Online Appendix for SABIO: An Implementation of MIP and CP for Interactive Soccer Queries

Robinson Duque[1], Juan Francisco Díaz[1], and Alejandro Arbelaez[2]

[1] Universidad del Valle, Cali, Colombia
{robinson.duque, juanfco.diaz}@correounivalle.edu.co
[2] Insight Centre for Data Analytics, University College Cork, Ireland
{alejandro.arbelaez}@insight-centre.org

General Description

Soccer fans usually have questions related to their favourite teams and most of the time they are subject to media speculations that are sometimes proved wrong by either fans who can find contradicting scenarios using ink and paper or by people with certain skills to create models and programs. We attempt to present a general MIP model that can be used to simulate different scenarios and problems such as the elimination problem [1,2,3,4], the score vector problem [5], promotion and relegation [3,6], etc. where fans can combine four different kind of queries to analyze soccer:

- *Game Results.* These kind of queries let users impose constraints about particular matches in any fixture. (e.g., Barcelona ends in a tie with R.Madrid).
- *Position in Ranking.* These kind of queries let users impose constraints about the positions of the teams at the end of a tournament (e.g., R.Madrid will be in a better position than 3).
- *Relative Position.* These kind of queries let users impose constraints about the positions of two teams (e.g., R.Madrid will be in a worse position than Barcelona).
- *Final Points.* These kind of queries let users impose constraints about the expected final points of the teams at the end of a tournament (e.g., Barcelona score at the end of the tournament is 75).

Appendix A MIP Model for Soccer Queries

In this appendix, we describe a MIP formulation for soccer competitions. In the following we describe the variables and notations used in our model, we differentiate between five categories of variables: basic formulation, game result queries, position in ranking queries, relative position queries and final point queries. While a few of these variables had already been used in the literature [7,8,9], we recall that many variables had to be added to formulate certain features of our model, e.g., flexibility of single and double round-robin competition.

Basic Formulation Variables: these variables capture basic information to formulate a model for soccer competitions.

- n: number of teams in the competition;
- T: set of team indexes in the competition;
- i, j: team indexes, such that $(i, j \in T)$;
- p_i: initial points of team i. If i has not played any games, then $p_i = 0$;
- g_{ij}: number of remaining games between a pair of teams i and j;
- w_{ij}: number of games that team i wins over team j ;
- t_{ij}: number of games that team i ties with team j ;
- l_{ij}: number of games that team i loses over team j;
- tp_i: total points of team i at end of the competition;
- pos_i: position of team i at the end of the competition;
- $worstPos_i$: upper bound for pos_i;
- $bestPos_i$: lower bound for pos_i;

Game Result Queries: we use this set of variables to represent user defined assumptions of the remaining games, e.g., Barcelona ends in a tie with R. Madrid.

- Q: set of game result queries for pairs of teams (i, j). Q is defined as a set of triples $nq_a = (wc_{ij}, tc_{ij}, lc_{ij})$ and $0 \leq a \leq |Q|$;
- wc_{ij}: minimum number of games that team i wins over team j;
- tc_{ij}: minimum number of games that team i ties with team j;
- lc_{ij}: minimum number of games that team i loses over team j;

Position in Ranking Queries: we use this set of variables to represent queries of teams at the end of the competition.

- P: set of possible position in ranking queries, defined as a set of triples $np_b = (i, opr_i, ptn_i)$ and $0 \leq b \leq |P|$;
- opr_i: logical operator ($opr_i \in \{<, \leq, >, \geq, =\}$) to constrain team i;
- ptn_i: denoting the expected position for team i; $1 \leq ptn_i \leq n$;
- L: denoting the set of team indexes included in all the triples $np_b \in P$ such that $np_b = (i, opr_i, ptn_i)$ and $i \in L$ and $L \subseteq T$;
- geq_{ij}: boolean variable indicating if team j has greater or equal total points as i: if $tp_j \geq tp_i$ then $geq_{ij} = 1$; otherwise $geq_{ij} = 0 \quad (\forall i \in L, \forall j \in T)$;
- eq_{ij}: boolean variable indicating if two different teams i and j tie in points at the end of the competition: if $tp_j = tp_i$ and $i \neq j$ then $eq_{ij} = 1$; otherwise $eq_{ij} = 0 \quad (\forall i \in L, \forall j \in T)$.

