European Journal of Operational Research 308 (2023) 1249–1267

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Decision Support

The international timetabling competition on sports timetabling (ITC2021)

David Van Bulck[a,b,*], Dries Goossens[a,b]

[a] Faculty of Economics and Business Administration, Ghent University, Ghent, Belgium
[b] FlandersMake@UGent – Core lab CVAMO, Ghent, Belgium

ARTICLE INFO

Article history:
Received 20 April 2022
Accepted 24 November 2022
Available online 29 November 2022

Keywords:
OR in sports
Sports scheduling
Combinatorial optimization competition
Instance space analysis
Instance generation

ABSTRACT

The fifth International Timetabling Competition ITC2021 aims to instigate further research on automated sports timetabling. The competition's problem consists of constructing a compact double round-robin tournament with 16 to 20 teams while respecting various hard constraints and minimizing the penalties from violated soft constraints. This paper focuses on the organization of the ITC2021 competition, with a particular focus on the generation of a set of artificial though challenging, realistic, and diverse problem instances. For the latter, we present a set of features describing the structure of the problem instances, and use these features to construct the so-called instance space for sports timetabling. Several gaps in this space hint that existing problem instances from the literature are not very diverse. We therefore propose a novel integer programming approach to determine the feature values that cover these gaps, and show how to generate associated problem instances. Finally, we provide an overview of the participants and their contributions.

1. Introduction

In essence, sports timetabling is deciding on a suitable time slot for each of the games to be played in the tournament such that no team plays more than one game per time slot. Sports timetabling problems are often computationally challenging, even for competitions with a small number of teams. Furthermore, they are characterized by a wide diversity of constraints, and conflicting interests of many stakeholders (e.g. clubs, broadcasters, sponsors, police). Besides the business involved, its relevance for society is considerable, ranging from fans watching major events like the Olympics to parents organizing their personal schedules around their children's sporting hobbies.

In the late1970′s, sports timetabling found its way to academic papers (e.g. Ball & Webster, 1977), and - motivated by a large number of innovative applications in practice - gradually developed into a sizeable research field (Kendall, Knust, Ribeiro, & Urrutia, 2010). To this day, however, many sports timetabling contributions in the literature read as a case study, describing a single instance for which a tailored algorithm is developed and compared to a manual solution. Due to confidentiality agreements, these real-life problem instances are rarely shared, and consequently, few realistic benchmark instances are available.[1] Moreover, the state-of-the-art does not include a general solution method, capable of dealing with the wide variety of constraints that are typically present in sports timetabling. Such general method would nevertheless be useful for sports associations, confronted with a variety of timetabling problems, constraints that differ from season to season, and very different priorities in professional and amateur levels. Furthermore, given proper benchmark instances, general methods would be useful to assess the added value of dedicated methods. Indeed, it would be good practice to show how dedicated methods score better on a subset of the benchmark, or alternatively, to show how more general methods score (statistically) worse on dedicated benchmarks (see also Ceschia, Di Gaspero, & Schaerf, 2022). In this context, we believe there was a need to organize an international timetabling competition, labelled ITC2021, on sports timetabling. The main objective of ITC2021 was to promote the development and proper benchmarking of more general solution methods by providing a set of challenging, diverse, and realistic problem instances.

* Corresponding author.
E-mail address: david.vanbulck@ugent.be (D. Van Bulck).

[1] One notable exception is the travelling tournament problem, which minimizes the total team travel in a timetable. For this problem, substantial algorithmic progress has been reported after Easton, Nemhauser, & Trick (2001) made a set of artificial benchmark instances publicly available, and for which best results can be submitted to a website maintained by professor Michael Trick (see http://mat.tepper.cmu.edu/TOURN/).

https://doi.org/10.1016/j.ejor.2022.11.046
0377-2217/

Table 1
A compact double round-robin timetable for a single league with 6 teams. Each game is represented by an ordered pair in which the first element is the home team, and the second element is the away team.

s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}
(1,2)	(2,5)	(2,4)	(2,3)	(6,2)	(4,2)	(5,2)	(2,1)	(3,2)	(2,6)
(3,4)	(4,1)	(1,6)	(5,1)	(4,5)	(6,1)	(1,4)	(4,3)	(1,5)	(5,4)
(5,6)	(6,3)	(5,3)	(6,4)	(1,3)	(3,5)	(3,6)	(6,5)	(4,6)	(3,1)

The ITC2021 competition focused on the construction of (single-league) compact double round-robin (2RR) timetables, while respecting various hard constraints and minimizing the penalties arising from soft constraints. In a double round-robin tournament, each team faces every other team twice: once at its home venue, and once at the venue of the opponent (i.e. away). The literature distinguishes between compact timetables, which use the minimal number of time slots needed, and relaxed timetables, which use more time slots than strictly needed. In the compact case, a time slot is usually referred to as a round (i.e. a period during which a team can play at most one game), typically corresponding to a weekend or a week. In addition to the focus on compact 2RRs, ITC2021 assumes that the number of teams is even, and consequently, that each team plays exactly one game per time slot. Although many other tournament formats are conceivable (e.g. relaxed tournaments, multi-league timetabling, or knock-out formats), compact 2RR tournaments are well-researched and very common in practice (see e.g. Goossens & Spieksma, 2012). An example of a compact 2RR timetable is given in Table 1.

The main challenge in organizing ITC2021 is to come up with a set of problem instances that are challenging, feasible, realistic and yet diverse. To accomplish this, we first devise features describing the structure of problem instances from the literature, and use them to construct a so-called high-dimensional instance space. Using principal component analysis, we reduce the high-dimensional instance space into a visually interpretable two-dimensional space. We determine the boundaries of the region in which real-life instances occur, and identify a diverse set of coordinates that cover this region. We propose a novel integer programming approach to determine the (high-dimensional) feature values that match these coordinates, and show how to generate associated problem instances. Finally, we check that the instances are not too easy nor too difficult to solve by tackling them with an integer programming and a constraint programming approach, as well as a fix-and-optimize matheuristic. Problem instances are made available using the RobinX XML data format, which was specifically designed by Van Bulck, Goossens, Schönberger, & Guajardo (2020b) to make the description and communication of sports timetabling benchmark instances much easier. Moreover, this data format offers the flexibility to express the large amount and wide variety of constraints that are typically present in real-life instances.

ITC2021 started in October 2020, and closed by the end of April 2021. Throughout the competition, problem instances were released in three waves (early, middle, and late). While there is no fundamental difference between these groups of instances, the available time to solve them differed from more than six months (early) to two weeks (late), and the contribution on the ranking of the participants was larger for the late (middle) as compared to the middle (early) instances. Solutions were compared only on the basis of the sum of penalties from their violated soft constraints; only solutions satisfying all hard constraints were considered. All problem instances, together with an online solution validator, were made available on the competition website (www.itc2021.ugent.be). There were no restrictions on the hardware or (commercial) solvers used, nor on the computation time, other than that solutions had to be submitted before the end of the competition.

We are much indebted to the organizers of various other international timetabling competitions that have been organized before. All of these competitions have proven beneficial to the research community in terms of bringing together researchers from different areas, as well as stimulating the development of new solution approaches and comparing them. Furthermore, they ‘have brought forth most of the standard formulations and benchmarks and ‘have given the necessary initial boost in terms of infrastructure and promotion (Ceschia et al., 2022). The first ITC was organized in 2002 and focused on (a simplified version of) the university course timetabling problem (Paechter, Gambardella, & Rossi-Doria, 2003). The next ITC competition (2007) aimed to further develop interest in the general area of educational timetabling and involved three problems: curriculum-based timetabling, examination timetabling, and post-enrollment timetabling (McCollum, 2007; McCollum et al., 2010). With high-school timetabling, the ITC highlighted yet another educational timetabling problem in 2011 (Post, Di Gaspero, Kingston, McCollum, & Schaerf, 2011; 2016). The fourth ITC was again devoted to university course timetabling: it introduced the combination of student sectioning together with time and room assignment of events in courses (Müller, Rudová, & Müllerová, 2018; 2019). In between, there were two international nurse rostering competitions in 2010 (Haspeslagh, De Causmaecker, Schaerf, & Stølevik, 2014) and 2014 (Ceschia, Dang, De Causmaecker, Haspeslagh, & Schaerf, 2019), as well as a cross-domain heuristic search challenge (CHeSC 2011), where the challenge was to design a high-level search strategy that controls a set of problem-specific low-level heuristics, which would be applicable to different problem domains (Burke et al., 2011).

The remainder of this paper is as follows. Section 2 provides a description of the problem we address in ITC2021. Section 3 explains in detail how we generated the ITC2021 problem instances. Next, Section 4 explains and motivates the most important competition rules, and provides an overview of the participants and a brief description of their algorithmic approaches. Conclusions follow in Section 5.

2. Problem description

Let us denote with T the set of n teams that participate in the 2RR, and with S the set of time slots (i.e. rounds). Since we assume that n is even and that the set of time slots is compact, we have that $|S| = 2n - 2$. In all problem instances, n is either 16, 18, or 20; we believe that this number is common in real-life (see e.g. Goossens & Spieksma, 2009) and corresponds with a problem size that typically cannot be solved to optimality by state-of-the-art techniques. The season of a compact 2RR is often split into two halves, such that the first and the last $n - 1$ time slots each represent a single round-robin tournament (even though the order of the opponents that each team faces may differ between both halves). We call a timetable that follows this format ‘phased’.

Apart from the basic constraints that require that all games of the 2RR are scheduled and that each team plays at most one game per time slot, the ITC2021 problem instances feature nine (somewhat simplified) constraint types from the classification framework developed by Van Bulck et al. (2020b). We believe that the majority of the constraints that appear in real-life can be modelled with this selection of constraint types. Constraints are either hard or soft, where hard constraints represent fundamental properties of the timetable that can never be violated and soft constraints represent preferences that should be satisfied whenever possible. The objective in the ITC2021 problem instances is to minimize the overall (weighted) sum of deviations from violated soft constraints while respecting all hard constraints.

The problem instances are expressed using the RobinX data format developed by Van Bulck et al. (2020b). The main intention of this XML data format is to promote problem instance data sharing and reuse among different users and software applications, which is exactly what the timetabling competition envisioned. The XML data format is open, human readable (i.e. no binary format), software and platform independent, and flexible enough to store the problem instances which is why we believe it minimizes the specification burden while maximizing accessibility.

In the remainder of this section, we describe each of the constraint types (grouped into four constraint classes) and refer to Appendix A for a more detailed description of the constraints, their representation in the competition file format, and the calculation of the deviation from soft constraints.

2.1. Capacity constraints

Capacity constraints force a team to play home or away and regulate the total number of games played by a team or group of teams during a given period in time. We consider four different capacity constraints (CA1, CA2, CA3, CA4), each of which can be hard or soft. Constraints CA1 impose an upper limit on the number of home games (or away games) a given team plays during a given set of time slots. We use CA1 to model 'place constraints' that forbid a team to play at home during a particular time slot (e.g. stadium availability), and to balance the home-away status of games over teams and time (e.g. a minimal number of home games per team during the start and end of the season). Constraints CA2 generalize CA1 in the sense that they impose an upper limit on the number of home games (or away games) for a given team *against a given set of other teams* during a given set of time slots. This constraint could for instance be used to state that a bottom team plays at most one away game against a strong team during the last four time slots of the season. Constraints CA3 impose a limit on the maximal sequence of consecutive home or away games. In ITC2021, a CA3 constraint that is hard states that no team plays more than two home games (or away games) in a row (there are thus at most two CA3 hard constraints). When it is soft, it states that a given team should play no more than two home games (away games, or games) against a specific set of teams – usually a certain strength group – during every four consecutive rounds. Constraints CA4 impose an upper limit on the number of home games (away games, or games) between teams from a first set against teams from a second set during a given set of time slots. We for instance use this constraint to limit the number of games between teams from the same strength group, or to limit the total number of home games for a set of teams during a specific time slot (e.g. because teams share a stadium or are located in the same geographical region).

2.2. Game constraints

A GA1 hard or soft constraint enforces or forbids the assignment of a (set of) game(s) to one or more specific time slots. Examples include the police that forbid to play high risk games during time slots in which other major events are planned, or games that should take place during a 'derby time slot'. It is the only type of game constraint from the RobinX framework that we consider.

2.3. Break constraints

If a team plays a game with the same home-away status as its previous game, we say it has a break. As an example, team 2 in Table 1 has a break in time slot s_3, s_4, s_6, and s_7. Among others, the timing and frequency of breaks may be important as breaks may have an adverse impact on game attendance (see Forrest & Simmons, 2006). Constraints BR1 can be hard or soft and impose an upper limit on the total number of breaks for a given team and set of time slots (e.g. no breaks near the beginning or end of the season). In ITC2021 instances, we use a BR2 constraint to limit the total number of breaks in the timetable, hence at most one BR2 constraint (hard or soft) is present. Note that when formulated as hard constraints, the CA3 constraints forbid that any team has two consecutive breaks.

2.4. Fairness and separation constraints

To increase the fairness and attractiveness of the tournament, we consider the FA2 and SE1 soft constraints. Soft constraint FA2 expresses the preference that the timetable is 2-ranking-balanced, meaning that the difference in played home games between any two teams is at most two at any point in time. Soft constraint SE1, on the other hand, states that there should be at least 10 time slots between each pair of games involving the same teams. Given this interpretation, which is based on real-life instances, only one constraint of each type exists.

3. Generating a diverse set of problem instances

Over recent years, Smith-Miles and co-authors have developed a framework known as instance-space analysis that, among others, can be used to visualize how similar or dissimilar problem instances are with regard to each other (see e.g. Kletzander, Musliu, & Smith-Miles, 2021; Smith-Miles, Baatar, Wreford, & Lewis, 2014; Smith-Miles & Bowly, 2015; Smith-Miles & Lopes, 2012). To be suitable for an optimization competition, we believe that problem instances should (*i*) challenge existing algorithms such that progression towards new solutions methods is made, (*ii*) be feasible and allow to find (suboptimal) solutions with reasonable effort (at least for the majority of the instances so as to encourage participants to enter the competition), (*iii*) be as diverse as possible such that they cover the entire spectrum of constraints found in real-life and that algorithms generalize well outside the competition, and (*iv*) be as similar as possible to real-life problem instances so as to bridge the gap between theory and practice.

