Sports League Scheduling: Enumerative Search for Prob026 from CSPLib

Jean-Philippe Hamiez and Jin-Kao Hao

LERIA, Université d'Angers, UFR Sciences
2, boulevard Lavoisier, 49045 Angers CEDEX 01, France
{Jean-Philippe.Hamiez, Jin-Kao.Hao}@univ-angers.fr

Abstract. This paper presents an enumerative approach for a sports league scheduling problem. This simple method can solve some instances involving a number T of teams up to 70 while the best known constraint programing algorithm is limited to $T \leq 40$. The proposed approach relies on interesting properties which are used to constraint the search process.

1 Introduction

This paper deals with "Prob026" from CSPLib [1], also known as the "balanced tournament design" problem in combinatorial design theory [2, pages 238–241]. It seems to be first introduced in [3].

- There are $T = 2n$ teams (i.e. T even). The season lasts $W = T - 1$ weeks. Weeks are partitioned into $P = T/2$ slots (periods);
- $c_{\mathcal{H}}$ constraint: All teams play each other exactly once ($\mathcal{H}$alf competition);
- $c_{\mathcal{W}}$ constraint: All teams play in each $\mathcal{W}$eek;
- $c_{\mathcal{P}}$ constraint: No team plays more than twice in the same $\mathcal{P}$eriod.

Various techniques were used to tackle Prob026: Integer programming [4,5] ($T \leq 12$), basic local search [4] ($T \leq 14$), local search with a many-valued propositional logic encoding [6] ($T \leq 16$), randomized deterministic complete search [7] ($T \leq 18$), local search with classical propositional logic encoding [8] ($T \leq 20$), constraint programming with powerful filtering algorithm [9] ($T \leq 24$), multiple threads [10] ($T \leq 28$), constraint programming [11] ($T \leq 30$), constraint programming with problem transformation [12] and tabu search [13] ($T \leq 40$).

Note that solutions exist for all $T \neq 4$ [14]. Furthermore, direct constructions have already been proposed when $(T - 1) \bmod 3 \neq 0$ or $T/2$ is odd [14,15,16]. This leaves open the cases where $T \bmod 12 = 4$.

In this paper, we present `EnASS`, an `En`umerative `A`lgorithm for `S`ports `S`cheduling for Prob026. Given T, `EnASS` starts building a particular conflicting schedule (called $\overline{s}$) verifying a set $\mathcal{R}$ of properties (or $\mathcal{R}$equirements). The set S of solutions is generated using $\overline{s}$ in a simple exhaustive way with backtracks and observed to identify new properties. $\mathcal{R}$ is then updated to solve Prob026 for larger T or to accelerate the resolution. Despite the exponential-time complexity of `EnASS`, we manage to build particular $\mathcal{R}$ sets that enable `EnASS` to find solutions to Prob026 for most T up to 70 in a reasonable amount of time. Note that similar ideas have been recently used for constraint reasoning [17].

F. Benhamou (Ed.): CP 2006, LNCS 4204, pp. 716–720, 2006.

2 Reducing the Complexity

Since any valid schedule can be thought of as a particular permutation of the $T(T-1)/2$ matches, the search space size is $[T(T-1)/2]!$. In other words, the search space size grows as the factorial of the square of $T/2$.

Patterned one-factorization [2, page 662, example 4.33] can be used to verify $c_{\mathcal{H}}$ and $c_{\mathcal{W}}$, the goal of `EnASS` being then to satisfy the last constraint $c_{\mathcal{P}}$. Form a regular polygon with the first $T-1$ teams. Draw W sets of $P-1$ parallels connecting vertices in pairs starting with each w side. Each set, augmented with the pair of missing teams, corresponds to the matches to place in week w [18]. Let $\overline{s}$ be the tournament obtained (in linear-time complexity) with this technique, where $\overline{s}\langle p, w\rangle$ is the match scheduled in period p and week w in $\overline{s}$. See [15] for a full detailed description and the formal model used to build $\overline{s}$.

Prob026 has symmetries that can be combined [19]: renumbering of the teams, permutation of weeks or / and periods. They can be avoided using patterned one-factorization and fixing the first week.

