Briskorn, Dirk; Drexl, Andreas

Scheduling sports leagues using branch- and price

Manuskripte aus den Instituten für Betriebswirtschaftslehre der Universität Kiel, No. 609

Provided in Cooperation with:
Christian-Albrechts-University of Kiel, Institute of Business Administration

Manuskripte
aus den
Instituten für Betriebswirtschaftslehre
der Universität Kiel

No. 609

Scheduling Sports Leagues Using Branch-And-Price

Dirk Briskorn, Andreas Drexl

October 2006

Dirk Briskorn, Andreas Drexl
Christian–Albrechts–Universität zu Kiel,
Institut für Betriebswirtschaftslehre,
Olshausenstr. 40, 24098 Kiel, Germany,
http://www.bwl.uni-kiel.de/bwlinstitute/Prod
briskorn@bwl.uni-kiel.de, andreas.drexl@bwl.uni-kiel.de

Abstract

A single round robin tournament can be described as a league of a set T of n teams (n even) to be scheduled such that each team plays exactly once against each other team and such that each team plays exactly once per period resulting in a set P of $n-1$ periods. Matches are carried out at one of both opponents' stadiums. A team playing twice at home or twice away in two consecutive periods is said to have a break in the latter of both periods. There is a vast field of requests arising in real world problems. For example, the number of breaks is to be minimized due to fairness reasons. It is well known that at least $n-2$ breaks must occur. We focus on schedules having the minimum number of breaks. Costs corresponding to each possible match are given and the objective is to minimize the sum of arranged matches' cost. Then, sports league scheduling can be seen as a hard combinatorial optimization problem. We develop a branch & price approach in order to find optimal solutions.

Keywords: Sports league scheduling, round robin tournaments, break, branch& price

1 Introduction

Round robin tournaments (RRT) covers a huge variety of different types of sports league schedules arising in practice. The focus in this paper is on single RRTs where scheduling is temporally constrained which means that a minimum number of periods are given the matches have to be scheduled in. We consider a set T of n teams. If n is odd we easily can add a dummy team and, hence, we can assume, that n is even without loss of generality.
In a single RRT each team plays exactly once against each other team, either at home or away. Furthermore, a team $i \in T$ has to play exactly once in each period and, hence, we have a set P of $n-1$ periods altogether.
Note that special types of RRTs where a pair of teams meets more than once can be covered by single RRTs as outlined in [5]. Such RRTs are called mirrored if they are divided into rounds where each round forms a single RRT and, furthermore, team i plays at home against team j in the p^{th} period of round r with $r \geq 2$ if and only if team j plays at home against team i in the p^{th} period of round $r-1$. Hence, there are several real world examples of tournaments we can consider by all what follows:

- Single RRTs often are carried out in major sports events such as FIFA soccer world cup before the play off rounds start.
- Mirrored double RRTs (which can be covered by single RRT) are predominant in professional soccer leagues.

Illustrative examples for single RRTs and mirrored double RRTs are given in tables 1 and 2, respectively. Here i-j denotes that team i plays at home against team j.
Since each match has to be carried out at one of the both opponents' venues breaks come into play. We say team i has a break in period p if and only if i plays twice at home or away, respectively, in periods $p-1$ and p. For the sake of fairness among teams the most common goal concerning breaks is to minimize the number of their occurences. It is well known from [8] that the number of breaks can be no less than $n-2$ in a single RRT. The single RRT provided in table 1 has 4 breaks for 6 teams and, therefore, has a minimum number of breaks.

period	1	2	3	4	5
match 1	3-4	4-5	2-4	4-6	6-5
match 2	5-2	1-3	5-1	3-5	4-1
match 3	6-1	2-6	6-3	1-2	2-3

Table 1: Single RRT for $n = 6$

period	1	2	3	4	5	6	7	8	9	10
match 1	1-2	5-6	3-4	4-5	5-1	2-1	6-5	4-3	5-4	1-5
match 2	5-3	1-4	2-5	3-1	4-2	3-5	4-1	5-2	1-3	2-4
match 3	4-6	2-3	1-6	2-6	3-6	6-4	3-2	6-1	6-2	6-3

Table 2: Mirrored double RRT for $n = 6$

Models for sports league scheduling have been the topic of extensive research. A whole stream of papers is based on the analogy between sports league scheduling and edge coloring of complete graphs. Examples are [8], [9], [10], [11], [12], [13], and [14]. [6] and [14] analyze the relationship between sports league scheduling and multi-mode resource constrained project scheduling. [5] line out the similarity of structures of single RRTs and planar three index assignments. [2], [3], and [25], [26] examine particular formulations.
The remainder of this paper is organized as follows. In section 2 we motivate a sports league scheduling problem and develop two model formulations. Based on one of these models we propose a branch-and-price (B&P) approach in order to obtain sports leagues schedules having a minimum number of breaks in section 3. In section 4 we line out computational results and, finally, section 5 contains conclusions.

2 Problem Setting And Models

2.1 Problem Definition

In most approaches in order to schedule sports leagues the goal is to find an arbitrary schedule obeying given structural requirements such as providing a minimum number of breaks. Consequently, there is no difference in solution quality depending on the period a pair of teams meets as long as the solution is feasible according to the structural requirements. In fact, there are several aspects arising in real world sports leagues as well as in scheduling theory leading to a pair of teams to meet more favorable in period $p \in P$ than in period $p' \in P \setminus \{p\}$:

- Teams usually have preferences for playing at home in certain periods.
- Pairs of teams might have preferences for playing against each other in a specific period.
- Since a major objective of the organizers of a tournament is to maximize attendance it is straightforward to consider the economic value of the estimated attendance of the single RRT.
- Often, a stadium is owned by some public agency and teams do have to pay a fee for each match taking place in that particular stadium. This fee might depend on the specific period.

