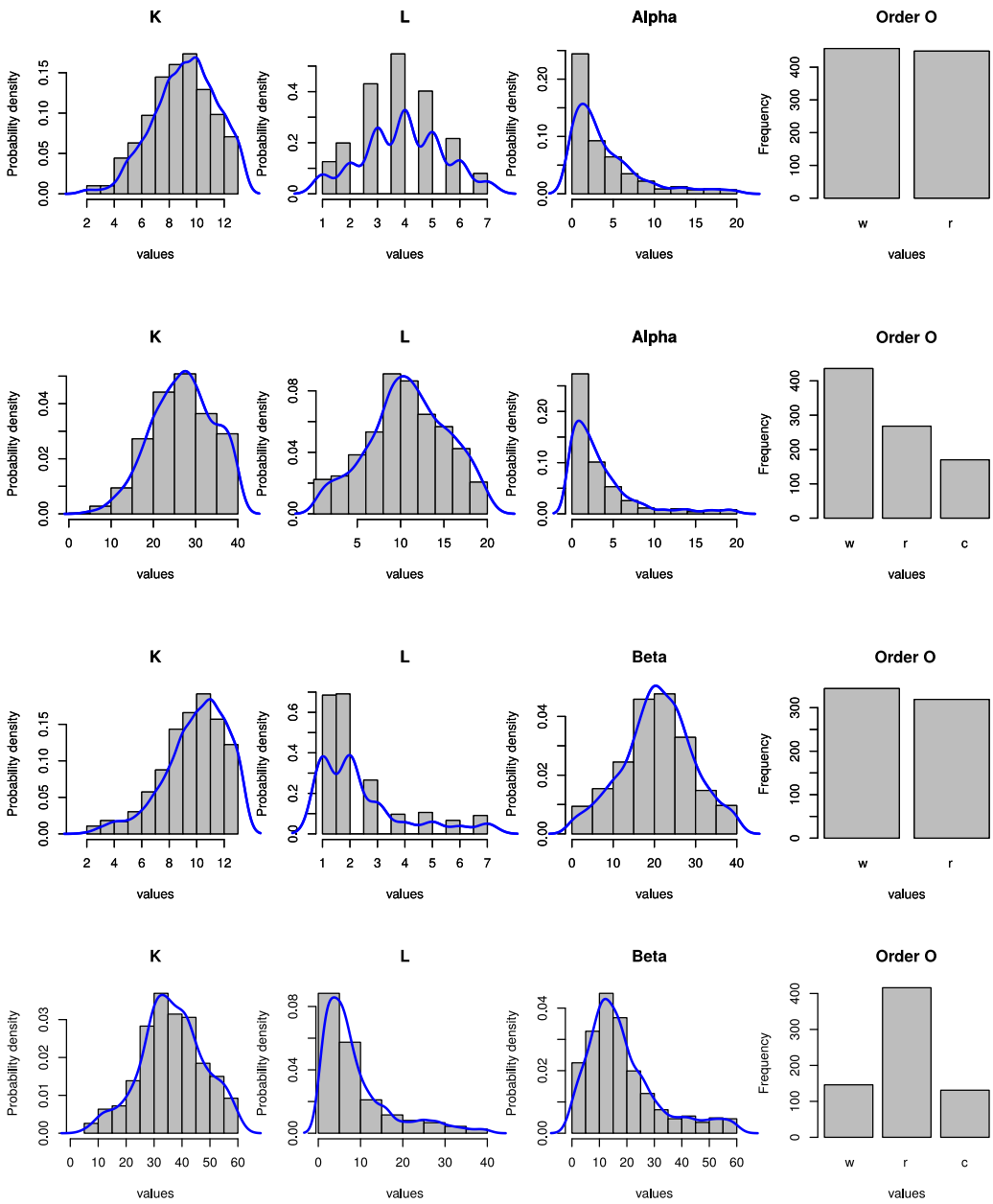

Figure 3. Frequency of parameters as sampled by `irace` to tune the heuristics for the ARTP objective. From top to bottom: team R&F, time R&F, team F&O, and time F&O.