Prob026 solutions verify this property: In each p period, two different "$\mathcal{D}$eficient" [14] teams (a 2-set $\mathcal{D}_p$) appear exactly once. Furthermore, if one considers any $p' \neq p$ period, then $\forall t \in \mathcal{D}_p$, t appears twice in period p'. More formally, if $c_{\mathcal{D}}$ refers to this implicit constraint, then: $c_{\mathcal{D}}(p) \Leftrightarrow \forall p' \neq p, \mathcal{D}_{p'} \cap \mathcal{D}_p = \emptyset$.

3 Prob026: A Constraint Satisfaction Problem

Let $x = \langle p, w\rangle$ be any assignment of a match in period p and week w. Values of this variable type are of (t, t') pattern, meaning that team t meets team t' in period p and week w, noted $x \mapsto (t, t')$. So, the set X of variables is $X = \{x = \langle p, w\rangle, 1 \leq p \leq P, 1 \leq w \leq W\}$. Domains are defined according to the comments from the previous section: $\forall x = \langle p, 1\rangle \in X, d_x = \{\overline{s}\langle p, 1\rangle\}$ and $\forall x = \langle p, w\rangle \in X(w > 1), d_x = \{\overline{s}\langle \overline{p}, w\rangle, 1 \leq \overline{p} \leq P\}$. Since $\overline{s}$ verifies already $c_{\mathcal{W}}$ and $c_{\mathcal{H}}$, the set of constraints is only composed of the implicit $c_{\mathcal{D}}$ constraint (see Sect. 2) and $c_{\mathcal{P}}$: For each team t and each period p, we impose the constraint $c_{\mathcal{P}}(t, p) \Leftrightarrow |\{x = \langle p, w\rangle \mapsto (t, t'), 1 \leq w \leq W, t' \neq t\}| \leq 2$.

4 `EnASS`: Overall Procedure

Let $w_f = 2$ and $w_l = W$ be the first (respectively last) week that `EnASS` considers when filling any period and $\mathcal{R} = \mathcal{R}_0 = \{c_{\mathcal{P}}, c_{\mathcal{D}}\}$.

`EnASS` requires three parameters: p and w identify the current variable, $\overline{p}$ specifies the value assignment tried. The function returns TRUE if a solution is found, FALSE otherwise. `EnASS` is called first, after building $\overline{s}$, with $(p, w, \overline{p}) = (1, 2, 1)$ meaning that it tries to fill period 1 of week 2 with the $\overline{s}\langle 1, 2\rangle$ match. Note that we only give here the pseudo-code of `EnASS` for finding a first solution since it can easily be modified to return the entire set of all-different solutions.

$\texttt{EnASS}(p, w, \overline{p})$:

1. If $p = P + 1$ then return TRUE: A solution is obtained since all periods are filled and valid according to $\mathcal{R}$;
2. If $w = w_l + 1$ then return $\texttt{EnASS}(p + 1, w_f, 1)$: Period p is filled and valid according to $\mathcal{R}$, try to fill next period;
3. If $\overline{p} = P+1$ then return FALSE: Backtrack since no value remains for $\langle p, w\rangle$;
4. If $\exists\, 1 \le p' < p / \langle p', w\rangle = \overline{s}\langle \overline{p}, w\rangle$ then return $\texttt{EnASS}(p, w, \overline{p}+1)$: Value already assigned to a variable, try next value;
5. $\langle p, w\rangle \leftarrow \overline{s}\langle \overline{p}, w\rangle$: Try to assign a value to the current variable;
6. If $\mathcal{R}$ is locally verified and $\texttt{EnASS}(p, w + 1, 1) =$ TRUE then return TRUE: The assignment leads to a solution;
7. Undo step 5 and return $\texttt{EnASS}(p, w, \overline{p} + 1)$: $\mathcal{R}$ is locally violated or next calls lead to a failure, backtrack and try next value.

We will refer to this complete `EnASS` function with $\texttt{EnASS}_0$. All `EnASS` functions were coded in `C` (`cc` compiler) and ran on an Intel PIV processor (2 Ghz) Linux station. A time limit of 3 hours was imposed.

$\texttt{EnASS}_0$ solved Prob026 for all $T \le 32$ in less than three minutes except for $T = 24$. This clearly outperforms [4,5,6,7,8,9,10,11] and competes well with [12,13].