- In terms of more complex models the single RRT might be used as a subproblem, e.g., within a Lagrangean relaxation or a column generation framework. Then, dual information have to be considered.

In order to cover those aspects mentioned above we associate cost $c_{i,j,p}$ with each possible match of team i playing at home against team j in period p. We define the cost of a single RRT as the sum of arranged matches' costs. Then, we can state problem "MinCostMinBreakScheduleProblem" in Definition 1.

Definition 1 *Given a set of teams T, a set of periods P with $|P| = |T| - 1$, and cost $c_{i,j,p}, i \in T, j \in T \setminus \{i\}, p \in P$ the "MinCostMinBreakScheduleProblem" is to find the minimum cost tournament among all single RRT having a minimum number of breaks.*

There are generation schemes for single RRTs having the minimum number of breaks, see [2] for example. However, to the best of our knowledge each generation scheme will fail if cost cost minimization is a goal to be considered.

2.2 Basic Integer Programming Model

We employ two types of binary variables $x_{i,j,p} \forall i, j \in T, i \neq j, p \in P$ and $br_{i,p} \forall i \in T, p \in P$. Variables $x_{i,j,p}$ represent matches: $x_{i,j,p}$ is equal to 1 if and only if team i plays at home against team j in period p. Variables $br_{i,p}$ represent breaks: $br_{i,p}$ is equal to 1 if and only if team i has a break in period p. Hence, we have an overall number of $n\left((n-1)^2 + n - 2\right)$ binary variables.
We construct model "IP" in order to represent problem "MinCostMinBreakScheduleProblem" by means of Integer Programming.

IP

$$\min \sum_{i \in T} \sum_{j \in T \setminus \{i\}} \sum_{p \in P} c_{i,j,p} x_{i,j,p} \tag{1}$$

$$\text{s.t.} \sum_{p \in P} (x_{i,j,p} + x_{j,i,p}) = 1 \qquad \forall\, i, j \in T, i < j \tag{2}$$

$$\sum_{j \in T \setminus \{i\}} (x_{i,j,p} + x_{j,i,p}) = 1 \qquad \forall\, i \in T, p \in P \tag{3}$$

$$\sum_{j \in T \setminus \{i\}} (x_{i,j,p-1} + x_{i,j,p}) - br_{i,p} \leq 1 \qquad \forall\, i \in T, p \in P^{\geq 2} \tag{4}$$

$$\sum_{j \in T \setminus \{i\}} (x_{j,i,p-1} + x_{j,i,p}) - br_{i,p} \leq 1 \qquad \forall\, i \in T, p \in P^{\geq 2} \tag{5}$$

$$\sum_{i \in T} \sum_{p \in P^{\geq 2}} br_{i,p} \leq n - 2 \tag{6}$$

$$x_{i,j,p} \in \{0,1\} \qquad \forall\, i, j \in T, i \neq j, p \in P \tag{7}$$

$$br_{i,p} \in \{0,1\} \qquad \forall\, i \in T, p \in P^{\geq 2} \tag{8}$$

Objective function (1) represents the goal of cost minimization. Restrictions (2) and (3) form a single RRT by letting each pair of teams meet exactly once and forcing each team to play

exactly once per period. Restrictions (4) and (5) set binary variable $br_{i,p}$ to 1 if team i plays twice at home or twice away in periods $p-1$ and p. Term (6) assures that no more than $n-2$ breaks are arranged. Since this is the minimum possible number $br_{i,p}$ is set to 1 if and only if team i has a break in period p. The overall number of restrictions given in (2) to (6) is $n\left(\frac{3(n-1)}{2}+2(n-2)\right)+1$.

2.3 Column Generation Model

When tackling the problem defined in section 2.1 represented by the model proposed in section 2.2 we suffer from several difficulties.

- The problem size is determined by $O(n^3)$ variables and $O(n^2)$ constraints.
- The solution of the LP relaxation in general is highly fractional and the value of the LP relaxation is poor.
- Cost oriented node order strategies are difficult to implement since fixing variables has intractable consequences for other variables due to compact structure of single RRTs with the minimum number of breaks.

In order to improve the model's provided lower bound and create more meaningful variables we propose a column generation (CG) model in the following. We define a matchday (MD) as a set of matches where each team contributes exactly once and a scheduled MD as a MD being assigned to a specific period. Furthermore, we consider the set of scheduled MDs PM and associate a binary variable y_m with each scheduled MD $m \in PM$. Therefore, the number of variables is given as $|PM| = (n-1)\frac{n!}{\frac{n}{2}!} = (n-1)\prod_{i=\frac{n}{2}+1}^{n} i$. The period $m \in PM$ is assigned to is denoted by $p(m)$. We say a pair (i,j) of teams is contained in $m \in PM$ (denoted by $(i,j) \in m$) if a match between teams i and j is part of m. Additionally, an ordered pair $(\overrightarrow{i,j})$ is contained in $m \in PM$ (denoted by $(\overrightarrow{i,j}) \in m$) if a match of team i at home against team j is part of m. Cost c_m of scheduled MD $m \in PM$ is defined by $c_m = \sum_{(\overrightarrow{i,j})\in m} c_{i,j,p(m)}$. The CG Masterproblem is given by (9) to (16).

Objective function (9) represents the goal of finding the minimum cost tournament. Constraint (10) forces each pair of teams to meet at most once. As far as single RRTs are concerned constraint (10) is equivalent to (2). Equality (11) assures exactly one MD being arranged in each period of the tournament and it corresponds to (3) since each team $i \in T$ participates in each scheduled MD $m \in PM$ exactly once. Constraints (12), (13), and (14) are equivalent to (4), (5), and (6).

