European Journal of Operational Research 307 (2023) 313–327

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Decision Support

# Multi-league sports scheduling with different leagues sizes

Miao Li [a,*], Morteza Davari [b], Dries Goossens [a,c]

[a] Department of Business Informatics and Operations Management, Ghent University, Tweekerkenstraat 2, Gent 9000, Belgium
[b] SKEMA Business School, Université Côte d'Azur, Avenue Willy Brandt, Lille 59777, France
[c] FlandersMake@UGent – Core Lab CVAMO, Ghent, Belgium

ARTICLE INFO

*Article history:*
Received 18 July 2021
Accepted 3 October 2022
Available online 12 October 2022

*Keywords:*
OR In sports
Multi-league timetabling
Venue capacity
Round robin
Home-away patterns

ABSTRACT

This paper introduces a general multi-league sports scheduling problem where timetables for multiple leagues must be determined simultaneously, a practical and challenging problem in amateur and youth sports. We consider round robin leagues with different numbers of teams, and hence also requiring different numbers of rounds. As the number of simultaneous home games that clubs can organize for their teams is limited by the capacity of their venue, the objective is to minimize capacity violations. Along with a mixed integer programming model which is formulated to optimize the starting round of each league as well as to settle when teams have their home games, we develop various methods to construct an initial solution, and a heuristic with several local search and perturbation components to improve on this. Extensive computational experiments reveal that our heuristic can efficiently provide high-quality solutions for artificial and realistic instances. We also illustrate the impact of using different sets of home-away patterns on the total venue capacity violations. Results on a real-life application from the Belgian national football association indicate that schedules based on our heuristic allow teams to substantially decrease their venue capacity without causing meaningfully more violations.



## 1. Introduction

Sports scheduling, i.e. deciding which match is to be played when, is an indispensable part of every sports competition. Various papers describe the scheduling of a single league as a highly challenging problem, which requires reconciling numerous constraints of various stakeholders using advanced methods (see Kendall, Knust, Ribeiro, & Urrutia, 2010 and Van Bulck, Goossens, Schönberger, & Guajardo, 2020 for an overview). In amateur and youth sports, however, the sports scheduling problem typically involves multiple leagues (also called divisions), which are categorized based on the age or skill of the players. This number of leagues may well be several hundreds, involving thousands of teams, and tens of thousands of matches to be scheduled. Moreover, since teams from the same club share the same venue, a capacity problem at each club results. Indeed, if a team is the home team for its match, its club should host the match, and the maximum number of matches a club can host simultaneously is limited by its number of terrains. We say that a club has a venue capacity violation if at some point in time, it has more teams that are scheduled a home game than it can accommodate. As the teams of a club play in different leagues, this capacity problem creates dependencies between the leagues.

Some studies in the literature mention a practical application which involves more than one league, typically a first and second professional division. Usually, however, these leagues are independent of each other (e.g. Durán, Durán, Marenco, Mascialino, & Rey, 2019), or scheduled independently (e.g.Bartsch, Drexl, & Kröger, 2006; Goossens, 2018). In the latter case, while their authors recognize that there are dependencies between the leagues with respect to the schedule (e.g. teams from different leagues sharing a venue), they apply a hierarchical procedure, where the more important league is scheduled first. Once the schedule of the top league is determined, the schedule of the other league is computed taking into account restrictions resulting from the first league's schedule. While this may be a reasonable way to deal with a couple of professional leagues where the interests of one league clearly dominate those of the other, it is not a viable approach to schedule hundreds of equivalent leagues.

In this paper, we investigate the simultaneous scheduling of multiple sport leagues in a way that clubs' capacity violations are minimized, based on a few crucial assumptions. We assume that each league plays a so-called round robin tournament, a popular format where each team plays each other team the same number of times (usually once at their home venue, and once at the venue of the opponent, i.e. a double round robin). Matches are grouped into rounds (typically weekends), noting that no team can play more than once per round. Furthermore, we assume that the matches of a league are scheduled using the minimal required

* Corresponding author.
*E-mail address:* miao.li@ugent.be (M. Li).

https://doi.org/10.1016/j.ejor.2022.10.010
0377-2217/

| | Rounds | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| $h_1$ | A | H | A | H | A | H | A | H | A | H |
| $h_2$ | A | H | H | A | H | H | A | A | H | A |
| $h_3$ | H | A | A | H | A | A | H | H | A | H |
| $h_4$ | A | H | A | H | H | H | A | H | A | A |
| $h_5$ | H | A | H | A | A | A | H | A | H | H |
| $h_6$ | H | A | H | A | H | A | H | A | H | A |

**Fig. 1.** A HAP set for a 2RR consisting of six teams.

number of rounds, i.e. according to a so-called compact schedule, and that leagues consist of a (possibly different) even number of teams. In this common setting, it does not happen that one or more teams of a league have no match (i.e. are 'bye'), while the other teams in the league play. Consequently, all teams in a league have played the same number of games at each point in the season, and have the same information on how many points they should still collect to achieve their goal, which is considered fair (Van Bulck & Goossens, 2020). This approach also makes it easier to postpone games in case of unforeseen events (see Yi, Goossens, & Talla Nobibon, 2020). Finally, we assume that leagues are played in consecutive rounds, without interruption (except for e.g. Christmas or national holidays, which apply to all leagues). Based on conversations with football, hockey and basketball federations in Belgium and the Netherlands, we believe these assumptions are in line with common practice.

To handle multi-league tournaments in practice, league organizers typically use the following approach. First, the teams are grouped into leagues. The main goal here is to minimize the resulting total travel distance while respecting competitive balance constraints and other considerations. This problem is known as the sports team grouping problem (Toffolo, Christiaens, Spieksma, & Berghe, 2019), or the sports team realignment problem (e.g. Saltzman & Bradford, 1996, Mitchell, 2003, Ji & Mitchell, 2005, Recalde, Severín, Torres, & Vaca, 2018). In order to avoid collusion and match-fixing, league organizers strive to compose leagues such that they have at most one team per club (e.g. Burrows & Tuffley, 2015; Durán, Guajardo, López, Marenco, & Zamorano, 2021; Schönberger, 2015). Second, the organizer decides on a starting round for each league, and next determines for each team on which round it plays a home game. This is reasonable, as the club's venue capacity (and possibly also availability) is the most important constraint in youth and amateur sports. It is done by assigning each team to a so-called Home-Away Pattern (HAP). A home-away pattern indicates for each round whether its team has a home game (H), or an away game (A). Fig. 1 illustrates a set of HAPs for a 2RR with six teams.

Third and finally, combining the HAP assignment with a compatible opponent table, which specifies each team's opponent for each round, the schedule follows. This is illustrated in Fig. 2. Note that not just any set of HAPs will allow a schedule. Despite being a well-researched topic (see Briskorn, 2008; Goossens & Spieksma, 2011; Horbach, 2010; Miyashiro, Iwasaki, & Matsui, 2003; Van Bulck & Goossens, 2020), the complexity of deciding whether a schedule exists for a given HAP set is still open. Several classes of HAP sets are known in the literature; we discuss some of the most important classes in Section 6.

As the computational complexity of this problem is insurmountable to practitioners, certainly for a realistic size, many turn to splitting up their clubs beforehand into smaller subsets (regions), creating artificial borders which turn out detrimental for finding good league compositions. Furthermore, they apply ad hoc and manual rescheduling of matches of those clubs that complain most about their capacity violations, which results in many matches played on unfavorable moments. Our paper attempts to provide a method to handle large-scale multi-league scheduling problems.

The focus lies on the second step in the scheduling process, taking the composition of the leagues as input. We tackle the problem of (i) determining for each league when it starts, and (ii) deciding for each team to which HAP it should be assigned. We assume that a feasible HAP set is given for each league, and that it allows an opponent schedule that offers an adequate solution to the third step. Indeed, compared to professional leagues, the order of the opponents or the precise round in which a particular match is scheduled is far less important in amateur and youth leagues. Regardless, solving the second step allows to further deal with the leagues independently, and if necessary to optimize to some extent which particular opponent is faced when. In any case, the third step has no impact on capacity violations. A detailed problem description is provided in Section 3.

We see the following key contributions of this work. First, we adopt a more general view on multi-league sports scheduling than what is common in the literature, in the sense that (i) we take into account that leagues may have different sizes (i.e. numbers of teams), and (ii) we optimize the starting round of each league (instead of considering it as given). We provide a mixed integer programming formulation as well as a lower bound for this problem. Second, we assess how different HAP sets affect the venue capacity violations under various round robin formats. Finally, we propose a heuristic that can handle hundreds of leagues, which we demonstrate in an elaborate computational study involving randomly generated as well as realistic instances. To the best of our knowledge, this is the first computational study that tackles problems of this (realistic) scale, as opposed to previous contributions typically handling at most half a dozen leagues. Moreover, based on a case study with data from the Royal Belgian Football Association, we obtain insights on how improved multi-league scheduling can help clubs to avoid excessive investments in venue capacity.

The remainder of this paper is organized as follows. Section 2 reviews the literature on venue capacity constraints and multi-league sports scheduling. A formal definition of the problem is given in Section 3, while Section 4 presents a mixed integer programming (MIP) formulation and a lower bound. Section 5 describes a tailored heuristic solution approach. Section 6 analyzes the impact of HAP sets, followed by an extensive computational experiment in Section 7 and a conclusion in Section 8.

## 2. Related literature

In single-league sport scheduling, a wide variety of constraints has been studied. The classification of sports scheduling problems by Van Bulck et al. (2020) mentions capacity constraints, defined as constraints that determine whether a team plays home or away or regulate the total number of games played by a set of teams, as one of five types of constraints and by far the most common (96% of the problems they collected from the literature involve capacity constraints). The typical setting is that each team has its own venue; capacity constraints manifest themselves through restricted availabilities of these venues. This has been studied for sports like baseball (e.g. Russell & Leung, 1994), Australian football (e.g. Kyngäs et al., 2017), table tennis (e.g. Schönberger, Mattfeld, & Kopfer, 2004), football (e.g. Rasmussen, 2008), or indoor football (e.g. Van Bulck, Goossens, & Spieksma, 2019). In some cases (e.g. Goossens & Spieksma, 2009, Della Croce & Oliveri, 2006), teams that share a stadium are reported, whose home games are then scheduled alternately. Alternatively, a league is scheduled using a limited number of venues, which are not linked to a particular team or contestant, known as the multiple venue sport schedul-

| Rounds | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 6 vs 1 | 1 vs 3 | 2 vs 4 | 1 vs 2 | 2 vs 3 | 1 vs 6 | 3 vs 1 | 1 vs 5 | 2 vs 1 | 3 vs 2 |
| 5 vs 2 | 2 vs 6 | 5 vs 1 | 3 vs 5 | 4 vs 1 | 2 vs 5 | 6 vs 2 | 4 vs 2 | 5 vs 3 | 1 vs 4 |
| 3 vs 4 | 4 vs 5 | 6 vs 3 | 4 vs 6 | 6 vs 5 | 4 vs 3 | 5 vs 4 | 3 vs 6 | 6 vs 4 | 5 vs 6 |

**Fig. 2.** A schedule consistent with the HAP set from Fig. 1 where team $i$ has been assigned to $h_i$.

**Table 1**
A comparison of the literature on multi-league sports scheduling.

| Reference | League sizes | No. leagues | No. teams | Objective | Schedule type | Sport (Practice) |
|---|---|---|---|---|---|---|
| de Werra et al. (1990) | different | 2 | 28 | balance breaks | Cpt, 2RR | basketball |
| Kendall (2008) | different | 4 | 92 | min. total travel distance | Cpt, 2RR | football |
| Moody et al. (2010) | different | 3 | 23 | min. congestion and breaks | Cpt\Rel, 1RR\2RR | general |
| Grabau (2012) | different | 8 | 64 | min. additional games | Rel | softball (✓) |
| Burrows & Tuffley (2015) | different | 2 | 22 | max. common fixtures | Cpt, 1RR\2RR | rugby (✓) |
| Schönberger (2015) | identical | 2 | 12 | min. congestion and breaks | Rel, 2RR | general |
| Schönberger (2017) | identical | 3 | 30 | min. rest day violations | Cpt\Rel, 2RR | table tennis |
| Davari et al. (2020) | identical | - | - | min. capacity violations | Cpt, 2RR | general |
| Durán et al. (2021) | different | 2 | 28 | balance travel distance | Cpt, 1RR | football (✓) |
| This study | different | 400 | 4040 | min. capacity violations | Cpt, 2RR | general |

***Note:*** ' - ' signifies the information is not specified. Cpt = compact schedule. Rel = relaxed schedule.

ing problem (see e.g. Urban & Russell, 2003). Capacity issues for this setting have been studied for e.g. table tennis and volleyball (Su, Chiu, & Cheng, 2013), and tennis (Della Croce, Tadei, & Asioli, 1999). An extreme case is the so-called asynchronous tournament, where there is only one venue available on which all matches must be scheduled (see e.g. Suksompong, 2016, Knust, 2008).