5 Invariants in Prob026

We describe here exact `EnASS` variants that are no more complete since they work on a subset of the $\texttt{EnASS}_0$ solutions space.

Some solutions to Prob026 verify the following $r_\Rightarrow$ property: assume that $\langle p, w\rangle$ has been fixed to a match x with $w_f \le w \le P$, then x and the $\langle p, T - w + 1\rangle$ match appear in the same period in $\overline{s}$. More formally, $\forall\, w_f \le w \le P, r_\Rightarrow(p, w) \Leftrightarrow \langle p, w\rangle = \overline{s}\langle \overline{p}, w\rangle \Rightarrow \langle p, T - w + 1\rangle = \overline{s}\langle \overline{p}, T - w + 1\rangle$.

This leads to $\texttt{EnASS}_1$ which comes from $\texttt{EnASS}_0$ by setting $w_l = P$ and adding the $r_\Rightarrow$ requirement to $\mathcal{R}_0$: $\mathcal{R}_1 = \{c_\mathcal{P}, c_\mathcal{D}, r_\Rightarrow\}$.

Columns 2–4 in Table 1 give results obtained with $\texttt{EnASS}_1$: Number $|S_1|$ of solutions ("$\ge n$" indicates that $\texttt{EnASS}_1$ found n solutions when reaching the time limit), time (including the $\overline{s}$ construction) and number of backtracks to reach a first solution. "-" marks mean that $\texttt{EnASS}_1$ found no solution within the time limit or $|\text{BT}|$ is larger than the maximal value authorized by the system.

$\texttt{EnASS}_1$ clearly outperforms $\texttt{EnASS}_0$ and [12,13]. However, other invariants are needed to tackle larger instances within the time limit. For this purpose, we reinforce the set $\mathcal{R}$ of requirements by adding the following two properties:

1. r_I: Inverse weeks w_f and W. More formally, $\forall\, w \in \{w_f, W\}, r_I(w) \Leftrightarrow \forall\, 1 \le p \le P, \langle p, w\rangle = \overline{s}\langle P - p + 1, w\rangle$;
2. r_V: Matches (t, T) form a "V" like pattern. More formally, $\forall\, 1 \le p < P, r_V(p) \Leftrightarrow \langle p, p + 1\rangle = \overline{s}\langle P, p + 1\rangle$ and $\langle p, T - p\rangle = \overline{s}\langle P, T - p\rangle$.

This leads to EnASS$_2$ with $\mathcal{R}_2 = \{c_{\mathcal{P}}, c_{\mathcal{D}}, r_{\Rightarrow}, r_I, r_V\}$. Naturally, an additional step must be added in EnASS (between steps 1 and 2) due to r_V and w_f has to be set to 3.

Columns 5–7 in Table 1 give results obtained with EnASS$_2$. Note that no result is reported for $T \bmod 4 = 0$ or $T > 70$ since EnASS$_2$ failed in these cases within the time limit.

Table 1. Computational results (times in seconds)

T	EnASS$_1$			EnASS$_2$		
	$\|S_1\|$	Time	|BT|	$\|S_2\|$	Time	|BT|
32	$\geq$ 3 657 013	<1	332 306	-	-	-
34	$\geq$ 2 173 500	<1	1 342 216	≥ 1	<1	130 149
36	$\geq$ 1 122 145	<1	2 160 102	-	-	-
38	$\geq$ 692 284	5.34	13 469 359	≥ 1	<1	2 829 421
40	$\geq$ 523 804	6.25	16 393 039	-	-	-
42	$\geq$ 339 383	107.69	256 686 929	≥ 1	2.11	7 836 823
44	$\geq$ 236 614	876.91	1 944 525 360	-	-	-
46	$\geq$ 119 383	1 573.31	3 565 703 651	≥ 1	<1	1 323 929
48	$\geq$ 90 009	542.79	1 231 902 706	-	-	-
50	$\geq$ 19 717	6 418.52	-	≥ 1	13.75	47 370 701
54	-	-	-	≥ 1	10.59	29 767 940
58	-	-	-	≥ 1	269.88	827 655 311
62	-	-	-	≥ 1	279.38	494 071 117
66	-	-	-	≥ 1	7 508.51	1 614 038 658
70	-	-	-	≥ 1	8 985.05	-

6 Conclusion

We presented EnASS, an Enumerative Algorithm for Sports Scheduling, for Prob-026 from CSPLib. Based on this basic procedure, we derived two effective exact algorithms to constraint the search process by integrating solutions properties.