Note that constraints (10) and (11) are of the set packing and of the set partitioning type, respectively. This observation is picked up in section 3.1.

Since the number of variables is exponential in the number of teams it is straightforward to reduce the number of variables and initialize (9) to (16) with a subset $PM' \subset PM$ as a restricted master problem of a CG process. Then, we obtain the subproblem to find scheduled MDs which can improve the current restricted master's optimal solution. According to the fundamental work of [16] this means finding the scheduled MD having minimum reduced cost (or, at least, negative reduced cost). Let $\alpha_{i,j}$, β_p, $\gamma_{i,p}$, $\delta_{i,p}$, and ϵ be dual variables according to (10), (11), (12), (13), and (14), respectively. Therefore, we first state reduced cost $\bar{c}_m$ of MD $m \in PM$.

CG Masterproblem

$$\min \sum_{m \in PM} c_m y_m \tag{9}$$

$$\text{s.t.} \sum_{m \in PM,(i,j) \in m} y_m \leq 1 \quad \forall\, i,j \in Ts, i < j \tag{10}$$

$$\sum_{m \in PM,p(m)=p} y_m = 1 \quad \forall\, p \in P \tag{11}$$

$$\sum_{j \in T, j \neq i} \sum_{m \in PM,(\overrightarrow{i,j}) \in m, p-1 \leq p(m) \leq p} y_m - br_{i,p} \leq 1 \quad \forall\, i \in T, p \in P^{\geq 2} \tag{12}$$

$$\sum_{j \in T, j \neq i} \sum_{m \in PM,(\overrightarrow{j,i}) \in m, p-1 \leq p(m) \leq p} y_m - br_{i,p} \leq 1 \quad \forall\, i \in T, p \in P^{\geq 2} \tag{13}$$

$$\sum_{i \in T} \sum_{p \in P} br_{i,p} \leq n-2 \tag{14}$$

$$y_m \in \{0,1\} \forall\, m \in PM \tag{15}$$

$$br_{i,p} \in \{0,1\} \forall\, i \in T, p \in P^{\geq 2} \tag{16}$$

$$\bar{c}_m = c_m - \sum_{(i,j) \in m, i<j} \alpha_{i,j} - \beta_{p(m)} - \sum_{(\overrightarrow{i,j}) \in m} \left(\underbrace{\gamma_{i,p(m)} + \delta_{j,p(m)}}_{\text{if } p(m) \geq 2} \underbrace{+\gamma_{i,p(m)+1} + \delta_{j,p(m)+1}}_{\text{if } p(m) \leq |P|-1} \right) \tag{17}$$

Note that we consider dual variables according to (12) and (13) with $p(m)$ only if $p(m) > 1$ (since there is no restriction (12) or (13) for $p = 1$). Analogously, dual variables according to (12) and (13) with $p(m)+1$ are taken into account only if $p(m) < |P|$ (since there is no restriction (12) or (13) for $p = |P|+1$).
We can break equation (17) down to the contribution $\tilde{c}_{i,j,p}$ of each match of team i at home against team j in period p to the reduced cost of scheduled MD $m \in PM$ as outlined in (18).

$$\tilde{c}_{i,j,p} = c_{i,j,p} - \alpha_{i,j} - \frac{2\beta_p}{|T|} \underbrace{-\gamma_{i,p} - \delta_{j,p}}_{\text{if } p \geq 2} \underbrace{-\gamma_{i,p+1} - \delta_{j,p+1}}_{\text{if } p \leq |P|-1} \tag{18}$$

Consequently, the problem to find the minimum reduced cost scheduled MD can be represented by means of IP modelling as model MDP.
Objective function (19) is to minimize arranged matches' cost according to cost $\bar{c}_{i,j}$. Constraint (20) assures that each team participates in exactly once. Restrictions (21) and (22) set cost $\bar{c}_{i,j}$ to reduced cost according to the chosen period p with $y_p = 1$. M is set to $\max_{i,j \in T, i \neq j, p \in P} |c_{i,j,p}|$. Equation (23) assures that the MD is assigned to exactly one period. MDP is a quadratic mixed IP. In order to reduce computational effort we decompose MDP into $|P|$ subproblems $\text{MD}_p, p \in P$, representing the objective to find the minimum reduced

CG Subproblem MDP

$$\min \sum_{i \in T} \sum_{j \in T \setminus \{i\}} \overline{c}_{i,j} \overline{x}_{i,j} \tag{19}$$

$$\text{s.t.} \sum_{j \in T : j \neq i} (\overline{x}_{i,j} + \overline{x}_{j,i}) = 1 \qquad \forall i \in T \tag{20}$$

$$(1 - \overline{y}_p)M + \widetilde{c}_{i,j,p} \geq \overline{c}_{i,j} \qquad \forall i, j \in T : j \neq i, p \in P \tag{21}$$

$$(\overline{y}_p - 1)M + \widetilde{c}_{i,j,p} \leq \overline{c}_{i,j} \qquad \forall i, j \in T : j \neq i, p \in P \tag{22}$$

$$\sum_{p \in P} \overline{y}_p = 1 \tag{23}$$

$$\overline{x}_{i,j} \in \{0,1\} \qquad \forall i, j \in T : j \neq i \tag{24}$$

$$\overline{y}_p \in \{0,1\} \qquad \forall p \in P \tag{25}$$

$$\overline{c}_{i,j} \in \mathbb{R} \qquad \forall i, j \in T : j \neq i \tag{26}$$

cost MD assigned to p. MD_p is equivalent to MD if $y_p = 1$ holds. Therefore, MD can be represented as MDP'.

