An Integrated Constraint Programming Approach to Scheduling Sports Leagues with Divisional and Round-Robin Tournaments

Jeffrey Larson[1,2], Mikael Johansson[1], and Mats Carlsson[3]

[1] Automatic Control Lab, KTH, Osquldas väg 10, SE-100 44 Stockholm, Sweden
{jeffreyl,mikaelj}@kth.se
[2] Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, USA
jmlarson@anl.gov
[3] SICS, P.O. Box 1263, SE-164 29 Kista, Sweden
matsc@sics.se

Abstract. Previous approaches for scheduling a league with round-robin and divisional tournaments involved decomposing the problem into easier subproblems. This approach, used to schedule the top Swedish handball league Elitserien, reduces the problem complexity but can result in suboptimal schedules. This paper presents an integrated constraint programming model that allows to perform the scheduling in a single step. Particular attention is given to identifying implied and symmetry-breaking constraints that reduce the computational complexity significantly. The experimental evaluation of the integrated approach takes considerably less computational effort than the previous approach.

1 Introduction

A double round-robin tournament (DRRT), where every team plays every other team once home and once away, is one of the most common formats for a broad range of sporting events. Football leagues around the world (EPL, Serie A, La Liga, Bundesliga, Eredivisie, Allsvenskan, CONCACAF, and many others) base their season schedules on DRRTs; the format has also been used by Super Rugby, Indian Premier League (cricket), and even the Chess World Championship (where each player played every other player once as black and once as white). As the format is ubiquitous, considerable research into scheduling DRRTs efficiently and fairly has been conducted [20,21]. A comprehensive literature survey and an introduction to sports scheduling terminology can be found in [11].

There are, of course, many reasons why any league may wish to transition away from a DRRT. For example, the top Danish football league contains only 12 teams, and consists of a triple round-robin tournament to offer a sufficient number of games over the season [12]. The Swedish handball league, Elitserien, augments its traditional double round-robin format by dividing teams into two divisions, each of which hold a single round-robin tournament before a league-wide DRRT. Compared to the amount of research that has been devoted to DRRTs, these more "exotic" league formats have

H. Simonis (Ed.): CPAIOR 2014, LNCS 8451, pp. 144–158, 2014.

received limited attention in the literature (see, e.g., [21, Section 5.2] for a notable exception). This paper develops a constraint programming (CP) approach to league scheduling for the combination of divisional and round-robin play employed by Elitserien.

Since DRRT scheduling is already difficult (cf. [2] for NP-completeness results), one might be led to believe that augmentations of DRRT are even harder to address. While it is true that their complexity can make them hard to schedule, they can also allow for extra degrees of freedom or impose new constraints which make it easier to construct a schedule. For example, Elitserien requires that teams meeting three times (those in the same division) must not play at the same venue in consecutive meetings. We denote this requirement the *Alternative Venue Requirement*, or AVR for short. As we will see, this requirement ultimately makes constructing a season schedule easier.

When scheduling sport leagues, it is often common to break the problem into subproblems that can be addressed more easily. For example, the schedule-then-break approach of Trick [20] is widely used in the scheduling of DRRTs. We adopted this approach in an earlier paper [7], where we developed a schedule for Elitserien by first constructing and enumerating a set of home-away patterns sets (HAP sets), not all of which were schedulable with respect to the AVR. A series of increasingly restrictive necessary conditions removed unschedulable HAP sets. Next, for each HAP set, tournament templates (a tournament containing generic numbers and not actual team names) were generated and ranked according to a number of factors, including carry-over effects [18], that are not easy to optimize for directly. The construction of a template was an essential step in scheduling the league, because any change in the schedule required approval from the team owners who are accustomed to working with templates. After a template was agreed upon, we used an integer programming approach to assign actual teams to the numbers in the template in a manner satisfying various constraints (e.g., venue availability or desired derby matches).

Of course, such a decomposed approach (first building a HAP set, then fixing a single template, and finally constructing a schedule) can result in a suboptimal league schedule depending on which template was chosen and the particular venue availabilities and desires that occur in a given year. Even if the league owners agree to adjust the template when the availabilities and desires become known, exploring all possibilities to find the optimal solution would result in a computationally expensive generate-and-test scheme. In contrast, this paper proposes a constraint programming approach for integrated scheduling of Elitserien that completely eliminates the intermediate steps and quickly generates a provably optimal schedule. A particular effort is made to identify and break symmetries in the problem, and substantial speed-ups are obtained over the previous approaches.

For example, for the specific requirements imposed by Elitserien for the 2013-2014 season, the CP approach implemented on a standard desktop was able to find an optimal schedule and prove its optimality in less than half a minute. Although not all leagues will allow for identical symmetry breaking tricks, we believe the approach presented in this paper is general enough to be applicable to many league formats.

Constraint programming has certainly been used for sports scheduling before. Examples of decomposed CP approaches to finding DRRT schedules include [5,19,18].

