Referee Assignment in Sports Leagues

Alexandre R. Duarte[1], Celso C. Ribeiro[2], Sebastián Urrutia[3], and Edward H. Haeusler[1]

[1] Department of Computer Science, Catholic University of Rio de Janeiro, Rua Marquês de São Vicente 225, Rio de Janeiro, RJ 22453-900, Brazil.
[2] Department of Computer Science, Universidade Federal Fluminense, Rua Passo da Pátria 156, Niterói, RJ 24210-240, Brazil.
[3] Department of Computer Science, Universidade Federal de Minas Gerais, Av. Antônio Carlos 6627, Belo Horizonte, MG 31270-010, Brazil.
{aduarte,hermann,celso}@inf.puc-rio.br, surrutia@dcc.ufmg.br

Abstract. Optimization in sports is a field of increasing interest. Combinatorial optimization techniques have been applied e.g. to game scheduling and playoff elimination. A common problem usually found in sports management is the assignment of referees to games already scheduled. There are a number of rules and objectives that should be taken into account when referees are assigned to games. We address a simplified version of a referee assignment problem common to many amateur leagues of sports such as soccer, baseball, and basketball. The problem is formulated by integer programming and its decision version is proved to be NP-complete. To tackle real-life large instances of the referee assignment problem, we propose a three-phase heuristic approach based on a constructive procedure, a repair heuristic to make solutions feasible, and a local search heuristic to improve feasible solutions. Numerical results on realistic instances are presented and discussed.

1 Introduction

Optimization in sports is a field of increasing interest. Some applications have been reviewed by Ribeiro and Urrutia [15]. Combinatorial optimization techniques have been applied e.g. to the traveling tournament problem [1, 5, 14, 17], to playoff elimination [16], and to the scheduling of a college basketball conference [13]. Easton et al. [6] reviewed scheduling problems in sports.

A common problem usually found in amateur sports management is the assignment of referees to games already scheduled. Sport games are regulated by rules that depend on the sport and tournament. The *o*fficiating crew is a group of referees that is responsible to ensure that all rules are respected in a game. The number of referees compounding a crew may vary, depending on the sport, league, and tournament: soccer games usually require three referees, while basketball games require two. Each member of an officiating crew is said to occupy a *r*efereeing position in a game. For example, in a regular soccer game, there are one head umpire and two side judges, totalizing three refereeing positions to be filled with referees. In some applications, managers make pre-assignments to satisfy some specific requirements. The referee assignment problem consists in assigning referees to the empty refereeing positions (not yet assigned) for all games of a league or tournament.

There are a number of rules and objectives that should be taken into account when referees are assigned to games. Games in higher divisions may require higher-skilled referees. Since referees may officiate several games during the day, travel feasibility and travel times between the facilities where the games take place have to be considered. Additionally, and especially in some amateur children leagues, some of the referees are players or their relatives. In this case, a natural constraint is that a referee cannot officiate a game in which he/she or a relative is scheduled to play.

Real-life versions of this problem appear in regional amateur leagues in the United States. Amateur leagues of several sports, such as baseball, basketball and soccer, have hundreds of games every weekend in different divisions. In a single league in California there might be up to 500 soccer games in a weekend, to be refereed by hundreds of certified referees. In the MOSA (Monmouth & Ocean Counties Soccer Association) league, New Jersey, boys and girls of ages 8 to 18 make up six divisions per age and gender group with six teams per division, totalizing 396 games every Sunday.

Referee assignment problems in other contexts have been addressed in [7, 8, 18]. Dinitz and Stinson [4] considered a problem involving referee assignment to tournament schedules, connecting room squares and balanced tournament designs. We address a basic version of a referee assignment problem common to many amateur leagues of sports such as soccer, basketball, and baseball, among others. In the next section, we state the problem considered in this work. Section 3 presents an integer programming formulation to this referee assignment problem. The decision version of the problem is proved to be NP-complete in Section 4. The proposed solution strategy is described in Section 5. In Section 6, computational results illustrating the application of the proposed approach to solve real-size randomly generated instances are shown. Concluding remarks and further extensions of this work are reported in the last Section.

2 Problem statement

We consider the general problem, in which each game has a number of refereeing positions to be assigned to referees. The games are previously scheduled and the facilities and the time in which each game takes place are known beforehand. In our approach, referees are assigned to empty (i.e., not pre-assigned) refereeing positions, not to games. This allows not only to handle referee assignment problems in sports requiring different numbers of referees, but also in tournaments where games of the same sport may need different numbers of referees due to the game division or importance. Games with pre-assigned referees to some refereeing positions can also be handled by this approach. Each refereeing position to be filled by a referee is called an empty refereeing slot.