CG Subproblem MD_p

$$\min \sum_{i \in T} \sum_{j \in T \setminus \{i\}} \widetilde{c}_{i,j,p} \overline{x}_{i,j} \tag{27}$$

$$\text{s.t.} \sum_{j \in T : j \neq i} (\overline{x}_{i,j} + \overline{x}_{j,i}) = 1 \qquad \forall i \in T \tag{28}$$

$$\overline{x}_{i,j} \in \{0,1\} \qquad \forall i, j : j \neq i \tag{29}$$

CG Subproblem MDP'

$$\min_{p \in P} \text{MD}_p \tag{30}$$

Clearly, MD_p is a minimum cost perfect matching problem which is known to be solvable in polynomial time, see [15] and [7].

3 Branch-And-Price Approach

Since solving the LP relaxation of (9) to (16) to optimality does not necessarily lead to integer solutions we propose a branching scheme in the following. Furthermore we line out details of the CG process and provide a lower bound in order to stop the column generation process before optimality for the current node's problem is reached.

3.1 Branching Strategy

We propose a branching scheme being divided into two layers. In the first stage we fix branch variables according to characteristics of single RRTs having a minimum number of breaks. These characteristics are well studied in [11] and [22], for example. [4] develop a static branching scheme based on these characteristics. For nodes of the search tree having depth d lower than n branching is carried out by creating a single child node for each possible break for team $d+1$. Obviously, at depth d we have to consider those breaks already set for teams 1 to d on the path from the root node to the current node. The following rules for deciding whether to create a child node corresponding to a specific break (defined by period and venue) are taken from [4] and repeated here for the sake of completeness:

I A home-break (away-break) at period 1 is possible if no home-break (away-break) has been set for period 1 on the path from the root node to the current node.

II If exactly one break is already set in period p we can set a home-break (away-break) at MD p if the existing one is an away-break (home-break).

III We can set a break in period $p \geq 2$ where no break has been set on the path from the root node to the current node if the number of $\frac{n}{2}$ periods having breaks does not get exceeded and p does not complete a sequence of three periods having breaks.

Since in a single RRT having a minimum number of breaks each team has exactly one break (two teams must have their breaks in period 1 which makes these two breaks artificial) the consequence of setting a break determines each period's venue of the team we set the break for. Accordingly, branching by setting a specific break is implemented by fixing each match variable corresponding to the team at hand but not obeying the venue of this team for a period to 0. Hence, the subproblem's structure is not affected. This branching idea has proven to be quite effective in cutting subtrees being infeasible according to the minimum number of breaks structure.

Note, that after finishing the branching scheme's first stage we obtain a partition of teams into teams playing at home and teams playing away for each period. Therefore, CG subproblem MD_p is reduced to the minimum cost bipartite matching problem and can be solved according to the basic work in [18].

Since fixing breaks for each team does not guarantee integer solutions to (9) to (16) we propose a second stage for branching if the optimal solution to the current node is fractional. Here, the branching idea is motivated by the structure of the master problem which is a type of set partitioning problem as outlined in section 2.

[1] give an exhaustive overview and emphasize a method developed by [24]. This strategy has been successfully applied to B&P frameworks, e.g. in [21] and [27]. The rule prescribes to find two columns c and c' having fractional variable values in the current optimal solution. Second, two rows r and r' must be determined such that c covers r and r' and c' covers exactly one of them. As shown in [1] c, c', r, r' always exist in a standard set partitioning problem if the current solution is not integral. Two subproblems are created by either forcing r and r' to be covered by the same column or forcing r and r' to be covered by different columns. Projecting this idea to our problem we obtain the branching strategy outlined in the following.

If r is of type (10) and r' is of type (11) pairing $(i,j)\widehat{=}r$ is contained in MDs assigned to $p\widehat{=}r'$ as well as in MDs assigned to other periods having variable values greater than zero. Forcing both rows to be covered by the same column and different columns means fixing (i,j) to p and forbidding (i,j) to be carried out in p, respectively.

In the first child node pairing (i, j) is fixed to be carried out in period p. Therefore, solutions to the subproblem corresponding to p must incorporate pairing (i, j). This can be implemented by fixing $\overline{x}_{i,j} + \overline{x}_{j,i} = 1$. Only one of these two variables has not been fixed to 0 during the first stage of the branching scheme, hence, the other one is fixed to 1 now. Next, we solve the remaining minimum cost bipartite matching problem corresponding to period p based on the set $T \setminus \{i, j\}$. Those subproblems corresponding to periods in $P \setminus \{p\}$ must not incorporate pairing (i, j). Therefore, we set $\overline{x}_{i,j} = \overline{x}_{j,i} = 0$. In the second child node pairing (i, j) can not be carried out in period p. This is implemented by setting $\overline{x}_{i,j} = \overline{x}_{j,i} = 0$ in the subproblem corresponding to period p. In period $p' \neq p$ (i, j) might be carried out, hence, the subproblem corresponding to p' is not affected by the branching step.
We have to decide how to select a branching candidate if more than one is given. We propose to select the candidate leading to the lowest possible cost of the pairing (i, j) to be fixed in period p.
Clearly, these two branching layers lead to integer solutions since fixing teams' venues as well as fixing the period a match is carried out means fixing the whole schedule.