Hybridized with other methods, CP has been used to minimize travel distance in sports tournaments [1,3,10]. In [16], Régin uses a CP approach to minimizing the number of breaks (consecutive home or away games) in sports schedules. Alternative CP models of sports scheduling were discussed in [8]. However, case studies solved by *integrated* CP approaches are scarce in the literature; perhaps the problems have been assumed to be intractable without decomposition into simpler subproblems.

The outline of the paper is as follows. In Section 2, we state the requirements on the schedule to be computed. In Section 3, we present a global constraint model of the problem, and discuss the additional seasonal constraints and preferences that arise every year. In Section 4, we report the results from our computational experiments. Section 5 concludes the paper.

2 Problem Statement and Basic Tournament Properties

The requirements on Elitserien's schedule can be broadly classified into two categories: the first category (which we call *structural constraints*) addresses the schedule format and fairness in terms of breaks (consecutive home or away games), periods without games (called byes), the alternating venue requirement, and the sequence of home and away games; the second category (which we refer to as *seasonal constraints*) concerns stadium and referee availabilities, the desire to support various match-ups (such as rivalries), wishes from the media, etc. Historically, Elitserien has determined their schedule by first proposing a tournament template which addresses the structural constraints. This tournament template has numbers in place of actual teams in the schedule. Every year, the league collects information about unavailabilities and particular wishes from the clubs and assigns teams to numbers in the tournament template to form the season schedule.

Although we will develop an integrated scheduling approach which accounts for all the above concerns, it is still useful to keep the distinction between the structural and seasonal constraints. In this way, we can examine the combinatorial properties and symmetries of the tournament template generation process itself and make a direct computational comparison with the approach used in [7]. We can also clearly expose the different symmetry-breaking techniques that apply to the tournament template design and the integrated scheduling respectively.

Elitserien poses the following structural constraints on its tournament template:

C1. Both 7-team divisions must hold an SRRT to start the season.
C2. This must be followed by a DRRT between the entire league. The DRRT is organized into two SRRTs, where the second SRRT is the mirrored complement of the first: the order is reversed, home games become away games and vice versa.
C3. There must be a minimum number of breaks in the schedule.
C4. Each team has one bye during the season to occur during the divisional RRT.
C5. At no point during the season can the number of home and away games played by any team differ by more than 1.
C6. Any pair of teams must have consecutive meetings occur at different venues (AVR).
C7. Each division must have 3 pairs of complementary schedules.

Elitserien considers any consecutive home or away matches to be breaks. Therefore, a team playing away-bye-away or home-bye-home constitutes a break, as does a team ending the SRRT with the same type of game (home or away) as they start the DRRT with.

The structural constraints C1-C7 are hard constraints that do not change from year to year. However, the teams themselves might change, and every year sees different constraints on venue availabilities, match-ups, and derbies. Thus, in addition to the structural constraints, the integrated scheduling approach also accounts for the following seasonal constraints.

C8. Each division must contain a prescribed set of teams.
C9. Specific pairs of teams in each division have to be assigned complementary schedules (typically teams that come from the same city, or even share the same arena).
C10. To increase the visibility of handball, the league arranges derbies in specific periods. Elitserien derby constraints consist of a single period and a set of teams, out of which as many matches as possible should be formed. Alternatively, a single team, a single period, and a set of possible opponents are given.
C11. Venue unavailabilities have to be respected to the highest extent possible.

The league considers constraints C8-C10 hard, but allow some flexibility with constraint C11. This flexibility arises because, although a venue might be available for the target dates of a specific game round, the league allows the teams flexibility to move a game date a few days forward or backward in time. For example, a game scheduled for Saturday can be played on Friday or Sunday, depending on venue, referee, and team availabilities.

The problem that we address in this paper is how to generate a schedule that satisfies requirements C1-C10 and violates a minimum number of venue unavailabilities (requirement C11).

2.1 Basic Tournament Properties

To motivate our approach to tournament template and schedule design, we recall a basic property of Elitserien's tournament format established in [7]. The first proposition [7, Proposition 2.1] gives a lower bound on the minimum number of breaks required to schedule a league combining divisional and round-robin play.

Proposition 1. *In an n-team league (n even) with a schedule consisting of two concurrent divisional RRTs followed by two consecutive full-league RRTs, if only one bye is allowed and it must occur during the divisional RRT, any schedule must have at least $2n - 4$ breaks.*

In [7], it was also shown that it is possible to construct schedules that achieve this lower bound. This is accomplished by combining the unique divisional RRT from [4] with the fact that HAP sets must break in odd periods if they ensure that C5 is satisfied: the cumulative number of home and away games played never differs by more than 1 at any point in the season [17]. The home-away pattern for a tournament combining divisional and round-robin play can thus be constructed by combining the divisional

RRT home-away patterns in Fig. 1(a) with two copies of a full-season RRT home-away pattern in Fig. 1(b) without introducing additional breaks. To construct a given team HAP, Part I consists of one pattern from Fig. 1(a). This is followed by Part II, which consists of one pattern from Fig. 1(b). Part III is the reflected complement of the Part II pattern; i.e., if Part II ends `AHH` then Part III starts `AAH`. The schedule will also mirror this pattern. If Part II of the schedule ends with team 1 playing at team 2, Part III will start with team 2 hosting team 1.