In the remainder of this section, we explain how to use instance space analysis to generate the ITC2021 competition instances that have the above properties. Section 3.1 proposes a set of features to describe sports timetabling problem instances in a high-dimensional instance space, which is then transformed into a two-dimensional instance space where the diversity of problem instances can be visually inspected. Based on problem instances previously presented in the literature, we also determine the part of the two-dimensional space where real-life problem instances are likely projected. Section 3.2 determines a set of target coordinates and uses a novel integer programming (IP) approach to derive a set of (high-dimensional) feature vectors projected at these coordinates, such that the associated problem instances are diverse and real-world-like. Finally, Section 3.3 proposes an instance generator which transforms a feature vector into a feasible problem instance that challenges existing solvers. At the time the competition was announced, none of the information from this section was available to the participants.

Table 2
Overview of the 18 features in F considered for the instance generation of the ITC2021 competition.

Name	Description
$f_{\vert T\vert}$	Number of teams (16, 18, or 20)
f_{P}	Boolean which is one if the tournament is phased and 0 otherwise
f^{H}_{CA1}	Number of CA1 hard constraints (others: f^{H}_{CA2}, f^{H}_{CA3}, f^{H}_{CA4}, f^{H}_{GA1}, f^{H}_{BR1}, f^{H}_{BR2})
f^{S}_{CA1}	Number of CA1 soft constraints (others: f^{S}_{CA2}, f^{S}_{CA3}, f^{S}_{CA4}, f^{S}_{GA1}, f^{S}_{BR1}, f^{S}_{BR2}, f^{S}_{FA2}, f^{S}_{SE1})

Table 3
Overview of the three-field notations for the real-life problem instances.

Name	Contributor	No.	Teams	Description
BEL	Goossens & Spieksma (2009)	3	18	2RR, C, P $\mid$ BR1^{H}, BR2^{H}, CA1^{H,S}, CA2^{S}, CA3^{H,S}, CA4^{H,S}, GA1^{S}, SE1^{S} $\mid$ SC
PRIN	Lewis & Thompson (2011)	10	12-18	2RR, C, ∅ $\mid$ CA1^{H}, CA2^{H,S}, CA3^{H,S}, CA4^{H}, SE1^{S} $\mid$ SC
ECUA	Recalde, Torres, & Vaca (2013)	1	12	2RR, C, P $\mid$ BR1^{H,S}, CA2^{H}, CA3^{H,S}, CA4^{H}, SE1^{S} $\mid$ SC
FIN	Kyngäs & Nurmi (2009)	1	14	2RR, C, P $\mid$ BR1^{S}, BR2^{S}, CA1^{H,S}, CA3^{S}, CA4^{S}, FA2^{S}, GA1^{H}, SE1^{S} $\mid$ SC
GER	Bartsch, Drexl, & Kröger (2006)	3	18	2RR, C, P $\mid$ BR1^{H}, BR2^{H}, CA1^{H,S}, CA4^{H}, GA1^{H}, SE1^{S} $\mid$ SC
ART	Nurmi et al. (2010)	16	10-16	2RR, C, {P,∅} $\mid$ BR1^{H}, BR2^{S}, CA1^{H,S}, CA3^{S}, CA4^{H,S}, GA1^{S}, SE1^{S} $\mid$ SC
SOUTHA	Durán, Guajardo, & Sauré (2017)	1	10	2RR, C, P $\mid$ BR1^{S}, CA1^{H}, CA3^{H}, SE1^{S} $\mid$ SC
ITA	Cocchi et al. (2018)	1	14	2RR, C, P $\mid$ BR1^{H}, BR2^{S}, CA1^{H,S}, CA2^{H,S}, CA3^{H}, CA4^{H}, FA2^{S}, GA1^{S}, SE1^{S} $\mid$ SC
RRT	Horbach, Bartsch, & Briskorn (2012)	33	10-22	2RR, C, P $\mid$ BR1^{H}, CA1^{S}, CA4^{H}, GA1^{S}, SE1^{S} $\mid$ SC
NOR	Hausken, Andersson, Fagerholt, & Flatberg (2012)	8	14-16	2RR, C, P $\mid$ BR1^{H}, BR2^{S}, CA1^{S}, CA3^{S}, CA4^{H,S}, GA1^{H,S}, SE1^{S} $\mid$ SC

3.1. Visualizing the instance space

Let us define with F the set of problem instance features, where each feature describes a measurable property or characteristic of a problem instance (e.g. the total number of teams). Features are useful to express how similar problem instances are. For a general advise of how to devise features, we refer to Smith-Miles & Lopes (2012). For the ITC2021 competition, we use as features the number of teams and the three-field notation $\alpha|\beta|\gamma$ of RobinX. A problem instance in this notation is of type $\alpha_1, \alpha_2, \alpha_3|\beta|\gamma$, where α_1 gives the tournament format, α_2 the compactness of the tournament, α_3 the symmetry structure, β the constraints that are present (distinguishing hard and soft constraints), and γ the objective function (see Van Bulck et al., 2020b). However, as we only consider compact double round-robin tournaments where the objective is to minimize the sum of soft constraint penalties while respecting all hard constraints, we ignore the α_1, α_2, and γ fields. Moreover, instead of simply denoting whether or not a specific type of hard or soft constraint is present, we count for each constraint the number of times it appears in the problem instance. For instance, if a problem instance contains 20 hard constraints of type CA1, then $f_{\mathrm{CA1}^H} = 20$. For an overview of the features, we refer to Table 2.

Next, we collect a representative set of real-life problem instances to determine whether the artificial problem instances to be generated are real-world-like. To this purpose, we select all compact 2RR problem instances from the RobinX archive by Van Bulck, Goossens, Schönberger, & Guajardo (2020a) that have an even number of teams between 10 and 22 and that have a soft constraint minimization objective. This way, a total of 77 problem instances were obtained which, depending on the contributors, can be divided into ten different groups of instances. For each problem instance i, we then extract its $|F|$-dimensional feature vector $\mathbf{f_i}$. We thereby ignore all constraints that do not appear in the ITC2021 competition, and replace any symmetry structure by the phased structure plus an SE1 hard constraint. Table 3 gives an overview of the real-life problem instances. For each group of instances, the table provides the name of the instance group, a reference to the contributors, the number of problem instances in the group (often related to different seasons of the same competition), the range of the number of teams in the instances, and a description in terms of the three-field notation of RobinX.

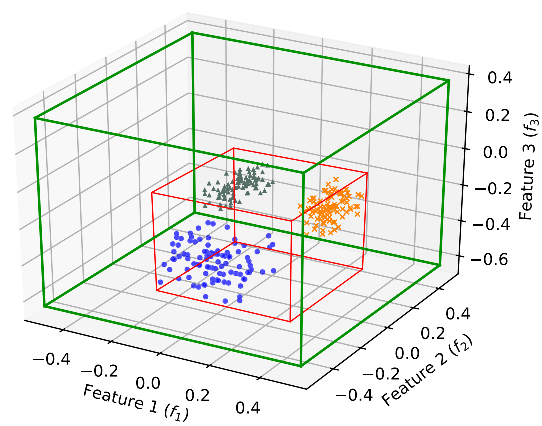

Fig. 1. Visualization of type 1 (blue circles), type 2 (gray triangles), and type 3 (orange crosses) problem instances via their feature vector in the high-dimensional space (i.e. a 3-dimensional space if there are 3 features). The inner box in red represents the target instance space, whereas the outer box in green represents the valid instance space.

In order to assess instance diversity, it helps to visualize the feature vectors in the high-dimensional instance space: if the feature vectors of two problem instances are similar, then the rectilinear or Euclidean distance between the problem instances in the high-dimensional instance space is small. Hence, clusters of feature vectors likely denote problem instances that are related to one another (e.g. the same sport or different seasons of the same competition; see e.g. the cluster of blue circles in Fig. 1). An intriguing question is what part of the instance space is spanned by all problem instances that can be expressed with the syntax of ITC2021. We refer to this region as the valid instance space. Smith-Miles & Bowly (2015) determine the boundaries of the valid instance space by deriving upper and lower bounds on the value of each feature. The bounds can then be represented as inequalities in the high dimensional instance space, which together define a bounding box in which all problem instances are situated (see the green outer box in Fig. 1). We use a similar idea to define what we call the *target instance space*, where we use as bounds the minimal and maximal feature values observed in the real-life problem instances (see the red inner box in Fig. 1, and the descriptive statistics in Table 4). As it is hard to differentiate the set of real-life problem instances

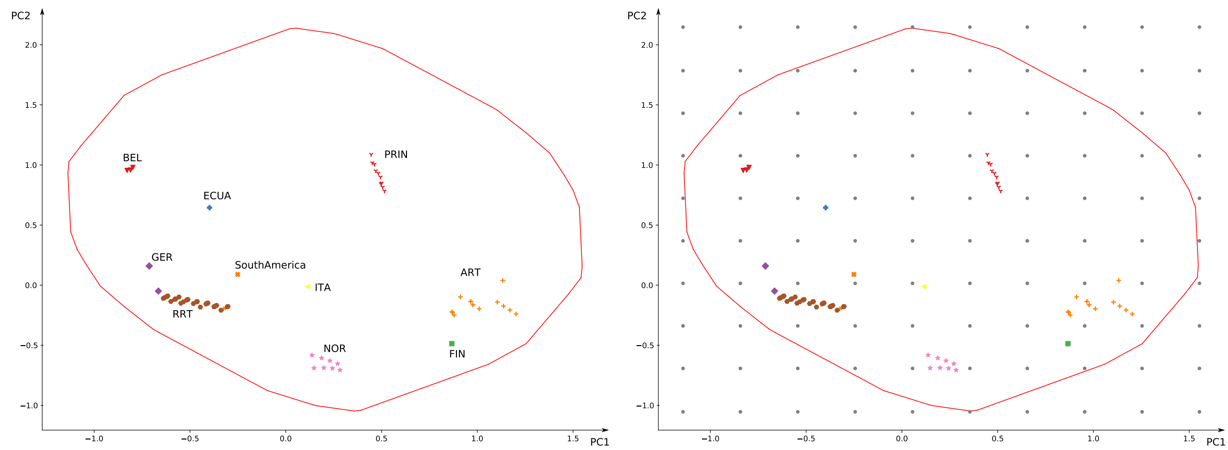

Fig. 2. Left: Projection of the problem instances in the 2D-space. The target instance space is represented by the red convex hull. Right: target coordinates in the lattice.

Table 4
Descriptive statistics of the feature values observed in the real-life problem instances. In the decile computation for the features related to frequency of constraints, the value '0' has been ignored (making that the 10% decile can be lower than the mean, see Section 3.2).

	$f_{\lvert T\rvert}$	f_P	f^H_{CA1}	f^S_{CA1}	f^H_{CA2}	f^S_{CA2}	f^H_{CA3}	f^S_{CA3}	f^H_{CA4}	f^S_{CA4}	f^H_{GA1}	f^S_{GA1}	f^H_{BR1}	f^S_{BR1}	f^H_{BR2}	f^S_{BR2}	f^S_{FA2}	f^S_{SE1}
min	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10% decile	10	0	5	15	6	12	1	12	18	18	2	1	12	10	1	1	1	1
mean	14.83	0.68	5.56	16.35	3.90	27.92	0.27	20.36	32.06	19.01	1.78	64.99	26.14	0.60	0.08	0.31	0.03	0.94
90% decile	20	1	42	32	72	620	2	112	85	340	34	126	44	24	1	1	1	1
max	22	1	116	60	112	620	2	126	198	374	34	4368	602	24	1	1	1	1

from artificially generated problem instances that are projected in the target instance space (they have similar feature values; see also Lopes & Smith-Miles, 2013), we consider all problem instances in the target instance space as real-world-like.

In order to visualize the embeddings of the feature vectors when the number of features is higher than three, we need a dimension reduction technique to transform points in the high-dimensional instance space to e.g. a two-dimensional (2D) instance space. To assess instance dissimilarity and to discover clusters of problem instances, ideally we have that the topology of the original high-dimensional space is preserved as much as possible in the low dimensional space such that instances that are close in the low-dimensional space are also close in the high-dimensional space (see Smith-Miles et al., 2014). Moreover, in order to derive insights, the low-dimensional instance space must be intuitive to analyse (e.g. by considering a projection that results in linear trends of feature values over the two-dimensional space, see Muñoz, Villanova, Baatar, & Smith-Miles, 2018). Smith-Miles et al. (2014) and Smith-Miles & Bowly (2015) propose to reduce the dimensions with Principal Component Analysis (PCA) which essentially constructs a linear transformation from the high-dimensional space to a lower dimensional space, maximally preserving the original variance (for an introduction to PCA, see Abdi & Williams, 2010). Before applying PCA, however, for each instance i we first perform min-max normalization $(f_{i,j} - \min_j)/(\max_j - \min_j)$ and mean centering $(f_{i,j} - \mu_j)$ such that each feature $j \in F$ has range 1 (or zero if $\max_j = \min_j$) and mean zero. The min-max normalization ensures that each feature has an equal share in determining the direction of maximal variation, whereas the mean centering makes that a feature vector in which all feature values attain their average is projected at the origin of the two-dimensional space.

Considering the problem instances from Table 3, the principal components can be computed as the eigenvectors of the data's covariance matrix. The first two principal components combined explain 55% of the variance in the data. The projection weights $\boldsymbol{W}^\intercal$ of these components are given in Eq. (1).

$$\boldsymbol{W}^\intercal F = \begin{bmatrix} -0.239 & 0.119 \\ -0.666 & -0.383 \\ 0.063 & 0.132 \\ -0.197 & -0.171 \\ 0.006 & 0.096 \\ -0.077 & 0.187 \\ -0.015 & 0.489 \\ 0.224 & 0.418 \\ -0.122 & 0.167 \\ -0.045 & 0.122 \\ 0.033 & -0.143 \\ 0.003 & -0.000 \\ -0.034 & -0.006 \\ 0.019 & -0.008 \\ -0.152 & 0.190 \\ 0.565 & -0.482 \\ 0.033 & -0.300 \\ -0.193 & 0.043 \end{bmatrix}^\intercal \begin{bmatrix} f_{\lvert T\rvert} \\ f_P \\ f^H_{\mathrm{CA1}} \\ f^S_{\mathrm{CA1}} \\ f^H_{\mathrm{CA2}} \\ f^S_{\mathrm{CA2}} \\ f^H_{\mathrm{CA3}} \\ f^S_{\mathrm{CA3}} \\ f^H_{\mathrm{CA4}} \\ f^S_{\mathrm{CA4}} \\ f^H_{\mathrm{GA1}} \\ f^S_{\mathrm{GA1}} \\ f^H_{\mathrm{BR1}} \\ f^S_{\mathrm{BR1}} \\ f^H_{\mathrm{BR2}} \\ f^S_{\mathrm{BR2}} \\ f^S_{\mathrm{FA2}} \\ f^S_{\mathrm{SE1}} \end{bmatrix} \quad (1)$$

Fig. 2 (left) shows the PCA embedding of the problem instances in the 2D instance space. It is interesting to see that the problem instances provided by the same contributors (often different seasons of the same competition) indeed form clusters of problem instances in the 2D instance space. The target instance space is denoted by the red convex hull; as noted by Smith-Miles & Bowly (2015), this hull can be computed by taking the convex hull of the projections of the inequalities that define the bounding box in the high-dimensional instance space.

