No. 1006

Solving Sports Scheduling Problems Using Network Structure

by

Ayami Suzuka, Yasufumi Saruwatari
and Akiko Yoshise

October 2002

Solving Sports Scheduling Problems Using Network Structure

Ayami Suzuka[1], Yasufumi Saruwatari[2], and Akiko Yoshise[3]

[1] Graduate School of Systems and Information Engineering, University of Tsukuba, Ibaraki, Japan,
asuzuka@sk.tsukuba.ac.jp

[2] Institute of Policy and Planning Sciences, University of Tsukuba, Tokyo, Japan,
saru@gssm.otsuka.tsukuba.ac.jp

[3] Institute of Policy and Planning Sciences, University of Tsukuba, Ibaraki, Japan,
yoshise@sk.tsukuba.ac.jp [⋆]

Abstract. This paper aims to provide a widely applicable technique for solving Sports Scheduling Problems. We give a set of conditions which are often imposed in practice, and define a problem, Sports Scheduling Problem in General Form (abbreviated to SSPGF). It is easy to formulate the problem as an integer program, but solving the program may entail a tremendous computational cost in general. For the purpose of reducing the cost, we extract a subproblem having a network structure and propose a branch-and-bound algorithm. Computational experiments show that our algorithm solves the SSPGF correctly.

Keywords: sports scheduling, round-robin, network structure, branch-and-bound method

1 Introduction

Sports competition timetables have been often made by hand and individually, because of a lot of requirements concerning maintaining fairness, athletic facilities, commercial interests and so on.

Recently, several systematic algorithms have been developed in order to avoid such a time-consuming work. De Werra [7, 8] introduce characterizations and graph-theoretical properties of some particular schedules, which are exploited in a schedule for Dutch major soccer league [13]. Ferland and Fleurent [9] develop an interactive decision support system for constructing schedules for the National Hockey League and other hockey leagues. Campbell and Chen [6], and Nemhauser and Trick [11] study the problems arising in basketball leagues, based on integer programming and enumeration techniques.

Most of these works employ case-oriented approach and hence, it is difficult to apply one of proposed methods to other problems as it is. On the other hand,

[⋆] Research supported in part by Grant-in-Aid for Scientific Research ((C2)13650061) of the Ministry of Education, Culture, Sports, Science and Technology of Japan.

we see that some requirements have been imposed commonly regardless of the kind of sports competition. In this paper, we give a set of such requirements and introduce a problem, the Sports Scheduling Problem in General Form (abbreviated to SSPGF) of finding a solution which satisfies all of the requirements.

As we will see in Section 2, the SSPGF can be formulated as an integer program. Therefore, several branch-and-bound algorithms are applicable for solving the SSPGF. In fact, our algorithm is based on the branch-and-bound technique, but completely different from typical algorithms such as using the LP relaxation: Our intention is to give a graph representation of the SSPGF and to extract a well-structured subproblem in order to develop an efficient branch-and-bound algorithm.

Our paper is organized as follows.

In Section 2, we define the SSPGF in detail. We assume that teams play a round-robin tournament in the competition, i.e., each team is expected to play against every other team a fixed number of times throughout the competition. The assumption leads us to three operational requirements (C.1-C.3) on the schedule. In addition, we impose three fairness requirements (C.4-C.6), which are sometimes indispensable to improve the quality of the schedule. We also assume that the number of attendance for each match may depend on some factors assigned to the match, but the organization which hosts the competition knows an expected number of attendance for each case, and hopes to maximize the total expected number of attendance of the competition. According to these considerations, we formulate the SSPGF as an optimization problem with binary variables.

In Section 3, we provide a branch-and-bound algorithm for solving the SSPGF. To do this, we prepare a graph and show that a subproblem of the SSPGF can be regarded as a problem of finding a longest path in the graph. We also define a branching rule for finding an optimal solution to the SSPGF.

Some computational experiments of our algorithm are reported in Section 4. These results show that our algorithm works well and finds an optimal solution to the SSPGF with reasonable computation time. Thus we can conclude that our algorithm is promising for practical use in sports scheduling.

Section 5 is concluding remarks.

2 Problem

In this section, we define our problem, the *Sports Scheduling Problem in General Form* (abbreviated to SSPGF) and formulate it as an integer program.

2.1 Definition of SSPGF

We suppose that the competition is a round-robin tournament of $2r$ rounds and that

- the set P of $2n$ teams and

– the set T of m terms in a season

are given a priori. We also suppose that each team has a specific home. We define SSPGF as a problem of finding a schedule under the following 6 conditions. The conditions can be divided into two categories:

– Operational requirements, and
– Fairness requirements.

The Operational requirements are concerned with the fundamental principle of round-robin tournaments:

C.1 Each team must play exactly one match in a term.
C.2 Every match between i and j must be held at either i's or j's home.
C.3 The number of matches between i and j must be $2r$ for any i, j.

The above conditions C.1 and C.3 claim that the number of terms m should be $m = 2r(2n-1)$. On the other hand, we consider the following requirements as the Fairness requirements which should be taken into consideration for maintaining fairness of the competition.

C.4 Among the matches between i and j the number of matches at i's (resp. j's) home must be r for any i, j.
C.5 At most one match between i and j can be held in w consecutive terms for any i, j.
C.6 No team is allowed to play at its home in more than h consecutive terms.
C.6' No team is allowed to play at the opponent's home, i.e., away, in more than a consecutive terms.