BAHAHAH	AHAHAHAHAHAHA
HBAHAHA	AHAHAHAHAHAHH
AHBAHAH	AHAHAHAHAHHAH
HAHBAHA	AHAHAHAHHAHAH
AHAHBAH	AHAHAHHAHAHAH
HAHAHBA	AHAHHAHAHAHAH
AHAHAHB	AHHAHAHAHAHAH
or	HAAHAHAHAHAHA
BHAHAHA	HAHAAHAHAHAHA
ABHAHAH	HAHAHAAHAHAHA
HABHAHA	HAHAHAHAAHAHA
AHABHAH	HAHAHAHAHAAHA
HAHABHA	HAHAHAHAHAHAA
AHAHABH	HAHAHAHAHAHAH
HAHAHAB	
(a)	(b)

Fig. 1. Left: Two HAP sets for a 7-team, no-break RRT. Right: HAP set satisfying Elitserien's requirements for a 14-team, 12-break RRT. These HAP sets are unique up to permutation of the rows.

HAP sets created as the patterns in Fig. 1 have many attractive properties. Taking the unique (up to permutation of the rows) no-break, 7-team tournament HAP set and its complement ensures that 7 teams play at home and 7 teams play away in period 8 without introducing a break. If we did not take the complement, we would have 8 teams needing to play at home without introducing a break in period 8, an impossibility. Since we are reflecting and taking the complement of Part II to schedule Part III, and breaks only occur during odd periods (to ensure the number of home and away games never differ by more than 1 at any point in the season), there are no breaks in period 9. This implies no team has a break to end the season; in other words, every team plays at home one of the last two periods of the season.

At first glance, the reflecting and taking the complement of Part II to form Part III forces teams to play the same team in periods 20 as they do in period 21 (at the opposite venue). This could be undesirable, depending on the league, but it is a non-issue for Elitserien. Period II ends before Christmas, allowing for a month-long break for Champions League competitions before Period III starts at the beginning of February.

2.2 Tournament Specific Properties

We now recall a number of properties of the unique HAP set satisfying Elitserien's requirements shown in Fig. 1. These properties will be instrumental for the development of efficient implied and symmetry-breaking constraints:

P1. In each division, one bye occurs in each period.
P2. It follows from C5 that breaks can only occur on odd periods.
P3. The three pairs of complementary schedules per division required by C7 must have breaks that are pairwise aligned, e.g.:

```
HBAHAHA | HAHAHAHAHAAHA
AHBAHAH | AHAHAHAHAHHAH

HAHBAHA | HAHAAHAHAHAHA
AHAHBAH | AHAHHAHAHAHAH

HAHAHBA | HAAHAHAHAHAHA
AHAHAHB | AHHAHAHAHAHAH
```

P4. Two HAPs can only be complementary if the byes occur in adjacent periods [7, Proposition 3.3]. It is apparent by visual inspection of Fig. 1 that two non-adjacent sequences are non-complementary in at least one of the periods 1 through 8.
P5. If the byes are placed as in Fig. 1, the required three pairs of complementary schedules must include teams 2, 4 and 6 of the given division.
P6. If the byes are placed as in Fig. 1, the first row of Part I is complementary to the first column for each division.

3 Constraint Model

We now describe the integrated constraint programming model in detail. We first define the variables to be used in the CP, and then the essential constraints to ensure the resultant schedule will satisfy Elitserien's structural requirements C1-C7. We next highlight some implied constraints and symmetry breaking properties that we find greatly reduce the search effort. Lastly, we model the league's seasonal constraints so we can construct the entire schedule in an integrated approach. The constraint model was encoded in MiniZinc 1.6 and executed with Gecode 3.7.0 as back-end. Full details of our experiments are given in Section 4.

3.1 Problem Variables

We first note that it is sufficient to consider only the first part of the DRRT because its second half is the mirrored complement of the first half. According to C1-C2, let $t \in 1..14$ denote a team and $p \in 1..20$ denote a period. The tournament template corresponds to the array of variables $T[t,p] \in -14..14$, where $T[t,p] < 0$ stands for team t playing away in period p, $T[t,p] > 0$ if it plays at home, and $T[t,p] = 0$ if it

has a bye. The HAP set corresponds to the array of variables $H[t,p] \in \{\texttt{A}, \texttt{B}, \texttt{H}\}$. We also need an array $O[t,p] \in 1..14$ where $O[t,p]$ stands for the opponent of team t in period p. $O[t,p] = t$ if and only if it has a bye in that period. Finally, we need an array $B[t] \in 0..20$ which stands for the period in which the break for team t occurs, or 0 if team t has no break in its schedule.