3.2. Deriving target three-field notations

Given the two-dimensional instance space, we now derive the (high-dimensional) feature vector of the problem instances to be generated. To this purpose, we first determine a set of target coordinates in the 2D-space where we would like that the problem instances are projected. We do this by determining the smallest $r \times r$ lattice such that the lattice covers the entire target instance space and that the number of lattice points within the target instance space is at least the desired number of problem instances to be generated (45 in the ITC2021 case; see Fig. 2, right).

Although the lattice ensures that the problem instances are diverse, a simple inverse transformation from the 2D-space to the high-dimensional instance space does not suffice to derive a set of high dimensional feature vectors projected at these coordinates. Indeed, such a simple transformation would ignore the relation between different features (e.g. in the ITC2021 competition, there is either one BR2 hard constraint, one BR2 soft constraint, or no BR2 constraint at all). In fact, it does not even ensure that the feature vector is within the high dimensional target space. Moreover, as the projection is linear, this likely results in a feature vector where most features have value zero and all other features have very high or low values. Instead, we want to make sure that the number of non-zero features is within reasonable bounds and that extremely low or high feature values are avoided.

This motivates the use of an IP model where for each pair of target coordinates $\boldsymbol{z} = (z_1, z_2)$ in the convex hull of the 2D-space, we look for a feature vector in the high-dimensional instance space of which the projection is as close as possible to $\boldsymbol{z}$. Our main decision variables are variables f_j that model the value of feature $j \in F$, and g_j that model the min-max normalized and mean-centred feature value of j. Let $F_{\text{hard}} \subseteq F$ and $F_{\text{soft}} \subseteq F$ denote the subset of hard and soft constraint labels in F, respectively. Dummy variables s_d and e_d respectively model the slack and excess on the projections from the target z_d along dimension $d \in \{1, 2\}$, and variable y_j with $j \in F_{\text{hard}} \cup F_{\text{soft}}$ is one if $f_j > 0$ and 0 otherwise. Parameters $w_{j,d}$ are fully determined by the PCA model (i.e. Eq. (1)) and give the projection weight of feature $j \in F$ along dimension $d \in \{0, 1\}$. Finally, non-negative parameters $\min_j$, $\max_j$, μ_j, $\delta_{1,j}$, and $\delta_{9,j}$ give the minimum, maximum, mean value, 10%, and 90% decile of feature j over the real-life problem instances (see Table 4 for the actual values).

$$\textbf{minimize} \quad \sum_{d \in \{1,2\}} (s_d + e_d)$$

$$\sum_{j \in F} g_j w_{j,d} + s_d - e_d = z_d \qquad \forall d \in \{1, 2\} \quad (2)$$

$$g_j = \frac{f_j - \min_j}{\max_j - \min_j} - \mu_j \qquad \forall j \in F \quad (3)$$

$$f_{\text{BR2}^H} + f_{\text{BR2}^S} \leqslant 1 \quad (4)$$

$$y_j \delta_{1,j} \le f_j \le y_j \delta_{9,j} \qquad \forall j \in F_{\text{hard}} \cup F_{\text{soft}} \quad (5)$$

$$\sum_{j \in F_{\text{hard}}} y_j \geqslant 3, \sum_{j \in F_{\text{soft}}} y_j \geqslant 3 \quad (6)$$

$$\sum_{j \in F_{\text{hard}}} f_j \geqslant 20, \sum_{j \in F_{\text{soft}}} f_j \geqslant 30 \quad (7)$$

$$g_j, s_d, e_d \ge 0 \qquad \forall j \in F, d \in \{1, 2\} \quad (8)$$

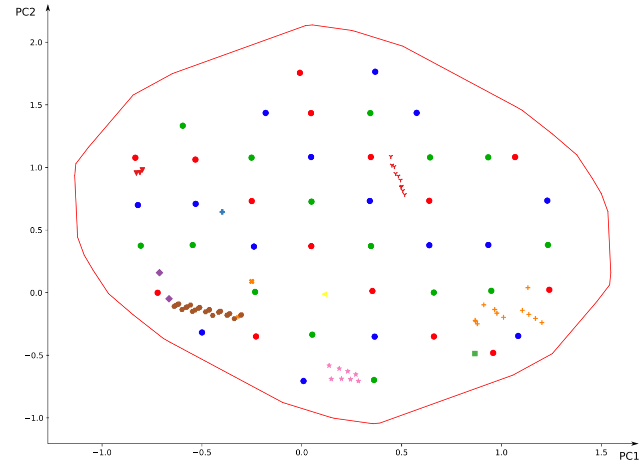

Fig. 3. Projections of the ITC problem instances in the two-dimensional instance space. Early instances are coloured in red, middle instances in green, and late instances in blue.

$$f_j \in \mathbb{N}^+ \qquad \forall j \in F \quad (9)$$

$$y_j \in \{0, 1\} \qquad \forall j \in F_{\text{hard}} \cup F_{\text{soft}} \quad (10)$$

$$f_{|T|} = 16 \quad (11)$$

The objective function minimizes the rectilinear distance from the projection of the feature vector to the target coordinates. Constraints (2) model the slack and excess of the projection along the two dimensions, and Constraints (3) model the relation between f_j and g_j. Constraints (4) express that there is at most one BR2 hard or soft constraint. Next, Constraints (5) regulate the value of the y_j variables. In particular, they express that when constraint type j is present, the total number of constraints of that type is at least the first decile and at most the ninth decile (this also explains why we ignore the value '0', i.e. the constraint type is not present, in the determination of the deciles, see Table 4). Next, Constraints (6) and (7) express that at least 3 hard and 3 soft constraint types must be present, and that there are at least 20 hard constraints and 30 soft constraints (regardless of the type). Constraints (8) to (10) are the non-negativity and integrality constraints, and Constraint (11) regulates the number of teams.

For each pair of target coordinates in the convex hull of the 2D-space, we repeat the generation of the three-field notation with the restriction that $f_{|T|}$ equals 18 and 20, instead of 16 (Constraint (11)). Next, we manually choose 15 feature vectors for each group of early, middle, and late problem instances such that each group contains three feature vectors with 16 teams, six with 18 teams, and six with 20 teams. Table 5 gives the resulting feature vectors for the early, middle, and late competition problem instances; the projection of these feature vectors in the two-dimensional instance space is given in Fig. 3. As we can see, each instance group is well spread over the instance space.

3.3. Problem instance generator

Once the feature vectors are generated, a compatible and feasible problem instance needs to be generated for each of them. This section proposes an instance generator that creates a feasible problem instance for each high-dimensional feature vector in Table 5. We do this by first constructing a number of so-called candidate home-away patterns (HAPs), where the HAP of a team specifies for

Table 5
Overview of the early, middle, and late high-dimensional feature vectors generated with IP for the different target coordinates in Fig. 2 (right).

				CA1		CA2		CA3		CA4		GA1		BR1		BR2		FA2	SE1
	Instance	Sym.	Teams	H	S	H	S	H	S	H	S	H	S	H	S	H	S	S	S
Early	1	1	16	8	29	16	0	0	0	0	21	31	5	13	20	0	0	0	0
	2	1	16	38	30	0	0	2	82	0	0	0	1	12	0	1	0	1	0
	3	1	16	24	0	72	21	0	112	0	0	34	51	18	0	0	1	1	0
	4	1	18	0	32	0	235	0	0	85	0	34	0	44	0	1	0	0	1
	5	1	18	41	27	36	331	2	111	81	117	23	0	23	0	1	0	0	1
	6	1	18	38	31	71	591	2	54	81	115	0	3	0	0	0	1	1	1
	7	0	18	42	31	30	620	1	112	84	340	5	55	12	0	1	0	0	1
	8	0	18	19	0	8	57	0	112	0	339	4	73	39	0	0	0	1	0
	9	0	18	39	0	14	0	0	88	0	0	14	2	23	10	0	1	1	0
	10	1	20	42	32	72	620	2	23	85	339	0	0	44	0	1	0	0	1
	11	0	20	42	32	72	620	2	112	85	340	0	3	44	0	1	0	0	1
	12	1	20	37	0	72	0	2	20	31	0	17	1	20	13	0	1	0	0
	13	0	20	41	31	27	257	1	110	0	0	10	24	20	10	1	0	0	0
	14	0	20	5	30	0	0	0	0	0	0	34	0	17	24	0	1	1	0
	15	0	20	42	0	72	620	2	112	71	340	0	126	0	24	0	1	1	0
	1	1	16	0	32	14	620	0	0	85	340	0	0	44	0	1	0	0	1
	2	1	16	42	32	72	620	2	112	85	340	0	126	44	0	1	0	0	1
	3	0	16	42	0	72	617	2	107	85	338	0	126	36	21	0	1	1	1
	4	1	18	31	18	17	0	1	0	0	41	25	85	23	24	0	0	0	0
	5	1	18	41	24	40	33	0	12	0	0	26	126	44	0	0	1	1	0
	6	1	18	39	0	41	0	2	111	30	27	6	1	44	13	0	1	0	1
	7	0	18	0	30	0	355	1	0	78	51	34	17	28	21	0	1	0	1
	8	0	18	16	0	0	27	2	108	0	34	12	41	32	14	0	0	0	0
	9	0	18	42	0	0	37	1	100	19	39	4	0	28	23	0	1	1	0
	10	1	20	41	15	71	363	0	0	46	262	0	74	39	0	1	0	0	0
	11	1	20	7	0	71	612	2	88	84	340	0	7	12	0	0	0	1	0
	12	1	20	0	32	28	168	1	13	0	0	5	4	29	21	0	1	1	1
	13	0	20	42	29	72	242	1	0	85	76	7	2	12	0	0	0	0	1
	14	0	20	5	18	11	319	2	112	0	338	6	19	38	10	1	0	1	0
	15	0	20	12	0	23	0	0	77	0	0	23	44	37	10	0	1	0	1
	1	0	16	42	32	72	198	1	13	82	283	0	15	38	0	0	0	1	0
	2	0	16	42	0	72	620	2	112	85	340	0	5	44	0	1	0	0	0
	3	0	16	42	0	72	326	1	43	0	60	0	7	12	0	0	1	1	1
	4	1	18	0	32	0	0	0	0	18	0	34	1	44	0	0	0	0	1
	5	1	18	0	19	69	614	2	0	81	109	23	4	0	0	1	0	1	0
	6	1	18	0	32	0	125	0	0	85	0	34	0	44	0	0	1	0	1
	7	0	18	42	32	40	601	1	61	0	0	5	43	37	0	1	0	0	1
	8	1	18	37	15	0	14	0	111	0	0	32	29	41	24	0	1	0	1
	9	0	18	40	0	20	250	2	112	0	0	0	18	40	20	0	1	1	0
	10	1	20	0	31	67	447	2	0	85	205	34	10	44	0	1	0	0	1
	11	1	20	6	0	16	274	0	88	0	0	17	3	12	0	1	0	1	0
	12	0	20	40	32	72	620	2	16	85	340	0	0	44	0	1	0	0	1
	13	0	20	14	32	72	15	2	0	81	71	0	13	0	0	0	1	1	1
	14	0	20	42	0	72	390	2	112	0	340	0	126	0	24	0	0	1	0
	15	0	20	5	0	0	0	0	15	0	0	34	0	12	24	0	1	1	0

each time slot whether a team plays at home or away, and then combine these HAPs into a set of HAPs by assigning exactly one HAP to each team (see Section 3.3.1). Given a HAP set, we then check its feasibility by creating a compatible timetable or proving that none exists (see Section 3.3.2). We note that this approach resembles the well-known First-Break-Then-Schedule approach (see Nemhauser & Trick, 1998). Next, we construct a problem instance around the HAP set and its compatible timetable by adding constraints as described by the feature vector (see Section 3.3.3). Finally, we check that the generated problem instances are neither too easy nor too challenging (see Section 3.3.4). For an overview of our approach, we refer to Fig. 4.

3.3.1. Home-away pattern set

In order to generate a HAP set, we start by enumerating home-away patterns using the Constraint Programming (CP) formulation below. In this formulation, variable h_s is one if the pattern contains a home game in time slot $s \in S$, and 0 otherwise. Moreover, b_s is 1 if the pattern has a break in time slot $s \in S \setminus \{1\}$ and is 0 otherwise.

$$\texttt{distribute}([n-1, n-1], [0, 1], [h_1, \ldots, h_{2(n-1)}]) \qquad (12)$$

$$(h_s = h_{s-1}) \Rightarrow (b_s = 1) \qquad \forall s \in S \setminus \{1\} \quad (13)$$

$$\sum_{s \in S \setminus \{1\}} b_s \leqslant 6 \qquad (14)$$

The global distribute constraint has syntax `distribute`(*cards*, *values*, *vars*), where *cards* and *values* are vectors with the same index set L and *vars* is a vector of decision variables. The constraint is satisfied if for each $l \in L$ exactly *cards*$[l]$ variables in *vars* have value *values*$[l]$. In our case, the constraint states that a pattern contains exactly $(n-1)$ home games and $(n-1)$ away games (i.e. $\sum_{s \in S} h_s = n-1$). Constraints (13) model the values of the b_s variables. Finally, Constraints (14) state that each pattern contains at most 6 breaks; if the feature vector contains a BR2 hard constraint, we limit the total number of breaks per pattern to 5. This upper limit on the number of breaks is based on experience in practice: although very often requested by league organizers, constraints on the number of breaks per team are hard to enforce by existing algorithms.

