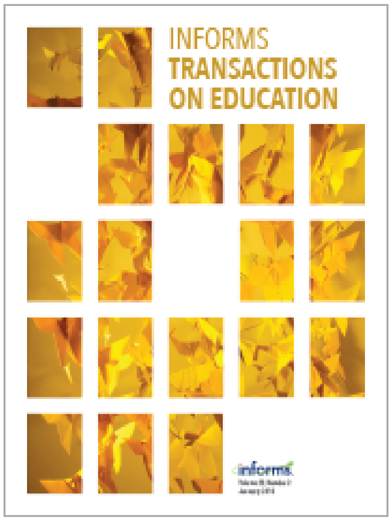

INFORMS Transactions on Education

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Puzzle—Integer Programming and League Table Puzzles

Martin J. Chlond,

Vol. 11, No. 3, May 2011, pp. 138–140
ISSN 1532-0545 | 11 | 1103 | 0138
INFORMS Transactions on Education

DOI 10.1287/ited.1110.0067

Puzzle

Integer Programming and League Table Puzzles

Martin J. Chlond
Lancashire Business School, University of Central Lancashire, Preston PR1 2HE, United Kingdom,
mchlond@uclan.ac.uk

1. Introduction

An interesting puzzle emerges naturally from the construction of league tables in team sports. A league table typically has seven columns containing data on games played, games won, drawn, lost, goals for, goals against and points for each team. In the game of soccer, points are awarded on the basis of three for a win, one for a draw, and none for a loss, and the teams are arranged in decreasing order of points gained. Is it possible, from this summary data, to reconstruct the actual scores for each of the matches played?

Small tables may occur in reality or may be carefully constructed such that there is only one possible set of scores that could produce the given table. In such cases the scores may be determined by the application of logical thinking.

2. Examples

In each of the examples presented below, each team plays each of the other teams at most once.

A simple example taken from "The Hidden Mathematics of Sport" by Eastaway and Haigh (2007) is in Table 1.

The completed table for Group G in the FIFA World Cup 2010 is reproduced in Table 2. This is the only one of the eight groups in which the table leads inevitably to the actual scores achieved in the tournament, and this may be verified by the interested reader. All other groups resulted in tables that have several possible solutions, only one of which, obviously, represents the actual solution.

The puzzles can be made more interesting if we extend the idea to situations in which not all matches have been played at the time the table was constructed, and also there is missing data. These can be much more difficult to solve and have quite a Sudoku-like feel to them. An example taken from Eastaway and Haigh (2007) is in Table 3.

Solutions to the above three puzzles are in the appendix.

3. IP Formulation

It is convenient to adopt the notion of home and away games, although the puzzle conditions do not require us to deduce this, and there is insufficient information to do so. This is merely a device to allow us to distinguish between teams when computing scores.

Define sets $Nteam = 1..n$ and $Column = 1..7$ in which n is the number of teams in the table. Also define M as a suitable upper bound on the score.

Triangular matrices of decision variables are defined as follows:

$Played_{i,j} = 1$ if match $\{i, j\}$ played, 0 otherwise
$Hscore_{i,j}$ (integer)—home score of match $\{i, j\}$
$Ascore_{i,j}$ (integer)—away score of match $\{i, j\}$
$Hwin_{i,j} = 1$ if match $\{i, j\}$ home win, 0 otherwise
$Awin_{i,j} = 1$ if match $\{i, j\}$ away win, 0 otherwise
$Draw_{i,j} = 1$ if match $\{i, j\}$ drawn, 0 otherwise

In each case the parameters are defined over the range $i = 1..n-1$ and $j = i+1..n$.

We also need to define the initial table of results as a set of decision variables to accommodate those puzzles that do not contain complete information.

$$Result_{i,j} = integer \quad \forall\, i \in Nteam, \quad j \in Column$$

Finally, define a parameter table to accommodate the details of a particular puzzle:

$$Given_{i,j} = integer \quad \forall\, i \in Nteam, \quad j \in Column,$$

where $Given_{i,j} = -1$ if the value is unavailable.

Table 1 Eastaway and Haigh's First Puzzle

	Played	Won	Drawn	Lost	Goals for	Goals against	Points
United	2	1	1	0	1	0	4
City	2	1	0	1	2	1	3
Albion	2	0	1	1	0	2	1

Table 2 World Cup 2010—Group G

	Played	Won	Drawn	Lost	Goals for	Goals against	Points
Brazil	3	2	1	0	5	2	7
Portugal	3	1	2	0	7	0	5
Ivory Coast	3	1	1	1	4	3	4
North Korea	3	0	0	3	1	12	0

Table 3 Eastaway and Haigh's First Puzzle

	Played	Won	Drawn	Lost	Goals for	Goals against	Points
Athletic	3	2	*	*	4	4	*
Rovers	*	1	0	*	3	0	*
Town	*	*	0	*	1	1	*
Wanderers	*	*	*	*	*	*	*

*Represents missing data.