Some papers discuss a setting in which the league is split (a priori) into divisions (also called conferences), and teams play a number of interdivisional matches besides matches against opponents from their own division (e.g. Carlsson, Johansson, & Larson, 2017, de Werra (1985), Hoshino & Kawarabayashi, 2011, Larson & Johansson, 2014). More rare are so-called minitournaments or pod series (e.g. the Finnish ice hockey (Nurmi, Kyngäs, & Goossens, 2013), NCAA softball (Saur, Starr, Husted, & Newman, 2012)) where some matches of the regular season are played in one venue during one weekend, mainly for marketing reasons or to reduce travel costs. Since these tournaments are either considered to be a single competition (with one resulting final ranking), or multiple competitions that can be scheduled independently (with inter and intra divisional matches on separate parts of the season), we do not consider them multi-league sports scheduling. Consequently, they are not included in Table 1, which lists the most relevant papers on multi-league sports scheduling, together with several distinguishing properties.

Pioneering research on multi-league sports scheduling was done by de Werra, Jacot-Descombes, & Masson (1990), who use properties of oriented factorizations of complete graphs to construct a timetable for two leagues for an Australian basketball organization. Some pairs of teams in these leagues share the same facilities and cannot play home games simultaneously. Taking into account a number of constraints, the schedule should have its breaks (i.e. consecutive home games, or consecutive away games, for the same team) spread as evenly as possible among the teams.

Kendall (2008) studies how fixtures can be scheduled over holiday periods (i.e. Boxing Day and New Year's Day) for the four major football leagues (divisions) in England to minimize the total travel distance. Dependencies between the leagues arise from limits in certain cities to the number of teams that can have a home game on the same day, and pairs of teams from different leagues that should not play simultaneous home games. An algorithm based on depth-first search followed by a local search is developed, which can produce better schedules than those currently used. Note that, although the leagues have different sizes, only two rounds of the season are scheduled.

Moody, Kendall, & Bar-Noy (2010) create timetables for several youth divisions with teams sharing venues that have a capacity limit. Remarkably, although leagues may have different sizes, the season consists of a specific number of games to be played by all teams (regardless of the league size). The objective is minimizing fixture congestion and breaks, as well as some refereeing issues. A sample instance with three clubs that have teams in two leagues (with four and six teams), and eight matches to be played by all teams, is presented. The authors develop a two-phase method with the first phase based on the so-called tiling followed by a local search phase, but do not mention computational results.

Grabau (2012) investigates a scheduling problem arising in recreational youth softball. Teams do not have their own venue, instead, each match must be scheduled on one of 12 fields (with different properties). The leagues do not play according to a round robin format, however, each team should play at least a prescribed number of home (away) matches; the objective is to minimize the number of home (away) games scheduled in excess of that number to satisfy other intertwined scheduling rules. An integer programming (IP) approach is proposed to solve an instance with eight leagues and 64 teams.

Burrows & Tuffley (2015) consider a scheduling problem faced by the Rugby Union competition in New Zealand, involving a first and a second division, played as a double and single round robin tournament respectively. The objective is to maximize the number of so-called common fixtures, which occur when teams from two clubs play against each other in the same round in both divisions. The paper does not contain computational results, instead, it proves an upper bound on the number of common fixtures that can be attained, together with a scheduling procedure that achieves this bound.

Schönberger (2015) addresses the simultaneous scheduling of round robin leagues. This problem, called the *championship scheduling problem* involves two types of inter-league constraints, namely respecting limited venue capacities as well as ensuring player substitution opportunities among several teams of one club. The objective is to evenly distribute the teams' matches over the season, as well as to avoid consecutive breaks. A tiny instance with two leagues, each comprising six teams (from four clubs), turns out a tough nut to crack for the proposed MIP approach, which fails

to find an optimal solution within 15 minutes. In a follow-up paper, Schönberger (2017) studies a similar championship timetabling problem (with the same inter-league constraints) in the context of non-commercial table tennis. The objective is slightly different: each team specifies the number of rest days it wants at least between two consecutive matches, after which the number of rest day violations is minimized. The paper focuses on the generation of benchmark instances for a setting with three leagues of ten teams each, involving 20 clubs. Using a relatively simple Monte Carlo timetabling approach, the impact of various problem instance parameter settings (e.g. venue availability and the degree of overlap between the leagues) on the difficulty of the resulting problem instances are characterized.

Durán et al. (2021) contribute a practical implementation of scheduling six youth football leagues, corresponding to various age categories of professional clubs in Argentina. The main goal is to balance the differences in distance travelled between the teams from the same club, which is pursued through two mathematical programming methods based on regional team clustering, and an analysis of the distances between the club's home venues.

Davari, Goossens, Beliën, Lambers, & Spieksma (2020) present a multi-league scheduling problem that involves the assignment of teams to home-away patterns, with minimizing the total capacity violations at the clubs as the objective. They describe a polynomial-time exact algorithm (without implementing it) for the special case where each league has the same even number of teams. They also discuss the computational complexity of two generalizations (all teams from the same club must play according to the same HAP; clubs' capacities differ throughout the season). While their multi-league scheduling problem is similar to ours, our problem incorporates optimizing the starting round for each league and considers different league sizes. Furthermore, Davari et al. (2020) do not include problem instances or a computational study.

In conclusion, the literature has mostly dealt with competitions involving just a few leagues, playing a double round robin format according to a compact (Cpt) schedule. Multi-league research on relaxed (Rel) schedules, where (many) more rounds than needed are used, is rare, likely due to the fact that this setting is uncommon in amateur and youth sports. None of these studies involve computational experiments on large-scale instances, nor do they consider optimizing when leagues should start. Our contribution intends to close that gap.

## 3. Problem description

We are given a set $C$ of clubs and a set $L$ of leagues, each with an even number of teams. A given set $T$ of teams is partitioned once into leagues in $L$ and once into clubs in $C$: $\bar{T}_\ell$ indicates which teams belong to which league, and $\hat{T}_c$ indicates which teams belong to which club (clubs can have any number of teams). Leagues may have different sizes; the size of league $\ell \in L$ is denoted by $s_\ell = |\bar{T}_\ell|$.

All leagues play a $k$-round robin tournament ($k$RR), i.e. a tournament where each team faces each other team $k$ times. We define a set of rounds $R$: each round refers to a period (e.g. day or weekend) on which matches can take place, but no team can play more than once on a round. The number of rounds in $R$ depends on the largest league size $s_{\max} = \max_{\ell \in L} s_\ell$ as follows: $R = \{1, 2, \ldots, k(s_{\max} - 1)\}$. Each league plays according to a compact schedule, i.e. using the minimum number of rounds needed. In other words, given that the leagues have an even number of teams, either all teams of a league play on a round, or none. For each league $\ell$, a set $R_\ell \subseteq R$ is given with rounds on which it can start; $R_\ell$ only includes rounds that allow league $\ell$ to finish before the last round. We assume that, once started, leagues must be played in consecutive rounds, without interruption. Leagues can however overlap during the season, as illustrated in Fig. 3.

Each club $c \in C$ has a given capacity $\delta_c$ corresponding to the number of matches it can host per round. Venue capacity violations happen whenever for any club and any round the number of teams of the club that play home exceeds the capacity. The violation of a club in a round is measured by a scalar value that is zero if there is no violation, or otherwise equal to the number of teams of that club that play home in that round minus its capacity.

For each league, a feasible HAP set $\mathcal{H}_\ell$ is given, such that leagues of the same size have identical HAP sets. More specifically, a binary parameter $U_{h,q}$ indicates whether the $q$-th match of HAP $h \in \mathcal{H}_\ell$ is a home (H) game or an away (A) game.

The Multi-League Scheduling Problem (MLSP) consists of determining, for each league $\ell$, a starting round $r_\ell \in R_\ell$ as well as an assignment of its teams $t \in T_\ell$ to HAPs $h \in \mathcal{H}_\ell$, such that the sum of capacity violations over the clubs is minimized. While the complexity of MLSP is open, Davari et al. (2020) show that this problem is polynomially solvable if all leagues have the same even size, and hence all start at the first round.

## 4. Model formulation and lower bound

We introduce binary decision variables $x_{t,h}$ that are equal to one if team $t$ is assigned to HAP $h \in \mathcal{H}_\ell$ and zero otherwise, and $y_{\ell,s}$ which are equal to one if league $\ell$ starts at round $s$ and zero otherwise. We also use the following auxiliary variables: binary variables $u_{t,r}$ that are equal to one if team $t$ plays home in round $r$ and zero otherwise, and variables $v_{c,r}$ that represent the violation of club $c$ in round $r$. A mixed integer programming formulation is then given as follows.

$$\min \sum_{c \in C} \sum_{r \in R} v_{c,r}$$

s.t.

$$\sum_{t \in \bar{T}_\ell} x_{t,h} = 1 \quad \forall \ell \in L, h \in \mathcal{H}_\ell \tag{1}$$

$$\sum_{h \in \mathcal{H}_\ell} x_{t,h} = 1 \quad \forall \ell \in L, t \in \bar{T}_\ell \tag{2}$$

$$\sum_{s \in R_\ell} y_{\ell,s} = 1 \quad \forall \ell \in L \tag{3}$$

$$u_{t,r} \geq x_{t,h} + \sum_{s \in R_\ell : s < r} U_{h,r-(s-1)} y_{\ell,s} - 1 \quad \forall \ell \in L, t \in \bar{T}_\ell, h \in \mathcal{H}_\ell, r \in R \tag{4}$$

$$v_{c,r} \geq \sum_{t \in \hat{T}_c} u_{t,r} - \delta_c \quad \forall c \in C, r \in R \tag{5}$$

$$v_{c,r} \geq 0 \quad \forall c \in C, r \in R \tag{6}$$

$$x_{t,h} \in \{0, 1\} \quad \forall \ell \in L, t \in \bar{T}_\ell, h \in \mathcal{H}_\ell \tag{7}$$

$$u_{t,r} \in \{0, 1\} \quad \forall \ell \in L, t \in \bar{T}_\ell, r \in R \tag{8}$$

$$y_{\ell,s} \in \{0, 1\} \quad \forall \ell \in L, s \in R_\ell \tag{9}$$

The objective function minimizes the total capacity violations. Constraints (1) and (2) enforce that in each league, each team is assigned to a HAP, and each HAP is assigned to a team. Constraints

(3) ensure that each league starts on one of the specified rounds. Constraints (4) compute $u_{t,r}$, ensuring that if team $t$ (which is assigned to some HAP $h$) plays home in round $r$, then $u_{t,r} = 1$. Note that whether or not HAP $h$ has a home game in round $r$ depends on the starting round of its league. The violations of each club in each round follow from Constraints (5) and (6). Finally, constraints (7)–(9) define the domains of the variables.

In order to obtain a lower bound on the total capacity violations, we ignore the assignment of teams to HAPs. Instead, for each club, we try to distribute its team's home games evenly throughout the season, such that the total capacity violation is minimized. Let us introduce $\gamma_{cr}$ as the number of teams from club $c$ playing a home game in round $r$. The violation of club $c$ in round $r$ is thus computed by $\max\{0, \delta_c - \gamma_{cr}\}$. Let $\beta_{c,\ell}$ denote the number of teams in club $c$ playing in league $\ell$ and $\Gamma_c$ denote the total number of home games to be hosted by club $c$, i.e. $\Gamma_c = \sum_{t\in\hat{T}_c} k(s_\ell^t - 1)/2$, with $s_\ell^t$ the size of the league featuring team $t \in \hat{T}_c$. The following formulation computes $v_1^{LB}$, resulting in a lower bound for our problem.

$$v_1^{LB} = \min \sum_{c\in C}\sum_{r\in R} v_{c,r}$$

s.t. constraints (3),(6),(9),

$$v_{c,r} \geq \gamma_{c,r} - \delta_c \quad \forall c \in C, r \in R \tag{10}$$

$$\gamma_{c,r} \leq \sum_{\ell\in L}\beta_{c,\ell} \sum_{i=0}^{\min(k(s_\ell-1)-1,r-1)} y_{\ell,r-i} \quad \forall c \in C, r \in R \tag{11}$$

$$\sum_{r\in R}\gamma_{c,r} = \Gamma_c \quad \forall c \in C \tag{12}$$

$$\gamma_{c,r} \in \mathbb{Z}^+ \quad \forall c \in C, r \in R \tag{13}$$

Constraints (10) regulate the capacity violations. Constraints (11) enforce that no club can host more home games on a round than the number of teams playing on that round it has, while Constraints (12) ensure that each club accommodates the correct number of home games.

A lifted variant of $v_1^{LB}$ is given as follows:

$$v_2^{LB} = \min \sum_{c\in C}\sum_{r\in R} v_{c,r}$$

s.t. constraints (3),(6),(9),(10)-(13)

$$\sum_{r=1}^{\tau}\gamma_{c,r} \geq \sum_{\ell\in L_c}\sum_{r=1}^{\tau-k(s_\ell-1)+1} y_{\ell,r}\lceil k(s_\ell-1)/4\rceil + \sum_{\ell\in L_c}\sum_{r=1}^{\tau-k(s_\ell-1)/2+1} y_{\ell,r}\lfloor k(s_\ell-1)/4\rfloor \quad \forall c \in C, \forall \tau \in R \tag{14}$$

$$\sum_{r=\tau}^{|R|}\gamma_{c,r} \geq \sum_{\ell\in L_c}\sum_{r=\tau}^{|R|-k(s_\ell-1)/2+1} y_{\ell,r}\lceil k(s_\ell-1)/4\rceil + \sum_{\ell\in L_c}\sum_{r=\max(1,\tau-(k(s_\ell-1)/2))}^{|R|-k(s_\ell-1)/2+1} y_{\ell,r}\lfloor k(s_\ell-1)/4\rfloor \quad \forall c \in C, \forall \tau \in R \tag{15}$$

The additional constraints enforce that in each league, a team plays at least half of its games (rounded down) at home in each half of the tournament.