3.2 Node Order Strategy

In preliminary test runs we investigated the performance of classical depth first search (DFS) and classical breadth first search (BFS) choosing the node having the lowest lower bound next. Since there is no efficient heuristic to transform node's optimal solutions into feasible solutions to the original problem DFS provides feasible solutions much faster than BFS does. Since we aim at optimality BFS providing exact solutions much faster than DFS is to be favored to DFS. Unfortunately, we suffer from lack of memory while solving all instances but the very smallest ones.
Therefore, we use beam search as an intermediate approach. Beam search originates from the artificial intelligence community, see [20] and [23]. Beam search means to explore a fixed number of nodes at each level of the search tree. This number is called beam width w. We select those w nodes having the lowest lower bounds for further exploration. All nodes but those w get discarded which makes the approach a heuristic one.
The larger w is the more nodes get explored and, therefore, solution quality and runtime as well memory requirements rise. If we let $w = 1$ just a single path from the root node is explored. This either leads to finding a single feasible solution or finishing the procedure without any feasible solution if the path leads to a node whose problem is infeasible. Between those extremes, beam search enables us to customize it depending on problem size and constraints taken into account in order to find a promising tradeoff between solution quality and memory requirements.
Obviously, the heuristic nature of beam search is a major drawback since we aim at the minimum cost solution. By slight variation we can turn the heuristic beam search into an exact method. Instead of discarding those nodes not selected for the beam we preserve them for a backtracking mechanism. If the approach backtracks on a specific level, again, the w most promising remaining nodes are selected and a new beam down from this level is established. Therefore, we have some kind of DFS order for beams. We experience counterrotating effects when changing w here too: The larger w is chosen the lower is the number of beams needed to reach the optimal solution as well as the overall running time of our algorithm. On the other hand the smaller w is chosen the lower are running time requirements to fully explore a beam and overall memory requirements.

3.3 Column Generation Process

Column management covers three activities: generating initial columns in order to obtain the first restricted master problem as beginning for the iterativ column generation approach, generating pleasant columns during the column generation approach, and deleting columns not pleasant anymore in order to reduce the restricted master problem's size.

3.3.1 Generating Initial Columns

Initial columns for the restricted master problem have to provide at least one feasible solution to the problem represented by the current node. The constraints in the current node are composed of constraints of the original problem and of constraints resulting from branching steps executed on the path from the root node to the current node of the branching tree. If no feasible solution is provided no dual variables corresponding to a feasible solution are given. These are substantially needed for the column generation process.
Each constraint in the restricted master problem originates from the original problem. We construct feasible solutions according to structure of single RRTs by using a generation scheme originating from [17], namely the polygon technique. Constraints according to the minimum number of breaks structure are taken into account by the generation scheme given in [25].
The constraints contained in the pricing problem CG subproblem MD_p are fulfilled by each single RRT and, hence, by both generation schemes. Constraints derived from branching decisions on the other hand can not be considered by both generation schemes. Scheduled MDs not obeying these branching constraints are called infeasible MDs in the following. We incorporate infeasible MDs in the restricted master problem since we can not guarantee initial feasible MDs providing a solution to the master problem by using the chosen generation scheme. We have to make sure that the CG process will ban infeasible MDs from the solutions while optimizing. Therefore, we set cost c_m to $M_{vio} = \frac{n(n-1)}{2} \max_{i,j \in T, i \neq j, p \in P}(c_{i,j,p})$ if scheduled MD m is infeasible.
Instead of generating a single scheduled MD per period by using the generation scheme we propose to construct $|P|$ scheduled MDs per period. The generation scheme yields $|P|$ different MDs forming a single RRT. We assign each of them to each period and obtain $|P|^2$ different scheduled MDs. The underlying idea is to increase probability of finding a good solution as first solution to the restricted master problem by covering more possible solutions.
Alternatively, we suggest to employ the optimal solution's MDs of the father node as initial columns. When doing this we have to check whether a MD m is feasible according to constraints of the parent node but gets infeasible by the constraint derived from the branching step. If so we set $c_m = M_{vio}$ in the current node. This methodology is justified by the fact that the father node and the current node differ in only one branching step. Therefore, the father node's optimal solution might just slightly differ from the optimal solution or at least a good solution of the current node. Obviously, we can not use this idea for the root node.
We tested all variants outlined above. The following strategy proofed to be the most effective one (measured in running times) and is the one we restrict all what follows to:

- For the root node we construct $|P|^2$ initial columns by assigning each MD obtained from the generation scheme of [25] to each period.
- For each other node we pass the columns contained in the father node's optimal solution as well as those from the generation scheme as initial columns to the child nodes.

3.3.2 Generating Columns

Since we have $|P|$ decomposed problems MD_p resulting from the pricing problem MD there are several decisions to be made about the solution process.
In order to find the overall minimum reduced cost scheduled MD we have to solve all single decomposed problems iteratively. Hence, we determine $|P|$ scheduled MDs all of which might have negative reduced cost. Consequently, we propose to insert all scheduled MDs having negative reduced cost into the restricted master.
Alternatively, we can solve decomposed problems until we have found the first scheduled MD having negative reduced cost. Therefore, we solve the decomposed problems in decreasing order of dual variable β_p. β_p is a constant element of the reduced cost of MDs assigned to period p and, therefore, the problem MD_p having highest β_p is most likely to produce a scheduled MD having negative reduced cost.
Testing both variants made clear that solving each single decomposed problem is more effective.

3.3.3 Discarding Columns

While optimizing the master problem lots of scheduled MDs are generated and are inserted into the restricted master problem. Nearly all of them turn useless later on when they are not part of the current optimal solution's base anymore. In order to reduce solution times for the restricted master problem it is substantial to prevent the restricted master from growing to large. There are two popular ideas to choose columns to be deleted from the restricted master problem:

- A column is deleted whenever it has not been part of the base for a given number of iterations.
- A column is deleted whenever its reduced cost are larger than a given threshold.

According to our test the best choice for our problem is to delete a scheduled MD when it has not been part of the base for 5 iterations.