In what follows, we will ignore the requirement C.6' since it is not essential in the SSPGF due to symmetry between the requirements C.6 and C.6'.

In terms of commercial aspects, we set our goal on maximizing the total attendance of the competition. The attendance for a match may depend on opponents, terms and homes, and we assume that it is estimated statistically by data of the matches of the past seasons. We define the estimated attendance of the match between team $i, j \in P;\ i \neq j$ at i's home in term t by c^t_{ij}.

2.2 Integer Program of SSPGF

The SSPGF can be formulated into an integer program with binary variables $y^t_{ij}(i, j \in P;\ i \neq j, t \in T)$:

$$y^t_{ij} = \begin{cases} 1: & \text{If team } i \text{ plays against team } j \text{ at its home in term } t. \\ 0: & \text{otherwise} \end{cases}$$

The binary property of y^t_{ij} directly implies that the requirement C.2 is satisfied. Consider the following integer program (IP):

$$
\begin{array}{llllll}
\text{(IP)}\max & \displaystyle\sum_{t\in T}\sum_{i\in P}\sum_{\substack{j\in P\\ j\neq i}} & c_{ij}^t y_{ij}^t & & & (1)\\
\text{s. t.} & \displaystyle\sum_{\substack{j\in P\\ j\neq i}}\{y_{ij}^t + y_{ji}^t\} & = & 1 & \forall i\in P,\ \forall t\in T & (2)\\
& \displaystyle\sum_{t\in T} y_{ij}^t & = & r & \forall i,j\in P; i\neq j & (3)\\
& \displaystyle\sum_{t=s}^{s+w-1}\{y_{ij}^t + y_{ji}^t\} & \leq & 1 & \forall i,j\in P; i\neq j,\ \forall s\in\{1,2,\cdots,m-(w-1)\} & (4)\\
& \displaystyle\sum_{t=s}^{s+h}\sum_{\substack{j\in P\\ j\neq i}} y_{ij}^t & \leq & h & \forall i\in P,\ \forall s\in\{1,2,\cdots,m-h\} & (5)\\
& y_{ij}^t & \in & \{0,1\} & \forall i\in P,\ \forall j\in P,\ j\neq i & (6)
\end{array}
$$

Constraints in (2) state that each team have exactly one match in a term. Constraints in (3) mean that the number of matches between i and j at i's home is exactly r, i.e., the total number of matches between i and j is exactly $2r$. Constraints in (4) ensure that at most one match between i and j can be held in w consecutive terms for any i, j, and (5) states that no team can play at its home in more than h consecutive terms. Constraints in (6) are the integrality constraints. Thus we can see that every optimal solution to (IP) above meets all of the requirements C.1–C.6 in the SSPGF. The problem (IP) is an integer program with $4mn^2 + 4n^2 + 3mn$ constraints and $4mn^2$ variables. The goal of maximizing the total estimated attendance will be achieved by the objective function of (IP).

3 Algorithm

In this section, we propose our algorithm based on the branch-and-bound method for solving the SSPGF.

As we have shown in the previous section, the SSPGF can be formulated as an integer program (IP). Thus we can adopt several decomposition techniques, e.g.,

LP-relaxation : the subproblems consist of Constraints (2)–(5),
1-factorization [7] : the subproblems consist of Constraints (2) and (6).

Among others, we can see that the 1-factorization technique is frequently employed in many algorithms proposed in the field of sports scheduling.

Our approach described below is completely different from the above decomposition techniques: We prepare a graph and show that a subproblem of the

SSPGF can be represented as a problem of finding a longest path in the graph. We describe the graph representation and our algorithm in the succeeding subsections.

3.1 Graph representation and subproblems

In the remaining of this paper, we assume that the competition employs the *double* round-robin tournament, i.e., $r = 1$. We discuss how we may extend the results to the general $2r$ round-robin tournament at the last of this subsection.

For each $i \in P$ and $j \in P$, define a new variable $t^i_j \in T$ which means that the term $t = t^i_j$ is assigned for the match between i and j held at i's home.

Here, the requirements C.2–C.4 is equivalent to the condition that for each $i \in P$ and $j \in P,\ i \neq j$, the match between i and j must be held exactly once at i's home and exactly once at j's home, respectively, under the assumption of the double round-robin tournament. Hence, if we can assign each term $t \in T$ to the variables $t^i_j (i \in P, j \in P, i \neq j)$ with no contradiction to the requirements C.1, C.5 and C.6, then we fix a schedule which is a feasible solution to the SSPGF.

This can be represented in a graph as follows. Let $\{i, j\}$ $(i < j)$ be a combination of teams. For each $\{i, j\}$, we first define a complete bipartite graph $\overline{G_{ij}} = (V_{ij} \cup V_{ji}, V_{ij} \times V_{ji})$ where $V_{ij} = \{v^t_{ij} | t = t^i_j \in T\},\ V_{ji} = \{v^t_{ji} | t = t^j_i \in T\},\ V_{ij} = V_{ji} = T$ (see Fig. 1(A)).