The following result shows that, under certain circumstances, $v_1^{LB}$ can be computed more easily.

**Proposition 1.** *If there exists a vector $\hat{\mathbf{y}}$ satisfying (3) such that*

$$\sum_{\ell\in L}\beta_{c,\ell}\sum_{i=0}^{\min(k(s_\ell-1)-1,r-1)} \hat{y}_{\ell,r-i} \geq \frac{\sum_{t\in\hat{T}_c} k(s_\ell^t-1)}{2|R|} \quad \forall c \in C, r \in R,$$

*then $v_1^{LB}$ can be computed using the following compact formula:*

$$v_c^{LB} = \sum_{c\in C}\max\left(\sum_{t\in\hat{T}_c} k(s_\ell^t-1)/2 - \delta_c|R|, 0\right). \tag{16}$$

The proof is given in Appendix A.

## 5. A heuristic approach

In this section, we propose a heuristic algorithm for MLSP. It starts with the construction of a feasible initial solution (Section 5.1). Then, in each iteration, we attempt to improve the solution by choosing a club and optimizing the capacity violation for that specific club (CLUB-OPT). To avoid repeated selection of the same club, we use a tabu list. Next, we choose a league (taking into account another tabu list), generate associated minimum weight bipartite matching problems for different starting rounds of the selected league (GEN-MATCH), and then optimally solve these problems (SOLVE-MATCH) (see Section 5.2). Solutions are evaluated based on their capacity violation over all clubs and rounds. After some iterations without improvement, we perform a more time-consuming edge removal sub-procedure (EDGE-REM) to improve the solution. Finally, we apply two perturbation techniques (see Section 5.3) whenever the procedure reaches a local optimum: starting round reset (SR-RESET) and assignment shuffle (AS-SHUFFLE).

The main framework of our heuristic is outlined in Algorithm 1, which involves the following notation. $S^{\text{in}}$, $S$, $S^*$, and $S^G$ represent the initial solution, the current solution, the periodic best solution, and the global best solution, respectively. The algorithm stops as soon as the time limit is reached. Also, EDGE-REM is called if $S^*$ has not been improved and EDGE-REM was not called in the last MAXER iterations. If $S^*$ has not been improved and perturbation was not applied in the last MAXPT iterations, one of the two perturbation procedures (SR-RESET and AS-SHUFFLE) is called.

### 5.1. Initial solution

As displayed in Algorithm 1, the first step is to construct an initial solution. A feasible solution consists of an assignment of HAPs to teams and a starting round for each league. To generate the initial solution, an initial assignment of HAPs to teams is obtained by one of the following three methods: the random method, the sequential assignment method, and the equal league size decomposition method. Each of these methods has two versions, depending on how the initial starting rounds are set. We compare the resulting six methods in terms of solution quality in Section 7.

**Random method ( RAN ):** A feasible assignment of teams to HAPs is randomly generated for each league. There are two versions of this method: $\text{Ran}_s$ and $\text{Ran}_d$. The starting round of each league $\ell$ is set to the earliest round in $R_\ell$ in the $\text{Ran}_s$ version, and randomly selected from $R_\ell$ in the $\text{Ran}_d$ version. This procedure is executed ten times with varying random seeds; the solution with the least total violation is selected as the initial solution.

**Sequential assignment method ( SEQ ):** The problem of assigning teams to HAPs for each single league can be cast to a minimum weighted bipartite matching. The set of teams and the set of HAPs form the two sets of vertices and there is an edge connecting each 'team' vertex to each 'HAP' vertex. Each edge connecting a 'team' vertex and a 'HAP' vertex has a non-negative cost, which is the incremental violation induced by assigning the team to the HAP,

**Algorithm 1** Pseudo code for the MLSP heuristic.

**Input:** An instance of MLSP
**Output:** Schedule $S^G$

Create an initial feasible solution $S^{\text{in}}$.
$S \leftarrow S^{\text{in}}$, OBJ$(S^*) \leftarrow \infty$, OBJ$(S^G) \leftarrow \infty$,
TABULIST-C $\leftarrow \varnothing$ and TABULIST-L $\leftarrow \varnothing$.

**while** the time limit is not reached **do**

  Pick a club $c \notin$ TABULIST-C; update TABULIST-C with $c$.
  $S \leftarrow$ CLUB-OPT$(S, c)$
  **if** OBJ$(S) <$ OBJ$(S^*)$ **then** $S^* \leftarrow S$

  Pick a league $\ell \notin$ TABULIST-L; update TABULIST-L with $\ell$.
  **for** $r \in R_\ell$ **do**
    $G \leftarrow$ GEN-MATCH$(S, \ell, r)$
    $M \leftarrow$ SOLVE-MATCH$(G)$
    Update $S$ based on $M$
    **if** OBJ$(S) <$ OBJ$(S^*)$ **then** $S^* \leftarrow S$
  **end for**

  **if** $S^*$ hasn't changed in the last MAXER iterations **then**
    **if** EDGE-REM wasn't called for MAXER iterations **then**
      $S \leftarrow$ EDGE-REM$(S^*)$;
      **if** OBJ$(S) <$ OBJ$(S^*)$ **then** $S^* \leftarrow S$
    **end if**
  **end if**

  **if** OBJ$(S^*) <$ OBJ$(S^G)$ **then** $S^G \leftarrow S^*$

  **if** $S^*$ hasn't changed in the last MAXPT iterations **then**
    **if** perturbation wasn't applied for MAXPT iterations **then**
      OBJ$(S^*) \leftarrow \infty$
      Choose one perturbation procedure from:
        1. $S \leftarrow$ SR-RESET$(S)$
        2. $S \leftarrow$ AS-SHUFFLE$(S)$
    **end if**
  **end if**

**end while**

given a (partial) assignment of teams to HAPs in other leagues. The Kuhn-Munkres algorithm (KMA) (Kuhn, 1956; Munkres, 1957) is a well-known algorithm that can be used to find a matching with minimum total weight in polynomial time. The detailed procedure of generating a bipartite matching instance corresponding to a league is elaborated in Section 5.2.2, however, here we use the method in a constructive way. We sequentially choose leagues (in a non-increasing order of their sizes, and randomly among leagues with the same size) and for each league, KMA is applied to assign teams to HAPs. We have a version (Seq$_s$) of this method where the earliest round in $R_\ell$ is selected as the starting round for league $\ell$, and a version (Seq$_d$) where we solve a bipartite matching instance for each round in $R_\ell$, and select the starting round which causes the least total violation for this league.

Note that using the club capacities $\delta_c$ to compute the incremental violation is not very meaningful in the early stages of this procedure, since most clubs would be well below their capacity (and hence the assignment of teams to HAPs would essentially be random). Therefore, each time after scheduling a league, we (dynamically) set the capacity of each club $c$ equal to half the number of its teams that have already been scheduled (rounded up), and at most $\delta_c$.

**Equal league size method ( EQU ):** The polynominal algorithm proposed by Davari et al. (2020) optimally assigns teams to HAPs for the case where all leagues have the same size (and start on the same round). Accordingly, we first decompose our instance into a number of sub-instances with equal league size. For each sub-instance, we construct the corresponding graph as described by Davari et al. (2020), for which we then construct an Euler circuit to find a 2-factorization. Finally we associate each 2-factor with a pair of complementary HAPs (i.e., patterns such that one has a home game each time the other has an away game, and vice versa), leading to an assignment of teams to HAPs for each sub-instance. One variant of this method (Equ$_s$) simply has each league starting at the earliest round in $R_\ell$. Another variant (Equ$_d$) traverses all possible starting rounds for the leagues after solving each sub-instance, and then chooses a starting round with the lowest capacity violation. Note that all leagues in the same sub-instance start at the same round.

### 5.2. Local search

Once an initial solution is created, it is improved by three local search components: the single-club and single-league optimization form a basic neighborhood search and are performed in every iteration, while the computationally more demanding edge removal is used only when certain conditions are met.

#### 5.2.1. Single-club optimization

The single-club optimization procedure starts by selecting a club, based on roulette wheel selection, which is then added to a tabu list. A club's selection probability is given by the ratio of the capacity violation at this club to the total capacity violation over all non-tabu clubs (clubs in the tabu list cannot be selected). Assuming an empty tabu list in the example of Fig. 4, clubs $c_3$, $c_4$, $c_6$, $c_7$, and $c_8$ will be selected with a probability of 3/15.

Each team $t$ belonging to the selected club is processed in random order. We swap its HAP and the HAP of each other team $t'$ in the same league. A move is accepted if it decrease the total capacity violations of the corresponding two clubs of $t$ and $t'$. Fig. 5 provides an example where $c_3$ is the selected club (consisting of teams $t_8$, $t_9$ and $t_{10}$) and $t_9$ is its team currently being processed. One of the moves considered is swapping the HAP of $t_9$ with the HAP of $t_{16}$, belonging to $c_6$. This move will be accepted if the resulting total capacity violation of clubs $c_3$ and $c_6$ is less than 6. This process is repeated until all teams from the selected club have been processed. Note that single-club optimization does not change the starting round of any league.

#### 5.2.2. Single-league optimization

The single-league optimization procedure starts with the selection of a league, where – similarly to the single-club optimization – we avoid repeated selection of the same leagues by using a combination of a tabu list and a roulette wheel selection approach. The probability for a league to be selected equals the total capacity violation of its associated clubs divided by the total violation over all non-tabu leagues if the league is not in the tabu list, and zero otherwise. In the example of Fig. 4 (and assuming an empty tabu list), the total violations of leagues $\ell_1$, $\ell_2$, $\ell_3$ and $\ell_4$ lead to selection probabilities 15/48, 15/48, 9/48 and 9/48, respectively.

We now describe how we optimize the HAP assignment of league $\ell$. For each starting round $r \in R_\ell$, an associated instance of the minimum weighted bipartite matching problem is generated as follows. We construct a bipartite graph $G$ with two disjoint sets of vertices: one set containing a vertex for each team $t \in \bar{T}_\ell$, the other a vertex for each HAP $h \in \mathcal{H}_\ell$. Its edge set $E$ contains an edge for each pair $(t, h) \in \bar{T}_\ell \times \mathcal{H}_\ell$.

Each edge is associated with a team, and as such with a club. If no club has multiple teams in the league, we compute for each edge $e \in E$ its weight taking into account the current allocation