Relative Position Queries: we use these variables to represent queries about relative positions between two teams, e.g., Barcelona will be in a better position than R. Madrid.

- R: set of possible relative position queries defined as a set of triples $nr_c = (i, op_{ij}, j)$ and $0 \leq c \leq |R|$;
- op_{ij}: denoting a logical operator ($op_{ij} \in \{<, \leq, >, \geq, =\}$) to constrain a pair of teams i and j.

Final Point Queries: (also known as score queries) we use these variables for queries about the final points of the teams, e.g., Barcelona scores at the end of the competition 75 points.

– S: set of possible final point queries defined as a set of tuples $ns_d = (i, s_i)$ and $0 \leq d \leq |S|$;
– s_i: denoting the wanted final points of team i.

A.1 MIP Model Formulation

Basic Soccer Model: Constraints (1), (2), and (3) describe a basic soccer model with a valid (win, tie, lose) assignment for every game between a pair of teams (i, j) with g_{ij} games left to play. We recall that g_{ij} must be equal to g_{ji}, the games a team i wins over team j (i.e., w_{ij}) must be also equal to the games team j loses over team i (i.e., l_{ji}). Respectively, $t_{ij} = t_{ji}$, and $l_{ij} = w_{ji}$. In this scenario, a team can get (3 points – win, 1 point – tie, 0 points – loss) in every game, then the total number of points of team i can be calculated by adding its initial points p_i and the points obtained against every other team. Depending on the competition and the current state of the tournament g_{ij} is set to 0 (no games left between the teams), 1 or 2. Unlike previous work in [9] which is limited to single round-robin competitions, this representation is flexible to represent both single and double round-robin competitions.

$$w_{ij} + t_{ij} + l_{ij} = g_{ij} \quad \forall_{i,j \in T} \wedge g_{ij} \geq 0 \tag{1}$$

$$w_{ij} = l_{ji} \wedge t_{ij} = t_{ji} \wedge l_{ij} = w_{ji} \quad \forall_{i,j \in T} \tag{2}$$

$$tp_i = p_i + \sum_{j=1, j \neq i}^{n} 3 \cdot w_{ij} + t_{ij} \quad \forall_{i,j \in T} \tag{3}$$

Game Results Queries: our model will allow users to include assumptions of the outcome of remaining games in the competition, e.g., team i wins over team j in at least one of the two remaining games between the two teams. Taking this into account we extend our basic model with constraint (4).

$$\begin{aligned} (wc_{ij} + tc_{ij} + lc_{ij} \leq g_{ij}) &\ \forall nq_a \in Q \wedge nq_a = (wc_{ij}, tc_{ij}, lc_{ij}) \\ (w_{ij} \geq wc_{ij}) \wedge (t_{ij} \geq tc_{ij}) \wedge (l_{ij} \geq lc_{ij}) &\ \forall nq_a \in Q \wedge nq_a = (wc_{ij}, tc_{ij}, lc_{ij}) \\ & wc_{ij}, tc_{ij}, lc_{ij} \geq 0 \end{aligned} \tag{4}$$

Position in Ranking Queries: A position in ranking query involves a set of constrained teams $L \subseteq T$ and indicates whether a given team can be above, below, or at a given position ptn_i, constraint (5) depicts the five possibilities:

$$\forall np_b \in P \wedge np_b = (i, opr_i, ptn_i) \begin{cases} pos_i = ptn_i, & \text{if } opr_i \text{ is } = \\ pos_i \leq ptn_i - 1, & \text{if } opr_i \text{ is } < \\ pos_i \leq ptn_i, & \text{if } opr_i \text{ is } \leq \\ pos_i \geq ptn_i + 1, & \text{if } opr_i \text{ is } > \\ pos_i \geq ptn_i, & \text{if } opr_i \text{ is } \geq \end{cases} \tag{5}$$

Constraint (6) indicates the number of teams j that finish the competition with better or equal points as team i. This number (i.e., $worstPos_i$) is the upper bound for pos_i as expressed in constraint (8).

$$
\begin{aligned}
geq_{ij} &= \begin{cases} 1, & \text{if } tp_j \geq tp_i \\ 0, & otherwise \end{cases} \quad \forall j \in T \land \forall i \in L \\
worstPos_i &= \sum_{j=1}^{n} geq_{ij} \quad \forall j \in T \land \forall i \in L
\end{aligned} \tag{6}
$$