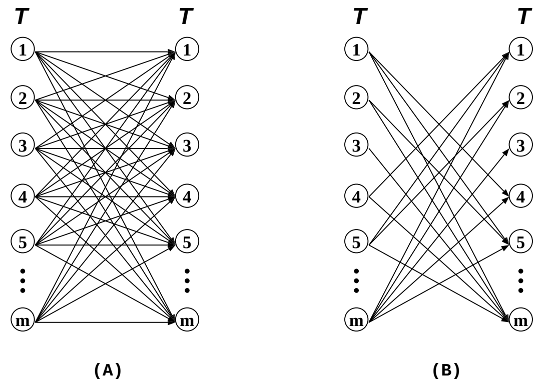

Fig. 1. (a) The graph $\overline{G_{ij}}$. (b) The graph $\overline{G_{ij}}$ after eliminating unnecessary arcs.

Choosing a vertex $v^t_{ij} \in V_{ij}$ $(v^{t'}_{ji} \in V_{ji})$ in the left (right) hand side implies that $t^i_j = t$ $(t^j_i = t')$. Thus choosing an arc between v^t_{ij} and $v^{t'}_{ji}$ in the graph $\overline{G_{ij}}$ means that we fix the matches between i and j completely in the schedule. Moreover, if we choose an arc in $\overline{G_{ij}}$ for every combination $\{i, j\}$, then we obtain a schedule which satisfies the requirements C.2-C.4 in the SSPGF.

On the other hand, by taking the requirement C.5 into consideration, we can eliminate some unnecessary arcs from $\overline{G_{ij}}$. The requirement C.5 implies that the interval $|t_j^i - t_i^j|$ of the matches must satisfy $|t_j^i - t_i^j| \geq w$ for every combination $\{i, j\}$. Thus the arc $(v_{ij}^t, v_{ji}^{t'}) \in V_{ij} \times V_{ji}$ should be eliminated if $|t - t'| < w$, for each graph $\overline{G_{ij}}$ (see Fig. 1(B)). The resultant graph $\overline{G_{ij}}$ can be given by $\overline{G_{ij}} = (V_{ij} \cup V_{ji}, \overline{A_{ij}})$ where $\overline{A_{ij}} = (V_{ij} \times V_{ji}) \setminus \{(v_{ij}^t, v_{ji}^{t'}) | \quad |t - t'| < w\}$.

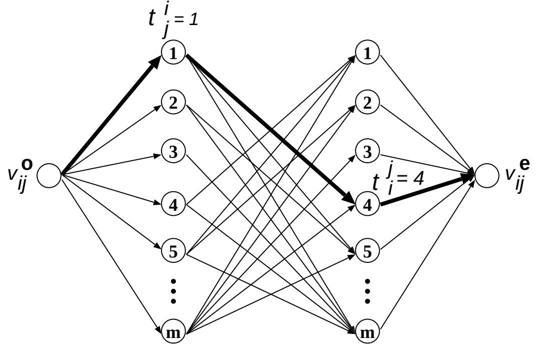

Fig. 2. An example flow on G_{ij}.

To represent this as a network flow problem, for each graph $\overline{G_{ij}}$, we add a pair of dummy vertices denoted by v_{ij}^{o} and v_{ij}^{e} and the arcs associated with the vertices $\overline{A_{ij}^{o}} = \{(v_{ij}^{o}, v_{ij}^t) | t \in T\}$ and $\overline{A_{ij}^{e}} = \{(v_{ji}^t, v_{ij}^{e}) | t \in T\}$. Define the new graph $G_{ij} = (W_{ij}, A_{ij})$ where $W_{ij} = \{V_{ij} \cup V_{ji} \cup \{v_{ij}^{o}, v_{ij}^{e}\}\}$ and $A_{ij} = \{\overline{A_{ij}} \cup \overline{A_{ij}^{o}} \cup \overline{A_{ij}^{e}}\}$. For each G_{ij}, a flow from v_{ij}^{o} to v_{ij}^{e} shows that the variables t_j^i and t_i^j satisfy $t_j^i = v_{ij}^t$ and $t_i^j = v_{ji}^{t'}$, respectively. Fig. 2 shows an example of the flow in G_{ij}. A flow denoted by the thick line in Fig. 2 means that the matches between i and j will be held at

- i's home in *the* 1*st* term, and
- j's home in *the* 4*th* term.

At this present, our graph $\bigcup_{\{i,j\}} G_{ij}$ is just a union of all subgraphs G_{ij} of each combination $\{i, j\}$ (see Fig. 3(A)). As we will see below, there still exist several arcs which can be eliminated.

Note that each flow originated from v_{ij}^{o} goes through only in the subgraph G_{ij}. In view of finding a feasible schedule, it is desirable that every flow knows which vertices are chosen in other flows. Combining this with the graph $\bigcup_{\{i,j\}} G_{ij}$ is difficult at the stage where each flow passes a vertex $v_{ij}^t \in V_{ij}$, since the flow should keep its identity of $\{i, j\}$ to proceed the next vertex $v_{ji}^t \in V_{ji}$. However, at the next stage, i.e., after each flow passes a vertex $v_{ji}^t \in V_{ji}$, the identity of

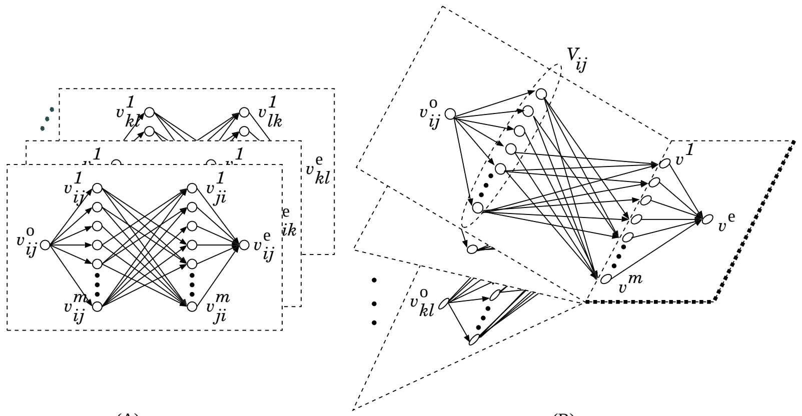

Fig. 3. (A) The graph $\bigcup_{\{i,j\}} G_{ij}$, (B) The graph G after eliminating unnecessary arcs.