Let $S = \{1, \ldots, n\}$ be the set of refereeing slots. Each refereeing slot $j \in S$ has to be filled by a referee with a previously determined minimum skill level q_j. Let $R = \{1, \ldots, m\}$ be the set of referees, represented by their indices. Each referee $i \in R$ has a certain skill level, denoted by p_i, defining the refereeing slots in which he/she can officiate. Referees may declare their unavailability to officiate at certain time slots. Furthermore, each referee $i \in R$ establishes M_i as the maximum number of games he/she is able to officiate and T_i as the target number of games he/she is willing to officiate. Travels are not allowed, i.e. referees that officiate more than one game in the same day must be assigned to games that take place at the same facility. Moreover, referees that are also players have a hard facility assignment constraint: they must officiate at the same facility where they play.

The *Referee Assignment Problem* (RAP) consists in assigning referees to all refereeing slots associated to games scheduled in a given time interval (typically, a day or a weekend), minimizing the sum over all referees of the absolute value of the difference between the target and the actual number of games assigned to each referee and satisfying a set of hard constraints listed below:

(a) all refereeing slots must be filled for all games;
(b) referees cannot officiate more than one game in overlapping time slots;
(c) referees cannot officiate games in time slots where they declared to be unavailable;
(d) referees must meet the minimum skill level established for each refereeing slot;
(e) referees cannot officiate more than a given maximum number of games; and
(f) each referee can officiate in only one facility.

3 Integer programming model

The problem described in the previous section can be formulated by integer programming. We denote by d_i the absolute value of the difference between the target and the actual numbers of games assigned to referee $i \in R$. The following variables are used in the formulation:

$$x_{ij} = \begin{cases} 1, & \text{if referee } i \in R \text{ is assigned to slot } j \in S \\ 0, & \text{otherwise.} \end{cases}$$

Furthermore, $C(j) \subseteq S$ denotes the set of refereeing slots conflicting with slot $j \in S$, i.e. refereeing slots that take place at different facilities than or overlapping with j. Also, $U(i) \subseteq S$ represents the set of refereeing slots to which referee $i \in R$ cannot be assigned due to a lower skill level or to his/her unavailability. The RAP integer programming model can be formulated as:

$$\text{minimize} \sum_{i=1}^{m} d_i \tag{1}$$

subject to:

$$d_i = |T_i - \sum_{j=1}^{n} x_{ij}| \quad \forall i = 1, \ldots, m \tag{2}$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad \forall j = 1, \ldots, n \tag{3}$$

$$\sum_{j=1}^{n} x_{ij} \leq M_i \quad \forall i = 1, \ldots, m \tag{4}$$

$$x_{ij} + x_{ij'} \leq 1 \quad \forall i = 1, \ldots, m, \quad \forall j = 1, \ldots, n, \quad \forall j' \in C(j) \tag{5}$$

$$\sum_{j \in U(i)} x_{ij} = 0 \quad \forall i = 1, \ldots, m \tag{6}$$

$$x_{ij} \in \{0, 1\} \quad \forall i = 1, \ldots, m, \quad \forall j = 1, \ldots, n. \tag{7}$$

The objective function (1) states that the sum over all referees of the slack between their target and actual numbers of scheduled games is minimized. Constraints (2) enforce that d_i is equal to the absolute value of the difference between the target and actual numbers of games assigned to referee $i \in R$. Constraints (3) ensure that every refereeing slot must be assigned to exactly one referee. Constraints (4) establish the upper bound to the number of refereeing slots that can be assigned to each referee. Constraints (5) ensure that refereeing slots with timetabling conflicts or taking place at different facilities cannot be assigned to the same referee. Constraints (6) prevent assignments that violate minimum skill level and unavailability restrictions (alternatively, all variables x_{ij} with $j \in U(i)$ may simply be removed from the model). Constraints (7) establish the integrality of the decision variables. We notice that constraints (3) and (4) are those characterizing a generalized assignment problem [9].

4 NP-completeness

We consider the following feasibility decision problem (DRAP):
Problem: REFEREE ASSIGNMENT
Input: Set S of refereeing slots, set R of referees, and the maximum

number of games to be officiated by each referee.
Question: Is there an assignment of referees in R to refereeing slots in S satisfying constraints (a) to (f)?

Theorem 1. *DRAP is NP-complete.*

Proof. DRAP is clearly in NP, since the feasibility of any assignment can be checked in time polynomial in $|R|$ and $|S|$. To prove its NP-completeness, we use a transformation from the problem of Partition into Bounded Independent Sets on interval graphs (PBIS). Given an undirected graph $G = (V, E)$ and integer numbers k and k', this problem consists in deciding whether there exists a partition of V into k independent sets $I_1, \dots, I_k$, with $|I_i| \leq k'$ for $1 \leq i \leq k$. PBIS is NP-complete even if G is an interval graph, see [2].