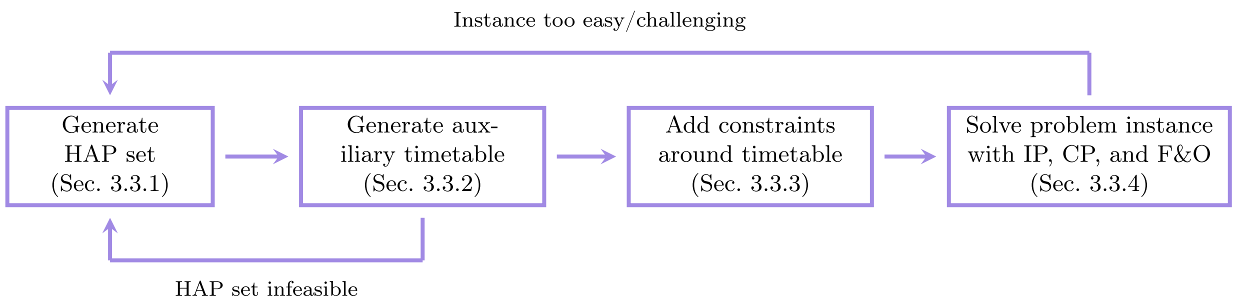

Fig. 4. Overview of how a problem instance is generated for a given feature vector. The first step generates a HAP set with some constraints on the number of breaks per team, while the second constructs an auxiliary timetable compatible with this HAP set. In the third step, we add all constraints from the feature vector to the problem instance in such a way that the auxiliary timetable satisfies the hard constraints (allowing to ignore these constraints in the first two steps). Finally, the fourth step checks that the resulting problem instance is not too easy nor too challenging by solving the resulting problem instance with a straightforward IP, CP, and F&O approach.

In case the feature vector contains at least one hard constraint of type CA3, we replace Constraints (12) by (15).

$$\texttt{sequence}(1, 2, 3, [n-1, n-1], [0, 1], [h_1, \ldots, h_{2(n-1)}]) \quad (15)$$

The global sequence constraint has syntax $\texttt{sequence}(min, \max, width, cards, values, vars)$ and is satisfied if for each $l \in L$ exactly $cards[l]$ variables in $vars$ have value $values[l]$ and for each subsequence of length $width$ at least min and at most max variables have value $values[l]$. In our case, this constraint thus regulates the required number of home and away games and additionally forbids consecutive breaks (i.e. $1 \leqslant h_s + h_{s+1} + h_{s+2} \leqslant 2$ for all $s \in S : s+2 \leqslant |S|$).

We use CPLEX CP OPTIMIZER version 12.10 in combination with a multi-point search strategy and random variable and value selection to generate a diverse set of 2000 home-away patterns. Assuming that the number of teams given by the feature vector is n, we subsequently use Integer Programming (IP) to select n patterns which will constitute the HAP set. However, to ensure that different runs of the generator result in different instances, we offer to the IP solver only a selection of 500 uniformly selected patterns. We denote this subset with Q, and describe each pattern $q \in Q$ with parameters $h'_{q,s}$ which is 1 if pattern q contains a home game on time slot $s \in S$ and zero otherwise. Using variables p_q which are 1 if pattern q is selected and 0 otherwise, we generate a HAP set by solving the following IP formulation.

$$\sum_{q \in Q} p_q = n \quad (16)$$

$$\sum_{q \in Q : h'_{q,s}=1} p_q = n/2 \qquad \forall s \in S \quad (17)$$

Constraint (16) selects exactly n patterns, and Constraints (17) state that during each time slot exactly half of the selected patterns has a home game (a necessary condition for a compatible timetable to exist). We solve model (16)-(17) with ILOG CPLEX version 12.10.

3.3.2. Auxiliary timetable

Once a HAP set is found, we need to check whether it is feasible (a problem known as the HAP set feasibility problem, see e.g. Van Bulck & Goossens, 2020). We do this by solving the following IP model with ILOG CPLEX which constructs a compatible timetable (described by variables $x_{i,j,s}$ which are 1 if team $i \in T$ plays a home game against team $j \in T \setminus \{i\}$ in time slot $s \in S$, and 0 otherwise) or proves that no such timetable exists. Without loss of generality, we assume that the pattern with the lowest index is assigned to team 1, and so on. This results in parameters $h'_{i,s}$ which are 1 if team i plays home in time slot $s \in S$, and 0 if i plays away in s.

$$x_{i,j,s} = 0 \qquad \forall i, j \in T : i \neq j, s \in S : h'_{i,s} = 0 \vee h'_{j,s} = 1 \quad (18)$$

$$\sum_{j \in T \setminus \{i\}} (x_{i,j,s} + x_{j,i,s}) = 1 \qquad \forall i \in T, \forall s \in S \quad (19)$$

$$\sum_{s \in S} x_{i,j,s} = 1 \qquad \forall i, j \in T : i \neq j \quad (20)$$

$$\sum_{s=1}^{n-1} (x_{i,j,s} + x_{j,i,s}) = 1 \qquad \forall i, j \in T : i < j \quad (21)$$

$$x_{i,j,s} \in \{0, 1\} \qquad \forall i, j \in T : i \neq j, \forall s \in S \quad (22)$$

Constraints (18) reduce the number of variables in the system by stating that game (i, j) can only take place during time slots on which team i plays home and team j plays away. Next, Constraints (19) and (20) state that each team plays exactly one game per time slot and that each game of the double round-robin tournament is scheduled exactly once. In case that the tournament needs to be phased, we additionally add Constraints (21). Finally, Constraints (22) are the binary constraints on the $x_{i,j,s}$ variables.

If no feasible timetable exists, we backtrack to the pattern set generation phase and generate a different candidate HAP set by adding the following cut where parameter p'_i is 1 if pattern $i \in Q$ is in the infeasible HAP set and 0 otherwise.

$$\sum_{\substack{i \in Q: \\ p'_i = 1}} p_i \leqslant n - 1 \quad (23)$$

We repeat this process until either a feasible HAP set is found, or all candidate HAP sets have been enumerated in which case we select another set of 500 randomly chosen patterns and repeat the procedure. For all considered feature vectors, this process took only a few seconds.

3.3.3. Constraint generation

We now construct a feasible problem instance around the auxiliary timetable obtained in the previous section by adding the required number of constraints as given by the feature vector. This is done by parameterizing the constraints from Section 2 in such a way that the auxiliary timetable always respects all hard constraints, whereas the soft constraints are parameterized in a random way. In other words, the auxiliary timetable is a feasible (but not necessarily optimal) solution to the problem instance which we create.

As an example, recall that a CA1 hard constraint imposes an upper limit on the number of home games a given team plays during a given set of time slots. In order to generate a CA1 hard constraint, we sample a random team $i \in T$ and a subset of time slots $S' \subseteq S$. Setting the upper bound to $\sum_{s \in S'} h'_{i,s}$ then results in a CA1 hard constraint which is satisfied by the auxiliary timetable. To generate a CA1 soft constraint, we instead select S' at random and we set the upper bound to $\lfloor |S'|/2 \rfloor$. We refer to Appendix B for more details on how to generate the other constraints.

3.3.4. Empirical problem hardness

In order to check that the problem instances are not too easy nor too difficult to solve, we developed a straightforward Integer Programming formulation (IP), a Constraint Programming formulation (CP), and a fix-and-optimize matheuristic. The motivation for IP and CP is to confirm that the problem instances cannot be solved to optimality with a straightforward implementation of a generic solver equipped with only limited computation time. The fix-and-optimize matheuristic, on the other hand, was chosen because of its high popularity in previous ITC competitions, and should reflect what participants are able to achieve with rather limited effort.

The main decision variable in the IP formulation is a binary variable $x_{i,j,s}$ which is one if team $i \in T$ plays against team $j \in T \setminus \{i\}$ in time slot $s \in S$ (see Appendix C). The CP formulation has two main decision variables: $o_{i,s} = j$ if team $i \in T$ plays against team $j \in T \setminus \{i\}$ in time slot $s \in S$, and $h_{i,s} = 1$ if team $i \in T$ plays at home in slot $s \in S$, and 0 if i plays away (see Appendix D). Constraint programming approaches search for a feasible solution by instantiating the variables one by one. The order in which to choose the variables is defined by the variable selection strategy. If we select the opponent variables first ('Oppon.') we mimic the well-known first-schedule-then-break strategy, whereas we mimic the first-break-then-schedule strategy if we select the pattern variables first ('Pattern').

We do not claim that these IP and CP formulations are the most efficient ones (for more advanced formulations, see e.g. Briskorn & Drexl, 2009; Ribeiro, 2012, and Trick, 2005). Rather, as the competition does not impose any run time limits, we want to make sure that a trivial IP or CP formulation does not solve the problem instances to optimality. On the other hand, the participants could be discouraged from working on the competition if even finding a feasible solution turns out to be extremely challenging. Inspired by the work by Van Bulck & Goossens (2021), we therefore also developed a relatively simple fix-and-optimize (F&O) improvement matheuristic. We believe that the F&O heuristic represents what would be possible to implement by the participants at the initial stage of the competition.

In order to construct a feasible solution, the F&O heuristic considers four different strategies. The first three strategies ignore all soft constraints, and then use the opponent CP, pattern CP, or IP formulation to construct a feasible solution. The fourth strategy starts from the so-called canonical schedule, which is a popular and well-studied schedule (see e.g. Goossens & Spieksma, 2012). It ignores all soft constraints, and relaxes all hard constraints violated by the canonical schedule into a new set of soft constraints. It then tries to solve this modified problem instance by using the improvement operators explained below, and returns to the original problem instance as soon as a zero-cost solution has been found.

After the initialization phase, the F&O heuristic tries to improve upon the initial solution by solving a series of somewhat easier IP formulations. Let us denote with parameters $x'_{i,j,s}$ the incumbent solution from the previous iteration or the initial solution in case of the first iteration. At each iteration, we then choose with uniform probability one of the following optimization problems to be solved with ILOG CPLEX and a runtime of 300 seconds (parameter d is initialized at 1 and increases with 1 every 200 iterations).

HAP(d) Draw with uniform probabilities a random subset of teams $T' \subseteq T$, $|T'| = \min(d, n)$. All games that do not involve any team from T' are fixed (i.e. $x_{i,j,s} = x'_{i,j,s}$ $\forall i, j \in T \setminus T' : i \neq j, \forall s \in S$), whereas all games that involve at least one team in T' are free to be optimized. However, the home-away pattern set must stay the same (i.e. $\sum_{j \in T \setminus \{i\}} x_{i,j,s} = \sum_{j \in T \setminus \{i\}} x'_{i,j,s}$ $\forall i \in T, \forall s \in S$).

Opponent(d) Same as HAP(d), but now the opponent schedule must stay the same (i.e. $(x_{i,j,s} + x_{j,i,s}) = (x'_{i,j,s} + x'_{j,i,s})$ $\forall i, j \in T : i \neq j, \forall s \in S$).

FixedGames(d) Same as Opponent(d) and HAP(d), but without the restriction that the HAP set or opponent schedule must stay the same.

FixedSlots(d) Draw with uniform probabilities a random subset of time slots $S' \subseteq S$, $|S'| = \min(\lfloor 1.5d \rfloor, |S|)$. All games not scheduled during time slots in S' are fixed (i.e. $x_{i,j,s} = x'_{i,j,s}$ $\forall s \in S \setminus S', \forall i, j \in T : i \neq j$), whereas all games scheduled in S' are free to be optimized.

Table 6 shows the computational results when running the monolithic IP and CP formulation and the F&O heuristic with an overall computation time of 1 hour, 16 gigabyte of RAM, and 8 threads on a CentOS 7.4 GNU/Linux based system with an Intel E5-2680 2.5 GHz processor. All IP formulations were solved with ILOG CPLEX version 12.10, and the CP formulations were solved with CPLEX CP OPTIMIZER version 12.10. For the IP model, we also show the best lower bound found within 1 hour. Table 6 shows that none of the problem instances can be solved with proven optimality by a straightforward IP or CP formulation. In fact, even just finding a feasible solution turns out to be challenging: the IP formulation finds a feasible solution for only 12 out of 45 problem instances, whereas the opponent and pattern CP formulations find feasible solutions for 16 and 15 problem instances respectively. This convinces us that the problem instances are not too easy. Neither are the problem instances overly challenging, as the F&O heuristic finds a feasible solution for the vast majority of the instances, and even finds an optimal solution to instance Late 4. The fact that the algorithms find solutions with different objective values, of which the best one is considerably lower than the objective value of the auxiliary timetable (Column 'Aux'), also suggests that the problem instances are neither overconstrained in the sense that only one or very few feasible solutions exist (making it hard for participants to compete based on solution quality). In the end, we opted not to remove and regenerate any of the problem instances as the instance set seemed to have the right difficulty.

4. Competition rules and results

The competition website (www.itc2021.ugent.be) contains the competition rules, as well as all problem instances and their best known solutions. It also offers an open-source validation tool and tutorial on the problem format, in order to lower the threshold for participation and further research as much as possible. It serves as a permanent repository for ITC2021. In this section, we discuss the most important rules, as well as the results obtained by the participants.

4.1. Competition rules and timing

Over the years, various organizers of the international timetabling competitions have wrapped their mind around developing competition rules. We are much indebted to them, as their experience has crystallized into the rules that were used for the ITC2019 competition (Müller, Rudová, & Müllerová, 2019), which we largely adopted for ITC2021. We believe the general idea behind these rules is that they guarantee an efficient and transparent competition in the sense that they do not require the organizer to run the participants' code, nor put any unnecessary burden on the

Table 6
Best found solutions for different algorithms on the set of early problem instances. The different columns for the F&O heuristic denote which method was used to generate the initial feasible solution: IP (for which we also give the best lower bound found), opponent CP, pattern CP, or relaxation of hard constraints into soft constraints. Symbol '✗' indicates that no feasible solution was found.