$\{i, j\}$ is no longer needed since every flow is only expected to go to a dummy node v^{e}_{ij} whose role does not depend on the combination $\{i, j\}$. Thus, we can identify the vertices v^{e}_{ij} in each G_{ij} for all $\{i, j\}$ with a single vertex, say v^{e}, and the vertices $v^{t}_{ji} \in V_{ji}$ for each $\{i, j\}$ with a single vertex v^t for each $t \in T$.

For each $t \in T$, we also replace for each set of arcs $\{(v^t_{ji}, v^{\mathrm{e}}_{ij}) \in \overline{A^{\mathrm{e}}_{ij}} | \{i, j\}\}$ by an arc (v^t, v^{e}). The resulting graph, denoted by $G = (V, A)$, is given as shown in Fig. 3(B). Here V and A are given by

$$V = \bigcup_{\{i,j\}} (W_{ij} \setminus (V_{ji} \cup \{v^{\mathrm{e}}_{ij}\})) \cup \{v^t | t \in T\} \cup \{v^{\mathrm{e}}\},$$
$$A = \bigcup_{\{i,j\}} (A_{ij} \setminus \overline{A^{\mathrm{e}}_{ij}}) \cup \{(v^t, v^{\mathrm{e}}) | t \in T\}.$$

By this shrink of the graph, $O(n^2 m) = O(n^3)$ number of arcs can be eliminated, but the total number of the arcs is of the same order between the graph $\bigcup_{\{i,j\}} G_{ij}$ and the graph G. Fig. 4 shows an alternative representation which is isomorphic to the graph G.

For each arc $(p, q) \in A_{ij}$, we assign a triplet $(c_{pq}, u_{pq}, \ell_{pq})$ where c_{pq} is the cost, u_{pq} is the upper bound and ℓ_{pq} the lower bound of capacity as follows: We set c_{pq} to the estimated attendance for the match corresponding to the arc (p, q) if $q = v^t_{ij}$ or $p = v^t_{ji}$, otherwise we set $c_{pq} = 0$. We set $u_{pq} = 1$ and $\ell_{pq} = 0$ for every arc $(p, q) \in A_{ij}$. By setting $c_{pq} = 0, u_{pq} = n$ and $\ell_{pq} = 0$ for $(p, q) \in \{(v^t, v^{\mathrm{e}}) | t \in T\}$, a flow with value 1 originating from each v^{o}_{ij} on G satisfies C.2–C.5, where for each vertex $v \in V$ the supply s_v is given by $s_v = 1$ if $v = v^{\mathrm{o}}_{ij}$, $s_{v^{\mathrm{e}}} = -n(2n-1)$ and otherwise $s_v = 0$. A lower bound of the SSPGF can be obtained by solving the Longest Path Problem(LPP) where x_{pq} is a flow of arc (p, q):

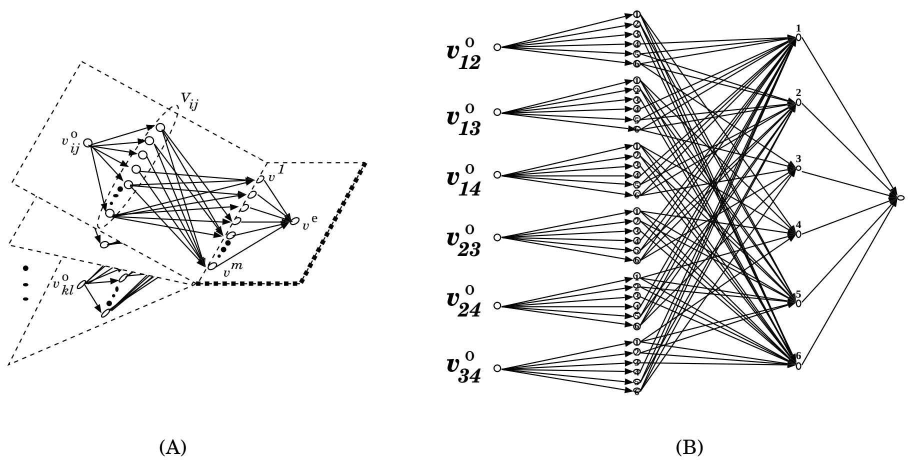

Fig. 4. An alternative representation which is isomorphic to G.