3.3.4 Lower Bound

One major drawback of column generation is that optimality of solutions is often proven much later than the solution is found (also called tailing-off effect). Hence, many iterations are spent useless (in terms of finding the master's optimal solutions). Consequently, we propose a lower bound according to the current restricted master's optimal solution and the optimal solutions to the pricing problems MD_p. Whenever we compute a lower bound for the optimal solution to the current node which is higher than the best feasible solution's value known so far we cancel the column generation process. The subtree corresponding to the current node is pruned, then.
With respect to the B&P scheme we consider the set $PM_f \subseteq PM$ of scheduled MDs being feasible according to the current node. Note that each branching step is represented by fixing variables of matching subproblems and, therefore, can be fully represented by reduction of PM. Furthermore, we have to take into account the set PM_{inf} of scheduled MDs introduced into the restricted master as initial columns and being infeasible according to the fixed variables of the current node. Let $PM' = PM_f \cup PM_{inf}$.
[19] proposes a lower bound for the current master's optimal solution when the master problem has a set partitioning structure. We pick this idea up and adapt it to (9) to (16).

Theorem 1 *Given an optimal solution to the restricted master problem having value z_{cur} a lower bound of the optimal solution's value to the master problem is given by*

$$z_{cur} + \sum_{p \in P} \min\left(\min_{m \in PM_f, p(m)=p} \bar{c}_m, 0\right) + \sum_{i \in T} \sum_{p \in P^{\geq 2}} (\gamma_{i,p} + \delta_{i,p} - \epsilon)\, br_{i,p}.$$

Proof. Subtracting the right hand side of (11) and the left hand side of (11), respectively, multiplied by corresponding dual variables β_p from objective function (9) leads to equation (31).

$$\sum_{m \in PM'} c_m y_m - \sum_{p \in P} \beta_p = \sum_{m \in PM'} c_m y_m - \sum_{p \in P} \beta_p \sum_{m \in PM', p(m)=p} y_m \tag{31}$$

Doing the same for (10), (12), (13), and (14) yields (32). Note that all dual variables used in this second step are less or equal to 0.

$$\begin{aligned}
&\sum_{m \in PM'} c_m y_m - \sum_{p \in P} \beta_p - \sum_{i \in T} \sum_{j \in T, j>i} \alpha_{i,j} - \sum_{i \in T} \sum_{p \in P^{\geq 2}} \gamma_{i,p} - \\
&\sum_{i \in T} \sum_{p \in P^{\geq 2}} \delta_{i,p} - (n-2)\,\epsilon \geq \sum_{m \in PM'} c_m y_m - \sum_{p \in P} \beta_p \sum_{m \in PM', p(m)=p} y_m - \\
&\sum_{i \in T} \sum_{j \in T, j>i} \alpha_{i,j} \sum_{m \in PM', (i,j) \in m} y_m - \epsilon \left(\sum_{i \in T} \sum_{p \in P} br_{i,p} \right) - \\
&\sum_{i \in T} \sum_{p \in P^{\geq 2}} \gamma_{i,p} \left(\sum_{j \in T, j \neq i} \sum_{m \in PM', (\vec{i,j}) \in m, p-1 \leq p(m) \leq p} y_m - br_{i,p} \right) - \\
&\sum_{i \in T} \sum_{p \in P^{\geq 2}} \delta_{i,p} \left(\sum_{j \in T, j \neq i} \sum_{m \in PM', (\vec{j,i}) \in m, p-1 \leq p(m) \leq p} y_m - br_{i,p} \right)
\end{aligned} \tag{32}$$

The terms being subtracted from $\sum_{m \in PM'} c_m y_m$ on the left hand side of (32) sum up to the dual solution's value and, therefore, to the current optimal solution's value of the restricted master due to duality theory. Furthermore, we extract scheduled MDs' reduced cost on the right hand side and obtain (33).

$$\begin{aligned}
\sum_{m \in PM'} c_m y_m - z_{cur} \geq \sum_{m \in PM'} y_m \Bigg(c_m - \sum_{(i,j) \in m, j>i} \alpha_{i,j} - \beta_{p(m)} - \gamma_{i,p(m)} - \\
\delta_{i,p(m)} - \gamma_{i,p(m)+1} - \delta_{i,p(m)+1} \Bigg) + \sum_{i \in T} \sum_{p \in P^{\geq 2}} (\gamma_{i,p} + \delta_{i,p} - \epsilon)\, br_{i,p}
\end{aligned} \tag{33}$$

With $\sum_{m \in PM', p(m)=p} y_m = 1$ for each $p \in P$ and $\min_{m' \in PM', p(m')=p} \bar{c}_{m'} \leq \bar{c}_m$ for all $m \in PM', p(m) = p$ we transform inequality (33) into (34).

$$\begin{aligned}
\sum_{m \in PM'} c_m y_m \geq z_{cur} + \sum_{p \in P} \min_{m \in PM', p(m)=p} \bar{c}_m + \\
\sum_{i \in T} \sum_{p \in P^{\geq 2}} (\gamma_{i,p} + \delta_{i,p} - \epsilon)\, br_{i,p}
\end{aligned} \tag{34}$$

Note that $\bar{c}_m \geq 0$ holds for each $m \in PM_{inf}$ and $\bar{c}_m \in \mathbb{R}$ holds for each $m \in PM_f$. From this we can conclude equation (35).

$$\min_{m \in PM', p(m)=p} \bar{c}_m = \min\left(\min_{m \in PM_f, p(m)=p} \bar{c}_m, 0\right) \tag{35}$$