We build an instance of DRAP where the set $S = \{1, \dots, |V|\}$ has exactly $|V|$ refereeing slots, each of them associated with a different game. All games take place at the same date and facility. The minimum skill level associated to each refereeing slot $j \in S$ is set as $q_j = 1$. Let $R = \{1, \dots, k\}$ be the set of available referees and set $M_i = k'$, $p_i = 1$, and $U(i) = \emptyset$ for every $i \in R$. The linear time recognition algorithm of Corneil et al. [3] is used to build an interval representation of G. Each interval of the latter is mapped to one refereeing slot, whose starting and ending times coincide with the starting and ending points of the corresponding interval.

We now prove that given the interval graph G and the integer numbers $k, k' \in N$, there is a partition of V into independent sets $I_1, ..., I_k$ with $|I_i| \leq k'$ for $1 \leq i \leq k$ if, and only if, there is a feasible assignment of the referees in R to the set of refereeing slots S built as above, subject to constraints (a) to (f), with $q_j = 1$ for all $j \in S$, $M_i = k'$ and $p_i = 1$ for all $i \in R$.

First, suppose we have a partition of $G = (V, E)$ into independent sets $I_1, \dots, I_k$, with $|I_i| \leq k'$ for $1 \leq i \leq k$. The slots assigned to referee $i \in R$ are exactly those corresponding to the vertices in I_i. This association guarantees that constraints (a) and (b) are satisfied. Constraints (c) and (f) are trivially satisfied, since $U(i) = \emptyset$, for all $i \in R$, and all games take place at the same facility. Since $p_i = 1$, for all $i \in R$, and $q_j = 1$, for all $j \in S$, constraint (d) is also trivially satisfied. Finally, constraint (e) is satisfied since $|I_i| \leq k' = M_i$ for all $i \in R$.

We now consider a feasible solution to an instance of DRAP. We construct the interval graph $G = (V, E)$ by assigning each refereeing slot $j \in S$ to a vertex $j \in V$. There is an edge $(j, j') \in E$ for each pair j and j' of overlapping refereeing slots. The partition of V into the independent sets $I_1, \dots, I_k$ is such that vertex j belongs to the independent set I_i if referee i is assigned to refereeing slot j. As the number of slots assigned to each referee is bounded by k', this partition into bounded independent sets is feasible. □

5 Solution approach

We propose a three-phase heuristic approach to tackle real-life large instances of the referee assignment problem. The first phase consists in applying a greedy heuristic to find an initial solution, possibly violating some constraints. The second phase is a repair heuristic, which is applied whenever necessary to make the initial solution feasible. Finally, another heuristic is used in the third phase to improve the feasible solution. The algorithms used in the second and third phases are based on principles similar to the Iterated Local Search (ILS) metaheuristic [10–12]. Algorithm 1 shows the general scheme of this approach.

The next subsection presents the constructive algorithm used to build initial solutions to the subsequent phases (repair and improvement). Section 5.2 describes the ILS scheme which is the basis for both the repair and improvement phases. Details of the local search procedure used within the ILS scheme are commented in Section 5.3. Sections 5.4 and 5.5 discuss some issues of the ILS scheme that are particular to each of the repair or improvement heuristics.

```
Algorithm $RefereeAssignmentHeuristic$(MaxIterations1, MaxIterations2)
Solution $\leftarrow$ $BuildGreedySolution()$;
if not isFeasible(Solution) then
    Solution $\leftarrow$ $RepairHeuristic$(Solution, MaxIterations1);
end
if isFeasible(Solution) then
    Solution $\leftarrow$ $ImprovementHeuristic$(Solution, MaxIterations2);
    return Solution;
else
    return no feasible solution was found;
end
```

Algorithm 1: Referee assignment heuristic.