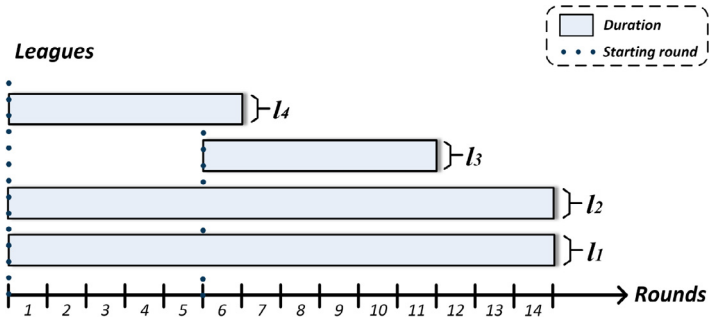


**Fig. 3.** A small example with four 2RR leagues of size 8, 8, 4, and 4, respectively, and $|R| = 14$.

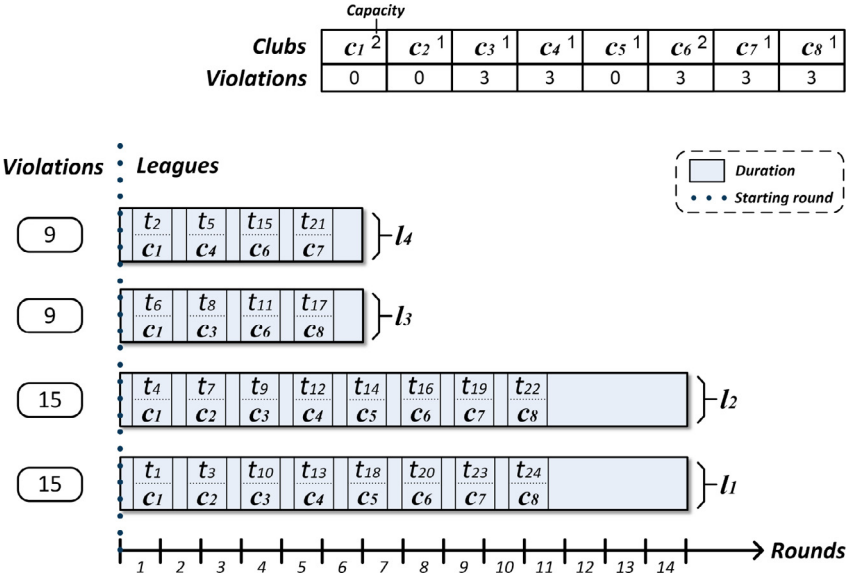


**Fig. 4.** An illustration of club and league selection on the example of Fig. 3..

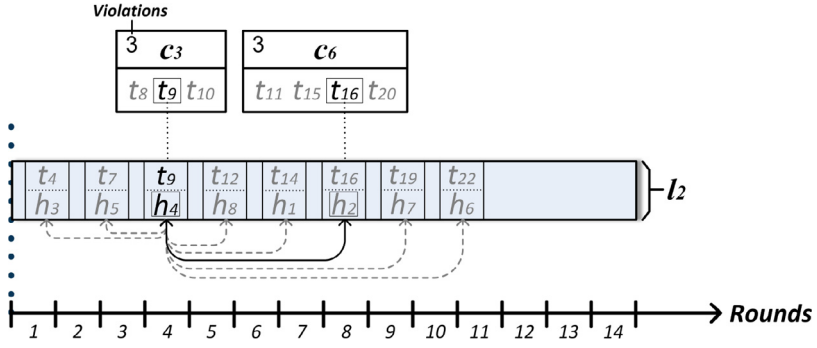


**Fig. 5.** An illustration of a HAP swap in the single-club optimization component for club $c_3$.

of the other teams of the edge's club. Consider the example of Fig. 6. Suppose that league $\ell_4$ is the selected league, which starts at round 2. The weights for the edge connecting team $t_{15}$ with HAP $h_{12}$ in the resulting bipartite graph (depicted in Fig. 7, left) are computed as follows. With team $t_{15}$ belonging to club $c_6$, whose capacity is two, we compute the total violation of $c_6$ assuming $t_{15}$ is assigned to HAP $h_{12}$, while considering the assignment of the other teams of club $c_6$ (i.e. $t_{11}$, $t_{16}$ and $t_{20}$) to their respective HAPs fixed as given in the current solution. For this example, the weight for the edge associated with $(t_{15}, h_{12})$ is two; all weights are given in the matrix in Fig. 7 (right). Note that for a different starting round, the resulting graph will only differ in its weights.

Once the selected league's associated matching problem is constructed for a certain starting round, we optimally solve it using the KMA, resulting in an assignment of teams to HAPs. For instance, the full lines in Fig. 7 (left) depict an optimal assignment of teams participating in league $\ell_4$ to HAPs in $\mathcal{H}_{\ell_4}$. Finally, the current solution is updated based on the new assignment of teams to HAPs and the new starting round, and the procedure is repeated for a next starting round in $R_\ell$.

In the unusual case that a club has more than one team in the same league, we propose to create a series of bipartite graphs, where each time the HAP assignment of all but one of these teams is fixed. In this way, the cost of the remaining team no longer depends on the outcome of the matching problem. After solving each

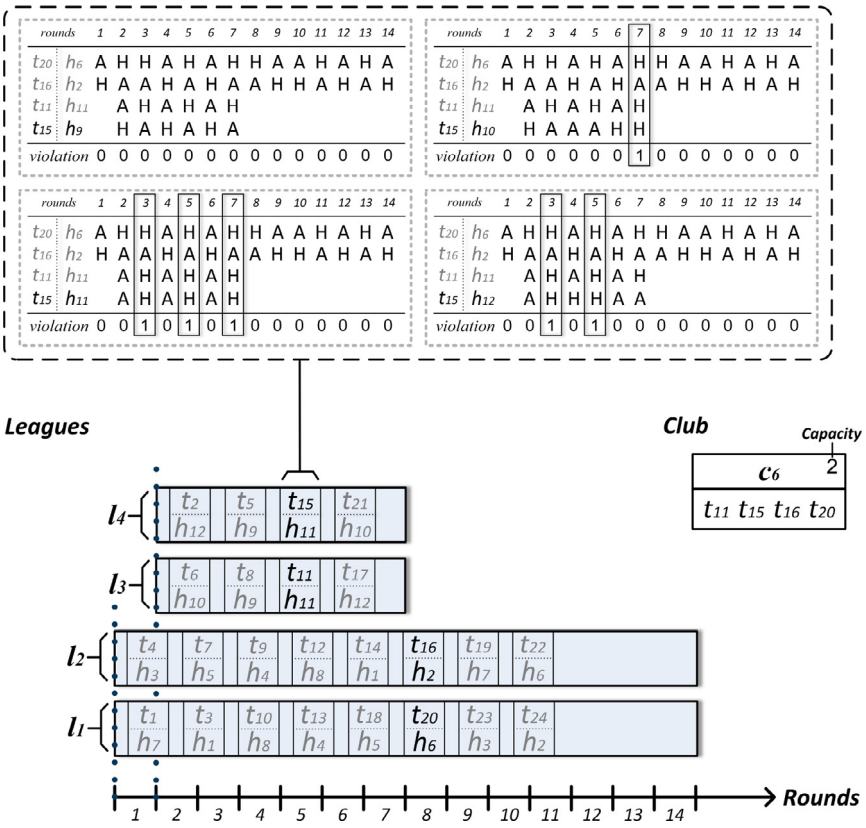


**Fig. 6.** An illustration of cost computation on edge weights for the case where clubs have at most one team per league.

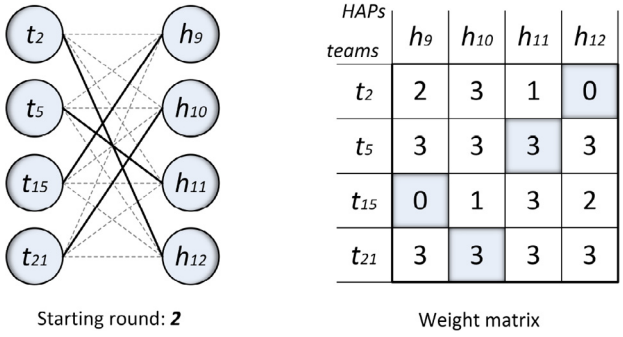


**Fig. 7.** A perfect matching of assignment for league $\ell_4$, with starting round 2.

of these weighted matching problems, the HAP assignment with the lowest violation is retained.

### 5.2.3. Edge removal

The edge removal (EDGE-REM) procedure tries to improve a given input solution ($S^{\text{in}}$) via iteratively generating and solving minimum weighted bipartite matching instances. Algoritm 2 depicts details of this procedure. Contrary to the single-league optimization procedure, the idea is now to consecutively solve these matching instances for a series of leagues $\ell \in L$. Leagues are sorted in a non-increasing order of their associated violations, based on the input solution $S^{\text{in}}$, into a list $L_v$, and then selected one by one in that order. The matching instances are constructed as described in Section 5.2.2, however, one instance is generated for each team $t \in \bar{T}_\ell$, where the edge corresponding to team $t$'s assignment in the current solution is removed. If team $t$ belongs to a club with multiple teams in this league, we remove edges between each team from that club in this league and the HAP that was assigned to $t$.

**Algorithm 2** EDGE-REM procedure.

**Input:** $S^{\text{in}}$
**Output:** $S^{\text{out}}$

$S \leftarrow S^{\text{in}}$, $S^{\text{out}} \leftarrow S^{\text{in}}$
Create a sorted list $L_v$ of leagues
**for** $\ell \in L_v$ **do**
  $r$: the starting round of league $\ell$ in solution $S$.
  **for** $t \in \bar{T}_\ell$ **do**
    $h$: the assigned HAP for team $t$ in solution $S$.
    $G \leftarrow$ GEN-MATCH$(S, \ell, r)$
    $G' \leftarrow$ REMOVE-EDGE$(G, (t, h))$
    $M \leftarrow$ SOLVE-MATCH$(G')$
    Update $S$ based on $M$
    **if** OBJ$(S) \leqslant$ OBJ$(S^{\text{out}})$ **then**
      $S^{\text{out}} \leftarrow S$.
    **end if**
  **end for**
  **if** OBJ$(S^{\text{out}}) <$ OBJ$(S^{\text{in}})$ **then**
    **break**
  **end if**
**end for**

We use KMA to solve each matching instance, and update the current solution accordingly. If after processing any league, a better solution than the input solution is obtained, the procedure terminates and returns this improved solution. If the procedure does not manage to improve the input solution, it will return the last encountered solution with the same objective value as that of the input solution. As this solution will likely be different from the input solution, EDGE-REM also contributes to the diversification in

the search process. Finally, observe that the starting rounds of the leagues remain unchanged in this procedure.

### 5.3. Perturbation

In order to improve the chance of escaping local optima, two perturbation techniques, namely *starting round reset* and *assignment shuffle*, are introduced.

#### 5.3.1. Starting round reset

Since the starting round of a league can only be changed in the single-league optimization phase, where the starting round of all other leagues is fixed, it may be difficult for the heuristic to escape a local optimum. The purpose of starting round reset (SR-RESET) is to randomly perturb the starting round for a subset of the leagues. Once its conditions are met, SR-RESET will randomly select a starting round for each of the selected leagues, different from its starting round in the current solution (unless there is only one possibility).

#### 5.3.2. Assignment shuffle

The assignment shuffle (AS-SHUFFLE) also randomly selects a subset of the leagues. Each selected league will see its teams randomly reassigned to HAPs, while its starting round remains unchanged. Hence, for each of these leagues, the current HAP assignment will be ruined and replaced with a possibly quite different one, enabling to escape local optima.

## 6. The impact of HAP sets on capacity violation

Our problem assumes that for each league, a (feasible) HAP set is given. In this section, we attempt to gain some understanding of how these given HAP sets impact the resulting capacity violation.

HAP sets are typically studied from a single round robin perspective (e.g. Knust & Lücking, 2009, De Werra, 1980, Lambers, Goossens, & Spieksma, 2022). A critical property of a HAP is the occurrence of so-called *breaks*. If a team has two consecutive home games, or two consecutive away games, in rounds $r$ and $r+1$, it is said to have a break in round $r+1$. If all HAPs include at most one break, the HAP set is a *single-break HAP set*. Breaks are generally perceived as undesirable, which is a reason why many sports leagues make use of this type of HAP set (see e.g. Goossens & Spieksma, 2012 for football). Furthermore, De Werra (1981) shows that a feasible single-break HAP set exists for any $2n$ teams. Consequently, we focus on single-break HAP sets.

Any single-break HAP set for $2n$ teams can be characterized by the rounds $r_1, r_2, \ldots, r_n$ in which a break occurs for at least one of the HAPs in the set. We define the difference between consecutive rounds with breaks as $d_i = r_{i+1} - r_i \;\; (i = 1, \ldots, n)$ where $r_{n+1} \equiv r_1 + 2n - 1$. Hence, the sequence $D = (d_1, d_2, \ldots, d_n)$ represents the break-gaps, i.e. the number of rounds between two consecutive breaks. This representation of a feasible, single-break HAP set is referred as the $D$-notation (Hansen, 1981; Knust & Lücking, 2009). While each HAP set uniquely corresponds to a $D$-sequence, the opposite is not true. For any $2n$ teams, $2n-1$ distinct HAP sets correspond with the same feasible $D$-notation or cyclic permutation of it. These HAP sets are defined by the round on which the $D$-notation starts, the so-called *base round*. For example, in the case of $2n = 8$, and a family of HAP sets given by $D = (2, 2, 2, 1)$, breaks occur in rounds 1, 3, 5 and 7 ($r_1 = 1$, $r_2 = 3$, $r_3 = 5$, $r_4 = 7$) with base round 1. The corresponding HAP set for 1RR is given in the left half of Fig. 8. Alternatively, for base round 5, the corresponding HAP set has breaks in rounds 2, 4, 5, and 7 ($r_1 = 5$, $r_2 = 7$, $r_3 = 2$, $r_4 = 4$), see Fig. 8 (right). For the 2RR tournament, the second half of each HAP corresponds to its first half with reversed home-away status.