LPP

$$\max \quad \sum_{(p,q)\in A} c_{pq}x_{pq}$$

s. t.

$$\sum_{q:(p,q)\in A} x_{pq} - \sum_{q:(q,p)\in A} x_{qp} = \begin{cases} 1, & p = v^{\mathrm{o}}_{ij},\ \forall i,j \in P,\ i<j, \\ -n(2n-1), & p = v^{\mathrm{e}}, \\ 0, & p \in V \setminus \{v^{\mathrm{o}}_{ij}, v^{\mathrm{e}} | i,j \in P,\ i<j\}, \end{cases}$$

$$\ell_{pq} \le x_{pq} \le u_{pq},\ \forall (p,q) \in A.$$

Fig. 5 shows an example of flow, which is a solution to the subproblem. From any solution to the LPP, we can generate a schedule which satisfies C.2–C.5.

Now we mention how the number of matches in each term t expected by the subproblems is affected by the shrink of the graph described above. For each term t, let $U(t)$ be the total number of the upper bounds u_{pq} associated with the arc (p,q) which are connected to the vertices v^t_{ij} or v^t_{ji} in the graph $\bigcup_{\{i,j\}} G_{ij}$. Then $U(t)$ is computed by

$$\begin{aligned} U(t) &\equiv \sum_{\{i,j\}} \left\{ \sum_{\substack{(p,q)\in\overline{A^{\mathrm{o}}_{ij}},\\ q=v^t_{ij}}} u_{pq} + \sum_{\substack{(p,q)\in\overline{A^{\mathrm{e}}_{ij}},\\ p=v^t_{ji}}} u_{pq} \right\} \\ &= n(2n-1) + n(2n-1) \\ &= 2n(2n-1), \end{aligned} \tag{7}$$

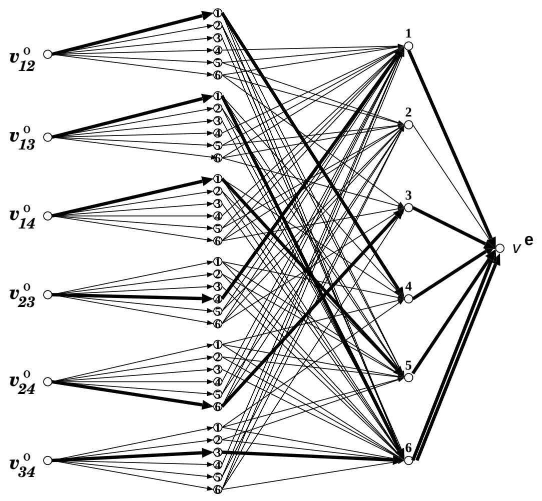

Fig. 5. An example of flow, which is a solution of the subproblem. $(2n = 4)$

and gives us an upper bound of the total number of matches exactly held in term t, denoted by $N(t)$ where

$$\begin{aligned} N(t) &\equiv \sum_{\{i,j\}} \left|\{(p,q) \in A_{ij} | x_{pq} = 1,\ p = v^t_{ij}\ or\ q = v^t_{ji}\}\right| \\ &\leq U(t). \end{aligned} \tag{8}$$

Note that the condition C.1 forces the value of $N(t)$ to be n. The amount of the gap between the values $N(t)$ and $U(t)$ can be regarded as an index to measure how the subproblem approximates the SSPGF. The corresponding number $U'(t)$ in the shrunken graph G is given by

$$\begin{aligned} U'(t) &= \sum_{\{i,j\}} \sum_{\substack{(p,q)\in \overline{A^o_{ij}}, \\ q=v^t_{ij}}} u_{pq} + u_{v^t v^e} \\ &= n(2n-1) + n \end{aligned} \tag{9}$$

and we can see that the amount of the gap is reduced to be nearly half. This shows a merit of the shrink of the graph.

In the above discussions, we assume that the competition employs the double round-robin tournament. However, by following augmentation we see that the results above can be extended to the general $2r$ round-robin tournament: We make r copies of V_{ij} and V_{ji} with indices for every combination $\{i, j\}$, let these sets be $V^1_{ij}, V^2_{ij}, \cdots V^r_{ij}, V^1_{ji}, V^2_{ji}, \cdots, V^r_{ji}$, where $V^1_{ij} = V^2_{ij} = \cdots V^r_{ij} = V^1_{ji} = V^2_{ji} = \cdots = V^r_{ji} = T$. Then construct a $2r$-partite graph $\overline{G^{2r}_{ij}} = (\overline{V^{2r}}, \overline{A^{2r}})$ instead of $\overline{G_{ij}}$, where $\overline{V^{2r}} = \{V_{ij}^1 \cup V_{ij}^2 \cup \cdots V_{ij}^r \cup V_{ji}^1 \cup V_{ji}^2 \cup \cdots \cup V_{ji}^r\}$, and $\overline{A^{2r}} = \{\{V_{ij}^1 \times V_{ji}^1\} \cup \{V_{ji}^1 \times V_{ij}^2\} \cup \cdots \cup \{V_{ji}^{r-1} \times V_{ij}^r\} \cup \{V_{ij}^r \times V_{ji}^r\}\}$. Unfortunately a flow on the graph $\overline{G_{ij}^{2r}}$ may not satisfy C.5. We have not developed other augmentation techniques which can maintain C.2–C.5 yet.

3.2 Branch-and-bound algorithm

As we have described above, every flows on the graph G gives a schedule satisfies the conditions C.2–C.5. We define our branching rules to find an optimal solution based on the best bound search so that the other condition C.1 and C.6 are also satisfied. The branching rules are stated in Fig. 6. We repeat branching procedures according to the rules while the conditions C.1 is violated. In what follows, we describe the rules in detail.