Computational results showed that these algorithms clearly outperform [4,5,6,7,8,9,10,11,13] and the best known constraint programming approach [12] which is limited to $T \leq 40$: EnASS solved Prob026 in a reasonable amount of time for all $T \leq 50$ and, for $50 < T \leq 70$, solutions have been generated for some T values.

EnASS is a simple enumerative algorithm with backtrack. One possible way to solve Prob026 for larger T or to speed up EnASS could be to use more elaborated backtracking techniques.

Acknowledgements

We would like to thank the reviewers of the paper for their useful comments. Special thanks go to Professor J.H. Dinitz from the University of Vermont for having suggested to us the origins of Prob026 and some important previous works.

References

1. Gent, I., Walsh, T.: CSPLib: A benchmark library for constraints. Volume 1520 of LNCS. Springer-Verlag (1998) 480–481 `http://www.csplib.org/`.
2. Colbourn, C., Dinitz, J., eds.: The CRC Handbook of Combinatorial Designs. Volume 4 of Discrete Mathematics and Its Applications. CRC Press (1996)
3. Gelling, E., Odeh, R.: On 1-factorizations of the complete graph and the relationship to round-robin schedules. Congressus Numerantium **9** (1974) 213–221
4. McAloon, K., Tretkoff, C., Wetzel, G.: Sports league scheduling. In: Proceedings of the Third ILOG Optimization Suite International Users' Conference. (1997)
5. Gomes, C., Selman, B., McAloon, K., Tretkoff, C.: Randomization in backtrack search: Exploiting heavy-tailed profiles for solving hard scheduling problems. In: Proceedings AIPS'98, AAAI Press (1998) 208–213
6. Béjar, R., Manyà, F.: Solving combinatorial problems with regular local search algorithms. Volume 1705 of LNAI. Springer-Verlag (1999) 33–43
7. Gomes, C., Selman, B., Kautz, H.: Boosting combinatorial search through randomization. In: Proceedings AAAI/IAAI'98, AAAI Press/MIT Press (1998) 431–437
8. Béjar, R., Manyà, F.: Solving the round robin problem using propositional logic. In: Proceedings AAAI/IAAI'00, AAAI Press/MIT Press (2000) 262–266
9. Régin, J.C.: Modeling and solving sports league scheduling with constraint programming. In: Proceedings INFORMS'98. (1998)
10. Wetzel, G., Zabatta, F. Technical report, City University of New York, USA (1998)
11. Van Hentenryck, P., Michel, L., Perron, L., Régin, J.C.: Constraint programming in OPL. Volume 1702 of LNCS. Springer-Verlag (1999) 98–116
12. Régin, J.C.: Constraint programming and sports scheduling problems. In: Proceedings INFORMS'99. (1999)
13. Hamiez, J.P., Hao, J.K.: Solving the sports league scheduling problem with tabu search. Volume 2148 of LNAI. Springer-Verlag (2001) 24–36
14. Schellenberg, P., van Rees, G., Vanstone, S.: The existence of balanced tournament designs. Ars Combinatoria **3** (1977) 303–318
15. Hamiez, J.P., Hao, J.K.: A linear-time algorithm to solve the Sports League Scheduling Problem (prob026 of CSPLib). Discrete Applied Mathematics **143** (2004) 252–265
16. Haselgrove, J., Leech, J.: A tournament design problem. American Mathematical Monthly **84** (1977) 198–201
17. Gomes, C., Sellmann, M.: Streamlined constraint reasoning. Volume 3258 of LNCS. Springer-Verlag (2004) 274–289
18. Lockwood, E.: American tournaments. The Mathematical Gazette **20** (1936) 333
19. Flener, P., Frisch, A., Hnich, B., Kiziltan, Z., Miguel, I., Pearson, J., Walsh, T.: Breaking row and column symmetries in matrix models. Volume 2470 of LNCS. Springer-Verlag (2002) 462–476