In [6], the authors show that, if the constraint model uses opponent variables, like ours does, then an SRRT can be codified by two types of constraints, the filtering algorithms of which are crucial to performance. First, every period consists of a matching (or one-factor) of the teams. This is captured by the *symmetric alldifferent* [15] (a.k.a. *one-factor*) constraint; see (5) below. Second, the complete set of opponents for a given team i is the entire set of teams without team i. This is captured by the *alldifferent* constraint [13]; see (6) below. Alternatives to opponent variables are discussed in [8].

3.2 Structural Constraints

The T array is channeled to the O and H arrays straightforwardly:

$$T[t,p] = \begin{cases} -O[t,p], & \text{if } H[t,p] = \texttt{A} \\ O[t,p], & \text{if } H[t,p] = \texttt{H} \\ 0, & \text{if } H[t,p] = \texttt{B} \end{cases}, \forall t, \forall p. \tag{1}$$

The definition of a break and the channeling between the B and H arrays are captured by the following constraint:

$$B[t] = \sum_{p \in 8..19} p \times (H[t,p] = H[t,p+1]), \forall t. \tag{2}$$

As mentioned above, the channeling between the O and H arrays is captured by the following constraint:

$$O[t,p] = t \Leftrightarrow H[t,p] = \texttt{B}, \forall t, \forall p. \tag{3}$$

The possible HAP for any team is constrained by C3-C5 and by the fact that we know the set of sequences that must make up a HAP set satisfying Elitserien's requirements; see Fig. 1. This is easily captured by a regular expression e. The corresponding finite automaton is shown in Fig. 2, and the corresponding *regular* constraint [9] is posted on every row of H:

$$\mathit{regular}([H[t,p] \mid p \in 1..20], e), \forall t. \tag{4}$$

As mentioned above, it is a fundamental RRT constraint, usually encoded with *symmetric alldifferent*, that every period consist of a matching of the teams. Unfortunately, *symmetric alldifferent* is not among the standard MiniZinc global constraints, and has no native support in Gecode. Fortunately, is is easily emulated by MiniZinc's *inverse* global constraint, which is supported by Gecode:

$$O[O[t,p],p] = t, \forall t, \forall p, \tag{5}$$

i.e.,

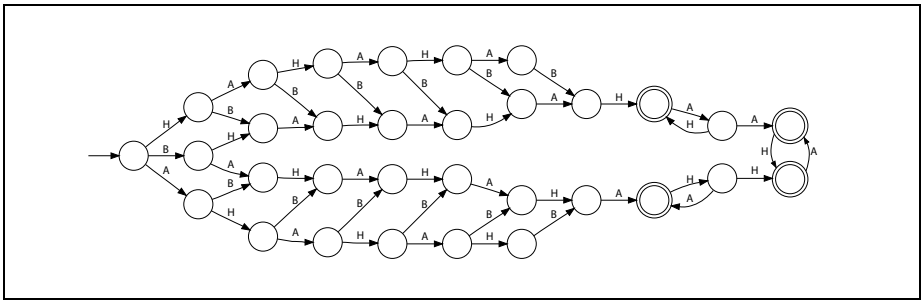

Fig. 2. The finite automaton for valid HAP. As usual, accepting states are indicated by double circles. When restricted to sequences of length 20, it accepts the combinations of rows in Fig. 1.

$$inverse([O[t,p] \mid t \in 1..14], [O[t,p] \mid t \in 1..14]), \forall p.$$

Each team must meet every other team during Part I and Part II. As mentioned above, this is another fundamental RRT constraint, easily expressed with *alldifferent*:

$$alldifferent([O[t,p] \mid p \in 1..7]) \wedge alldifferent([O[t,p] \mid p \in 8..20]), \forall t. \tag{6}$$

Home and away must match for every team and its opponent, everywhere. This is yet another fundamental RRT constraint:

$$\begin{pmatrix} (H[t,p] = \texttt{A} \wedge H[O[t,p],p] = \texttt{H}) \vee \\ (H[t,p] = \texttt{B} \wedge H[O[t,p],p] = \texttt{B}) \vee \\ (H[t,p] = \texttt{H} \wedge H[O[t,p],p] = \texttt{A}) \end{pmatrix}, \forall t, \forall p. \tag{7}$$

To encode C6, we note that it is satisfied if and only if every row of the tournament template contains distinct nonzero values. And since every team has exactly one bye, we can use *alldifferent*:

$$alldifferent([T[t,p] \mid p \in 1..20]), \forall t \tag{8}$$

Finally, to model C7, let $i \oplus j$ denote the fact that teams i and j have complementary schedules:

$$i \oplus j \Leftrightarrow B[i] = B[j] \wedge (H[i,p] \neq H[j,p], \forall p \in 1..20).$$

Then we have:

$$\begin{array}{l} \exists \{i,j,k,l,m,n\} \subset 1..7 \text{ such that } (i \oplus j) \wedge (k \oplus l) \wedge (m \oplus n) \text{ and} \\ \exists \{i,j,k,l,m,n\} \subset 8..14 \text{ such that } (i \oplus j) \wedge (k \oplus l) \wedge (m \oplus n). \end{array} \tag{9}$$