5.1 Greedy constructive heuristic

The first phase of our approach attempts to build a feasible solution. Its main principle consists in assigning first the referees that are also players to the facilities where they have a game. Next, while there are unassigned refereeing slots and unassigned referees, the heuristic greedily selects a facility with unassigned refereeing slots, obtains an unassigned referee and assigns refereeing slots to this referee without violating any constraint. Finally, if any refereeing slot remains unassigned, the solution is completed with infeasible assignments.

```
1  Algorithm BuildGreedySolution()
2  $S^u \leftarrow \{j = 1, \dots, n : \sum_{i=1}^{m} x_{ij} = 0\}$;
3  $R^{HF} \leftarrow \{i = 1, \dots, m :$ referee i plays at least one game$\}$;
4  $R^{NHF} \leftarrow R - R^{HF}$;
5  while $R^{HF} \neq \emptyset$ do
6      Randomly select a referee $i \in R^{HF}$;
7      $R^{HF} \leftarrow R^{HF} - \{i\}$;
8      Let $f$ be the facility where referee $i$ plays a game;
9      forall $j \in S^u$ : refereeing slot $j$ takes place at facility $f$ do
10         if referee $i$ can be assigned to refereeing slot $j$ then
11             $x_{ij} \leftarrow 1$;
12             $S^u \leftarrow S^u - \{j\}$;
13         end
14     end
15 end
16 while $S^u \neq \emptyset$ and $R^{NHF} \neq \emptyset$ do
17     $\overline{p} \leftarrow \max_{i \in R^{NHF}}\{p_i\}$;
18     Let $f$ be the facility with the strongest need for referees with
       skill level equal to $\overline{p}$;
19     Randomly select a referee $i \in R^{NHF}$ with $p_i = \overline{p}$;
20     $R^{NHF} \leftarrow R^{NHF} - \{i\}$;
21     forall $j \in S^u$ : refereeing slot $j$ takes place at facility $f$ do
22         if referee $i$ can be assigned to refereeing slot $j$ then
23             $x_{ij} \leftarrow 1$;
24             $S^u \leftarrow S^u - \{j\}$;
25         end
26     end
27 end
28 if $S^u \neq \emptyset$ then
29     forall $s_j \in S^u$ do
30         Let $f$ be the facility where $s_j$ takes place;
31         Randomly select a referee $i \in R$ officiating at facility $f$;
32         $x_{ij} \leftarrow 1$;
33         $S^u \leftarrow S^u - \{j\}$;
34     end
35 end
36 return Solution : referee $i \in R$ is assigned to refereeing slot $j \in$
   $S$ if and only if $x_{ij} = 1$;
```

Algorithm 2: Greedy constructive heuristic.

The pseudo-code of this heuristic is presented in Algorithm 2. We denote by S^u the set of all unassigned refereeing slots, by R^{HF} the set of referees associated with a hard facility constraint, and by R^{NHF} the set of referees with no hard facility constraint, i.e. $R = R^{HF} \cup R^{NHF}$. These sets are initialized respectively in lines 2, 3, and 4. The loop in lines 5 to 15 is performed until all referees associated with hard facility constraints have been examined and assigned to as many refereeing slots as possible. Next, the loop in lines 16 to 27 attempts to fill the remaining unassigned refereeing slots with referees without hard facility constraints.

A greedy criterion is applied in line 18 to select a facility f with the strongest need for referees with a certain skill level $\overline{p}$ computed in line 17. The computation of the greedy criterion is based on two measures: (a) an estimate of the minimum number of referees with skill level $\overline{p}$ needed to officiate at facility f and (b) the number of unassigned refereeing slots in facility f with minimum skill level less than or equal to $\overline{p}$. Finally, if unassigned refereeing slots still remain at line 28, then the loop in lines 29 to 34 makes infeasible assignments to complete the solution.

5.2 ILS-based scheme

Both the repair and the improvement heuristics use similar ILS schemes. They start by applying a first improving local search to the initial solution. Since the local search involves moves that change referee assignments for only one facility at a time, it should be applied to every facility.

Then, for a given number of iterations, a perturbation involving one pair of facilities is applied to the current solution. Each perturbation is followed by two applications of the local search procedure, once to each of the facilities of the pair involved in the perturbation. The solution obtained by local search is accepted if it satisfies a given acceptance criterion. This scheme is illustrated by the pseudo-code of Algorithm 3.

```
1  Algorithm ILS_Scheme(Solution, MaxIterations)
2  foreach facility $f$ do
3      Solution ← LocalSearch($f$, Solution);
4  end
5  for $i = 1, \dots,$ MaxIterations do
6      NewSolution ← Perturbation(Solution);
7      Let $f_1$ and $f_2$ be the facilities involved in the perturbation;
8      NewSolution ← LocalSearch($f_1$, NewSolution);
9      NewSolution ← LocalSearch($f_2$, NewSolution);
10     Solution ← AcceptanceCriterion(Solution, NewSolution);
11 end
12 return Solution;
```

Algorithm 3: ILS-based scheme.

We describe next the local search procedure and its associated neighborhoods, followed by the repair and improvement heuristics.

5.3 Local search and neighborhoods

Solutions built by the constructive heuristic are not necessarily optimal or even feasible. A local search algorithm successively replaces the current solution by a better one in a neighborhood of the first, terminating at a local optimum. In a *first improving* strategy, the current solution is replaced by the first neighbor whose cost function value improves that of the current solution. We consider two neighborhoods for local search:

– swap moves: referees assigned to two refereeing slots are swapped (such moves do not change the number of games assigned to each referee) and
– replace moves: the referee assigned to a refereeing slot is replaced by another referee (such moves increase by one the number of games assigned to one referee and decrease by one the number of games assigned to the other).

As referees cannot be assigned to games at different facilities (hard constraint), only moves involving referees that officiate at the same facility (or do not officiate at all) are allowed (otherwise, and unless two referees were assigned to exactly one game each, a move involving referees that officiate at different facilities would imply at least one constraint violation). Such restricted neighborhoods considering only moves involving the same facility allow the acceleration of the local search.