**Table 2**
Capacity violations for HAP set combinations from the canonical family under 2RR (1RR).

| $D$-notation | 2221 | | | | | | |
|---|---|---|---|---|---|---|---|
| 22222221 | 8-C-1 | 8-C-2 | 8-C-3 | 8-C-4 | 8-C-5 | 8-C-6 | 8-C-7 |
| 16-C-1 | **21** (**0**) | 23 (2) | 23 (2) | 23 (4) | 23 (2) | 23 (2) | 21 (0) |
| 16-C-2 | 21 (0) | **21** (**0**) | 23 (2) | 23 (2) | 23 (4) | 23 (2) | 23 (2) |
| 16-C-3 | 21 (0) | 21 (0) | **23** (**1**) | 23 (2) | 23 (3) | 23 (2) | 23 (2) |
| 16-C-4 | 22 (0) | 22 (0) | 24 (1) | **22** (**1**) | 24 (3) | 22 (2) | 24 (2) |
| 16-C-5 | 22 (0) | 22 (1) | 24 (1) | 22 (1) | **24** (**2**) | 22 (2) | 24 (2) |
| 16-C-6 | 22 (1) | 22 (1) | 24 (2) | 22 (1) | 24 (2) | **22** (**1**) | 24 (2) |
| 16-C-7 | 22 (1) | 22 (0) | 24 (2) | 22 (2) | 24 (2) | 22 (1) | **24** (**1**) |
| 16-C-8 | 24 (0) | 24 (0) | 26 (2) | 24 (2) | 26 (3) | 24 (1) | 26 (1) |
| 16-C-9 | **24** (**0**) | 24 (0) | 26 (2) | 24 (1) | 26 (3) | 24 (1) | 26 (2) |
| 16-C-10 | 24 (0) | **24** (**0**) | 26 (1) | 24 (1) | 26 (3) | 24 (2) | 26 (2) |
| 16-C-11 | 22 (0) | 22 (1) | **24** (**1**) | 22 (1) | 24 (2) | 22 (2) | 24 (2) |
| 16-C-12 | 22 (1) | 22 (1) | 24 (2) | **22** (**1**) | 24 (2) | 22 (1) | 24 (2) |
| 16-C-13 | 22 (1) | 22 (0) | 24 (2) | 22 (2) | **24** (**2**) | 22 (1) | 24 (1) |
| 16-C-14 | 22 (0) | 22 (0) | 24 (2) | 22 (2) | 24 (3) | **22** (**1**) | 24 (1) |
| 16-C-15 | 21 (0) | 21 (0) | 23 (2) | 23 (2) | 23 (3) | 23 (2) | **23** (**1**) |

Knust & Lücking (2009) count the number of feasible $D$-sequences for up to 20 teams. Two families of HAP sets are of special interest. One is the so-called *canonical* HAP set, whose $D$-notation is $D = (2, 2, \ldots, 2, 1)$. Despite its flaws (Lambers et al., 2022; Lambrechts, Ficker, Goossens, & Spieksma, 2017), the canonical HAP set is reportedly the most popular single-break HAP set in practice (see e.g. Goossens & Spieksma (2012)). The other is the *flexible* HAP set; its $D$-notation is $D = (3, 1, 2, \ldots, 2, 1)$. This HAP set allows the largest diversity of the schedules that are compatible with it. More in particular, when counting for each match on how many different rounds it could be scheduled based on the HAP set, and then summing this over all matches, one obtains the so-called *spread* of the HAP set (Lambers et al., 2022). Lambers et al. (2022) show that the flexible HAP set yields the highest spread among the single-break HAP sets, offering more options to tackle other scheduling constraints (besides venue capacity violations).

We illustrate the impact of these HAP sets on capacity violations by means of a simple instance containing a (large) league with 16 teams and two (small) leagues with 8 teams. Hence, the round set is $R = \{1, 2, \ldots, 30\}$; for the small leagues, we take $R_\ell = \{1, 2, \ldots, 17\}$ as possible starting rounds. There are 16 clubs; each club consists of two teams, one playing in the large league, one in a small league. Each club has a venue capacity of one. We consider all combinations of HAP sets from the canonical (C) and flexible (F) families, however, using the same HAP set for the small leagues. For each combination, the capacity violations are computed using the MIP model described in Section 4, solved to optimality with CPLEX (see Section 7 for implementation details; the computation time was negligible). The results are summarized in Tables 2 andTables A.1–A.3 (see Appendix), where HAP sets are denoted with their league size, family (C or F), and base round, respectively. The table gives the capacity violation for each combination for a 2RR tournament, and a 1RR tournament (between parentheses).

As can be seen, the smallest capacity violation (21) in the 2RR setting is obtained when using a canonical HAP set for the large league, both for canonical (Table 2) and flexible (Table A.1) HAP sets in the small leagues. Similarly, a setting without violations can be obtained in the 1RR setting, when the canonical HAP set is used for the small leagues (Tables 2 and A.2). In other words, using canonical HAP sets in all leagues allows the smallest violation in both 1RR and 2RR tournaments. The tables also illustrate the importance of the base round. Let's call HAP sets *in-line* if they have all their break-gaps in the same rounds (assuming that they start simultaneously). The in-line HAP sets are indicated in bold in the tables; note that only HAP sets of the same family are in-line. An

| | Rounds | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| $h_1$ | **H** | A | H | A | H | A | **H** |
| $h_2$ | H | **A** | **A** | H | A | H | A |
| $h_3$ | H | A | H | **A** | **A** | H | A |
| $h_4$ | H | A | H | A | H | **A** | **A** |
| $h_5$ | **A** | H | A | H | A | H | **A** |
| $h_6$ | A | **H** | **H** | A | H | A | H |
| $h_7$ | A | H | A | **H** | **H** | A | H |
| $h_8$ | A | H | A | H | A | **H** | **H** |

| | Rounds | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| $h_1$ | A | H | A | **H** | **H** | A | H |
| $h_2$ | H | A | H | A | H | **A** | **A** |
| $h_3$ | **A** | **A** | H | A | H | A | H |
| $h_4$ | A | H | **A** | **A** | H | A | H |
| $h_5$ | H | A | H | **A** | **A** | H | A |
| $h_6$ | A | H | A | H | A | **H** | **H** |
| $h_7$ | **H** | **H** | A | H | A | H | A |
| $h_8$ | H | A | **H** | **H** | A | H | A |

**Fig. 8.** Two canonical HAP sets with distinct base rounds for a 1RR consisting of eight teams.

**Table 3**
Overview of instance types and their properties.

| Instance type | Instance ID | No. leagues | League sizes | | | | | | | No. clubs | No. teams |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 16 | 14 | 12 | 10 | 8 | 6 | 4 | | |
| Small-scale | 3-1 | 3 | - | 1 | - | - | 1 | 1 | - | 14 | 28 |
| | 3-2 | 3 | 1 | - | - | - | 1 | 1 | - | 16 | 30 |
| | 5-1 | 5 | - | - | - | 1 | 1 | - | 3 | 10 | 30 |
| | 5-2 | 5 | - | - | - | - | 2 | 3 | - | 9 | 34 |
| | 10-1 | 10 | - | - | - | 1 | 1 | - | 8 | 12 | 50 |
| | 10-2 | 10 | - | - | - | - | 4 | - | 6 | 12 | 56 |
| Medium-scale | 25-1 | 25 | 3 | - | 5 | - | 8 | - | 9 | 20 | 208 |
| | 25-2 | 25 | 7 | - | 8 | - | 5 | - | 5 | 40 | 268 |
| | 50-1 | 50 | 12 | - | 14 | - | 24 | - | - | 40 | 552 |
| | 50-2 | 50 | 14 | - | 16 | - | 20 | - | - | 60 | 576 |
| | 75-1 | 75 | 15 | - | 25 | - | 35 | - | - | 60 | 820 |
| | 75-2 | 75 | 20 | - | 25 | - | 30 | - | - | 80 | 860 |
| Large-scale | 100-1 | 100 | 25 | - | 30 | - | 45 | - | - | 50 | 1120 |
| | 100-2 | 100 | 30 | - | 30 | - | 40 | - | - | 100 | 1160 |
| | 200-1 | 200 | 30 | - | 30 | 50 | 90 | - | - | 120 | 2060 |
| | 200-2 | 200 | 35 | - | 35 | 55 | 75 | - | - | 150 | 2130 |
| | 400-1 | 400 | 40 | - | 50 | 80 | 230 | - | - | 250 | 3880 |
| | 400-2 | 400 | 50 | - | 60 | 100 | 190 | - | - | 300 | 4040 |
| Real-life | RBFA | 328 | 29 | 6 | - | 2 | 278 | 13 | - | 370 | 2870 |

***Note:*** ' - ' indicates the instance type does not contain leagues of the corresponding size.

interesting finding is that an optimal solution can be found among in-line HAP sets. This is intuitive, because HAP sets with coinciding breaks make it easier to balance the venue usage of teams from the same club. Still, having in-line HAP sets is not sufficient for obtaining an optimal solution, not even if they have the same base round. In conclusion, the in-line combination (16-C-1, 8-C-1) yields a minimum capacity violation for this particular multi-league problem, which motivates us to use identical canonical HAP sets '2$n$-C-1' for all leagues in the computational experiments in the next section.

## 7. Computational study

In this section, we report the results of computational experiments with the MIP model and the MLSP heuristic, which are tested on a range of problem instances. The instance generation and the parameter configuration are described in Section 7.1. The performance of six initial solution methods and various components of the heuristic are analyzed in Section 7.2, where we compare the proposed MLSP heuristic against our mathematical model. Moreover, an application with real-life data from the Royal Belgian Football Association is discussed in Section 7.3. All experiments were executed on a personal computer with an Intel Core i7-10850H CPU, 2.70 GHz processor and 16.0 GB of internal memory. The MLSP heuristic is implemented in C++, and we use an $O(n^3)$ implementation of the Kuhn-Munkres algorithm (King, 2009). Our MIP model is implemented using CPLEX 20.1.0 within a C++ environment, using a time limit of 3600 seconds. We consider two settings: the default setting (CPLEX[1]), and an enhanced setting (CPLEX[2]) with emphasis on finding hidden feasible solutions and branching on the binary variables $y_{l,s}$ first. CPLEX[2] also starts using a solution polishing heuristic after 1200 seconds.

### 7.1. Instances and parameters

As discussed in Section 2, the literature includes a very limited set of instances, usually based on competitions with few leagues. The test case generation scheme by Schönberger (2017) is a notable exception, however, the instances generated there focus on a time-relaxed setting (i.e., leagues are played over many more rounds than strictly necessary), and include no more than three leagues. Hence, we developed an instance generator and created 18 sets of artificial instances, with a number of leagues ranging from 3 to 400, a number of clubs between 9 and 300, and a number of teams up to 4040. Each artificial instance type includes ten problem instances, all publicly available (see https://www.sportscheduling.ugent.be/RobinX/multiLeagueRepo.php); an overview of the instance types is given in Table 3. The table also mentions one real-life instance, for which we refer to Section 7.3.

The instance generator takes the number of teams ($|T|$), number of clubs ($|C|$), number of leagues, and the frequency of the league sizes as input parameters. Given the resulting values for $s_{\max}$ and $s_l$, we set the possible starting rounds for each league $\ell$ as $R_\ell = \{1, 2, \ldots, k(s_{\max} - s_\ell) + 1\}$. Next, our generator will determine the number of teams $|\hat{T}_c|$ for each club $c$, as an integer chosen randomly from the range $\lfloor(1-0.3)\times|T|/|C|\rfloor.\lfloor(1+0.3)\times|T|/|C|\rfloor$. If $\sum_{c\in C}|\hat{T}_c| > |T|$, we repeatedly and randomly choose clubs and reduce their sizes by one unit, until $\sum_{c\in C}|\hat{T}_c| = |T|$. Note that clubs can be chosen more than once in this stage. If

**Table 4**
Performance of the initial solution methods on small, medium, and large-scale instances.

| Instance type | $Ran_s$ | $Seq_s$ | $Equ_s$ | $Ran_d$ | $Seq_d$ | $Equ_d$ |
|---|---|---|---|---|---|---|
| 3-1 | 25.1 | 18.6 | 35.8 | 25.1 | 17.1 | 22.8 |
| 3-2 | 24.4 | 14.3 | 34.4 | 24.7 | 17.9 | 20.8 |
| 5-1 | 30.2 | 24.8 | 39.6 | 26.6 | 23.1 | 25.2 |
| 5-2 | 43.9 | 36.3 | 36.1 | 38.8 | 34.3 | 35.7 |
| 10-1 | 31.1 | 27.0 | 32.9 | 23.8 | 23.2 | 24.2 |
| 10-2 | 63.3 | 52.0 | 53.1 | 60.0 | 51.3 | 51.3 |
| 25-1 | 268.0 | 188.5 | 196.4 | 217.1 | 164.2 | 122.0 |
| 25-2 | 583.3 | 484.7 | 493.0 | 548.3 | 482.2 | 452.7 |
| 50-1 | 665.2 | 463.4 | 472.4 | 478.7 | 359.9 | 290.4 |
| 50-2 | 1003.7 | 788.6 | 802.2 | 899.3 | 720.1 | 663.4 |
| 75-1 | 912.1 | 632.9 | 644.9 | 661.0 | 508.8 | 415.8 |
| 75-2 | 1375.1 | 1048.3 | 1068.6 | 1171.0 | 910.5 | 854.3 |
| 100-1 | 853.1 | 550.2 | 556.3 | 529.8 | 404.2 | 288.5 |
| 100-2 | 1661.5 | 1226.4 | 1239.6 | 1429.0 | 1041.8 | 959.8 |
| 200-1 | 1862.5 | 1242.7 | 1272.0 | 1082.6 | 614.4 | 413.6 |
| 200-2 | 2318.5 | 1627.4 | 1679.3 | 1609.7 | 973.1 | 685.0 |
| 400-1 | 3857.5 | 2732.1 | 2808.9 | 2079.3 | 1172.8 | 1134.5 |
| 400-2 | 4552.6 | 3257.4 | 3353.9 | 2955.4 | 1740.4 | 1354.3 |

$\sum_{c\in C}|\hat{T}_c| < |T|$, we increase the club sizes similarly until $\sum_{c\in C}|\hat{T}_c| = |T|$. The club capacities are randomly chosen integer values in the range $\max(\lfloor|\hat{T}_c|/2\rfloor - 2, 1).\min(\lfloor|\hat{T}_c|/2\rfloor + 2, |\hat{T}_c|)$. Finally, the teams of the clubs are allocated to leagues according to the following procedure. We define an *eligible* club as a club that includes at least one non-allocated team. A league is *eligible* if it includes at least one non-allocated position. We repeatedly and randomly choose an eligible club $c$ and an eligible league $\ell$ and allocate a random non-allocated team $t$ from $c$ to $\ell$. We want to obtain a league composition where no club has multiple teams in the same league. Therefore, if there another team from club $c$ was already assigned to league $\ell$, we immediately apply a fixing sub-procedure: we repeatedly and randomly choose a league $\ell'$, none of whose assigned teams (so far) belong to $c$, and we loop over its assigned teams (denoted by $t'$) and swap the allocation of $t$ and $t'$ (i.e., team $t$ to league $\ell'$ and $t'$ to $\ell$). Note that such a swap might not be feasible if another team of the club that contains $t'$ was assigned to $\ell$; in the unlikely case when no feasible swap can be found, we backtrack. As soon as a feasible allocation is reached, the fixing sub-procedure halts.