3.3 Implied Constraints

Property P3 implies that each division must have six breaks. It was determined experimentally that posting this implied constraint improves propagation (see Section 4).

$$\sum_{t \in 1..7} (B[t] > 0) = 6 \text{ and } \sum_{t \in 8..14} (B[t] > 0) = 6. \tag{10}$$

The fact that the breaks must be pairwise aligned could also be posted as a constraint. This implied constraint, however, was experimentally determined to be useless:

$$\sum_{t \in 1..7} (B[t] = p) \text{ is even and } \sum_{t \in 8..14} (B[t] = p) \text{ is even}, \forall p.$$

It's a structural property that the numbers of home and away matches must always match. Contrary to Trick's observation [21, Section 6], this implied constraint was also experimentally found to be useless:

$$\sum_{t \in 1..14} (H[t,p] = \texttt{A}) = \sum_{t \in 1..14} (H[t,p] = \texttt{H}), \forall p.$$

We have Property P1, which is useful in itself, but which is subsumed by (15), as we shall see later:

$$\begin{array}{l} \textit{alldifferent}([p \mid H[t,p] = \texttt{B}, t \in 1..7, p \in 1..7]) \wedge \\ \textit{alldifferent}([p \mid H[t,p] = \texttt{B}, t \in 8..14, p \in 1..7]). \end{array} \tag{11}$$

Finally, we know from P2-P3 that out of the 14 teams, two teams must have a break each in periods 9, 11, 13, 15, 17 and 19, and two teams must have no break. This is efficiently encoded by an *global cardinality* constraint [14], and was also experimentally found to be useful:

$$\sum_{t \in 1..14} (B[t] = i) = 2, \forall i \in \{0, 9, 11, 13, 15, 17, 19\}. \tag{12}$$

3.4 Breaking Symmetries

The following constraints help remove many of the symmetries in the scheduling of Elitserien. There is an obvious home/away symmetry: given a solution, we can construct another solution by swapping home and away everywhere. This symmetry is easily broken:

$$H[1,2] = \texttt{H} \wedge H[2,1] = \texttt{A}. \tag{13}$$

A second source of symmetry is the fact that the two divisions can be swapped in the template. This symmetry can be broken by lexicographically ordering the break sequences:

$$[B[t] \mid t \in 1..7] \leq_{\text{lex}} [B[t] \mid t \in 8..14]. \tag{14}$$

There is third source of symmetry in the model: given a solution, we can construct another solution by swapping rows (teams) i and j of the same division in the arrays as well as values i and j (positive or negative) in O and T. To break this symmetry, we can fix the bye period for all teams like in Fig. 1, subsuming (11):

$$\begin{pmatrix} H[t,t] = \mathtt{B} & \wedge \\ O[t,t] = t & \wedge \\ O[t,p] \in (1..7) \setminus \{t\} \quad \forall p \neq t & \wedge \\ H[t+7,t] = \mathtt{B} & \wedge \\ O[t+7,t] = t+7 & \wedge \\ O[t+7,p] \in (8..14) \setminus \{t+7\} \quad \forall p \neq t & \end{pmatrix}, \forall t \in 1..7. \tag{15}$$

Having fixed the bye periods in such a manner, we can use Properties P4-P5 to construct a slightly stronger version of (9) that restricts the possible pairing of complementary schedules:

$$\begin{array}{l} (1 \oplus 2 \vee 2 \oplus 3) \wedge \\ (3 \oplus 4 \vee 4 \oplus 5) \wedge \\ (5 \oplus 6 \vee 6 \oplus 7) \wedge \\ (8 \oplus 9 \vee 9 \oplus 10) \wedge \\ (10 \oplus 11 \vee 11 \oplus 12) \wedge \\ (12 \oplus 13 \vee 13 \oplus 14) \end{array} \tag{16}$$

$$B[t] > 0, \ \forall t \in \{2,4,6,9,11,13\}. \tag{17}$$

Property P6 can be posted as an implied constraint, which was determined experimentally to improve propagation, but only marginally:

$$H[t,1] \neq H[1,t] \wedge H[t+7,1] \neq H[8,t], \forall t \in 2..7. \tag{18}$$

Finally, alternative symmetry breaking constraints are discussed in [20].

3.5 Seasonal Constraints

The constraints described in the previous section capture the generic Elitserien structural constraints i.e., a 14-team league with a schedule consisting of (Part I) two concurrent divisional SRRTs, followed by (Part II) an RRT between all teams, and (Part III) the reverse complement of Part II. Also, for every season there are a number of extra requirements and preferences:

Team Mapping (C8). Team *names* must be substituted for team *numbers*. This requires a level of indirection in the form of another array:

$$M[t] \in \begin{cases} 1..7 \ , & \text{for teams constituting Div 1} \\ 8..14 \ , & \text{for teams constituting Div 2} \end{cases}$$

where $M[t]$ is the team number (template row) assigned to team t. The M array can be treated in two different ways, both of which are evaluated in Section 4:

1. One can let M be an array of problem variables, which allows us to keep the symmetry breaking constraint (15, 16, 17, 18).
2. One can fix M before search, but replace the same symmetry breaking constraints by (9) and (11), which make no assumptions on the placement of byes.