Table 3 provides a more detailed overview of the absolute gap values for the different solution methods. The first column in this table displays the mean of the best lower bounds found by Gurobi run with three hours of computation time. The low values for the GPDI and break measures explain why the table shows the absolute gap instead of the more popular relative gap. The second column represents the mean absolute gap when solving the IP model without considering any objective and thus gives the quality of the initial solutions used in the R&F and F&O heuristics. The remaining columns display the average gap for each solution approach.

To verify the null hypothesis stating that the population mean of the absolute gap differs between two solution methods, Table 4 reports the p-values resulting from a

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

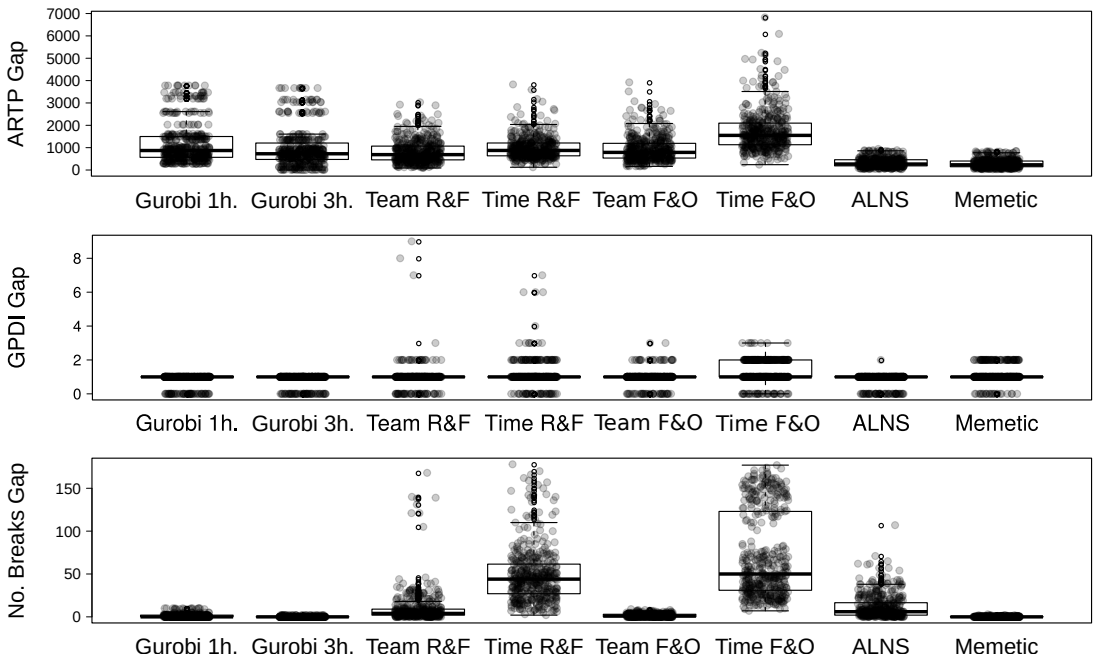

Figure 4. Boxplots of the absolute ARTP gap (top), GPDI gap (middle), and number of breaks gaps (bottom). Boxes represent the three quartiles, whiskers are drawn at 1.5 times the interquartile range, grey dots represent the solution quality for each solution, and black circles represent outliers.