First of all, we will find a team which violates for C.1 by Rule.1. If C.1 is violated in term t, i.e., $\exists t \in T,\ \sum_{(p,q)\in A^t} x_{pq} \neq n$, where A^t is the set of arcs which corresponds with the matches held in term t and given by

$$A^t = \{(v_{ij}^{\mathrm{o}}, v_{ij}^t)|\{i,j\}\} \cup (v^t, v^{\mathrm{e}})\},$$

then we choose a team i such that i matches more than one team in t, i.e., we choose $i \in P;\ \sum_{(p,q)\in A_i^t} x_{pq} > 1$, where A_i^t is the set of arcs which corresponds with matches of i in t,

$$\begin{aligned} A_i^t = \{(v_{ij}^{\mathrm{o}}, v_{ij}^t)|j \in P,\ i<j\} &\cup \{(v_{ji}^{\mathrm{o}}, v_{ji}^t)|j \in P,\ j<i\} \\ \cup\{(v_{ji}^t, v^{t'})|\ j \in P,\ j<i,\ t' \in T,\ |t-t'| \geq w\} \\ \cup\{(v_{ji}^t, v^{t'})|\ j \in P,\ i<j,\ t' \in T,\ |t-t'| \geq w\}. \end{aligned}$$

Next, as stated in Rule.2, we will find an arc (p^*, q^*) corresponding to a match which i plays in term t with the maximum attendance. Let the opponent be team k. The homes are determined by where the arc (p^*, q^*) lies. Let the home be i's if $(p^*, q^*) \in H_i^t \subseteq A_i^t$, where H_i^t is the set of arcs corresponding to the matches which are held at i's home in t, and given by

$$\begin{aligned} H_i^t = &\{(v_{ij}^{\mathrm{o}}, v_{ij}^t)|j \in P,\ i<j\} \\ &\cup \{(v_{ji}^t, v^{t'})|\ j \in P,\ j<i,\ t' \in T,\ |t-t'| \geq w\}, \end{aligned}$$

otherwise let it be j's home.

Then as Rule.3, we will construct two new subproblems: one is "team i plays against team k at $i's$ (or j's) home in term t", i.e., $x'_{p^*q^*} = 1$ and the other is "team i never plays against team k at i's (or j's) home in term t", i.e., $x'_{p^*q^*} = 0$. In the example in Fig. 5, team 1 plays against more than one team, say team $2, 3$ and 4, in the first term. If the match between 1 and 2 has the maximum attendance, we choose the arc $(v_{12}^{\mathrm{o}}, v_{12}^1)$ as (p^*, q^*), then construct following subproblems. In one of the subproblems, "team 1 plays against team 2 at i's home in the first term" must be fixed, and in the other, "team 1 never plays against team 2 at i's home in the first term", inversely.

We also add two extra rules, Rule.4 and Rule.5, to tighten our lower bounds. We call these two procedures, "pegging tests", because each variable fixed by Rule.4 and Rule.5 in a branching procedure corresponds with the match which can not be held in term t, we can fix them to 0 after the branching.

In the Rule.4, if the match between i and k is fixed in term t, i and k can not have other match in t by C.1, that is, for all $j \in P;\ j \neq i,\ j \neq k$ matches between i and j and matches between k and j in t can be prohibited. Every variables x_{pq} which correspond with all arcs $(p, q) \in \{A_i^t \cup A_k^t\} \setminus \{(p^*, q^*)\}$ must be fixed by 0. The resulting flow satisfies the condition C.1 for i and k.

Rule.5 is concerned with the feasibility checking of C.6. When a match between i and k at i's home is fixed in term t, we check the feasibility of C.6. If the result of fixing the match will coincide with one of these patterns shown in Table 1, no more matches can be held at i's home in the term has "★" because it makes the subproblem infeasible inevitably.

In order to prevent the violation of C.6, we scan every series of $h + 1$ terms $Z_z \in \mathcal{Z}$ for each $z \in S = \{s \in T | t - h \leq s \leq t\}$ where

$$\begin{aligned} \mathcal{Z} &= \{Z_z | z \in S\} \text{ and} \\ Z_z &= \{t - h - 1 + z, t - h - 1 + z + 1, \cdots, t - h - 1 + z + h\} \\ &= \{t - h - 1 + z, t - h + z, \cdots, t - 1 + z\}. \end{aligned}$$

If we recognize a pattern by scanning, i.e., $\exists z \in S; Z_z \in \mathcal{Z}, \sum_{s \in Z_z} \sum_{(p,q) \in H_i^s} x_{pq} = h$, let $\overline{H_i^z}$ be the set of arcs corresponding to home-games for i which can not be held in Z_z, and it is given by

$$\overline{H_i^z} = \{(p, q) | s \in Z_z,\ (p, q) \in H_i^s,\ x_{pq} \neq 1\}.$$

Then for each arc $(p, q) \in \overline{H_i^z}$ we fix the variables to 0, i.e., $x'_{pq} = 0$. After prohibiting, team i does not break C.6 in and around t.