Complementary Schedules (C9). For reasons of e.g., venue availability, certain pairs (i, j) of teams are required to have complementary schedules:

$$M[i] \oplus M[j]$$

Derbies (C10). A derby constraint is given as a period p and a set $\mathcal{Q}$ of four teams, out of which two pairs of playing teams must be formed. Alternatively, a set $\mathcal{T}$ of three teams is given, two of which must play each other:

$$\begin{pmatrix} (O[M[i],p] = j \wedge O[M[j],p] = i) & \vee \\ (O[M[i],p] = k \wedge O[M[k],p] = i) & \vee \\ (O[M[j],p] = k \wedge O[M[k],p] = j) & \end{pmatrix}, \text{ where } \mathcal{T} = \{i, j, k\}.$$

$$O[M[i], p] \in \{M[j] \mid j \in \mathcal{Q}\}, \forall i \in \mathcal{Q}.$$

Venue Unavailabilities (C11). These are soft constraints, turning the scheduling problem into an optimization problem. If the preferences are stated as an array:

$$N[t,p] = \begin{cases} 1\ , & \text{if team } t \text{ prefers not to play at home during period } p \\ 0\ , & \text{otherwise,} \end{cases}$$

then the cost function for all three parts of the schedule is:

$$\sum_{\substack{t \in 1..14, \\ p \in 1..20}} N[t,p] \times (H[M[t],p] = \texttt{H}) + \sum_{\substack{t \in 1..14, \\ p \in 21..33}} N[t,p] \times (H[M[t], 41-p] = \texttt{A})$$

4 Experiments

The constraint model was encoded in MiniZinc 1.6 and executed with Gecode 3.7.0 as back-end on a quad core 2.8 GHz Intel Core i7-860 machine with 8MB cache per core, running Ubuntu Linux. In the parallel runs, Gecode was given 8 threads in the usual way with the command line option `-p 8`:

```
mzn-gecode -p 8 ...
```

The MiniZinc models and instances can be found here:

```
http://www.sics.se/~matsc/Elitserien
```

In a first experiment, we solved the generic scheduling problem, involving the structural constraints encoded by constraints (1–18) except (9) and (11), which are implied by constraints (16) and (15), respectively, but no seasonal constraints. Searching on the H variables in row-major order, followed by the O variables in row-major order, the first solution was found using one thread in 70ms and 81 failures. With the symmetry

Table 1. HAP sets distribution according to how many templates satisfying C1-C7 they admit

Range	No. HAP sets
$[10^1, 10^2)$	3
$[10^2, 10^3)$	13
$[10^3, 10^4)$	39
$[10^4, 10^5)$	41
$[10^5, 10^6)$	7
$[10^6, 10^7)$	1
Total	104

breaking described in Section 3.4, it was already known that there exist 104 distinct, feasible HAP sets. We enumerated these sets, and counted the number of solutions per set. As shown in Table 1, the solution counts vary a lot, from 34 to 2249812. This would suggest that some HAP sets can accommodate seasonal constraints much better than others.

In a second experiment, we considered the optimization problem subject to structural as well as seasonal constraints, treating the mapping M as an array of problem variables. This allowed us to keep the symmetry breaking constraints (15–18), which are very effective. Constraints (13) and (14) are however not valid in this context and were disabled.

In this experiment, we generated 20 random instances because (a) our model has only been used for one year and we wished to verify that this instance was not an especially easy case to solve, and (b) the league requested that we not divulge the true team desires for privacy reasons. The structure of the random seasonal constraints were very similar to the real ones, though:

- The partitioning of teams into divisions (C8).
- One specific pair of teams in each division to be assigned complementary schedules (C9).
- One 3-team intra-division derby set, one 3-team inter-division derby set, and one 4-team inter-division derby set (C10).
- For each team t and period p, t prefers to not play at home during period p with probability 0.054, yielding on average 25 unavailabilities (C11), which is the number of unavailabilities requested by Elitserien teams for the season that we scheduled.

We searched on the H variables in column-major order, followed by the M variables, followed by the O variables in row-major order. Fig. 3 shows the results in terms of number of instances solved to optimality as a function of elapsed time, with one curve for 1 thread and one curve for 8 threads. For 8 threads, the median solve time to optimality was 15s, the average 42s, and the standard deviation 54s. These numbers are reasonable, and show that there are no extreme outliers among our random instances. We also observe a rather uniform speed-up of 3x-4x from parallelism.