The aim of the puzzles is to find scores for the games played that are consistent with the information given and, hence, no objective function is required. The following constraints enforce the conditions of the puzzles.

Insert *Given* data into *Result* table:

$$Result_{i,j} = Given_{i,j}, \quad \forall\, i \in Nteam,$$
$$j \in Column \mid Given_{i,j} \geq 0.$$

Scores exist only for games played:

$$Hscore_{i,j} \leq M\ Played_{i,j}, \quad \forall\, i \in 1..n-1, \quad j \in i+1..n,$$
$$Ascore_{i,j} \leq M\ Played_{i,j}, \quad \forall\, i \in 1..n-1, \quad j \in i+1..n.$$

The number of games played, won, drawn and lost by each team (that is, $\forall\, i \in Nteam$) are consistent with the respective columns in the *Result* table:

$$\sum_{j=i+1}^{n} Played_{i,j} + \sum_{j=1}^{i-1} Played_{j,i} = Result_{i,1},$$
$$\sum_{j=i+1}^{n} Hwin_{i,j} + \sum_{j=1}^{i-1} Awin_{j,i} = Result_{i,2},$$
$$\sum_{j=i+1}^{n} Draw_{i,j} + \sum_{j=1}^{i-1} Draw_{j,i} = Result_{i,3},$$
$$Result_{i,1} - Result_{i,2} - Result_{i,3} = Result_{i,4}.$$

"Goals for" and "goals against" for each team are consistent with the *Result* table:

$$\sum_{j=i+1}^{n} Hscore_{i,j} + \sum_{j=1}^{i-1} Ascore_{j,i} = Result_{i,5},$$
$$\sum_{j=i+1}^{n} Ascore_{i,j} + \sum_{j=1}^{i-1} Hscore_{j,i} = Result_{i,6}.$$

Points gained by each team are consistent with wins, losses, and draws:

$$3Result_{i,2} + Result_{i,3} = Result_{i,7}.$$

The following sets of constraints ensure that the scores obtained in all played games are consistent with whether the result was a home win, an away win, or a draw. (Note: In each case the constraints are defined over the range $i \in Nteam$, $j \in i+1..n$.)

If home win. then $Hwin_{i,j} = 1$, else 0:

$$Hscore_{i,j} - Ascore_{i,j} - M\ Hwin_{i,j} \leq 0,$$
$$Hscore_{i,j} - Ascore_{i,j} - (1+M)Hwin_{i,j} \geq -M.$$

If away win, then $Awin_{i,j} = 1$, else 0:

$$Ascore_{i,j} - Hscore_{i,j} - M\ Awin_{i,j} \leq 0,$$
$$Ascore_{i,j} - Hscore_{i,j} - (1+M)Awin_{i,j} \geq -M.$$

If game is played and neither of above, then $Draw_{i,j}$=1, else 0:

$$Hwin_{i,j} + Awin_{i,j} + Draw_{i,j} = Played_{i,j}.$$

A GLPK (2010) Mathprog implementation of the above formulation is included (league.mod) and may be used to solve the above puzzles and others.

A useful additional facility for puzzle setters would be the ability to test for uniqueness of a particular puzzle. This may be achieved by modifying the formulation such that a known solution is provided as input, and constraints added to ensure that any solution generated must be different in some way from this known solution. There would be no feasible solution to this modified IP for a puzzle with a unique solution. The code necessary to implement this modification is included in league.mod.

This type of puzzle was possibly invented by E. R. Emmet, and quite a number appear in his book "Puzzles for Pleasure" (Emmet 1972). These include many more challenging puzzles in which some of the information is known to be false and others in which the tables have been produced by combinations of truth-tellers, liars, and people who alternate between lying and truth-telling.

Supplementary Material

Files that accompany this paper can be found and downloaded from http://ite.pubs.informs.org/.

Appendix

Solution to puzzle in Table 1.

	City	Albion
United	1, 0	0, 0
City		2, 0

Solution to puzzle in Table 2.

	Portugal	Ivory Coast	N. Korea
Brazil	0, 0	3, 1	2, 1
Portugal		0, 0	7, 0
Ivory Coast			3, 0

Solution to puzzle in Table 3.

	Rovers	Town	Wanderers
Athletics	0, 3	1, 0	3, 1
Rovers		*	*
Town			1, 0

*Represents an unplayed match.

References

Eastaway, R., J. Haigh. 2007. *Beating the Odds: The Hidden Mathematics of Sport*. Robson Books, London.

Emmet, E. R. 1972. *Puzzles for Pleasure*. Bell Publishing Company, New York.

GLPK (Gnu Linear Programming Kit). 2010. Accessed on December 7, 2010, http://www.gnu.org/software/glpk/.