	Gurobi				R&F		F&O			
	LB	No obj.	1h	3h	Team	Time	Team	Time	ALNS	Memetic
ARTP	761	9,856	1,180	958	814	980	534	975	324	286
GPDI	1.15	7.84	0.86	0.86	1.00	1.13	0.97	1.28	0.87	1.11
Breaks	0.11	144.59	1.13	0.14	9.49	48.50	1.67	71.14	11.55	0.14

Table 3. Mean absolute gaps for the different solution methods. Gaps are based on the best lower bound found by Gurobi run with 3 hours of computation time (see column 'LB').

pairwise Wilcoxon rank sum test with Bonferonni's correction for multiple testing. The p-value gives the smallest level of significance at which the null hypothesis would be rejected. A small p-value therefore indicates strong evidence that one solution method systematically performs better in terms of the mean absolute gap; a large p-value, on the contrary, hints that the two solutions methods perform equally well.

Our computational experiments show that for all considered objectives the team-based variable partitioning methods score significantly better than the time-based variable partitioning methods. When comparing the team-based R&F and F&O, we find that F&O scores significantly better on the ARTP and breaks objective while there is no significant difference for the GPDI objective. With regard to the comparison of the R&F and F&O heuristics and the monolithic IP formulation solved with Gurobi, we find that the team-based R&F and F&O heuristic respectively outperform Gurobi run with one and three hours of computation time on the ARTP objective. While Gurobi outperforms the proposed heuristics on the GPDI and breaks objective, the differences are small despite the fact that the R&F and F&O heuristics are given considerably less computation time than Gurobi (2 minutes vs. 1 or 3 hours). When comparing the

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

	Gur. 1h	Gur. 3h	Team R&F	Time R&F	Team F&O	Time F&O	ALNS
ARTP							
Gurobi 3h.	< 2e-16	-	-	-	-	-	-
Team R&F	< 2e-16	0.11379	-	-	-	-	-
Time R&F	0.00091	0.08576	3.8e-15	-	-	-	-
Team F&O	< 2e-16	< 2e-16	< 2e-16	< 2e-16	-	-	-
Time F&O	0.00111	0.14026	5.3e-16	1.00000	< 2e-16	-	-
ALNS	< 2e-16	< 2e-16	< 2e-16	< 2e-16	< 2e-16	< 2e-16	-
Memetic	< 2e-16	< 2e-16	< 2e-16	< 2e-16	< 2e-16	< 2e-16	< 2e-16
GPDI							
Gurobi 3h.	-	-	-	-	-	-	-
Team R&F	1.0e-07	1.0e-07	-	-	-	-	-
Time R&F	< 2e-16	< 2e-16	2.8e-05	-	-	-	-
Team F&O	4.8e-09	4.8e-09	1	2.9e-05	-	-	-
Time F&O	< 2e-16	< 2e-16	< 2e-16	7.4e-06	< 2e-16	-	-
ALNS	1	1	1.9e-06	< 2e-16	7.1e-08	< 2e-16	-
Memetic	< 2e-16	< 2e-16	2.2e-07	1	2.4e-07	4.1e-07	< 2e-16
Breaks							
Gurobi 3h.	< 2e-16	-	-	-	-	-	-
Team R&F	< 2e-16	< 2e-16	-	-	-	-	-
Time R&F	< 2e-16	< 2e-16	< 2e-16	-	-	-	-
Team F&O	5.2e-09	< 2e-16	< 2e-16	< 2e-16	-	-	-
Time F&O	< 2e-16	< 2e-16	< 2e-16	< 2e-16	< 2e-16	-	-
ALNS	< 2e-16	< 2e-16	4.8e-05	< 2e-16	< 2e-16	< 2e-16	-
Memetic	< 2e-16	1	< 2e-16	< 2e-16	< 2e-16	< 2e-16	< 2e-16

Table 4. Pairwise Wilcoxon signed rank values for the mean objective values of Table 3. For the GPDI objective, no p-value is available (n/a) comparing Gurobi 1h with Gurobi 3h as the objective value for each instance was exactly the same.

performance of the proposed heuristics with the ALNS and memetic algorithm, we see that the team-based F&O heuristic scores slightly worse on the ARTP objective but it statistically outperforms the memetic algorithm on the GPDI objective and the ALNS heuristic on the breaks objective. An overview of the recommended heuristics for each of the objective functions is given in Table 5.