This is an improvement over the approach in [7], which first generated 80640 HAP sets satisfying C3-C5 but not necessarily schedulable, then applied necessary conditions

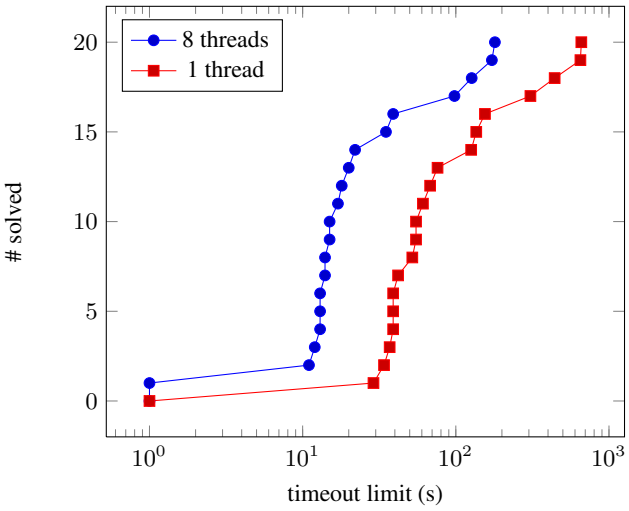

Fig. 3. Number of instances solved to optimality as function of timeout limit in seconds

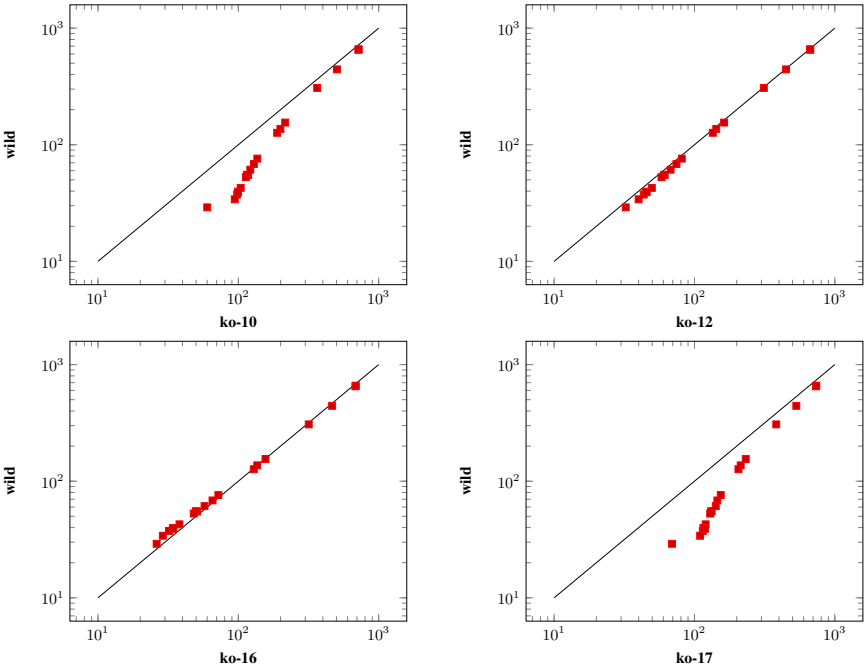

Fig. 4. Effect of knocking out a single implied constraint. Solve time to optimality in seconds, running on a single thread. The model variants are: **wild** – all constraints; **ko-10** – (10) knocked out; **ko-12** – (12) knocked out; **ko-16** – (16) knocked out and replaced by (9); **ko-17** – (17) knocked out.

for schedulability to rule out some 87% of the unschedulable HAP sets. An attempt was then made to convert the remaining HAP sets to templates by solving an integer program. The resultant templates were ranked in their carry-over effect to produce a template for the league. This template was then assigned teams with respect to the seasonal constraints C8-C11. Testing all HAP sets against the necessary conditions took nearly a day. Since the template was fixed before the seasonal constraints were available, it is likely that a suboptimal schedule was produced. Furthermore, a straightforward application of the approach in [7] to scheduling where a template does not need to be fixed a priori would clearly be inefficient: the 104 schedulable HAP sets admit 5,961,704 templates if constraints (13) and (14) are used, or 23,846,816 templates if they are not. Assuming that it takes 0.1 seconds per template to assign teams to numbers optimally, an optimistic estimate, finding the best schedule would take almost one month.

In order to gauge the effectiveness of implied and symmetry-breaking constraints (10, 12, 16, 17), we ran the same instances with the same model, except in each run, a given constraint was disabled. The results are shown as scatter plots in Fig. 4, running on a single thread for maximal runtime stability. We find that only constraints (10) and (17) had any significant impact, shortening the solve time by up to three times. An unexpected observation is that the speed-up gained from these extra constraints seems to decrease as the difficulty of the instance increases. Constraint (12) gave a marginal speedup, whereas replacing constraints (9) by (16) had a slightly detrimental effect.

In a third experiment, we considered the same optimization problem, but fixed the mapping M before search. So we had to disable constraints (15–18) and instead enable constraints (9) and (11), which make no assumptions on the placement of byes. This gave much worse results than the second experiment: on an instance that was solved with proof of optimality in 20 seconds on 8 threads in the second experiment, an optimal solution was found in about 3.5 minutes in the third experiment, with a proof of optimality obtained after 10 hours.