To give an example, we suppose that $h = 3$, and team i has two home-games in term $t - 2$ and $t + 1$, and another home-game for i has just assigned in term $t = s_3$ by this branching procedure, i.e., $\exists z = 2$, and case 2 in the Table 1 is occurring in $Z_2 = \{t - 2, t - 1, t, t + 1\}$. If one more home-game for i will be assigned in term $t - 1$, then C.6 is violated in the series of 4 terms Z_2. We set $\overline{H_i^2} = \{(v_{ij}^{\circ}, v_{ij}^{t-1})\}$ then fix $x'_{v_{ij}^{\circ} v_{ij}^{t-1}} = 0$.

Table 1. Pattern of series of matches at one's home

term	$\cdots$	s_1	s_2	s_3	$\cdots$	s_{h-1}	s_h	s_{h+1}	$\cdots$
case 1		★	H	H	$\cdots$	H	H	H	
case 2		H	★	H	$\cdots$	H	H	H	
$\vdots$									
case h		H	H	H	$\cdots$	H	★	H	
case $h+2$		H	H	H	$\cdots$	H	H	★	

Rule.1 If $\exists t \in T, \sum_{(p,q)\in A^t} x_{pq} \neq n$,

where $A^t = \{(v^{\text{o}}_{ij}, v^t_{ij}) | \{i,j\}\} \cup (v^t, v^{\text{e}})\}$

then choose $i \in P; \sum_{(p,q)\in A^t_i} x_{pq} > 1$,

$$\begin{aligned} \text{where } A^t_i = &\{(v^{\text{o}}_{ij}, v^t_{ij}) | j \in P,\ i<j\} \cup \{(v^{\text{o}}_{ji}, v^t_{ji}) | j \in P,\ j<i\} \\ &\cup \{(v^t_{ji}, v^{t'}) |\ j \in P,\ j<i,\ t' \in T,\ |t-t'| \geq w\} \\ &\cup \{(v^t_{ji}, v^{t'}) |\ j \in P,\ i<j,\ t' \in T,\ |t-t'| \geq w\}. \end{aligned}$$

Rule.2 Set $(p^*, q^*) \in \text{argmax}\{c_{pq} | (p,q) \in A^t_i,\ x_{pq} = 1\}$.

Rule.3 Construct two new subproblems:
- $x'_{p^*q^*} = 1$ and
- $x'_{p^*q^*} = 0$.

Rule.4 Prohibit some matches which can not be held in term t.
$\forall (p,q) \in \{A^t_i \cup A^t_k\} \setminus \{(p^*,q^*)\}, \quad x'_{pq} = 0$

Rule.5 Prevent the violations for C.6.
If $\exists z \in S = \{s \in T | t-h \leq s \leq t\}; Z_z \in \mathcal{Z}, \sum_{s \in Z_z} \sum_{(p,q)\in H^s_i} x_{pq} = h$,

$$\begin{aligned} \text{where} \quad \mathcal{Z} &= \{Z_z | z \in S\}, \\ Z_z &= \{t-h-1+z, t-h-1+z+1, \cdots, t-h-1+z+h\} \\ &= \{t-h-1+z, t-h+z, \cdots, t-1+z\}, \text{and} \\ H^t_i &= \{(v^{\text{o}}_{ij}, v^t_{ij}) | j \in P,\ i<j\} \cup \{(v^t_{ji}, v^{t'}) |\ j \in P,\ j<i,\ t' \in T,\ |t-t'| \geq w\}, \end{aligned}$$

let $\overline{H^z_i}$ be the set of arcs, $\overline{H^z_i} = \{(p,q) | s \in Z_z,\ (p,q) \in H^s_i,\ x_{pq} \neq 1\}$,
then $\forall (p,q) \in \overline{H^z_i} \quad x'_{pq} = 0$.

Fig. 6. Branching Rules

4 Computational Results

For experiments, we use Dell Precision Workstation 530 with Intel XEON 1.7GHz CPU and 512MB memory. Our program is written in C++ with LEDA version 4.3 and compiled by gcc version 2.95. The data sets used in this experiment are generate at random from 1000 to 10000, and the number of teams in the league is 6.

The algorithm solves all problems correctly. The computational results are summarized in Table 2. The first column is the index of the data sets. The second column shows the total CPU time in hours and minutes. #ITR denotes the total number of iterations. $\#\mathrm{ITR}_{\mathrm{ini}}$ and $\#\mathrm{ITR}_{\mathrm{opt}}$ indicates the number of iterations until we find the initial incumbent solution and the optimal solution, respectively. The three ratios in the 4th, 5th and 7th column are given by following equations:

$$\mathrm{r\_ITR}_{\mathrm{ini}} \equiv \frac{\#\mathrm{ITR}_{\mathrm{ini}}}{\#\mathrm{ITR}}$$

$$\mathrm{r\_VAL}_{\mathrm{ini}} \equiv \frac{\text{the initial incumbent val.}}{\text{the optimal val.}}$$

$$\mathrm{r\_ITR}_{\mathrm{opt}} \equiv \frac{\#\mathrm{ITR}_{\mathrm{opt}}}{\#\mathrm{ITR}}$$

#INC shown in the 9th column is the number of feasible solutions we found. In Table 2, "–" stands for the number less than 0.1%.