The local search procedure performed within the ILS scheme is divided into two phases, both of them using a first improving strategy. In the first phase, only improving moves are accepted. The second phase also accepts moves leading to solutions at least as good as the current one, using a list of forbidden moves to prevent cycles. Each phase is separated in two parts: first, only swap moves are considered; next, only replace moves.

5.4 Repair heuristic

The repair heuristic follows the ILS scheme in Algorithm 3, based on local search and perturbations. It attempts to make feasible the initial solution obtained by the greedy constructive heuristic. Constraint violations in the initial solution may concern time conflicts, referee unavailabilities, inadequate skill levels, or maximum numbers of games. The repair heuristic minimizes the number of constraint violations of an infeasible initial solution. A modified solution is feasible if and only if it has no constraint violations.

Solutions built by the constructive heuristic have the property that all referees officiate in at most one facility. Therefore, the local search considers only moves involving referees that officiate at the same facility (or do not officiate at all) and attempts to find a feasible solution by minimizing the number of constraint violations. Ties with respect to the number of constraint violations are broken in favor of the solution with the smaller objective function value (i.e., the absolute value of the difference between the target and the actual number of games assigned to each referee involved in a move).

The perturbation procedure within the repair heuristic changes the facility where one of the referees officiates, according to the following steps:

1. select a facility f with infeasible referee assignments;
2. select the highest minimum skill level ℓ^* over all refereeing slots in facility f assigned to referees with at least one violation;
3. determine a referee r that officiates at another facility f' (or does not officiate at all) whose skill level is greater than or equal to ℓ^*;
4. randomly select referees other than r that officiate at facility f' and assign them to the refereeing slots currently assigned to r;
5. assign referee r to any refereeing slot at facility f currently assigned to a referee with at least one violation.

The solution `NewSolution` obtained after a perturbation followed by local search is accepted by procedure *AcceptanceCriterion* if and only if it has fewer constraint violations or the same number of violations and a smaller objective function value than the current solution.

5.5 Improvement heuristic

Once a feasible solution is known, the improvement heuristic is performed as an attempt to reduce the current value of the objective function, i.e. to minimize the sum over all referees of the absolute value of the difference between the target and the actual number of games assigned to each of them. The improvement heuristic is also based on the ILS scheme presented in Section 5.2.

The local search used in the improvement heuristic differs slightly from that used in the repair heuristic: swap moves are not performed (because they cannot improve the objective function) and only moves and perturbations that preserve feasibility are considered.

The perturbations applied to the current solution within the improvement heuristic select two referees that officiate at different facilities and swap all their assignments, according to the following steps:

1. Choose a possible perturbation: select two referees officiating at different facilities. If the swap of all their assignments does not preserve feasibility, then go to the final step. Otherwise, temporarily perform the swap of all assignments of the two selected referees.
2. Look ahead: for each of the two selected referees whose target number of games is greater than the new (after the swap) number of games he/she will officiate, check if there are other refereeing slots in which

he/she could officiate at the new facility. This look ahead procedure only checks refereeing slots that are currently assigned to referees officiating more games than their targets and only until the referee under investigation does not officiate more games than his/her target. Whenever possible, temporarily replace the previously assigned referee by the new referee involved in the perturbation.
3. Accept the perturbation: if the perturbation applied to the two referees (swap of all their assignments), followed by all possible replace moves in the destination facility for each referee, decreases the objective function value, then the perturbation is accepted and all temporary changes are made final. Otherwise, go to step 4.
4. Return: if all pairs of referees have already been considered, then perform the swap of all assignments of the pair of referees that increases the least the objective function, while preserving feasibility. Otherwise, return to the first step to select a new pair of referees.

Solution `NewSolution` obtained after a perturbation followed by local search is always accepted, because the heuristic chooses either an improving perturbation or the one that deteriorates the least the current solution.

6 Computational results

Only very small instances with up to 40 games and 60 referees could be exactly solved by a commercial solver such as CPLEX 9.0, applied directly to the integer programming model presented in Section 3. In this section, we report computational results obtained on realistic, real-size randomly generated instances.

6.1 Test problems

Since the RAP is a new problem, no benchmark instances are available. Test instances have been randomly generated, following patterns similar to those observed in real-life soccer instances. They have up to 500 games and up to 1000 referees, with different numbers of referees, different numbers of facilities, and different patterns of the target number of games each referee is willing to officiate.

Each game lasts for two hours and is scheduled to start at any hour from 8 AM to 7 PM. A facility and a starting time are randomly assigned to each game. There are three refereeing slots to be assigned to referees in each game: one of them requires a higher skill level (the head umpire), while the two other require less skilled referees (the two side judges).