Instance	Aux.	IP		CP		F&O heuristic			
		LB	UB	Oppon.	Pattern	IP	Oppon.	Pattern	Rel.
Early 1	1898	1	✗	✗	✗	✗	✗	✗	753
Early 2	852	0	✗	✗	✗	✗	✗	✗	465
Early 3	1899	46	6060	3089	✗	1518	1380	1603	1467
Early 4	1828	0	✗	✗	✗	✗	✗	✗	✗
Early 5	4085	262	✗	✗	✗	✗	✗	✗	✗
Early 6	5214	0	✗	✗	✗	✗	✗	✗	✗
Early 7	8736	1306	✗	✗	✗	✗	✗	✗	✗
Early 8	3554	212	4604	3961	3344	1743	1825	1825	1834
Early 9	1557	0	5228	3488	2483	887	973	963	1013
Early 10	4919	321	✗	✗	✗	✗	✗	✗	✗
Early 11	8724	336	✗	✗	✗	✗	✗	✗	7364
Early 12	1125	0	✗	✗	✗	1780	✗	✗	✗
Early 13	1254	2	✗	✗	✗	450	✗	✗	388
Early 14	1103	1	9038	3889	3379	1072	949	976	1055
Early 15	5699	477	✗	6612	7154	6403	✗	7285	5811
Middle 1	6296	2912	✗	✗	✗	✗	✗	✗	✗
Middle 2	7465	2959	✗	✗	✗	✗	✗	✗	✗
Middle 3	10470	3350	✗	✗	✗	✗	✗	✗	✗
Middle 4	385	7	118	227	✗	36	29	31	31
Middle 5	1177	46	6627	2956	4691	1179	✗	1105	1031
Middle 6	2855	0	✗	✗	✗	✗	✗	✗	✗
Middle 7	6341	31	✗	✗	✗	3951	✗	✗	3893
Middle 8	1057	2	✗	920	887	407	379	465	444
Middle 9	2245	0	✗	3450	3015	1650	1680	1530	1670
Middle 10	2585	4	✗	✗	✗	✗	✗	✗	✗
Middle 11	4318	344	✗	✗	✗	✗	✗	✗	✗
Middle 12	2517	0	✗	3886	✗	3333	2506	✗	1873
Middle 13	4699	0	✗	✗	6398	3410	✗	2623	2461
Middle 14	3072	0	✗	✗	✗	2406	✗	✗	1938
Middle 15	6493	1	9612	7133	7061	1810	1542	1962	1830
Late 1	3301	1105	✗	✗	✗	✗	✗	✗	2509
Late 2	6239	2834	✗	✗	✗	✗	✗	✗	✗
Late 3	5714	401	8633	7273	6763	4179	3678	3719	3669
Late 4	956	0	68	✗	1023	0	✗	0	0
Late 5	2264	396	✗	✗	✗	✗	✗	✗	✗
Late 6	2454	5	✗	✗	✗	1371	✗	✗	1268
Late 7	6121	5	✗	✗	✗	3072	✗	✗	3123
Late 8	2656	74	5059	3680	4121	1539	1612	1534	1580
Late 9	2519	3	3485	3039	2625	1984	1920	1995	1844
Late 10	3187	1	✗	✗	✗	✗	✗	✗	✗
Late 11	1016	0	✗	✗	✗	✗	✗	✗	786
Late 12	7739	213	✗	✗	✗	✗	✗	✗	5964
Late 13	5744	0	✗	✗	✗	7996	✗	✗	6048
Late 14	3513	7	✗	3230	3400	✗	2501	2323	2180
Late 15	1400	0	6185	4180	2770	1150	1160	1070	930

participants, and that evaluation criteria are simple and unambiguous.

An important rule was that no limit was imposed on the computation time. In fact, the objective function value of the solution, as reported by the open-source validator on the competition website, was the only criterion that mattered. While computation time is obviously not unimportant, a fair comparison in terms of computation time is quite challenging (even more so for further research after the competition closed). It would have involved running all code on the same system, or compensating for differences in hardware, all of which would have required extra work from participants as well as organizers, and could easily have led to disputes. Moreover, from a practical point of view, sports timetabling problems are often not so time-critical, as there are often several days or even weeks available to obtain a good solution.

We also opted to allow the use of any commercial solver, again motivated by lowering the threshold of participation and reaching out to the largest possible research community. Due to the fact that we made sure that the instances were challenging computationally (see Section 3.3.4), we did not worry that the instances would solve with a straightforward formulation on e.g. state-of-the-art IP solvers, rendering the competition uninteresting.

Although we allowed parameter tuning, we required participants to use the same version of their algorithm for all instances, since we are looking for a solver that can cover a wide range of realistic problems. While their algorithm may analyse the problem instance and set parameters accordingly, it should apply this same procedure for all instances. In other words, participants should not set different parameters for different instances manually, but it is acceptable if their code is doing this automatically.

In total, we released three groups of 15 artificially generated problem instances each: early, middle, and late instances. As indicated in the timeline given in Fig. 5, the early group of instances were available as soon as the competition was officially announced (mid October 2020), while the middle group of instances was only

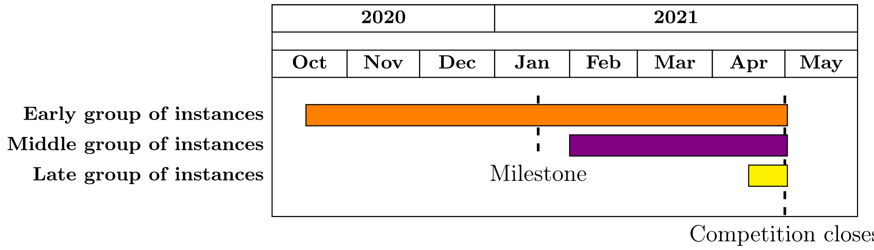

Fig. 5. Timeline for the International Timetabling Competition 2021.

Table 7
Overview of points awarded per position for instances from each group.

	Instance		
Position	Early	Middle	Late
1st	10	15	25
2nd	7	11	18
3rd	5	8	15
4th	3	6	12
5th	2	4	10
6th	1	3	8
7th		2	6
8th		1	4
9th			2
10th			1

released in February 2021. The late instances followed half April 2021, which gave the participants two weeks to come up with solutions.

The competition awarded points to each solution based on the position among its competitors and the instance group (early, middle or late). For each competitor and each instance, only the best submitted feasible solution was considered. The top six, eight, and ten competitors scored points according to the scale in Table 7; note that instances that were released later in the competition were worth more points. When two or more solutions tied for the same position, the points granted by these positions were split evenly between competitors (rounded up in case of fractional points). When a solver did not compute any feasible solution for some instance, it was awarded zero points for that instance. The winner of the competition is the participant with the highest total number of points over all instances.

In order to have a more lively competition, we organized a milestone event mid-January 2021 where participants had the possibility to submit their best solutions found at that time. Although optional, participation in the milestone was strongly encouraged as it provided participants with the feedback on where their algorithms ranked among their peers as well as a chance to win a

Table 8
Overview of competition participants, ordered according to their final rank in the competition.

Team	Participants	Institution (Country)	Search method
UoS	Carlos Lamas-Fernández Toni Martínez-Sykora Chris Potts	University of Southampton (UK)	Matheuristics (Lamas-Fernandez et al., 2021)
Udine	Roberto Maria Rosati Matteo Petris Luca Di Gaspero Andrea Schaerf	University of Udine (Italy)	Metaheuristics (Rosati et al., 2022)
Saturn	Daniil Sumin Ivan Rodin	Higher School of Economics (Russia)	IP Decomposition (Sumin & Rodin, 2021)
GOAL	George H. G. Fonseca Túlio A. M. Toffolo	Federal University of Ouro Preto (Brazil)	Matheuristics (Fonseca & Toffolo, 2022)
MODAL	Timo Berthold Thorsten Koch Yuji Shinano	Zuse Institute Berlin (Germany)	IP + Metaheuristics (Berthold, Koch, & Shinano, 2021)
TU/e	Jasper van Doornmalen Christopher Hojny Roel Lambers Frits Spieksma	Eindhoven University of Technology (Netherlands)	Matheuristics (van Doornmalen, Hojny, Lambers, & Spieksma, 2021)
DES	Antony E. Phillips Michael O'Sullivan Cameron Walker	University of Auckland (New Zealand)	Matheuristics (Phillips, O'Sullivan, & Walker, 2021)
Gionar	Giorgio Sartor Bjørnar Luteberget	SINTEF (Norway)	
DITUoI Arta	Angelos Dimitsas Christos Valouxis Christos Gogos	University of Ioannina (Greece)	CP + Metaheuristics (Dimitsas et al., 2022)
NHH	Ole Stordal Subba Elias	Norwegian School of Economics (Norway)	IP Decomposition (Subba & Stordal, 2021)
Aures	Arbaoui Taha Athmani Mohamed Elamine Henni Mohammed Terzi Mourad	Troyes University of Technology (France)	
UoR	Martin Lester	University of Reading (UK)	Pseudoboolean optimization (Lester, 2022)
Team zero	Swarup Ghadiali	IIT Bombay (India)	

Table 9
Final ranking of the ITC2021 participants; top 6 are finalists.

Team	Points				Feas. sol.	Best sol.
	Early	Middle	Late	Total		
1. UoS	121	178	297	596	45	21
2. Udine	75	114	235	424	44	4
3. Saturn	64	115	207	386	37	16
4. GOAL	38	72	133	243	37	4
5. MODAL	21	65	150	236	40	4
6. TU/e	41	47	136	224	38	2
7. DES	8	42	72	122	37	3
8. Gionar	25	16	68	109	40	3
9. DITUoI Arta	4	29	68	101	37	2
10. NHH	5	13	70	88	40	1
11. Aures	0	1	12	13	31	1
12. UoR	0	0	10	10	29	1
13. Team zero	0	0	5	5	26	0

Table 10
Overview of the best found results for each instance at the end of the competition.

Instance	Early		Middle		Late	
	Best found	Team	Best found	Team	Best found	Team
1	362	UoS	5177	UoS	1969	UoS
2	160	Saturn	7381	UoS	5400	UoS
3	1012	Saturn	9701	MODAL	2369	UoS
4	512	Saturn	7	*7 teams*	0	*11 teams*
5	3127	UoS	413	MODAL	1939	Saturn
6	3352	Saturn	1125	Saturn	923	UoS
7	4763	UoS	1784	Saturn	1558	Saturn
8	1064	GOAL	129	UoS	934	UoS
9	108	UoS	450	UoS	563	UoS
10	3400	UoS	1250	UoS	1988	Saturn
11	4436	UoS	2511	Saturn	207	Udine
12	380	Saturn	911	DES	3689	Saturn
13	121	UoS	253	Saturn	1820	GOAL
14	4	Gionar	1172	Udine	1206	Udine
15	3368	UoS	495	Saturn	20	Saturn & TU/e

free registration for the Mathsport International 2022 conference. Thanks to our sponsors, the EURO working groups PATAT and OR in Sports, we could split 1750 EUR prize money between the top 3 competitors, besides discounts on the registration for the PATAT 2022 conference.

4.2. Results

At the time of the final submission deadline, 13 research teams from 11 different countries submitted solutions to be taken into account for the final ranking; they are listed in Table 8. This is a solid participation, compared to the cross-domain heuristic search challenge (17 teams), the two international nurse rostering competitions (15 teams each), and the third and fourth international timetabling competition (5 teams each).

Six finalists were selected from the participating teams, whose final ranking was announced at the Mathsport International 2021 conference (Van Bulck, Goossens, Belien, & Davari, 2021), and is given in Table 9. Team UoS, from the University of Southamptom, has a clear lead over the team from the University of Udine (second) and team Saturn, from the Higher School of Economics, Moscow (third). Team UoS has the best score over the early, middle, and the late instances, and also won the milestone, before Udine and TU/e. Team UoS was also the only team to find a feasible solution for each instance, and they top the number of instances for which they found a best solution. While their victory is clearly well deserved, this does not mean that the other teams have no merits. Indeed, many teams found at least two best solutions and for more than half of the instances, one of the other teams beat team UoS.

Shortly after the competition ended, we queried the participants on the method they used (see Table 8). In the meantime, 10 participants have written preliminary descriptions of their approach (often in the proceedings of the PATAT 2021 conference and the PATAT 2021 special issue in Journal of Scheduling), and at the time of writing, many of them are working on a full paper. Team UoS (winners) used an IP-based matheuristic, in which they iteratively fix many variables in their IP model, in order to reduce the computation time (Lamas-Fernandez, Martinez-Sykora, & Potts, 2021). Team Udine (second) used a simulated annealing strategy, using five neighborhoods from the literature, as well as one neighborhood they developed specifically for the competition instances (Rosati, Petris, Di Gaspero, & Schaerf, 2022). Team Saturn (third) has used a more classic though novel IP decomposition approach (Sumin & Rodin, 2021). It is clear that a variety of methods has been applied in this competition, including e.g. pseudoboolean optimization (Lester, 2022), which had barely been explored in the context of sports timetabling before. We refer the readers to these publications for more details.

Table 10 gives an overview of the best results found by the participants at the time the competition ended. On all but three instances, a single best solution was found. Two instances, middle 4 and late 4, turned out less discriminating, since 7 and 11 groups respectively found the same best solution, which turned out optimal.

5. Conclusion

In this paper, we have described the organization and the outcome of the fifth international timetabling competition (ITC2021), which is the first to focus on sports timetabling. We see the generation of a set of challenging, realistic and diverse problem instances as the main challenge in organizing ITC2021. As part of this process, we proposed a novel integer programming approach to determine the (high-dimensional) feature values that correspond with target coordinates in the two-dimensional instance space. Ultimately, this allowed us to generate problem instances that fill the gaps in the instance space. While this approach has its limitations (e.g. constraints related to the presence of features are linear), it is applicable in a wider context than sports timetabling. Indeed, we think it could be a valuable method to generate a diverse set of problem instances via the use of instance generators that require parameters corresponding to features (e.g. number of vertices or graph density in a network generator), possibly linked by constraints, such as quadratic assignment, multi-dimensional knapsack, educational timetabling, graph colouring, or Boolean satisfiability (Bowly, Smith-Miles, Baatar, & Mittelmann, 2019). We believe that an increased focus on guaranteeing instance diversity would be of added value to the existing instance generators in timetabling problems.

We believe ITC2021 has been a success, which has considerably contributed to the (sports) timetabling community. This is motivated by the large number of teams that participated in the competition. Based on the fact that all participants managed to find a feasible solution for at least half of the instances (while no instance turned out too hard to find a feasible solution for all participants), and the spread of the best found solutions over the participants, we believe that the set of instances was not too easy nor too challenging to solve. Perhaps more than we anticipated at first, the results of this competition show that it is possible to build a generic solver that handles the wide variety of constraints that are common in sports timetabling. We hope that this motivates the research community to further explore the development of generic solvers, rather than dedicated algorithms that are applicable only to one very specific sports competition. At the same time, we learned that there is no single best solver. Figuring out which aspects of the problem are good predictors of the performance of an algorithm is an important future research topic that can build on the results of this competition.