Based on preliminary experiments with parameter configurations (not reported in this paper), we opted for the following settings in the MLSP heuristic. We set MAXER = 3, and MAXPT = 5 to determine how frequently the edge removal and the perturbation procedure are to be applied, respectively. For each instance type, the length of TABULIST-C (TABULIST-L) is set to 20% (40%) of the total number of clubs (leagues) involved. Both perturbation operators are performed on a subset that includes half of all leagues in the problem instance (randomly selected), and have an equal chance of being selected. For each league, we used the so-called canonical HAP set (for details, see Section 6).

### 7.2. Results

We evaluate the performance of the MIP model, the methods to generate an initial solution, and the proposed MLSP heuristic. The average computational results of ten problem instances per instance type (organized into small, medium, and large scale) are reported in Table 4. The first column gives the instance type, the second, third and forth columns give the average objective value obtained using the random method, the sequential assignment method, and the equal league size method, where all leagues start as early as possible (i.e., on the first round in $R_\ell$). The next three columns report on the same methods, however, this time considering each possible starting round. The computation time of all six initial solution methods is negligible. In Table 4, it is clear that considering all possible starting rounds offers an advantage over taking the earliest starting round for each league. Particularly for the medium and large instances, the $Equ_d$ method greatly reduces the number of capacity violations compared to the other five methods. We noticed however that MLSP obtains slightly better results with initial solutions based on the $Ran_s$ rather than $Equ_d$ (based on preliminary results not reported in this text), likely because it is more difficult for MLSP to escape a local optimum where the starting rounds are optimized. Looking at the smallest instances, the equal league size method performs worse than the sequential assignment method or even the random method. This is explained by the fact that these small instances have too few leagues of the same size, (involving just a few clubs with multiple teams in these leagues) for this method to be effective.

**Table 5**
Algorithmic performance on small-scale instances.

| Instance type | CPLEX[1] | | MLSP | | $v_1^{LB}$ |
|---|---|---|---|---|---|
| | Obj | Time (*s*) | Obj | Time (*s*) | |
| 3-1 | 7.0 | 66.33 | 7.0 | 0.02 | 0.0 |
| 3-2 | 5.0 | 16.55 | 5.0 | 0.11 | 0.0 |
| 5-1 | 10.6 | 588.10 | 10.6 | 0.79 | 5.5 |
| 5-2 | 34.2 | 803.71 | 34.2 | 0.01 | 34.2 |
| 10-1 | 9.0 | 678.28 | 9.0 | 3.97 | 9.0 |
| 10-2 | 44.6 | 528.63 | 44.6 | 0.93 | 44.6 |
| *Avg.* | | 446.93 | | 0.97 | |

**Table 6**
Results of the MLSP heuristic with different time limits on medium-scale instances.

| Instance type | Time limit (*s*) | | | | | |
|---|---|---|---|---|---|---|
| | 30 | 60 | 150 | 300 | 450 | 600 |
| 25-1 | 83.9 | 83.6 | 83.6 | 83.5 | 83.5 | 83.5 |
| 25-2 | 416.2 | 415.9 | 415.4 | 415.4 | 415.2 | 414.9 |
| 50-1 | 138.7 | 135.8 | 135.2 | 134.1 | 133.9 | 133.4 |
| 50-2 | 546.8 | 545.5 | 544.2 | 543.1 | 542.9 | 542.5 |
| 75-1 | 201.6 | 199.1 | 197.6 | 197.1 | 196.5 | 196.4 |
| 75-2 | 630.6 | 626.1 | 623.0 | 620.6 | 619.8 | 619.3 |

We now test the quality of heuristic solutions on small-scale instances against the optimal solutions obtained with the MIP model and the lower bound $v_1^{LB}$ (which equals $v_2^{LB}$ and $v_c^{LB}$ for all instances in this section). Table 5 exhibits the objective function value and the computation time (in seconds) averaged over 10 instances. The MIP model was solved to optimality using the default CPLEX settings; the MLSP heuristic made use of the initial solution generated by the $Ran_s$ method. Note that the MLSP heuristic used its allotted 60 seconds of computation time, but the table reports the computation time after which the optimal solution was reached. The final column in Table 5 offers the lower bound. The MLSP heuristic is able to produce the same optimal solutions found by CPLEX for all 60 problem instances, in a fraction of the time it takes CPLEX (to prove optimality). Besides, except for some of the small instance types, the lower bound $v_1^{LB}$ equals the optimal objective value.

For medium scale and large scale instance, CPLEX cannot obtain optimal solutions within one hour for any of the 120 instances, and 'out of memory' errors frequently occur with the largest instances. We have also generated and tested instances with only 1RR leagues. Despite the number of rounds being only half what a 2RR tournament needs, these instances still could not be solved to optimality within an hour.

We experimented with various time limits for the MLSP heuristic; the results are given in Tables 6 and Table 7. As expected, medium scale instances converge much more rapidly than large scale instances. In view of these results, we opted for a time limit

**Table 7**
Results of the MLSP heuristic with different time limits on large-scale instances.

| Instance type | Time limit (s) | | | | | |
|---|---|---|---|---|---|---|
| | 60 | 120 | 300 | 600 | 900 | 1200 |
| 100-1 | 48.9 | 47.1 | 46.1 | 43.4 | 43.1 | 42.7 |
| 100-2 | 720.5 | 715.3 | 713.1 | 709.2 | 708.7 | 708.3 |
| 200-1 | 91.2 | 90.1 | 85.7 | 83.4 | 81.4 | 80.4 |
| 200-2 | 362.9 | 359.0 | 354.6 | 350.7 | 349.0 | 347.2 |
| 400-1 | 235.3 | 209.3 | 200.8 | 196.1 | 193.0 | 191.7 |
| 400-2 | 724.2 | 655.6 | 629.4 | 621.8 | 615.9 | 614.1 |

of 450 seconds for medium scale instances, and 900 seconds for large scale instances.

The detailed results of the MIP model, the lower bound and the proposed MLSP heuristic on medium-scale and large-scale instances are summarized in Table 8. CPLEX[1] and CPLEX[2] indicate the average objective value and the best bound $v^{BB}$ obtained with the MIP model after one hour of computation time. The $v_1^{LB}$ column offers the lower bound calculated via (16). The MLSP column gives the average objective value obtained with the MLSP heuristic based on the initial solution generated with the $\text{Ran}_s$ method, and given a computation time of 450 seconds for the medium instances and 900 seconds for the large instances. The last column shows the improvement percentage of MLSP compared to the MIP solution found with CPLEX[2] (i.e., $\text{IMP}_c = (\text{CPLEX}^2\ \text{Obj} - \text{MLSP Obj})/\text{MLSP Obj}$). The results show that the objective values obtained by CPLEX[2] are in general better than to those yielded by CPLEX[1], though in some cases, the best bound found by the former is worse than the latter. Nevertheless, both bounds are clearly dominated by our lower bound $v_1^{LB}$. When the problem scale is relatively small, the MLSP heuristic outperforms CPLEX on average by more than 15%. On the largest instances, the relative performance of the MLSP heuristic further increases to over 110% on average. From a comparison with $v_1^{LB}$, we can conclude that the heuristic is indeed quite effective at producing high quality solutions.

The MLSP heuristic manages to improve on the initial solutions (see Table 4) considerably, especially for the larger instances. To assess the contribution of the edge-removal and perturbation strategies, we run the MLSP algorithm using only the single-league and single-club optimization components as a base setting (CL). To this base setting, we add either the edge removal (CLE) component, or the perturbation strategies assignment shuffle and starting round reset (CLP). A final setting involves the inclusion of all algorithmic components (CLEP). Since the use of a tabu list is a well-established technique (see e.g. Chelouah & Siarry, 2000), which has been successfully applied in a wide variety of problems, demonstrating its effectiveness is not reiterated in this study (i.e., tabu lists are included in each algorithmic setting). Fig. 9 depicts the results. We observe that, without the edge removal component, the perturbation strategies generally do not lead to improved solutions. More specifically, the average solution quality of the base setting slightly decreases by 1.1% when complemented only with SR-RESET and AS-SHUFFLE. Nevertheless, the solution quality improves with 48.6% if EDGE-REM is included, and even with 52.7% if all algorithmic components are added. The performance of EDGE-REM is impressive, making the relatively large computation time (compared to single-league and single-club optimization) worthwhile.

### 7.3. Case study: Royal Belgium Football Association

We obtained a real-life instance from the Royal Belgian Football Association (RBFA). This instance features 221 clubs and 2721 teams; it provides the league composition for all 328 leagues of the youth and amateur football divisions in the province East-Flanders for the season 2016–17. This instance was slightly altered to obtain leagues of even size only: for each of the 149 leagues with an odd number of teams, we added a dummy team with a corresponding dummy club, with a venue capacity of one. More details on this instance are presented in Table 3.

Unfortunately, the data provided does not include the capacity of the clubs. Hence, as a base case, each club's capacity is set equal to the maximum number of home games per round it organized according to the official schedule that was used for season 2016–17. Furthermore, we consider cases where the capacity of the clubs is changed with a factor $\beta$ chosen from $\{0.7, 0.8, 0.9, 1.0, 1.1, 1.2\}$. In the official schedule, all leagues with six or eight teams finished in the first half of the season, perhaps to keep the option open that some of these teams are added to newly created leagues to be contested in the second half of the season. However, we also consider a setting where every league can start at any round that allows them to finish before the end of the season. Consequently, we design two instance types to reflect these two options, denoted as *Half* and *Full* respectively; note that any solution for the *Half* setting is also a solution for the *Full* setting.

Each of the resulting 12 instances were solved using CPLEX and the MLSP heuristic, with a time limit of 3600 and 900 seconds respectively. The results and the violations resulting from the official schedule ('REAL') are reported in Table 9. The difference in violations between the CPLEX and MLSP solutions is somewhat less pronounced compared to the artificial instances, but still clearly in the advantage of the MLSP heuristic. CPLEX could not prove optimality for any of the instances within its time limit. However, our bound $v_c^{LB}$ (which equals $v_1^{LB}$ and $v_2^{LB}$ for the *Full* setting) shows that some instances were indeed solved to optimality.

The results show that decreasing the capacity of all clubs to 90% of their current capacity leads to a rather limited increase in capacity violations when using the MLSP heuristic. The MLSP heuristic can substantially reduce its capacity violations if the full season length can be used for scheduling the smallest leagues. In fact, in that case, only 80% of the current venue capacity is needed to obtain virtually the same number of violations. This means that, contrary to the intuition at the football association, clubs do not have to invest in more terrains to avoid conflicts. With a better scheduling approach, they could even cut their venue (rental) costs. Expanding the current capacity of all clubs ($\beta > 1$) does not significantly reduce the capacity violations. In fact, the capacity violations happen at a very limited number of (smaller) clubs; rather than increasing the capacity for all clubs, it makes sense to improve the venue capacity for these clubs.

In the setting where the small leagues need to finish in the first half of the season (*Half*) and $\beta \geqslant 1$, the official schedule has no violations, while the MLSP heuristic settles for up to 24 violations. The comparison is however not quite fair. Indeed, the capacities have been set based on the official schedule ('REAL'), which thus by definition yields zero violations. Moreover, in the official schedule, many matches have been postponed to other dates which are not included in the given set of rounds. The RBFA confirmed that they made these manual adjustments in order to deal with capacity violations in their original (compact) schedule; it was however not possible to reconstruct their original schedule. This explains why $v^{LB}$ is larger than the REAL solution, and why CPLEX and MSLP cannot find the schedule as it was actually played.