Obviously, (35) holds if $\min_{m \in PM_f, p(m)=p} \bar{c}_m \leq 0$. However, the opposite might hold due to the fact that solely infeasible scheduled MDs might form period p of the current restricted master's solution. If so, there are scheduled MDs in PM_{inf} having reduced cost equal to 0 since they contribute to the current optimal solution.
Employing (34) and (35) we obtain (36).

$$\sum_{m \in PM'} c_m y_m \geq z_{cur} + \sum_{p \in P} \min\left(\min_{m \in PM_f, p(m)=p} \bar{c}_m, 0\right) + \sum_{i \in T} \sum_{p \in P^{\geq 2}} (\gamma_{i,p} + \delta_{i,p} - \epsilon)\, br_{i,p} \tag{36}$$

Term (36) directly implies theorem 1. □
A fairly convenient property of this lower bound is that nearly no additional computational effort is to be made: Dual variables are known from the current optimal solution to the restricted master problem and the minimum reduced cost MD for each period is computed in order to find pleasant columns, anyway.
Furthermore, the lower bound formulation implies a customization to the current node of the search tree. Branching constraints are incorporated in the matching subproblem restricting the solution space of feasible scheduled MDs and, therefore, affect the lower bound according to the path from the root node to the current node.

4 Computational Results

In order to evaluate our B&P approach we employed Cplex 9.0 as benchmark. We executed the test runs on a 3.8 GHz Pentium IV PC with 3 GB RAM running Windows Server 2003. We construct the instances with cost $c_{i,j,p}$ randomly chosen from $[-10, 10]$. When measuring solutions' quality we transform all cost to be non-positive ($c_{i,j,p} = c_{i,j,p} - \max_{i,j,p} c_{i,j,p}$) and, therefore, can directly relate the corresponding solutions' values. 20 instances of each size are tested, except for problems having 10 teams where running times forced us to limit our study to 3 instances.
We employ Cplex using DFS as well as BFS as node order strategy. Our algorithm is tested both with exact and heuristic beam search using beam width $w = 15$. In order to get a fair comparison according to the times to find feasible solutions we used DFS in our approach as well. The focus is on running times in order to reach and proof optimality which are given in table 3 where "B&P beam e." and "B&P beam h." abbreviate exact and heuristic B&P approach, respectively.
The first thing to observe is that reaching optimality seems possible only for the smallest of instances. There is no problem when solving instances with up to 8 teams using Cplex. Employing DFS and BFS problems having 8 teams are solved in about 180 seconds and 50 seconds, respectively, on average. Our approach is clearly slower for these instances. This is true for each variant, thus for DFS and exact beam search as well as heuristic beam search.

size	Cplex DFS	Cplex BFS	B&P DFS	B&P beam e.	B&P beam h.
4	0.01	0.01	0.08	0.07	0.07
6	0.39	0.38	17.27	23.52	15.20
8	178.42	50.02	2971.59	2302.86	252.96
10	—	—	—	486200.00	1338.54
12	—	—	—	—	5129.67

Table 3: Running times optimal solutions

size	Cplex DFS	Cplex BFS	B&P DFS	B&P beam e./h.
6	0.00	0.00	0.54	2.20
8	6.98	2.5	2.91	95.21
10	—	—	—	730.00
12	—	—	—	4665.35

Table 4: Running times first solution

For larger problem instances Cplex using BFS runs out of memory after about 12 hours of running time giving an idea of the enormous searching tree's size. Obviously, Cplex does not suffer from lack of memory if DFS is employed as node order strategy. We tested 3 instances and stopped each optimization process after 6 days of running time. Not even one feasible solution was found for two instances. Cplex found a feasible solution for the third problem instance after about an hour. However, the optimality gap could not be lowered to less than 20% within 6 days of running time.

When applying our approach using exact beam search we solve all three instances to optimality within 6 days of running time. Although this seems to be an unacceptable amount of time we reach optimality where Cplex mostly fails to get feasible solutions at all.

In general running times of DFS are higher than those of exact beam search being higher than those of heuristic beam search. Note that there is an exception from this rule for instances having 6 teams. This might result from the cost oriented branching scheme being quite efficient for instances of this size. Because of its heuristic nature solution quality of the second beam search approach is of much interest. For problem instances having less than 8 teams we reach optimality for each single instance. For problems having 8 teams we find optimal solutions for 50% of the instances. Objective values of solutions obtained by heuristic beam search reach 99% of the optimal solution's value on average. No solution to instances having 10 teams is optimal. However, their objective values amount to 96% of optimal solutions' values on average. We expect this gap to grow if the heuristic beam search is applied to larger instances. Since solely finding good feasible solutions turned out to be pretty time consuming we additionally line out the amount of time in order to find feasible solutions in table 4.

We refuse to inspect instances having 4 teams since times to find feasible solutions are equal to or almost zero. Again, our approach can not compete with Cplex solving instances having 6 teams. However, our DFS approach outperforms Cplex using DFS for 8 teams and is close to Cplex using BFS. We do not distinguish between exact and heuristic beam search anymore since feasible solutions are found within the first beam and, therefore, results are identical. Obviously, beam search finds a feasible solution much slower.

Apart from the time in order to find feasible solutions their quality is a major concern. For those instances whose optimal solution is known we give the first solution's quality of the first

size	Cplex DFS	Cplex BFS	B&P DFS	B&P beam e./h.
6	92.8%	92.8%	86.1%	97.1%
8	77.4%	88.6%	82.7%	96.2%
10	—	—	—	94.5%

Table 5: Quality first solution

solution found by the approaches under consideration in table 5.
Clearly, Cplex using BFS provides better first solutions than Cplex using DFS does. Our DFS approach provides better first solutions than Cplex using DFS does in a significant lower amount of running time. Naturally, each approach's first solution's quality decreases if the number of teams is increased. Our beam search approaches provide the best first solution's quality to cost of highest amount of running time (see table 4) for instances having 6 and 8 teams. However, our beam search first solution's quality seems to decrease slower than those of the other approaches do. Furthermore, first solutions' quality for 10 teams is above all qualities provided by other approaches. Therefore, we find very good solutions in a reasonable amount of running time using our heuristic approach where Cplex rarely finds feasible solutions at all.