The skill level p_i and the maximum number of games M_i for each referee $i \in R$ are randomly generated in $\{1, \ldots, 8\}$ and $\{2, \ldots, 8\}$, respectively.

Two different patterns were used to generate the target number of games T_i for each referee $i \in R$. According to pattern P_0, T_i is randomly selected from $\{0, \ldots, M_i\}$. In the case of pattern P_1, T_i is proportional to $1/p_i$: the higher the referee skill level is, the lower his/her target number of games is. This reasoning allows to create some challenging instances in which the more qualified a referee is, the lower is the number of games he/she wants to officiate.

Table 1 presents the parameter values used to generate the test problems. Five different instances were generated for each of the 36 parameter combinations, in a total of 180 test problems. All test problems are available upon request to the authors.

Games	Referees	Facilities	Patterns
300	450, 525, 600	40, 50	P_0, P_1
400	600, 700, 800	55, 65	P_0, P_1
500	750, 875, 1000	65, 85	P_0, P_1

Table 1. Instances dimensions combinations.

6.2 Numerical results

The experimental results reported in this section were obtained on a 2.0 GHz Pentium IV processor with 512 Mbytes of RAM memory running Windows 2000$^{\text{TM}}$. All codes were implemented in C. The maximum number of iterations performed by the repair and the improvement heuristics was set at 1000 and 200, respectively.

Tables 2 to 7 summarize the results obtained for some classes of test problems by the heuristic approach. We only report results for the hardest problems, in which the number of games (500) is large and the number of referees is limited (problems with 1,000 referees have been discarded). Initial solutions are computed by the greedy heuristic. Computation times (in seconds) and objective function values are average results over ten runs for each instance. For each phase of the heuristic (construction, repair, improvement), we present its computation time (in seconds) and the objective function value of the solutions found. For the construction and repair phases, we also report the number of runs where a feasible solution was found.

The constructive heuristic ran in less than 0.1 second for all instances and found feasible solutions for most of them. The repair heuristic found a feasible solution in less than 20 seconds in almost all cases when the constructive heuristic failed. This is due to the effectiveness of the constructive algorithm. Table 8 depicts some illustrative results on instances with 500 games, 750 referees, and 85 facilities to support this claim. For each instance, we show that the total times to build feasible solutions starting from randomly generated assignments are much larger than those observed when the initial solution is computed by the greedy constructive algorithm. We also observed that the repair phase failed to build feasible solutions from randomly generated initial solutions for some instances, even after 10,000 iterations, but always succeeded when starting from a solution built by the constructive heuristic. We stress the importance of quick procedures for finding initial solutions for hard combinatorial problems in sports, as already noticed by Ribeiro and Urrutia [17].

The improvement phase improved the objective function value of feasible initial solutions by up to 63%. Instances with more facilities or fewer referees were harder in terms of computation times and building feasible solutions.

	Construction			Repair			Improvement	
Instance	time (s)	value	feas.	time (s)	value	feas.	time (s)	value
I_1	0.02	1286.20	10	—	—	—	32.34	619.60
I_2	0.02	1360.00	5	0.47	1338.00	10	31.81	623.40
I_3	0.02	1269.00	2	0.60	1247.00	10	33.87	621.60
I_4	0.03	—	—	1.14	1303.20	10	30.28	627.20
I_5	0.03	1302.67	3	1.40	1259.14	10	33.73	654.00

Table 2. 500 games, 750 referees, 65 facilities, and pattern P_0.

	Construction			Repair			Improvement	
Instance	time (s)	value	feas.	time (s)	value	feas.	time (s)	value
I_1	0.02	1752.75	8	0.66	1709.00	10	31.59	1022.60
I_2	0.02	1669.57	7	0.02	1675.67	10	30.34	888.60
I_3	0.02	—	—	5.91	1569.80	10	29.55	942.00
I_4	0.03	1777.00	1	1.53	1725.00	10	31.81	1033.80
I_5	0.02	1704.80	5	0.49	1704.80	10	28.17	952.00

Table 3. 500 games, 750 referees, 65 facilities, and pattern P_1.

	Construction			Repair			Improvement	
Instance	time (s)	value	feas.	time (s)	value	feas.	time (s)	value
I_1	0.03	—	—	11.27	1111.60	10	22.74	612.60
I_2	0.03	—	—	6.69	1231.60	10	24.18	715.20
I_3	0.03	—	—	11.33	1182.40	10	22.29	672.60
I_4	0.03	—	—	4.61	1229.00	10	23.45	692.80
I_5	0.03	—	—	3.39	1234.60	10	19.50	646.00

Table 4. 500 games, 750 referees, 85 facilities, and pattern P_0.