Finally, we want to point out that even though the competition has ended and prizes have been awarded, the set of instances generated for the competition remains important. Only two instances have been solved to optimality so far, so they remain challenging for future algorithmic approaches. In fact, this paper may serve as a general guide for anyone who wants to work on the problem instances in the future. It would also be natural to include these instances as a benchmark for future publications on sports timetabling algorithms. During the competition, the ITC website was accessed from all over the world, and even months after the competition closed, our website is still actively used, also beyond academia. For instance, the ITC2021 instances will be used in a hackathon organized by the company Baobab Soluciones. We are also aware of a number of ongoing master theses on this topic, and a number of papers have appeared by research teams that did not officially participate in the competition (see e.g. Alsmadi et al., 2022; Hutama & Muklason, 2021). Improved solutions and lower bounds have also been reported to us after the closing of the competition, and the competition website will keep track of these.

Acknowledgments

David Van Bulck is a postdoctoral research fellow funded by the Research Foundation - Flanders (FWO) [1258021N]. We are grateful to Jeroen Beliën and Morteza Davari for co-organizing ITC2021 and for their constructive feedback on this paper.

Appendix A. File format

Problem instances and solutions in the ITC2021 competition are expressed using the RobinX XML data format. In the remainder of this section, we outline the structure of both formats.

A.1. Meta data, resources, and structure

A general overview of the instance file format is given in Fig. A.6. The meta data of a problem instance (see Fig. A.7) consists of a unique instance name, the data type which is always artificial ('A'), the contributor which is always 'ITC2021', and the date on which the problem instance was released.

The resources in an ITC2021 problem instance consist of the teams, time slots, and a single league in which all teams play (see Fig. A.8). Recall from Section 2 that the total number of teams is either 16, 18, or 20. The total number of time slots is thus 30, 34, or 38, respectively.

The structure of the competition format is always a compact 2RR. The `gameMode` tag has value 'P' if the timetable needs to be phased, and has value 'NULL' otherwise (see Fig. A.9).

```
<Instance>
  <MetaData> ... </MetaData>
  <Resources> ... </Resources>
  <Structure> ... </Structure>
  <Constraints>
    <CapacityConstraints/>
    <GameConstraints/>
    <BreakConstraints/>
    <FairnessConstraints/>
    <SeparationConstraints/>
  </Constraints>
  <ObjectiveFunction>
    SC
  </ObjectiveFunction>
</Instance>
```

Fig. A6. Main structure of the problem instance XML file format.

```
<MetaData>
  <InstanceName>Example.xml</InstanceName>
  <DataType>A</DataType>
  <Contributor>ITC2021</Contributor>
  <Date year="2020" month="10"/>
</MetaData>
```

Fig. A7. XML specification for the meta data of the problem instance.

```
<Resources>
  <Leagues>
    <league id="0" name="League 0"/>
  </Leagues>
  <Teams>
    <team id="0" league="0" name="T1"/>
  </Teams>
  <Slots>
    <slot id="0" name="Slot 0"/>
  </Slots>
</Resources>
```

Fig. A8. Specification for the resources of the problem instance.

```
<Structure>
  <Format leagueIds="0">
    <numberRoundRobin>
      2
    </numberRoundRobin>
    <compactness>C</compactness>
    <gameMode>P</gameMode>
  </Format>
</Structure>
```

Fig. A9. Specification for the round-robin structure of the problem instance.

A.2. Constraints and objective

Note that the set of constraints is a partition of hard constraints (C_{hard}) and soft constraints (C_{soft}). For each constraint c, the `type` attribute denotes whether a constraint is hard ('HARD') or soft ('SOFT'). The validation of constraint c results in a vector of n_c integral numbers, called the deviation vector $D_c = [d_1 \ d_2 \ \dots \ d_{n_c}]$. The deviation vector of a violated constraint contains one or more non-zero elements. The objective in the ITC2021 problem instances is to minimize the penalties from violated soft constraints while respecting all hard constraints (value 'SC' in the `ObjectiveFunction` tag of Fig. A.6). More in particular, the objective is to minimize $\sum_{c \in C_{\text{soft}}} w_c \sum_{i=1}^{n_c} d_i$ where w_c is a weight given by the `penalty` attribute of each soft constraint.

In the remainder of this section, we give the syntax, deviation vector, and a specific example for each of the nine constraint types already briefly outlined in Section 2. For a more formal description of the constraints and their deviation vector, we refer to Van Bulck et al. (2020b).

< CA1 teams="0" max="0" mode="H" slots="0" type="SOFT"/>

The team in `teams` (always a singleton) plays at most `max` home games (`mode` = "H") or away games (`mode` = "A") during time slots in `slots`.
The team in `teams` triggers a deviation equal to the number of home games (`mode` = "H") or away games (`mode` = "A") in `slots` more than `max`.
Team 0 cannot play at home in time slot 0.

<CA2 teams1="0" max="1" mode1="HA" mode2="GLOBAL" teams2="1;2" slots ="0;1;2" type="SOFT"/>

The team in `teams1` (always a singleton) plays at most `max` home games (`mode1` = "H"), away games (`mode1` = "A"), or games (`mode1` = "HA") against teams (`mode2` = "GLOBAL"; the only mode we consider) in `teams2` during time slots in `slots`.
The team in `teams1` triggers a deviation equal to the number of home games (`mode1` = "H"), away games (`mode1` = "A"), or games (`mode1` = "HA") against teams in `teams2` during time slots in `slots` more than `max`.
Team 0 plays at most one game against teams 1 and 2 during the first three time slots.

<CA3 teams1="0" max="2" mode1="HA" teams2="1;2;3" intp="3" mode2="SLOTS" type="SOFT"/>

Each team in `teams1` plays at most `max` home games (`mode1` = "H"), away games (`mode1` = "A"), or games (`mode1` = "HA") against teams in `teams2` in each sequence of `intp` time slots (`mode2` = "SLOTS"; the only mode we consider).
Each team in `teams1` triggers a deviation equal to the sum of the number of home games (`mode1` = "H"), away games (`mode1` = "A"), or games (`mode1` = "HA") against teams in `teams2` more than `max` for each sequence of `intp` time slots.
Team 0 plays at most two consecutive games against teams 1, 2, and 3.

<CA4 teams1="0;1" max="3" mode1="H" teams2="2,3" mode2="GLOBAL" slots="0;1" type="SOFT"/>

Teams in `teams1` play at most `max` home games (`mode1` = "H"), away games (`mode1` = "A"), or games (`mode1` = "HA") against teams in `teams2` during time slots in `slots` (`mode2` = "GLOBAL") or during each time slot in `slots` (`mode2` = "EVERY").
The set `slots` (`mode2` = "GLOBAL") or each time slot in `slots` (`mode2` = "EVERY") triggers a deviation equal to the number of games (i, j) (`mode1` = "H"), (j, i) (`mode1` = "A"), or (i, j) and (j, i) (`mode1` = "HA") with i a team from `teams1` and j a team from `teams2` more than `max`.
Teams 0 and 1 together play at most three home games against teams 2 and 3 during the first two time slots.

<GA1 min="0" max="0" meetings="0,1;1,2;" slots ="3" type="SOFT"/>

At least `min` and at most `max` games from `meetings` = $\{i_1, j_1; i_2, j_2; \dots\}$ take place during time slots in `slots`.
The set `slots` triggers a deviation equal to the number of games in `meetings` less than `min` or more than `max`.
Game (0,1) and (1,2) cannot take place during time slot 3.

<BR1 teams="0" intp="0" mode2="HA" slots="1" type="SOFT"/>

The team in `teams` (always a singleton) has at most `intp` breaks (`mode2` = "HA", the only mode we consider) during time slots in `slots`.
The team in `teams` triggers a deviation equal to the difference in the sum of breaks during time slots in `slots` more than `intp`.
Team 0 cannot have a break in time slot 1.

<BR2 homeMode="HA" teams="0;1" mode2="LEQ" intp="2" slots ="0;1" type="SOFT"/>

The sum over all breaks (`homeMode` = "HA", the only mode we consider) for teams in `teams` (always containing all teams) is no more than (`mode2` = "LEQ", the only mode we consider) `intp` during time slots in `slots` (always containing all time slots).
The set `teams` triggers a deviation equal to the number of breaks in the set `slots` more than `intp`.
Team 0 and 1 together do not have more than two breaks during the first four time slots.

Each pair of teams in `teams` (always containing all teams) has a difference in played home games (`mode` = "H", the only

```
<Solution>
  <MetaData>
    <InstanceName>Example.xml</InstanceName>
    <SolutionName>ExampleSol.xml</SolutionName>
    <ObjectiveValue infeasibility="0" objective="2"/>
  </MetaData>
  <Games>
    <ScheduledMatch home="1" away="2" slot="1">
    ...
  </Games>
</Solution>
```

Fig. A10. Specification for a solution to a problem instance.

<FA2 teams="0;1;2" mode="H" intp="2" slots ="0;1;2;3" type="SOFT" penalty="10"/>

mode we consider) that is not larger than `intp` after each time slot in `slots` (always containing all time slots).
Each pair of teams in `teams` triggers a deviation equal to the largest difference in played home games more than `intp` over all time slots in `slots`.
The difference in home games played between the first three teams is not larger than 2 during the first four time slots.

<SE1 teams="0;1" min="5" mode1="SLOTS" type="SOFT" penalty="10"/>

Each pair of teams in `teams` (always containing all teams) has at least `min` time slots (`mode1` = "SLOTS", the only mode we consider) between two consecutive mutual games.
Each pair of teams in `teams` triggers a deviation equal to the sum of the number of time slots less than `min` or more than `max` for all consecutive mutual games.[2]
There are at least 10 time slots between the mutual games of team 0 and 1.

A.3. Solution file format

In order to store solutions, we also make use of RobinX (see Fig. A.10). The `metaData` tag stores the name of the instance XML file, the name of the generated solution, and the objective value which consists of the sum of violated hard constraints penalties (`infeasibility` attribute, which should be zero) and the sum of violated soft constraints penalties (`objective` attribute). Next comes a `games` tag which enumerates the time slot assigned to each game of the tournament. The competition website provides access to a validator, allowing participants to verify whether a solution expressed in this format satisfies all hard constraints and to determine its score on the objective function.

Appendix B. Adding constraints around the auxiliary timetable

We use the notation $q \sim Pr(q_1, q_2, \dots, q_n \mid r_1, r_2, \dots, r_n)$ to denote that a discrete random variable q follows a discrete probability distribution where each possible value q_i has a selection probability of $r_i / \sum_{1 \le k \le n} r_k$. Furthermore, we use function $U(a, b)$ to refer to the discrete uniform distribution with minimal value a and maximal value b (i.e. $U(a, b) = Pr(a, a+1, \dots, b-1, b \mid 1, 1, \dots, 1, 1)$). Given the auxiliary timetable where the HAP of team $i \in T$ in time slot $s \in S$ is denoted by parameters $h'_{i,s}$ and the assignment of opponents by parameters $x'_{i,j,s}$ with $j \in T \setminus \{i\}$ (see Section 3), the constraints are then generated as follows.

CA1 Draw a team $i \sim U(1, |T|)$ and set $\texttt{teams} = \{i\}$.
- Hard Draw a random subset of time slots $S' \subseteq S$ with $|S'| \sim Pr(1, 2, 3, 4 \mid 2, 1, 1, 1)$ and set $\texttt{slots} = S'$. Let $n_h = \sum_{s \in S'} h'_{i,s}$ and let $n_a = |S'| - n_h$. If $n_h < n_a$, set $\texttt{mode} =$ "H" and $\texttt{max} = \max(n_h, |S'| > 1)$; otherwise, set $\texttt{mode} =$ "A" and $\texttt{max} = \max(n_a, |S'| > 1)$.
- Soft Set $\texttt{penalty} = 1$ and set $\texttt{slots} = S'$ with $|S'| \sim Pr(1, 2, 3, 4, 5, 6 \mid 30, 14, 14, 14, 14, 14)$. With uniform probability, choose $\texttt{mode} =$ "H" or "A", and set $\texttt{max} = \max(\lfloor |S'|/2 \rfloor - 1, |S'| > 1)$.

CA2 Draw a team $i \sim U(1, |T|)$, set $\texttt{teams1} = \{i\}$, and set with uniform probability $\texttt{mode1} =$ "H", "A", or "HA".
- Hard If `mode1` is "HA", draw a random subset of time slots $S' \subseteq S$ with $|S'| \sim Pr(4, 5, 6 \mid 1, 1, 1)$ and set $\texttt{slots} = S'$; otherwise draw a time slot $s \sim U(1, |S|)$ and set $\texttt{slots} = \{s\}$. Let $T'_h = \{j \in T : x'_{i,j,s} = 1, s \in S'\}$, $T'_a = \{j \in T : x'_{j,i,s} = 1, s \in S'\}$, and $T' = T'_h \cup T'_a$. If `mode1` is "HA", draw a random subset of teams $U \subseteq T$ such that $|U \setminus T'| = \lceil |S'|/2 \rceil + 1$ and $|U \cap T'| = \lfloor |S'|/2 \rfloor - 1$, set $\texttt{teams2} = U$ and set $\texttt{max} = |\{x'_{i,j,s} : j \in T', s \in S'\} \cup \{x'_{j,i,s} : j \in T', s \in S'\}|$. If `mode1` is "H" or "A", with uniform probability, draw a team i from $T \setminus T'_h$ (or $T \setminus T'_a$), set $\texttt{teams2} = \{i\}$ and set $\texttt{max} = 0$.
- Soft Set `penalty=5`, draw a random subset of time slots $S' \subseteq S$ with $|S'| \sim Pr(6, 7, 8 \mid 1, 1, 1)$, set $\texttt{slots} = S'$, and draw a random subset of teams $U \subseteq T \setminus \{i\}$ with $|U| = |S'|$. If `mode1` is "H" or "A" set $\texttt{max} = \lfloor |\texttt{slots}|/2 \rfloor - 1$, and set $\texttt{max} = \lfloor |\texttt{slots}|/2 \rfloor - 2$ otherwise.