5 Conclusions

We propose a B&P approach in order to schedule sports leagues. The approach is quite reasonable since our restricted master problem is a variation of the well known set partitioning problem and our pricing problem is solvable in polynomial time.
Results are ambivalent. On the one hand, we can not compete with Cplex for problem instances having less than 10 teams. On the other hand, we can solve problem instances having 10 teams to optimality where Cplex fails. Furthermore, a heuristic variant of our approach provides good solutions (within 4% of optimality for less than 12 teams) in a small amount of time.
Attractive fields for further research are requirements resulting from real world sports leagues. Some can easily be incorporated into the pricing problem such as stadium availability.

References

[1] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch–And–Price: Column Generation For Solving Huge Integer Programs. *Operations Research*, 46:316–329, 1996.

[2] T. Bartsch. *Sportligaplanung – Ein Decision Support System zur Spielplanerstellung (in German)*. Deutscher Universitätsverlag, Wiesbaden, 2001.

[3] T. Bartsch, A. Drexl, and S. Kröger. Scheduling the Professional Soccer Leagues of Austria and Germany. *Computers & Operations Research*, 33:1907–1937, 2006.

[4] D. Briskorn and A. Drexl. Branching Based on Home–Away–Pattern Sets. In *GOR Proceedings 2006*. Springer, Berlin, Germany. Forthcoming.

[5] D. Briskorn, A. Drexl, and F. C. R. Spieksma. Round Robin Tournaments and Three Index Assignment. *Working Paper*, 2006.

[6] P. Brucker and S. Knust. *Complex Scheduling.* Springer, Berlin, 2006.

[7] W. Cook and A. Rohe. Computing Minimum–Weight Perfect Matchings. *INFORMS Journal on Computing*, 11:138–148, 1999.

[8] D. de Werra. Geography, Games and Graphs. *Discrete Applied Mathematics*, 2:327–337, 1980.

[9] D. de Werra. Scheduling in Sports. In P. Hansen, editor, *Studies on Graphs and Discrete Programming*, pages 381–395. North-Holland, Amsterdam, The Netherlands, 1981.

[10] D. de Werra. Minimizing Irregularities in Sports Schedules using Graph Theory. *Discrete Applied Mathematics*, 4:217–226, 1982.

[11] D. de Werra. On the Multiplication of Divisions: the Use of Graphs for Sports Scheduling. *Networks*, 15:125–136, 1985.

[12] D. de Werra. Some Models of Graphs for Scheduling Sports Competitions. *Discrete Applied Mathematics*, 21:47–65, 1985.

[13] D. de Werra, T. Ekim, and C. Raess. Construction of Sports Schedules with Multiple Venues. *Discrete Applied Mathematics*, 154:47–58, 1985.

[14] A. Drexl and S. Knust. Sports League Scheduling: Graph- and Resource–Based Models. *Omega*, to appear, 2006.

[15] J. Edmonds. Maximum Matching and a Polyhedron with (0,1) Vertices. *Journal of Research of the National Bureau of Standards Section B*, 69(B):125–130, 1965.

[16] P. C. Gilmore and R. E. Gomory. A Linear Programming Approach to the Cutting–Stock Problem. *Operations Research*, 9:849–859, 1961.

[17] T. P. Kirkman. On a Problem in Combinations. *Cambridge and Dublin Mathematics Journal*, 2:191–204, 1847.

[18] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[19] L. S. Lasdon, editor. *Optimization Theory in Large Systems.* North-Holland, Amsterdam, The Netherlands, 1970.

[20] B. T. Lowerre. *The HARPY Speeche Recognition System.* PhD thesis, Carnegie-Mellon University, USA, 1976.

[21] A. Mehrotra and M. A. Trick. A Column Generation Approach for Graph Coloring. *INFORMS Journal on Computing*, 8:344–354, 1996.

[22] R. Miyashiro, H. Iwasaki, and T. Matsui. Characterizing Feasible Pattern Sets with a Minimum Number of Breaks. In E. Burke and P. de Causmaecker, editors, *Proceedings of the 4th international conference on the practice and theory of automated timetabling*, Lecture Notes in Computer Science 2740, pages 78–99. Springer, Berlin, Germany, 2003.

[23] S. Rubin. *The ARGOS Image Understanding System.* PhD thesis, Carnegie-Mellon University, USA, 1978.

[24] D. M. Ryan and B. A. Foster. An Integer Programming Approach to Scheduling. In A. Wren, editor, *Computer Scheduling of Public Transport. Urban Passenger Vehicle and Crew Scheduling*, pages 269–280. North-Holland, Amsterdam, The Netherlands, 1981.

[25] J. A. M. Schreuder. Constructing Timetables for Sport Competitions. *Mathematical Programming Study*, 13:58–67, 1980.

[26] J. A. M. Schreuder. Combinatorial Aspects of Construction of Competition Dutch Professional Football Leagues. *Discrete Applied Mathematics*, 35:301–312, 1992.

[27] P. H. Vance, C. Barnhart, E. L. Johnson, and G. L. Nemhauser. Solving Binary Cutting Stock Problems by Column Generation and Branch-and-Bound. *Computational Optimization and Applications*, 3:111–130, 1994.