Constraint (7) indicates whether teams i and j ($i \neq j$) end up with the same points at the end of the competition. The lower bound for pos_i (i.e., $bestPos_i$) can be computed by subtracting from the upper bound, the total teams in the same position as team i.

$$
\begin{aligned}
eq_{ij} &= \begin{cases} 1, & \text{if } tp_j = tp_i \text{ and } i \neq j \\ 0, & otherwise \end{cases} \quad \forall j \in T \land \forall i \in L \\
bestPos_i &= worstPos_i - \sum_{j=1, j \neq i}^{n} eq_{ij} \quad \forall j \in T \land \forall i \in L
\end{aligned} \tag{7}
$$

Finally, constraint (8) sets the bounds for the position of team i and constraint (9) indicates that the positions for two teams must be different.

$$bestPos_i \leq pos_i \leq worstPos_i \quad \forall i \in L \tag{8}$$

$$(pos_i \neq pos_j) \quad \forall i, j \in L \land i \neq j \tag{9}$$

Relative Position Queries: this queries indicate whether a given team i will be above, below, or equal to another team j at the end of the tournament and constraint (10) depicts the five queries. In this particular case we use tp_i and tp_j. We consider that two teams i and j might tied up in the same position if they have the same points at the end of the competition. We recall that we don't use pos_i and pos_j due to constraint (9) which indicates that two teams must have different positions at the end of the competition.

$$
\forall nr_c \in R \land nr_c = (i, op_{ij}, j) \begin{cases} tp_i = tp_j, & \text{if } op_{ij} \text{ is } = \\ tp_i \leq tp_j - 1, & \text{if } op_{ij} \text{ is } < \\ tp_i \leq tp_j, & \text{if } op_{ij} \text{ is } \leq \\ tp_i \geq tp_j + 1, & \text{if } op_{ij} \text{ is } > \\ tp_i \geq tp_j, & \text{if } op_{ij} \text{ is } \geq \end{cases} \tag{10}
$$

Final Point Queries: To guarantee a set S of *final point queries*, the model should include the linear constraints from (11). Notice that the score s_i should have a lower bound of p_i and an upper bound of all the possible points if team i wins all its the remaining games g_{ij}:

$$(tp_i = s_i) \quad \forall ns_d \in S \wedge ns_d = (i, s_i)$$

$$p_i \leq s_i \leq p_i + 3 \cdot \sum_{j=1, j \neq i}^{n} g_{ij} \quad \forall ns_d \in S \wedge ns_d = (i, s_i) \tag{11}$$

Objective Function: Users might be interested in either minimizing or maximizing the total points for a given team as depicted in (12):

$$\text{maximize: } tp_i \quad (i \in T) \tag{12}$$

References

1. Schwartz, B.L.: Possible winners in partially completed tournaments. SIAM Review **8**(3) (1966) 302–308
2. Hoffman, A., Rivlin, T.: When is a team "mathematically" eliminated?, Princeton, NJ, Princeton Symposium on Mathematical Programming (1967) 391–401
3. Kern, W., Paulusma, D.: The new fifa rules are hard: complexity aspects of sports competitions. Discrete Applied Mathematics **108**(3) (2001) 317–323
4. Bernholt, T., Gälich, A., Hofmeister, T., Schmitt, N.: Football elimination is hard to decide under the 3-point-rule. In: MFCS. (1999) 410–418
5. Pálvölgyi, D.: Deciding soccer scores and partial orientations of graphs. Acta Univ. Sapientiae, Math **1**(1) (2009) 35–42
6. Kern, W., Paulusma, D.: The computational complexity of the elimination problem in generalized sports competitions. Discrete Optimization **1**(2) (2004) 205–214
7. McCormick, S.T.: Fast algorithms for parametric scheduling come from extensions to parametric maximum flow. Operations Research **47**(5) (1999) 744–756
8. Wayne, K.D.: A new property and a faster algorithm for baseball elimination. SIAM Journal on Discrete Mathematics **14**(2) (2001) 223–229
9. Ribeiro, C.C., Urrutia, S.: An application of integer programming to playoff elimination in football championships. International Transactions in Operational Research **12**(4) (2005) 375–386