Table 2. The Summary of Computational Results.

data set	CPU (h:m)	$\#\mathrm{ITR}_{\mathrm{ini}}$	$\mathrm{r\_ITR}_{\mathrm{ini}}$ (%)	$\mathrm{r\_VAL}_{\mathrm{ini}}$ (%)	$\#\mathrm{ITR}_{\mathrm{opt}}$	$\mathrm{r\_ITR}_{\mathrm{opt}}$ (%)	#ITR	#INC
1	3:37	2,597	0.1	89.5	54,441	3.0	1,808,501	5
2	18:58	640,398	4.3	89.3	5,654,361	37.7	14,984,509	13
3	10:12	17,879	0.3	89.1	2,417,819	45.4	5,326,227	10
4	5:39	90,241	2.0	77.7	2,536,635	55.2	4,599,231	9
5	22:27	1,063,090	5.6	85.9	4,338,884	22.7	19,125,389	14
6	95:30	149,860	0.2	79.9	59,300,579	39.2	81,964,927	19
7	12:23	437	–	83.7	4,275,369	42.2	10,130,817	8
8	2:52	8,933	0.4	88.0	1,026,300	45.3	2,264,687	12
9	16:58	679,831	5.0	83.7	2,149,349	15.8	13,576,650	10
10	32:44	9,075	–	78.0	10,396,527	38.5	27,025,407	14

We can see that the values $\mathrm{r\_ITR}_{\mathrm{ini}}$ are less than 6% and most of $\mathrm{r_VAL}_{\mathrm{ini}}$s are greater than 80%. These imply that

- the initial incumbent solution can be found in a very short time, and
- it is close on its optima,

which support the validity of the search direction. Furthermore, most of $\mathrm{r\_ITR}_{\mathrm{opt}}$ are less than 50%,

– an optimal solution also can be found at a relatively early stage.

The optimal solution can be found in reasonable CPU time, however, the checking of the optimality takes much time. In order to solve in more practical time, it might be expected to develop other efficient network representation, some extra procedures to obtain more tight lower bounds, and effective bounding rules.

5 Conclusion

In this study, we have introduced Sports Scheduling Problems in General Form. By generalization of SSP, the model is not case oriented. We also have proposed a branch-and-bound algorithm for solving SSPGF, where the subproblems, being the Longest Path Problems, are solved efficiently by use of network structure. In our approach, by the construction of the network subproblems, we can generate a sequence of solutions which always satisfy the condition of the number of matches in each term and integrality constraints. Applying our algorithm, we can solve SSPGF correctly. From computational results, we can see that the search direction is valid, however, the checking of the optimality takes much time.

As future works, we have to develop other network representation, some extra procedures to obtain more tight lower bounds, and effective bounding rules.

6 Acknowledgments

The authors would like to thank the anonymous referees and Prof. Yoshitsugu Yamamoto of University of Tsukuba in Japan for their careful reading and many useful comments on an earlier version of the paper. The authors also appreciate Prof. Tomomi Matsui of University of Tokyo in Japan for his valuable advices and helpful discussions.

References

1. R.K. Ahuja and T.L. Magnanti and J.B. Orlin, Network Flows, Prentice Hall, 1993.
2. R. Andreu and A. Corominas, "SUCCCES92: A DSS for scheduling the Olympic games," *Interfaces,* Vol.19, pp.1–12, 1989.
3. J. Armstrong and R.J. Willis, "Scheduling the cricket world cup – a case study," *Journal of the Operational Research Society,* Vol.44, pp.1067–1072, 1993.
4. J.C. Bean and J.R.Birge, "Reducing travelling costs and player fatigue in the National Basketball Association," *Interfaces,* Vol.10, pp.98–102, 1980.
5. W.O. Cain Jr., "The computer-assisted heuristic approach used to schedule the major league baseball clubs," Optimal Strategies in Sports, pp.32–41, North-Holland, Amsterdam, 1977.

6. R.T. Campbell and D.S.Chen, “A minimum distance basketball scheduling problem,” Management Science in Sports, pp.15–25, North-Holland, Amsterdam, 1976.
7. D. De Werra, “Scheduling in Sports,” O.R.Working Paper 45, Ecole Polytechnique Federale de Lousanne, 1979.
8. D. De Werra, “Geography, games and graphs,” *Discrete Applied Mathematics,* Vol.2, pp.327–337, 1980.
9. J.A. Ferland and C. Fleurent, “Computer aided scheduling for a sport league,” *INFOR,* Vol.29, pp.14–25, 1991.
10. K. Mehlhorn, and S. Näher, A LEDA Platform of Combinatirial and Geometric Computing, Cambridge University Press, New York, 1999.
11. G. Nemhauser and M. Trick, “Scheduling a major college basketball conference,” *Operations Research,* Vol.46, pp.1–8, 1997.
12. R.A. Russel and J.M.Y. Leung, “Divising a cost effective schedule for a baseball league,” *Operations Research,* Vol.42, pp.614–625, 1994.
13. J.A.M. Schreuder, “Constructing timetables for sports competitions,” *Mathematical Programming Study,* No.13, pp.58–67, 1980.
14. J.A.M. Schreuder, “Combinatorial aspects of construction of competition Dutch Professional Football Leagues,” *Discrete Applied Mathematics,* No.35, pp. 301–312, 1992.
15. T. Swan, UNIX Programmer’s Library GNU C++ Programming,2001.
16. R.J. Willis and B.J. Terrill, “Scheduling the Australian state cricket season using simulated annealing,” *Journal of the Operational Research Society,* No.45, pp. 276–280, 1994.
17. M. Wright, “Timetabling county cricket fixtures using a form of tabu search,” *Journal of the Operational Research Society,* No.45, pp. 758–770, 1994.