	Construction			Repair			Improvement	
Instance	time (s)	value	feas.	time (s)	value	feas.	time (s)	value
I_1	0.03	—	—	2.75	1670.60	10	25.85	1043.80
I_2	0.02	—	—	19.29	1649.50	8	26.15	1147.00
I_3	0.03	—	—	14.77	1586.60	10	24.65	1107.60
I_4	0.03	—	—	1.22	1602.80	10	25.59	1007.40
I_5	0.03	—	—	2.69	1611.20	10	24.60	1002.80

Table 5. 500 games, 750 referees, 85 facilities, and pattern P_1.

	Construction			Repair			Improvement	
Instance	time (s)	value	feas.	time (s)	value	feas.	time (s)	value
I_1	0.03	1582.80	10	—	—	—	45.07	574.40
I_2	0.02	1627.40	10	—	—	—	42.92	609.20
I_3	0.03	1535.40	10	—	—	—	44.62	558.20
I_4	0.03	1655.60	10	—	—	—	43.28	576.60
I_5	0.02	1626.00	10	—	—	—	43.31	619.20

Table 6. 500 games, 875 referees, 65 facilities, and pattern P_0.

	Construction			Repair			Improvement	
Instance	time (s)	value	feas.	time (s)	value	feas.	time (s)	value
I_1	0.03	2195.20	10	—	—	—	39.64	1091.80
I_2	0.03	2040.20	10	—	—	—	42.33	955.40
I_3	0.02	2153.40	10	—	—	—	41.34	1032.20
I_4	0.03	2173.40	10	—	—	—	42.54	1069.00
I_5	0.03	2137.60	10	—	—	—	39.73	1035.00

Table 7. 500 games, 875 referees, 65 facilities, and pattern P_1.

		Greedy			Random	
Instance	pattern	const. (s)	repair (s)	feas.	repair (s)	feas.
I_1	P_0	0.03	11.27	10	79.8	9
I_2	P_0	0.03	6.69	10	80.8	10
I_3	P_0	0.03	11.33	10	86.2	8
I_4	P_0	0.03	4.61	10	30.6	10
I_5	P_0	0.03	3.39	10	29.1	10
I_1	P_1	0.03	2.75	10	33.5	10
I_2	P_1	0.02	19.29	10	134.6	2
I_3	P_1	0.03	14.77	10	135.1	8
I_4	P_1	0.03	1.22	10	38.0	10
I_5	P_1	0.03	2.69	10	32.9	10

Table 8. Greedy vs. randomly generated initial solutions.

In another experiment, we compared the results obtained by the heuristic with those found by CPLEX 9.0 when applied to formulation (1)-(7) for some small instances that could be exactly solved within reasonable computation time. The heuristic always received the same computation time that CPLEX took to find the optimal solution. Some numerical results are summarized in Table 9. For each instance, we first give its identification and the pattern used for its generation. The two next columns display the optimal solution value and the computation time in seconds taken by CPLEX (and, consequently, given to the heuristic). Next, the table shows the average and the best solution values found by the heuristic over ten runs. The last column gives the time taken to find the best solution in the corresponding run. These results show that the heuristic was able to find the optimal solution for three out of the five test instances considered in this table. Furthermore, the times taken by the heuristic are significantly smaller than those observed with CPLEX, even for the small instances that the latter was able to solve to optimality.

		CPLEX		Heuristic		
Instance	pattern	optimum	time (s)	average	best	time (s)
I$_2$	P_0	43	164.00	47.00	44	18.99
I$_3$	P_0	18	200.00	20.80	18	3.05
I$_5$	P_0	44	137.00	45.20	44	11.45
I$_2$	P_1	65	128.00	67.20	65	15.32
I$_5$	P_1	72	47.00	82.40	75	8.83

Table 9. 33 games, 57 referees, 5 facilities, patterns P_0 and P_1.

In the last computational experiment, we replaced the linear objective function by a quadratic penalization. Table 10 details the differences between the target and the actual numbers of games assigned to each referee in the solutions obtained with the linear and quadratic cost functions for instance I$_3$ with 500 games, 750 referees, 85 facilities, and pattern P_1. It shows that more balanced solutions can be obtained when the quadratic cost function is used, in which the occurrences of larger differences are replaced by those of smaller differences concentrated at only one unit. The computation times of the constructive, repair, and improvement heuristics were not affected by the change of the objective function. We observe that 76 extremely privileged referees (i.e., those officiating exactly their target number of games) in the solution obtained with the linear cost function lose their privileges in the solution obtained with the quadratic cost function. Also, 23 referees that were far from their targets are now very close to them (i.e., their differences are now equal to one). The new solution obtained with the quadratic cost function is certainly more fair than that associated with the linear costs.

7 Concluding remarks

We introduced in this paper the referee assignment problem, a new optimization problem in sports. The problem was formulated as an integer model and the NP-completeness of its decision version was proved.