5. Conclusion

This study proposes relax-and-fix and fix-and-optimize heuristics to tackle fairness-related issues that occur in time-relaxed timetables. Our computational results show that these heuristics only use a fraction of the computational resources used by a monolithic integer programming solver, and that the quality of the timetables is comparable to those generated by other state-of-the-art heuristics. With regard to optimizing rest times and breaks, we find that fix-and-optimize works better than relax-and-fix and that a team-based variable partitioning works better than a time-based variable partitioning. The latter observation is likely due to the structure of the objective function which is mainly determined by the interplay of games of the same team. This is interesting since many methods in related research fields use a time-based variable partitioning.

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

Objective	Recommended heuristics
ARTP	ALNS, Memetic
GPDI	Team R&F, Team F&O, ALNS
Breaks	Team F&O, Memetic

Table 5. Overview of recommended heuristics for each of the objective functions.

Acknowledgement(s)

The computational resources (Stevin Supercomputer Infrastructure) and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by Ghent University, the Research Foundation Flanders (FWO), and the Flemish Government – department EWI. This work was supported by the Special Research Fund of Ghent University (BOF Ghent) [01N02516] and FWO [1258021N].

References

Chandrasekharan RC, Toffolo TA, Wauters T. 2019. Analysis of a constructive matheuristic for the traveling umpire problem. Journal of Quantitative Analysis in Sports. 15:41–57.

de Oliveira L, de Souza CC, Yunes T. 2014. Improved bounds for the traveling umpire problem: a stronger formulation and a relax-and-fix heuristic. Eur J Oper Res. 236:592 – 600.

Dorneles ÁP, de Araújo OCB, Buriol LS. 2014. A fix-and-optimize heuristic for the high school timetabling problem. Comput Oper Res. 52:29–38.

Durán G, Guajardo M, Sauré D. 2017. Scheduling the South American qualifiers to the 2018 FIFA World Cup by integer programming. Eur J Oper Res. 262:1109 – 1115.

Fischetti M, Lodi A. 2003. Local branching. Math Program. 98:23–47.

Forrest D, Simmons R. 2006. New issues in attendance demand: The case of the English football league. J Sports Econ. 7:247–266.

Kendall G, Knust S, Ribeiro CC, Urrutia S. 2010. Scheduling in sports: An annotated bibliography. Comput Oper Res. 37:1–19.

Kim T. 2019. Optimal approach to game scheduling of multiple round-robin tournament: Korea professional baseball league in focus. Comput Ind Eng. 136:95 – 105.

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

López-Ibáñez M, Dubois-Lacoste J, Cáceres LP, Birattari M, Stützle T. 2016. The irace package: Iterated racing for automatic algorithm configuration. Oper Res Perspect. 3:43–58.

Oliveira BB, Carravilla MA, Oliveira JF, Toledo FMB. 2014. A relax-and-fix-based algorithm for the vehicle-reservation assignment problem in a car rental company. Eur J Oper Res. 237:729–737.

Pochet Y, Wolsey LA. 2006. Production planning by mixed integer programming. Springer.

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

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

Van Bulck D, Goossens D. 2020. Handling fairness issues in time-relaxed tournaments with availability constraints. Comput Oper Res. 115:95–105.

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.

Van Bulck D, Goossens DR, Spieksma FCR. 2019. Scheduling a non-professional indoor football

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.

league: a tabu search based approach. Ann Oper Res. 275:715–730.
Yi X. 2021. Dealing with uncertainty in round robin sports scheduling. 4OR-Q J Oper Res. Available from: doi.org/10.1007/s10288-021-00472-3.

This is a preprint of an article that has been accepted for publication in Information Systems and Operational Research (INFOR), published by Taylor & Francis. The Version of Record of this article is available online at doi.org/10.1080/03155986.2021.1985902.