CA3 Recall that there are at most two CA3 hard constraints.
- Hard With uniform probability, choose $\texttt{mode1} =$ "H" or "A". Set $\texttt{teams1} = \texttt{teams2} = T$, $\texttt{max} = 2$, and $\texttt{intp} = 3$.
- Soft Set $\texttt{penalty} = 5$. Choose with uniform probability $\texttt{mode1} =$ "H", "A" or "HA", and draw a team $i \sim U(1, |T|)$ with $\texttt{teams1} = \{i\}$. Draw a random subset of teams $U \subseteq T$ with $|U| \sim Pr(5, 6, 7 \mid 1, 1, 1)$, and set $\texttt{teams2} = U$, $\texttt{max} = 2$, and $\texttt{intp} = 4$.

CA4 Scenario 1: simultaneous games between top teams (70% probability). Draw a random subset of teams $T' \subseteq T$ with $|T'| \sim Pr(4, 5, 6 \mid 1, 1, 1)$, set $\texttt{teams1} = \texttt{teams2} = T'$, `mode1`="H", and `mode2`="GLOBAL".
- Hard Collect a set of time slots S' with $|S'| = |T'|$ such that $a = |\{x'_{i,j,s} : i, j \in T', s \in S'\}|$ is minimal, and set $\texttt{slots} = S'$ and $\texttt{max} = \max(a, 2)$.

[2] If two teams play against each other in time slots s_1 and s_2 and there should be k time slots in between, deviation is given by $\max(k - (s_2 - s_1 - 1); 0)$ (and not $\max(k - (s_2 - s_1); 0)$ as stated in Van Bulck et al. (2020b)).

Soft Set $\texttt{penalty} = 5$, draw a random set of time slots $S' \subseteq S$ with $|S'| = |T'|$, and set $\texttt{slots} = S'$, $\texttt{max} = \lfloor |T'|/2 \rfloor - 1$.

Scenario 2: complementary HAPs (e.g. two teams share a stadium; 30% probability). Choose with uniform probability two teams $i, j \in T : i \neq j$, and set $\texttt{teams1} = \{i, j\}$, $\texttt{mode1} =$ "H", $\texttt{max} = 1$, $\texttt{teams2} = T$ and mode2="EVERY".

Hard Denote with $S' = \{s \in S : h'_{i,s} = 0 \vee h'_{j,s} = 0\}$, draw a time slot $s \in S'$, and set $\texttt{slots} = \{s\}$.

Soft Set $\texttt{penalty} = 5$, draw a time slot $s \sim U(1, |S|)$, and set $\texttt{slots} = \{s\}$.

GA1 Select with uniform probability a set of games G such that $|G| \sim Pr(1, 2, 3, 4 \mid 5, 1, 1, 1)$, and set $\texttt{meetings} = G$ and $S' = \{s \in S : \exists x'_{i,j,s} = 1, (i, j) \in G\}$. We now consider two scenarios: scenario 1 (forbidden slots; 60% probability) where we set $\texttt{max} = \lfloor |G|/2 \rfloor$, and scenario 2 (fixed slots; 40% probability) where we set $\texttt{min} = \lceil |G|/2 \rceil$.

Hard Collect a set of time slots $S' \subseteq S \setminus \{s \in S : \exists x'_{i,j,s} = 1, (i, j) \in G\}$ with $|S'| = |G|$, and set $\texttt{slots} = S'$.

Soft Collect a set of time slots $S' \subseteq S$ with $|S'| = |G|$, and set $\texttt{slots} = S'$.

BR1 Draw a team $i \sim U(1, |T|)$ and set $\texttt{teams} = \{i\}$.

Hard Draw a random subset of time slots $S' \subseteq S$ with $|S'| \sim Pr(1, 3, 6 \mid 1, 1, 1)$ and such that i has $\lfloor |S'|/3 \rfloor$ breaks in the auxiliary timetable. Set $\texttt{slots} = S'$ and $\texttt{intp} = \lfloor |S|/3 \rfloor$.

Soft Set $\texttt{penalty} = 5$ and draw a random subset of time slots S' with $|S'| \sim Pr(1, 3, 6 \mid 1, 1, 1)$. Set $\texttt{slots} = S'$ and $\texttt{intp} = \lfloor |S'|/3 \rfloor$.

BR2 Recall that there is at most one BR2 hard or soft constraint. Set $\texttt{teams} = T$ and $\texttt{slots} = S$.

Hard Denote with a the total number of breaks in the auxiliary timetable, and set $\texttt{intp} = a$.

Soft Set $\texttt{penalty} = 10$ and $\texttt{intp} = |T| - 2$.

FA2 Recall that we only consider the soft constraint mode of FA2, and that there is at most one FA2 soft constraint.

Soft Set $\texttt{penalty} = 10$, $\texttt{teams} = T$, $\texttt{slots} = S$, $\texttt{mode} =$ "H", and $\texttt{intp} = 2$.

SE1 Recall that we only consider the soft constraint mode of SE1, and that there is at most one SE1 soft constraint.

Soft Set $\texttt{penalty} = 10$, $\texttt{teams} = T$, $\texttt{min} = 10$, and $\texttt{mode1} =$ "SLOTS".

Appendix C. IP formulation

The following IP formulation is based on Briskorn & Drexl (2009). Some of the constraints in the ITC2021 format offer a set of options to specify different variants of the constraint. In this case, the formulation always assumes that the first option is chosen. For all other options, we assume that the reader can adapt the formulation accordingly.

Variables

$x_{i,j,s} = 1$ if home team $i \in T$ plays against away team $j \in T \setminus \{i\}$ in time slot $s \in S$, 0 else

$b_{i,s} = 1$ if team $i \in T$ has a break in time slot $s \in S \setminus \{1\}$ and $s - 1$, 0 else

$e_c =$ excess on constraint $c \in C_{\text{soft}} \cup C_{\text{hard}}$

$s_c =$ slack on constraint $c \in C_{\text{soft}} \cup C_{\text{hard}}$

$\mathbf{d_c} =$ deviation vector of constraint $c \in C_{\text{soft}} \cup C_{\text{hard}}$

$$\textbf{minimize} \quad \sum_{c \in C_{\text{soft}}} \texttt{penalty}_c \, \mathbf{d_c} \qquad \text{(C1)}$$

Round-robin constraints

$$\sum_{j \in T \setminus \{i\}} (x_{i,j,s} + x_{j,i,s}) = 1 \qquad \forall i \in T, s \in S \quad \text{(C2)}$$

$$\sum_{s \in S} x_{i,j,s} = 1 \qquad \forall i, j \in T, i \neq j \quad \text{(C3)}$$

$$\sum_{s=1}^{n-1} (x_{i,j,s} + x_{j,i,s}) = 1 \qquad \forall i, j \in T : i < j \quad \text{(C4)}$$

Capacity constraints

$$\sum_{j \in T \setminus \{i\}} \sum_{s \in \texttt{slots}_c} x_{i,j,s} - d_{c,i} \leq \texttt{max}_c \qquad \forall i \in \texttt{teams}_c, \forall c \in \text{CA1} \quad \text{(C5)}$$

$$\sum_{j \in \texttt{teams2}_c} \sum_{s \in \texttt{slots}_c} x_{i,j,s} - d_{c,i} \leq \texttt{max}_c \qquad \forall i \in \texttt{teams1}_c, \forall c \in \text{CA2} \quad \text{(C6)}$$

$$\sum_{j \in \texttt{teams2}_c} \sum_{p=s}^{s+\texttt{intp}_c-1} x_{i,j,p} - d_{c,i,s} \leq \texttt{max}_c$$
$$\forall i \in \texttt{teams1}_c, \forall s \in S : s < |S| - \texttt{intp}_c + 1, \forall c \in \text{CA3} \quad \text{(C7)}$$

$$\sum_{i \in \texttt{teams1}_c} \sum_{j \in \texttt{teams2}_c} \sum_{s \in \texttt{slots}_c} x_{i,j,s} - d_c \leq \texttt{max}_c \qquad \forall c \in \text{CA4} \quad \text{(C8)}$$

Game constraints

$$\texttt{min}_c - d_c \leq \sum_{(i,j) \in \texttt{meetings}_c} \sum_{s \in \texttt{slots}_c} x_{i,j,s} \leq \texttt{max}_c + d_c \qquad \forall c \in \text{GA1} \quad \text{(C9)}$$

Break constraints

$$\sum_{j \in T \setminus \{i\}} (x_{i,j,s-1} + x_{i,j,s}) - b_{i,s} \leq 1 \qquad \forall i \in T, s \in S \setminus \{1\} \quad \text{(C10)}$$

$$\sum_{j \in T \setminus \{i\}} (x_{j,i,s-1} + x_{j,i,s}) - b_{i,s} \leq 1 \qquad \forall i \in T, s \in S \setminus \{1\} \quad \text{(C11)}$$

$$\sum_{s \in \texttt{slots}_c} b_{i,s} - d_{c,i} \leq \texttt{intp}_c \qquad \forall i \in \texttt{teams}_c, \forall c \in \text{BR1} \quad \text{(C12)}$$

$$\sum_{s \in \texttt{slots}_c} \sum_{i \in \texttt{teams}_c} (b^h_{i,s} + b^a_{i,s}) - d_c \leqslant \texttt{intp}_c \qquad \forall c \in \text{BR2} \quad \text{(C13)}$$

Fairness and separation constraints

$$\sum_{k \in T} \sum_{\substack{p \in S: \\ p \leqslant s}} (x_{i,k,s} - x_{j,k,s}) - d_{c,\{i,j\}} \leqslant \texttt{intp}_c$$
$$\forall i, j \in \texttt{teams}_c : i \neq j, \forall s \in \texttt{slots}_c, \forall c \in \text{FA2} \quad \text{(C14)}$$

$$(\texttt{min}_c - (s_2 - s_1 - 1))(x_{i,j,s_1} + x_{j,i,s_1} + x_{i,j,s_2} + x_{j,i,s_2} - 1) \leqslant d_{c,\{i,j\}}$$
$$\forall i, j \in \texttt{teams}_c : i < j,$$
$$s_1, s_2 \in S : s_1 < s_2 \leqslant s_1 + \texttt{min}_c, \forall c \in \text{SE1} \quad \text{(C15)}$$

Binary constraints

$$x_{i,j,s} \in \{0, 1\} \qquad \forall i, j \in T : i \neq j, s \in S \quad \text{(C16)}$$

$$b_{i,s} \in \{0,1\} \quad \forall i \in T, s \in S \setminus \{1\} \quad (C17)$$

$$d_c \geq 0 \quad \forall c \in C_{\text{soft}} \quad (C18)$$

$$d_c = 0 \quad \forall c \in C_{\text{hard}} \quad (C19)$$

We only explain the round-robin constraints, and assume that all other constraints are self-explanatory. Constraints (C.2) state hat each team plays exactly one game per time slot. Furthermore, Constraints (C.3) state that each game of the 2RR is scheduled. Finally, if the tournament needs to be phased, Constraints (C.4) is added.

Appendix D. CP formulation

The following CP formulation again assumes that the first option of each constraint is chosen (see Appendix C), and is based on Easton (2003); Régin (2001); Trick (2003). Recall from Section 3.3.1 that $\texttt{distribute}(cards, values, vars)$ is a global constraint where *cards* and *values* are vectors with the same index set I and *vars* is a vector of decision variables. The constraint is satisfied if for each $i \in I$ exactly $cards[i]$ variables in *vars* have value $values[i]$. The global constraint `all-different(vars)` forces all variables in *vars* to take different values, and `count`(varArray, val) is a global constraint that counts the number of variables in *varArray* that are equal to *val*.

Decission variables

$o_{i,s} = j$ if team $i \in T$ plays against team $j \in T \setminus \{i\}$ in time slot $s \in S$

$h_{i,s} = 1$ if team $i \in T$ plays at home in slot $s \in S$, 0 if i plays away

$b_{i,s} = 1$ if team $i \in T$ has a break in time slot $s \in S$, 0 otherwise

$\mathbf{d_c}$ = deviation vector of constraint $c \in C_{\text{soft}} \cup C_{\text{hard}}$

$$\textbf{minimize} \quad \sum_{c \in C_{\text{soft}}} \texttt{penalty}_c \, d_c \quad (D1)$$

Round-robin constraints

$$o_{o_{i,s},s} = i \quad \forall i \in T, s \in S \quad (D2)$$

$$(h_{i,s} + h_{j,s} \neq 1) \Rightarrow o_{i,s} \neq j \quad \forall i, j \in T : i < j, s \in S \quad (D3)$$

$$(o_{i,s} = j) \Rightarrow (h_{i,s} + h_{j,s} = 1) \quad \forall i, j \in T : i < j, s \in S \quad (D4)$$

$$\texttt{distribute}([2], [j, \forall j \in T \setminus \{i\}], [o_{i,s}, \forall s \in S]) \quad \forall i \in T \quad (D5)$$

$$\sum_{s \in S} (o_{i,s} = j \wedge h_{i,s} = 1) = 1 \quad \forall i, j \in T : i \neq j \quad (D6)$$

$$\texttt{all-different}(o_{i,s}, s \in \{1, \dots, n-1\}) \quad \forall i \in T \quad (D7)$$

Pertinent constraints

$$o_{i,s} \neq i \quad \forall i \in T, s \in S \quad (D8)$$

$$\texttt{distribute}([n-1, n-1], [0,1], [h_{i,s} \forall s \in S]) \quad \forall i \in T \quad (D9)$$

$$\texttt{distribute}([n/2, n/2], [0,1], [h_{i,s} \forall i \in T]) \quad \forall s \in S \quad (D10)$$

$$\texttt{all-different}(o_{i,s}, i \in T) \quad \forall s \in S \quad (D11)$$

Capacity constraints

$$\texttt{count}([h_{i,s}, \forall s \in \texttt{slots}_c], 1) - d_{c,i} \leq \texttt{max}_c \quad \forall i \in \texttt{teams}_c, \forall c \in \text{CA1} \quad (D12)$$

$$\sum_{j \in \texttt{teams2}_c} \sum_{s \in \texttt{slots}_c} (h_{i,s} = 1 \wedge o_{i,s} = j)) - d_{c,i} \leq \texttt{max}_c \quad \forall i \in \texttt{teams1}_c, \forall c \in \text{CA2} \quad (D13)$$

$$\sum_{j \in \texttt{teams2}_c} \sum_{p=s}^{s+\texttt{intp}_c-1} (h_{i,p} = 1 \wedge o_{i,p} = j) - d_{c,i,s} \leq \texttt{max}_c$$
$$\forall i \in \texttt{teams1}_c, \forall s \in S : s < |S| - \texttt{intp}_c + 1, \forall c \in \text{CA3} \quad (D14)$$

$$\sum_{i \in \texttt{teams1}_c} \sum_{j \in \texttt{teams2}_c} \sum_{s \in \texttt{slots}_c} (h_{i,s} = 1 \wedge o_{i,s} = j) - d_c \leq \texttt{max}_c \quad \forall c \in \text{CA4} \quad (D15)$$