	Number of referees	
Difference from target	Linear penalties	Quadratic penalties
0	255	179
1	182	281
2	156	149
3	67	66
4	50	43
5	23	18
6	13	10
7	3	3

Table 10. Linear vs. quadratic objective functions.

A three-phase heuristic was proposed and implemented. Computational results on realistic instances showed the effectiveness of the greedy constructive heuristic combined with the repair heuristic to build feasible solutions. The improvement procedure used in the third phase was able to substantially improve solution quality. We also illustrated the importance of a quick construction procedure to build initial solutions.

We also compared the solutions obtained by the heuristic with those produced by CPLEX for some small instances that could be solved to optimality in reasonable computation times. The heuristic not only was able to find the optimal solutions for several instances, but also the computation times to find the best solution were significantly smaller than those observed with CPLEX.

Finally, we investigated and compared the behavior of an alternative quadratic objective function, which was able to find more fair solutions than the formulation with a linear cost function.

We are currently working on some extensions addressing further constraints of real-life applications, such as the existence of hard and soft links between some referees. In these situations, some referees may want to work with the same referees as partners in every game they officiate. This is the case when they are more confident to officiate together, but also when they want to travel in car pools or to officiate with relatives. Decision makers may also want referee assignments matching preferences regarding the facilities, divisions, and time slots where the referees officiate.

Another extension occurs when referees are able to officiate games in different facilities. In this case, travel times between facilities should also be considered for feasibility matters. They can also be incorporated to the objective function, so as that the minimization of the total traveling time turns out to be another objective. The minimization of the waiting times between consecutive games assigned to the same referee is also relevant.

The referee assignment problem has clearly the flavor of a multi-criteria optimization application. We are also investigating the use of multi-criteria methods coupled with a decision support system for its solution in practice.

References

1. A. Anagnostopoulos, L. Michel, P. Van Hentenryck, and Y. Vergados. A simulated annealing approach to the traveling tournament problem. *Journal of Scheduling*, 9:177–193, 2006.
2. H.L. Bodlaender and K. Jansen. Restrictions of graph partition problems - Part I. *Theoretical Computer Science*, 148:93–109, 1995.
3. D.G. Corneil, S. Olariu, and L. Stewart. The ultimate interval graph recognition algorithm? In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 175–180. SIAM, 1998.
4. J.H. Dinitz and D.R. Stinson. On assigning referees to tournament schedules. *Bulletin of the Institute of Combinatorics and its Applications*, 44:22–28, 2005.
5. K. Easton, G. L. Nemhauser, and M. Trick. The traveling tournament problem: Description and benchmarks. In *Proceedings of the 7th International Conference on Principles and Practice of Constraint Programming*, pages 580–584, London, 2001. Springer-Verlag.
6. K. Easton, G.L. Nemhauser, and M. Trick. Sports scheduling. In J.T. Leung, editor, *Handbook of Scheduling: Algorithms, Models and Performance Analysis*, pages 52.1–52.19. CRC Press, 2004.
7. J.R. Evans. A microcomputer-based decision support system for scheduling umpires in the American baseball league. *Interfaces*, 18:42–51, 1988.
8. J.R. Evans, J.E. Hebert, and R.F. Deckro. Play ball - The scheduling of sports officials. *Perspectives in Computing*, 4:18–29, 1984.
9. M.L. Fisher, R. Jaikumar, and L.N. Van Wassenhove. A multiplier adjustment method for the generalized assignment problem. *Management Science*, 32:1095–1103, 1986.
10. H.P. Lourenço, O. Martin, and T. Stützle. Iterated Local Search. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, pages 321–353. Kluwer Academic Publishers, 2002.
11. O. Martin and S.W. Otto. Combining simulated annealing with local search heuristics. *Annals of Operations Research*, 63:57–75, 1996.
12. O. Martin, S.W. Otto, and E.W. Felten. Large-step Markov chains for the traveling salesman problem. *Complex Systems*, 5:299–326, 1991.
13. G.L. Nemhauser and M.A. Trick. Scheduling a major college basketball conference. *Operations Research*, 46:1–8, 1997.
14. R.V. Rasmussen and M.A. Trick. Round robin scheduling - A survey. Technical report, Department of Operations Research, University of Aarhus, 2006.
15. C.C. Ribeiro and S. Urrutia. OR on the ball: Applications in sports scheduling and management. *OR/MS Today*, 31:50–54, 2004.

16. C.C. Ribeiro and S. Urrutia. An application of integer programming to playoff elimination in football championships. *International Transactions in Operational Research*, 12:375–386, 2005.
17. C.C. Ribeiro and S. Urrutia. Heuristics for the mirrored traveling tournament problem. *European Journal of Operational Research*, 179:775–787, 2007.
18. M.B. Wright. Scheduling English cricket umpires. *Journal of the Operational Research Society*, 42:447–452, 1991.