Finally, we want to point out that in this real-life case, the results by MLSP and CPLEX may overestimate the venue capacity violations. Indeed, recall that there are a number of leagues with an odd size, for which we added a dummy team. In our approach, a home game against such a dummy team would be included in the calculation of the capacity violations, while in fact it corresponds to a bye game. Hence, by carefully scheduling the opponents in the third step of the scheduling process (see Section 1), one could further reduce the venue capacity violations. For instance, such an

**Table 8**
Performance of the MIP model and the MLSP heuristic on medium-scale and large-scale instances.

| Instance type | CPLEX[1] | | CPLEX[2] | | $v_1^{LB}$ | MLSP Obj | $IMP_c(\%)$ |
|---|---|---|---|---|---|---|---|
| | Obj | $v^{BB}$ | Obj | $v^{BB}$ | | | |
| 25-1 | 96.6 | 0.4 | 86.0 | 0.0 | 75.6 | 83.5 | 2.99 |
| 25-2 | 431.5 | 73.2 | 420.2 | 65.6 | 408.9 | 415.2 | 1.20 |
| 50-1 | 348.8 | 1.5 | 178.6 | 1.5 | 102.6 | 133.9 | 33.38 |
| 50-2 | 715.1 | 25.7 | 579.9 | 25.5 | 511.9 | 542.9 | 6.82 |
| 75-1 | 560.2 | 0.0 | 274.2 | 0.0 | 141.6 | 196.5 | 39.54 |
| 75-2 | 977.2 | 27.0 | 688.3 | 27.0 | 559.2 | 619.8 | 11.05 |
| *Avg.* | | | | | | | 15.83 |
| 100-1 | 439.9 | 0.0 | 99.5 | 0.0 | 13.5 | 43.1 | 130.86 |
| 100-2 | 1241.8 | 16.5 | 839.2 | 16.5 | 634.7 | 709.0 | 18.36 |
| 200-1 | 744.0 | 0.0 | 214.7 | 0.0 | 32.3 | 81.0 | 165.06 |
| 200-2 | 1445.1 | 1.5 | 619.7 | 1.5 | 223.1 | 349.0 | 77.56 |
| 400-1 | 1735.0 | 0.0 | 536.9 | 0.0 | 87.5 | 193.0 | 178.19 |
| 400-2 | 2748.1 | 0.0 | 1197.3 | 1.5 | 379.9 | 615.9 | 94.40 |
| *Avg.* | | | | | | | 110.74 |

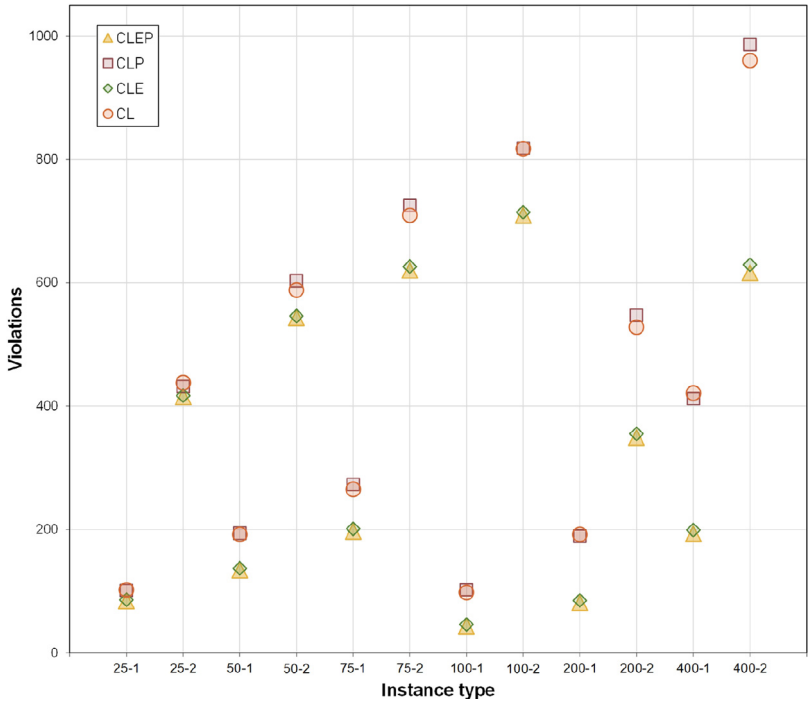


**Fig. 9.** Performance of various components of the MLSP heuristic.

**Table 9**
Results for the RBFA instances.

| $\beta$ | *Half* setting | | | | | | | *Full* setting | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | REAL | CPLEX[2] | $v^{BB}$ | MLSP | $v_1^{LB}$ | $v_2^{LB}$ | $v_c^{LB}$ | CPLEX[2] | $v^{BB}$ | MLSP | $v_c^{LB}$ |
| 1.2 | 0 | 20 | 11 | 20 | 13 | 19 | 13 | 13 | 0 | 13 | 13 |
| 1.1 | 0 | 20 | 0 | 20 | 13 | 19 | 13 | 13 | 0 | 13 | 13 |
| 1.0 | 0 | 24 | 0 | 25 | 13 | 19 | 13 | 13 | 0 | 13 | 13 |
| 0.9 | 529 | 427 | 0 | 84 | 15 | 25 | 13 | 14 | 0 | 14 | 13 |
| 0.8 | 1625 | 1218 | 0 | 481 | 32 | 180 | 13 | 18 | 0 | 15 | 13 |
| 0.7 | 3261 | 3016 | 96 | 1736 | 369 | 1091 | 165 | 177 | 90 | 168 | 165 |

optimization could reduce the total capacity violation for the full setting with $\beta = 1.0$ from 13 to 11.

## 8. Conclusions

In this paper, we first offered a comprehensive overview of problems, optimization methods, and applications related to multi-league sports scheduling. Motivated by common practice in amateur and youth sports competitions, our focus lies on minimizing venue capacity violations resulting from the limited number of simultaneous home matches that clubs can organize at their venue. We introduced a more general version of this problem, referred to as MLSP, in which we acknowledge that leagues can have different sizes, and that the competition organizer also needs to determine for each league when it starts.

Besides a lower bound, we developed a MIP model as well as a heuristic for MLSP. For the latter, we developed six different methods to obtain an initial solution and demonstrated the added value

of several local search and perturbation components to drastically improve on its initial solution. We generated a set of artificial instances with various sizes and properties. The heuristic yields solutions with considerably fewer capacity violations than the solutions obtained by solving the MIP model with CPLEX using enhanced settings (MIP emphasis on finding hidden feasible solutions, solution polishing, and setting the order of branching variables). A comparison with optimal solutions (for small instances), and with the lower bound (for larger intances) demonstrates that this heuristic is quite effective. Furthermore, it holds a clear advantage over CPLEX in terms of computation time, making it a viable approach for tackling problem instances with a realistic size.

A case study based on data obtained from the Royal Belgian Football Association confirms that optimizing the starting dates of the leagues over the full season can strongly reduce the number of violations. In fact, with careful planning of the leagues, clubs do not need to invest in more terrains. On the contrary, even an overall decrease of the club's capacities with 20% can be handled without causing a meaningful increase in capacity violations. Our method also allows the football association to pinpoint which clubs should increase their venue capacity, and which teams have overcapacity. While we demonstrated the advantage of our approach in dealing with the real instances provided by the Belgian Football Association, unfortunately, at the time of writing, we have not reached an agreement with the Belgian Football Association to implement our work in practice.

In this paper, we assume that leagues play according to a compact schedule, such that all its teams play on each round. An alternative policy would be to allow time-relaxed schedules, such that each round, several teams in the league have a bye game. While this spreads the leagues over a longer period of time in the season, it has the potential of reducing the capacity violations. Such policy, however, would require a method quite different from the one described in this paper, and forms an interesting direction for future research.

Finally, an interesting matter is the role of the selected HAP set(s) in multi-league scheduling. We have made some first steps in this direction by studying a specific setting with one large league and two small leagues. The results suggest that it is beneficial to stick to HAP sets of the same family in all leagues, and to make sure that their breaks are well aligned over different league sizes. Deriving general results on this matter would be a relevant topic for further research.

## Acknowledgement

The work was supported by the Chinese Scholarship Council (CSC) preference programme under grant no. 201906890037 and the Special Research Fund of Ghent University (BOF Ghent).

## Appendix A. Proof of Proposition 1

Given vector $\hat{\mathbf{y}}$ satisfying the condition of the proposition, we create a solution $(\hat{\gamma}, \hat{\mathbf{y}}, \hat{\mathbf{v}})$ as follows: we set

$$\hat{\gamma}_{c,r} = \frac{\sum_{t\in\hat{T}_c} k(s_\ell^t - 1)}{2|R|} \qquad \forall c \in C, r \in R,$$

and

$$\hat{v}_{c,r} = \max\left(\hat{\gamma}_{c,r} - \delta_c, 0\right) = \max\left(\frac{\sum_{t\in\hat{T}_c} k(s_\ell^t - 1)}{2|R|} - \delta_c, 0\right) \quad \forall c \in C, r \in R.$$

Based on the definition of $\Gamma_c$, we argue that both sets of constraints (11) and (12) are satisfied.

**Table A.1**
Capacity violations for HAP set combinations from the canonical and flexible families under 2RR (1RR).

| *D*-notation | 3121 | | | | | | |
|---|---|---|---|---|---|---|---|
| 22222221 | 8-F-1 | 8-F-2 | 8-F-3 | 8-F-4 | 8-F-5 | 8-F-6 | 8-F-7 |
| 16-C-1 | 22 (2) | 22 (3) | 23 (4) | 21 (2) | 21 (2) | 26 (4) | 25 (4) |
| 16-C-2 | 21 (2) | 21 (2) | 23 (4) | 22 (2) | 22 (2) | 25 (4) | 26 (4) |
| 16-C-3 | 22 (2) | 22 (2) | 23 (3) | 21 (2) | 21 (2) | 25 (3) | 25 (4) |
| 16-C-4 | 22 (2) | 22 (2) | 24 (3) | 22 (2) | 22 (2) | 25 (3) | 24 (3) |
| 16-C-5 | 23 (1) | 22 (2) | 24 (3) | 22 (2) | 22 (1) | 25 (3) | 24 (3) |
| 16-C-6 | 23 (1) | 22 (2) | 24 (4) | 22 (2) | 22 (1) | 26 (3) | 24 (3) |
| 16-C-7 | 23 (1) | 22 (2) | 24 (3) | 22 (2) | 22 (1) | 26 (3) | 24 (3) |
| 16-C-8 | 24 (2) | 24 (2) | 26 (3) | 24 (2) | 24 (1) | 26 (3) | 26 (3) |
| 16-C-9 | 24 (2) | 24 (2) | 26 (3) | 24 (2) | 24 (2) | 26 (3) | 26 (3) |
| 16-C-10 | 24 (1) | 24 (2) | 26 (3) | 24 (2) | 24 (2) | 26 (3) | 26 (3) |
| 16-C-11 | 22 (1) | 22 (2) | 24 (3) | 22 (2) | 23 (1) | 24 (3) | 26 (3) |
| 16-C-12 | 22 (1) | 22 (3) | 24 (4) | 22 (2) | 23 (1) | 24 (3) | 26 (3) |
| 16-C-13 | 22 (1) | 22 (2) | 24 (3) | 22 (2) | 23 (1) | 24 (3) | 25 (3) |
| 16-C-14 | 22 (2) | 22 (2) | 24 (3) | 22 (2) | 22 (2) | 24 (3) | 25 (3) |
| 16-C-15 | 21 (2) | 21 (2) | 23 (3) | 22 (2) | 22 (2) | 25 (4) | 25 (3) |

**Table A.2**
Capacity violations for HAP set combinations from the flexible and canonical families under 2RR (1RR).

| *D*-notation | 2221 | | | | | | |
|---|---|---|---|---|---|---|---|
| 31222221 | 8-C-1 | 8-C-2 | 8-C-3 | 8-C-4 | 8-C-5 | 8-C-6 | 8-C-7 |
| 16-F-1 | 22 (0) | 22 (1) | 24 (1) | 22 (1) | 24 (3) | 22 (2) | 24 (3) |
| 16-F-2 | 22 (1) | 22 (1) | 24 (2) | 23 (1) | 24 (3) | 23 (2) | 24 (3) |
| 16-F-3 | 22 (1) | 22 (1) | 24 (2) | 23 (2) | 24 (3) | 23 (2) | 24 (2) |
| 16-F-4 | 22 (1) | 22 (1) | 24 (2) | 22 (2) | 24 (4) | 22 (2) | 24 (2) |
| 16-F-5 | 22 (1) | 22 (1) | 24 (2) | 22 (2) | 24 (2) | 22 (2) | 24 (3) |
| 16-F-6 | 22 (1) | 22 (1) | 24 (2) | 22 (1) | 24 (2) | 22 (1) | 24 (3) |
| 16-F-7 | 22 (1) | 22 (1) | 24 (2) | 22 (1) | 24 (2) | 22 (1) | 24 (2) |
| 16-F-8 | 22 (1) | 22 (1) | 24 (3) | 22 (1) | 24 (2) | 22 (1) | 24 (2) |
| 16-F-9 | 22 (1) | 22 (1) | 24 (3) | 22 (2) | 24 (2) | 22 (2) | 24 (2) |
| 16-F-10 | 22 (1) | 22 (1) | 24 (2) | 22 (2) | 24 (4) | 22 (2) | 24 (2) |
| 16-F-11 | 22 (1) | 22 (1) | 24 (2) | 23 (2) | 24 (3) | 23 (2) | 24 (2) |
| 16-F-12 | 22 (1) | 22 (1) | 24 (3) | 23 (2) | 24 (3) | 23 (1) | 24 (2) |
| 16-F-13 | 22 (1) | 22 (0) | 24 (3) | 22 (2) | 24 (3) | 22 (1) | 24 (1) |
| 16-F-14 | 22 (0) | 22 (0) | 23 (1) | 22 (2) | 23 (2) | 22 (1) | 23 (1) |
| 16-F-15 | 22 (0) | 22 (0) | 23 (1) | 22 (1) | 23 (2) | 22 (2) | 23 (1) |

Summing over rounds and clubs results in

$$\sum_{c\in C}\left(|R|\max\left(\frac{\sum_{t\in\hat{T}_c} k(s_\ell^t - 1)}{2|R|} - \delta_c, 0\right)\right) = \sum_{c\in C}\max\left(\sum_{t\in\hat{T}_c} k(s_\ell^t - 1)/2 - \delta_c|R|, 0\right) = v_c^{LB}.$$

We conclude the proof by confirming that $(\hat{\gamma}, \hat{\mathbf{y}}, \hat{\mathbf{v}})$ must be equal to $v_1^{LB}$ as it provides a balanced utilization of the clubs.