Game constraints

$$\texttt{min}_c - d_c \leq \sum_{(i,j) \in \texttt{meetings}_c} \sum_{s \in \texttt{slots}_c} (o_{i,s} = j \wedge h_{i,s} = 1) \leq \texttt{max}_c + d_c \quad \forall c \in \text{GA1} \quad (D16)$$

Break constraints

$$b_{i,s} = (h_{i,s} = h_{i,s+1}) \quad \forall i \in T, s \in S \setminus |S| \quad (D17)$$

$$\sum_{s \in \texttt{slots}_c} (h_{i,s} = 1 \wedge b_{i,s} = 1) - d_{c,i} \leq \texttt{max}_c \quad \forall i \in \texttt{teams}_c, \forall c \in \text{BR1} \quad (D18)$$

$$\sum_{i \in \texttt{teams}_c} \sum_{s \in \texttt{slots}_c} (h_{i,s} = 1 \wedge b_{i,s} = 1) - d_c \leqslant \texttt{intp}_c \quad \forall c \in \text{BR2} \quad (D19)$$

Fairness and separation constraints

$$\texttt{count}(h_{i,s} \forall s \in \texttt{slots}_c, 1) - \texttt{count}(h_{j,s} \forall s \in \texttt{slots}_c, 1) - d_{c,\{i,j\}} \leqslant \texttt{intp}_c \quad \forall i, j \in \texttt{teams}_c : i \neq j, \forall s \in \texttt{slots}_c, \forall c \in \text{FA2} \quad (D20)$$

$$(o_{i,s_1} = j) \wedge (o_{i,s_2} = j) \Rightarrow d_{c,\{i,j\}} = \texttt{min}_c - (s2 - s1 - 1) \quad \forall i, j \in \texttt{teams}_c : i < j, s_1, s_2 \in S : s_1 < s_2 \leqslant s_1 + \texttt{min}_c, \forall c \in \text{SE1} \quad (D21)$$

$$d_c \geq 0 \quad \forall c \subseteq C_{\text{soft}} \quad (D22)$$

$$d_c = 0 \quad \forall c \subseteq C_{\text{hard}} \quad (D23)$$

We only explain the round-robin and pertinent constraints and assume that all other constraints are self-explanatory. Constraints (D.2) link the opponent variables by using a so-called element constraint which states that team i plays against team j in time slot s if and only if j plays against i on s. We add Constraints (D.3) if we instantiate the $h_{i,s}$ variables first, and add Constraints (D.4) if we instantiate the $o_{i,s}$ variables firsts. Both constraints link the $h_{i,s}$ and $o_{i,s}$ variables. Constraints (D.5) make use of the `distribute` constraint to state that every pair of opponents meets twice, and Constraints (D.6) state that each team meets every other team once at home. In case the tournament needs to be phased, we additionally add Constraints (D.7).

Constraints (D.8) further reduce the domain of the $o_{i,s}$ variables by stating that a team cannot play against itself. Constraints (D.9) and (D.10) respectively state that each team plays exactly $n-1$ games at home, and that exactly half of the teams play home in each time slot. Finally, Constraints (D.11) state that the opponents of each time slot are unique.

References

Abdi, H., & Williams, L. J. (2010). Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics, 2*, 433–459.

Alsmadi, M. K., Jaradat, G. M., Alzaqebah, M., Almarashdeh, I., Alghamdi, F. A., Mustafa, R., … Alqahtani, A. (2022). An enhanced particle swarm optimization for ITC2021 sports timetabling. *Computers, Materials & Continua, 72*, 1995–2014.

Ball, B. C., & Webster, D. B. (1977). Optimal scheduling for even-numbered team athletic conferences. *AIIE Transactions, 9*, 161–169.

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

Berthold, T., Koch, T., & Shinano, Y. (2021). MILP. try. repeat. In P. De Causmaecker, E. Özcan, & G. V. Berghe (Eds.), *Proceedings of the 13th international conference on the practice and theory of automated timetabling - PATAT 2021: Volume II* (pp. 403–411). PATAT.

Bowly, S., Smith-Miles, K., Baatar, D., & Mittelmann, H. (2019). Generation techniques for linear programming instances with controllable properties. *Mathematical Programming Computation, 12*, 389–415.

Briskorn, D., & Drexl, A. (2009). IP models for round robin tournaments. *Computers and Operations Research, 36*, 837–852.

Burke, E. K., Gendreau, M., Hyde, M., Kendall, G., McCollum, B., Ochoa, G., … Petrovic, S. (2011). The cross-domain heuristic search challenge – an international research competition. In C. A. C. Coello (Ed.), *Learning and intelligent optimization* (pp. 631–634). Berlin, Heidelberg: Springer Berlin Heidelberg.

Ceschia, S., Dang, N., De Causmaecker, P., Haspeslagh, S., & Schaerf, A. (2019). The second international nurse rostering competition. *Annals of Operations Research, 274*, 171–186.

Ceschia, S., Di Gaspero, L., & Schaerf, A. (2022). Educational timetabling: Problems, benchmarks, and state-of-the-art results. *European Journal of Operational Research*. https://doi.org/10.1016/j.ejor.2022.07.011.

Cocchi, G., Galligari, A., Nicolino, F. P., Piccialli, V., Schoen, F., & Sciandrone, M. (2018). Scheduling the Italian national volleyball tournament. *Interfaces, 48*, 271–284.

van Doornmalen, J., Hojny, C., Lambers, R., & Spieksma, F. (2021). A hybrid model to find schedules for double round robin tournaments with side constraints. In P. De Causmaecker, E. Özcan, & G. V. Berghe (Eds.), *Proceedings of the 13th international conference on the practice and theory of automated timetabling - PATAT 2021: Volume II* (pp. 412–419). PATAT.

Dimitsas, A., Gogos, C., Valouxis, C., Tzallas, A., & Alefragis, P. (2022). A Pragmatic Approach for Solving the Sports Scheduling Problem. *In Proc. 13th Int. Conf. Pract. Theory Autom. Timetabling, 3*, 195–207 PATAT.

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

Easton, K. (2003). *Using integer programming and constraint programming to solve sports scheduling problems*. Georgia Institute of Technology. USA, Ph.D. thesis

Easton, K., Nemhauser, G., & Trick, M. (2001). The traveling tournament problem description and benchmarks. In T. Walsh (Ed.), *Principles and practice of constraint programming — CP 2001* (pp. 580–584). Berlin, Heidelberg: Springer.

Fonseca, G. G., & Toffolo, T. A. M. (2022). A fix-and-optimize heuristic for the ITC2021 sports timetabling problem. *Journal of Scheduling, 25*, 273–286 Springer.

Forrest, D., & Simmons, R. (2006). New issues in attendance demand: The case of the English football league. *Journal of Sports Economics, 7*, 247–266.

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

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

Haspeslagh, S., De Causmaecker, P., Schaerf, A., & Stølevik, M. (2014). The first international nurse rostering competition 2010. *Annals of Operations Research, 218*, 221–236.

Hausken, M. D., Andersson, H., Fagerholt, K., & Flatberg, T. (2012). Retracted: Scheduling the Norwegian football league. *International Transactions in Operational Research, 20*, 59–77.

Horbach, A., Bartsch, T., & Briskorn, D. (2012). Using a SAT-solver to schedule sports leagues. *Journal of Scheduling, 15*, 117–125.

Hutama, R. R., & Muklason, A. (2021). Pembentukan solusi awal international timetabling competition 2021 [Initial solution formation for International Timetabling Competition 2021]. *Jurnal Teknik Informatika dan Sistem Informasi, 8*, 1939–1944.

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

Kletzander, L., Musliu, N., & Smith-Miles, K. (2021). Instance space analysis for a personnel scheduling problem. *Annals of Mathematics and Artificial Intelligence, 89*, 617–637.

Kyngäs, J., & Nurmi, K. (2009). Scheduling the finnish 1st division ice hockey league. In Proceedings of the twenty-second Florida artificial intelligence research society conference (pp. 195–200). Florida: AAAI Press.

Lamas-Fernandez, C., Martinez-Sykora, A., & Potts, C. N. (2021). Scheduling double round-robin sports tournaments. In P. De Causmaecker, E. Özcan, & G.V. Berghe (Eds.), Proceedings of the 13th international conference on the practice and theory of automated timetabling - PATAT 2021: Volume II (pp. 435–448). PATAT.

Lester, M. M. (2022). Pseudo-Boolean optimisation for RobinX sports timetabling. *Journal of Scheduling, 25*, 287–299 Springer.

Lewis, R., & Thompson, J. (2011). On the application of graph colouring techniques in round-robin sports scheduling. *Computers and Operations Research, 38*, 190–204.

Lopes, L., & Smith-Miles, K. (2013). Generating applicable synthetic instances for branch problems. *Operations Research, 61*, 563–577.

McCollum, B. (2007). A perspective on bridging the gap between theory and practice in university timetabling. In E. K. Burke, & H. Rudová (Eds.), *Practice and theory of automated timetabling VI* (pp. 3–23). Berlin, Heidelberg: Springer Berlin Heidelberg.

McCollum, B., Schaerf, A., Paechter, B., McMullan, P., Lewis, R., Parkes, A. J., … Burke, E. K. (2010). Setting the research agenda in automated timetabling: The second international timetabling competition. *INFORMS Journal on Computing, 22*, 120–130.

Muñoz, M. A., Villanova, L., Baatar, D., & Smith-Miles, K. (2018). Instance spaces for machine learning classification. *Machine Learning, 107*, 109–147.

Müller, T., Rudová, H., & Müllerová, S. (2018). University course timetabling and international timetabling competition 2019. In E. K. Burke, L. D. Gaspero, B. McCollum, N. Musliu, & E. Özcan (Eds.), *Proceedings of the 12th international conference on the practice and theory of automated timetabling*. Vienna: PATAT.

Müller, T., Rudová, H., & Müllerová, S. (2019). International timetabling competition (ITC2019). URL: http://www.itc2019.org/home.

Nemhauser, G. L., & Trick, M. A. (1998). Scheduling a major college basketball conference. *Operations Research, 46*, 1–8.

Nurmi, K., Goossens, D., Bartsch, T., Bonomo, F., Briskorn, D., Duran, G., … Wolf, R. (2010). A framework for scheduling professional sports leagues. In H. Katagir, L. Xu, & A. H. Chan (Eds.), *Ao,s-i.: vol. 5* (p. 1428). Springer.

Paechter, B., Gambardella, L., & Rossi-Doria, O. (2003). International timetabling competition (ITC2002). URL: http://sferics.idsia.ch/Files/ttcomp2002/.

Phillips, A. E., O'Sullivan, M., & Walker, C. (2021). An adaptive large neighbourhood search matheuristic for the ITC2021 sports timetabling competition. In P. De Causmaecker, E. Özcan, & G. V. Berghe (Eds.), *Proceedings of the 13th international conference on the practice and theory of automated timetabling - PATAT 2021: Volume II* (pp. 426–430). PATAT.

Post, G., Di Gaspero, L., Kingston, J., McCollum, B., & Schaerf, A. (2011). International timetabling competition (ITC2011). URL: http://www.utwente.nl/en/eemcs/dmmp/hstt/itc2011/.

Post, G., Di Gaspero, L., Kingston, J. H., McCollum, B., & Schaerf, A. (2016). The third international timetabling competition. *Annals of Operations Research, 239*, 69–75.

Recalde, D., Torres, R., & Vaca, P. (2013). Scheduling the professional Ecuadorian football league by integer programming. *Computers and Operations Research, 40*, 2478–2484.

Régin, J. C. (2001). Minimization of the number of breaks in sports scheduling problems using constraint programming. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 57*, 115–130.

Ribeiro, C. C. (2012). Sports scheduling: Problems and applications. *International Transactions in Operational Research, 19*, 201–226.

Rosati, R. M., Petris, M., Di Gaspero, L., & Schaerf, A. (2022). Multi-neighborhood simulated annealing for the sports timetabling competition ITC2021. *Journal of Scheduling, 25*, 301–319 Springer.

Smith-Miles, K., Baatar, D., Wreford, B., & Lewis, R. (2014). Towards objective measures of algorithm performance across instance space. *Computers and Operations Research, 45*, 12–24.

Smith-Miles, K., & Bowly, S. (2015). Generating new test instances by evolving in instance space. *Computers and Operations Research, 63*, 102–113.

Smith-Miles, K., & Lopes, L. (2012). Measuring instance difficulty for combinatorial optimization problems. *Computers and Operations Research, 39*, 875–889.

Subba, E., & Stordal, O. J. L. (2021). *Scheduling sports tournaments by mixed-integer linear programming and a cluster pattern approach: Computational implementation using data from the International Timetabling Competition 2021*. NHH Norwegian School of Economics. Master's thesis

Sumin, D., & Rodin, I. (2021). MILP based approaches for scheduling double round-robin tournaments. In P. De Causmaecker, E. Özcan, & G. V. Berghe (Eds.), *Proceedings of the 13th international conference on the practice and theory of automated timetabling - PATAT 2021: Volume II* (pp. 420–425). PATAT.

Trick, M. (2005). Formulations and reformulations in integer programming. In R. Barták, & M. Milano (Eds.), *Integration of AI and OR techniques in constraint programming for combinatorial optimization problems* (pp. 366–379). Berlin, Heidelberg: Springer.

Trick, M. A. (2003). Integer and constraint programming approaches for round-robin tournament scheduling. In E. Burke, & P. De Causmaecker (Eds.), *Practice and theory of automated timetabling IV* (pp. 63–77). Berlin, Heidelberg: Springer.

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

Van Bulck, D., & Goossens, D. (2021). Relax-fix-optimize heuristics for time-relaxed sports timetabling. *INFOR: Information Systems and Operational Research, 59*, 623–638.

Van Bulck, D., Goossens, D., Belien, J., & Davari, M. (2021). The fifth international timetabling competition (ITC2021): Sports timetabling. In *Mathsport international 2021* (pp. 117–122). University of Reading.

Van Bulck, D., Goossens, D., Schönberger, J., & Guajardo, M. (2020a). An instance data repository for the round-robin sports timetabling problem. *Management and Labour Studies, 45*, 184–200.

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