5 Conclusion

In this paper, we analyzed the situation where a league augments a traditional DRRT schedule by forming two divisions of teams, each of which hold an SRRT to start the season. This asymmetry (pairs of teams play three times if they are in the same division, twice otherwise) makes constructing feasible schedules an interesting problem. We highlighted the concerns of Elitserien, which we consider to be general enough to apply to many other leagues. We presented a constraint programming model for the problem, integrating the different phases that sports scheduling traditionally decomposes to, and showed a dramatic improvement over a previous approach using decomposition and integer programming. Non-decomposition approaches are rare in the sports scheduling literature. In our work, we made heavy use of implied and symmetry-breaking constraints, some of which turned out to be crucial to performance.

Acknowledgments. Jeffrey Larson and Mikael Johansson were funded in part by the Swedish Foundation for Strategic Research and the Swedish Science Council.

References

1. Benoist, T., Laburthe, F., Rottembourg, B.: Lagrange relaxation and constraint programming collaborative schemes for travelling tournament problems. In: Proceedings CPAIOR, vol. 1, pp. 15–26 (2001)
2. Briskorn, D.: Sports Leagues Scheduling. Lecture Notes in Economics and Mathematical Systems, vol. 603. Springer (2008)
3. Easton, K., Nemhauser, G.L., Trick, M.A.: The traveling tournament problem description and benchmarks. In: Walsh, T. (ed.) CP 2001. LNCS, vol. 2239, pp. 580–584. Springer, Heidelberg (2001)
4. Fronček, D., Meszka, A.: Round robin tournaments with one bye and no breaks in home-away patterns are unique. In: Multidisciplary Scheduling: Theory and Applications, pp. 331–340. MISTA, New York (July 2005) ISSN 2305-249X
5. Henz, M.: Scheduling a major college basketball conference—revisited. Operations Research 49(1), 163–168 (2001)
6. Henz, M., Müller, T., Thiel, S.: Global constraints for round robin tournament scheduling. European Journal of Operational Research 153(1), 92–101 (2004)
7. Larson, J., Johansson, M.: Constructing schedules for sports leagues with divisional and round-robin tournaments. Journal of Quantitative Analysis in Sports (to appear, 2014) doi:10.1515/jqas-2013-0090
8. Perron, L.: Alternate modeling in sport scheduling. In: van Beek, P. (ed.) CP 2005. LNCS, vol. 3709, pp. 797–801. Springer, Heidelberg (2005)
9. Pesant, G.: A regular language membership constraint for finite sequences of variables. In: Wallace, M. (ed.) CP 2004. LNCS, vol. 3258, pp. 482–495. Springer, Heidelberg (2004)
10. Rasmussen, R.V., Trick, M.A.: The timetable constrained distance minimization problem. In: Beck, J.C., Smith, B.M. (eds.) CPAIOR 2006. LNCS, vol. 3990, pp. 167–181. Springer, Heidelberg (2006)
11. Rasmussen, R.V., Trick, M.A.: Round robin scheduling - a survey. European Journal of Operational Research 188(3), 617–636 (2008)
12. Rasmussen, R.: Scheduling a triple round robin tournament for the best Danish soccer league. European Journal of Operational Research 185, 795–810 (2008)
13. Régin, J.C.: A filtering algorithm for constraints of difference in CSPs. In: 12th National Conference on Artificial Intelligence (AAAI-1994), pp. 362–367 (1994)
14. Régin, J.-C.: Generalized arc consistency for global cardinality constraint. In: Clancey, W.J., Weld, D.S. (eds.) AAAI/IAAI, vol. 1, pp. 209–215. AAAI Press / The MIT Press (1996)
15. Régin, J.-C.: The symmetric alldiff constraint. In: Dean, T. (ed.) IJCAI, pp. 420–425. Morgan Kaufmann (1999)
16. Régin, J.-C.: 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 (2001)
17. Ribeiro, C., Urrutia, S.: Scheduling the Brazilian soccer tournament with fairness and broadcast objectives. In: Burke, E.K., Rudová, H. (eds.) PATAT 2007. LNCS, vol. 3867, pp. 147–157. Springer, Heidelberg (2007)
18. Russell, R.A., Urban, T.L.: A constraint programming approach to the multiple-venue, sport-scheduling problem. Computers & Operations Research 33(7), 1895–1906 (2006)
19. Schaerf, A.: Scheduling sport tournaments using constraint logic programming. Constraints 4(1), 43–65 (1999)
20. Trick, M.A.: A schedule-then-break approach to sports timetabling. In: Burke, E., Erben, W. (eds.) PATAT 2000. LNCS, vol. 2079, pp. 242–253. Springer, Heidelberg (2001)
21. Trick, M.A.: Integer and constraint programming approaches for round-robin tournament scheduling. In: Burke, E.K., De Causmaecker, P. (eds.) PATAT 2002. LNCS, vol. 2740, pp. 63–77. Springer, Heidelberg (2003)