## Appendix B. Effect of various HAP set combinations on capacity violations

**Table A.3**
Capacity violations for HAP set combinations from the flexible family under 2RR (1RR).

| *D*-notation | 3121 | | | | | | |
|---|---|---|---|---|---|---|---|
| 31222221 | 8-F-1 | 8-F-2 | 8-F-3 | 8-F-4 | 8-F-5 | 8-F-6 | 8-F-7 |
| 16-F-1 | **22** (**1**) | 22 (3) | 24 (3) | 22 (2) | 22 (2) | 26 (3) | 24 (3) |
| 16-F-2 | 22 (1) | **22** (**1**) | 24 (4) | 22 (2) | 22 (2) | 26 (4) | 26 (3) |
| 16-F-3 | 22 (1) | 22 (1) | **24** (**2**) | 22 (2) | 22 (2) | 26 (4) | 26 (4) |
| 16-F-4 | 23 (2) | 22 (1) | 24 (2) | **22** (**1**) | 22 (2) | 26 (4) | 24 (4) |
| 16-F-5 | 24 (1) | 22 (2) | 24 (2) | 22 (1) | **22** (**1**) | 26 (4) | 24 (4) |
| 16-F-6 | 23 (1) | 22 (2) | 24 (3) | 22 (1) | 22 (1) | **26** (**2**) | 24 (3) |
| 16-F-7 | 22 (1) | 22 (1) | 24 (2) | 22 (1) | 22 (1) | 24 (2) | 24 (2) |
| 16-F-8 | 22 (1) | 22 (1) | 24 (3) | 22 (2) | 23 (1) | 24 (3) | **26** (**2**) |
| 16-F-9 | **22** (**1**) | 22 (1) | 24 (2) | 22 (2) | 24 (1) | 24 (4) | 26 (4) |
| 16-F-10 | 22 (2) | **22** (**1**) | 24 (2) | 22 (1) | 23 (2) | 24 (4) | 26 (4) |
| 16-F-11 | 22 (2) | 22 (2) | **24** (**2**) | 22 (1) | 22 (1) | 26 (4) | 26 (4) |
| 16-F-12 | 22 (2) | 22 (2) | 24 (4) | **22** (**1**) | 22 (1) | 26 (3) | 26 (4) |
| 16-F-13 | 22 (2) | 22 (2) | 24 (3) | 22 (3) | **22** (**1**) | 24 (3) | 26 (3) |
| 16-F-14 | 22 (2) | 22 (2) | 23 (2) | 22 (2) | 22 (2) | **24** (**2**) | 25 (3) |
| 16-F-15 | 22 (2) | 22 (2) | 23 (2) | 22 (2) | 22 (2) | 25 (3) | **24** (**2**) |

## References


Bartsch, T., Drexl, A., & Kröger, S. (2006). Scheduling the professional soccer leagues of Austria and Germany. *Computers & Operations Research, 33*(7), 1907–1937.

Briskorn, D. (2008). Feasibility of home away pattern sets for round robin tournaments. *Operations Research Letters, 36*(3), 283–284.

Burrows, W., & Tuffley, C. (2015). Maximising common fixtures in a round robin tournament with two divisions. *Australasian Journal of Combinatorics, 63*(1), 153–169.

Carlsson, M., Johansson, M., & Larson, J. (2017). Scheduling double round-robin tournaments with divisional play using constraint programming. *European Journal of Operational Research, 259*(3), 1180–1190.

Chelouah, R., & Siarry, P. (2000). Tabu search applied to global optimization. *European Journal of Operational Research, 123*(2), 256–270.

Davari, M., Goossens, D., Beliën, J., Lambers, R., & Spieksma, F. C. (2020). The multi-league sports scheduling problem, or how to schedule thousands of matches. *Operations Research Letters, 48*(2), 180–187.

De Werra, D. (1980). Geography, games and graphs. *Discrete Applied Mathematics, 2*(4), 327–337.

De Werra, D. (1981). Scheduling in sports, studies on graphs and discrete programming, p. hansen, ed. *Annals of Discrete Mathematics, 11*, 381–395.

Della Croce, F., & Oliveri, D. (2006). Scheduling the Italian Football League: An ILP-based approach. *Computers and Operations Research, 33*(7), 1963–1974.

Della Croce, F., Tadei, R., & Asioli, P. (1999). Scheduling a round robin tennis tournamentunder courts and players availability constraints. *Annals of Operations Research, 92*, 349–361.

Durán, G., Durán, S., Marenco, J., Mascialino, F., & Rey, P. A. (2019). Scheduling Argentina's professional basketball leagues: A variation on the travelling tournament problem. *European Journal of Operational Research, 275*(3), 1126–1138.

Durán, G. A., Guajardo, M., López, A. F., Marenco, J., & Zamorano, G. A. (2021). Scheduling multiple sports leagues with travel distance fairness: An application to Argentinean youth football. *INFORMS Journal on Applied Analytics, 51*(2), 136–149.

Goossens, D. (2018). Optimization in sports league scheduling: Experiences from the Belgian Pro League soccer. In *Proceedings of the 6th International Conference on Operations Research and Enterprise Systems* (pp. 283–293).

Goossens, D., & Spieksma, F. (2009). Scheduling the Belgian soccer league. *Interfaces, 39*(2), 109–118.

Goossens, D., & Spieksma, F. (2011). Breaks, cuts, and patterns. *Operations Research Letters, 39*, 428–432.

Goossens, D. R., & Spieksma, F. C. (2012). Soccer schedules in europe: An overview. *Journal of scheduling, 15*(5), 641–651.

Grabau, M. (2012). Softball scheduling as easy as 1-2-3 (strikes you're out). *Interfaces, 42*(3), 310–319.

Hansen, P. (1981), *Studies on graphs and discrete programming* (11, pp. 381–395). North-Holland.

Horbach, A. (2010). A combinatorial property of the maximum round robin tournament problem. *Operations Research Letters, 38*(2), 121–122.

Hoshino, R., & Kawarabayashi, K. (2011). A multi-round generalization of the traveling tournament problem and its application to Japanese baseball. *European Journal of Operational Research, 215*, 481–497.

Ji, X., & Mitchell, J. E. (2005). Finding optimal realignments in sports leagues using a branch-and-cut-and-price approach. *International Journal of Operational Research, 1*(1–2), 101–122.

Kendall, G. (2008). Scheduling English football fixtures over holiday periods. *Journal of the Operational Research Society, 59*(6), 743–755.

Kendall, G., Knust, S., Ribeiro, C. C., & Urrutia, S. (2010). Scheduling in sports: An annotated bibliography. *Computers & Operations Research, 37*(1), 1–19.

King, D. E. (2009). Dlib-ml: A machine learning toolkit. *The Journal of Machine Learning Research, 10*, 1755–1758.

Knust, S. (2008). Scheduling sports tournaments on a single court minimizing waiting times. *Operations Research Letters, 36*(4), 471–476.

Knust, S., & Lücking, D. (2009). Minimizing costs in round robin tournaments with place constraints. *Computers & Operations Research, 36*(11), 2937–2943.

Kuhn, H. W. (1956). Variants of the Hungarian method for assignment problems. *Naval Research Logistics Quarterly, 3*(4), 253–258.

Kyngäs, J., Nurmi, K., Kyngäs, N., Lilley, G., Salter, T., & Goossens, D. (2017). Scheduling the Australian Football League. *Journal of the Operational Research Society, 68*, 973–982.

Lambers, R., Goossens, D., & Spieksma, F. C. (2022). The flexibility of home away pattern sets. *Journal of Scheduling*, 1–11.

Lambrechts, E., Ficker, A. M., Goossens, D. R., & Spieksma, F. C. (2017). Round-robin tournaments generated by the circle method have maximum carry-over. *Mathematical Programming, 172*(1-2), 277–302.

Larson, J., & Johansson, M. (2014). Constructing schedules for sports leagues with divisional and round-robin tournaments. *Journal of Quantitative Analysis in Sports, 10*(2), 119–129.

Mitchell, J. E. (2003). Realignment in the national football league: Did they do it right? *Naval Research Logistics, 50*(7), 683–701.

Miyashiro, R., Iwasaki, H., & Matsui, T. (2003). Characterizing feasible pattern sets with a minimum number of breaks. In *Proceedings of the 4th international conference on the practice and theory of automated timetabling* (pp. 78–99).

Moody, D., Kendall, G., & Bar-Noy, A. (2010). Youth sports leagues scheduling. In *Proceedings of the 8th international conference on the practice and theory of automated timetabling* (pp. 283–293).

Munkres, J. (1957). Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics, 5*(1), 32–38.

Nurmi, K., Kyngäs, J., & Goossens, D. (2013). Scheduling a triple round robin tournament with minitournaments for the Finnish national youth ice hockey league. *Journal of the Operational Research Society, 65*, 1770–1779.

Rasmussen, R. V. (2008). Scheduling a triple round robin tournament for the best Danish soccer league. *European Journal of Operational Research, 185*, 795–810.

Recalde, D., Severín, D., Torres, R., & Vaca, P. (2018). An exact approach for the balanced k-way partitioning problem with weight constraints and its application to sports team realignment. *Journal of Combinatorial Optimization, 36*(3), 916–936.

Russell, R. A., & Leung, J. M. (1994). Devising a cost effective schedule for a baseball league. *Operations Research, 42*(4), 614–625.

Saltzman, R. M., & Bradford, R. M. (1996). Optimal realignments of the teams in the national football league. *European Journal of Operational Research, 93*(3), 469–475.

Saur, M., Starr, K., Husted, M., & Newman, A. (2012). Scheduling softball series in the Rocky Mountain athletic conference. *Interfaces, 42*, 296–309.

Schönberger, J. (2015). Scheduling of sport league systems with inter-league constraints. In *Proceedings of the 5th international conference on mathematics in sport* (pp. 171–176).

Schönberger, J. (2017). The championship timetabling problem-construction and justification of test cases. In *Proceedings of the 6th international conference on mathematics in sport* (pp. 330–339).

Schönberger, J., Mattfeld, D. C., & Kopfer, H. (2004). Memetic algorithm timetabling for non-commercial sport leagues. *European Journal of Operational Research, 153*, 102–116.

Su, L.-H., Chiu, Y., & Cheng, T. E. (2013). Sports tournament scheduling to determine the required number of venues subject to the minimum timeslots under given formats. *Computers & Industrial Engineering, 65*(2), 226–232.

Suksompong, W. (2016). Scheduling asynchronous round-robin tournaments. *Operations Research Letters, 44*, 96–100.

Toffolo, T. A., Christiaens, J., Spieksma, F. C., & Berghe, G. V. (2019). The sport teams grouping problem. *Annals of Operations Research, 275*(1), 223–243.

Urban, T., & Russell, R. A. (2003). Scheduling sports competitions on multiple venues. *European Journal of Operational Research, 148*(2), 302–311.

Van Bulck, D., & Goossens, D. (2020). Handling fairness issues in time-relaxed tournaments with availability constraints. *Computers & Operations Research, 115*, 104856.

Van Bulck, D., & Goossens, D. (2020). On the complexity of pattern feasibility problems in time-relaxed sports timetabling. *Operations Research Letters, 48*(4), 452–459.

Van Bulck, D., Goossens, D., Schönberger, J., & Guajardo, M. (2020). RobinX: A three-field classification and unified data format for round-robin sports timetabling. *European Journal of Operational Research, 280*(2), 568–580.

Van Bulck, D., Goossens, D. R., & Spieksma, F. C. (2019). Scheduling a non-professional indoor football league: a tabu search based approach. *Annals of Operations Research, 275*(2), 715–730.

de Werra, D. (1985). On the multiplication of divisions: The use of graphs for sports scheduling. *Networks, 15*(1), 125–136.

de Werra, D., Jacot-Descombes, L., & Masson, P. (1990). A constrained sports scheduling problem. *Discrete Applied Mathematics, 26*(1), 41–49.

Yi, X., Goossens, D., & Talla Nobibon, F. (2020). Proactive and reactive strategies for football league timetabling. *European Journal of Operational Research, 282*, 772–785.