A GENETIC ALGORITHM FOR SIMULTANEOUSLY SCHEDULING GAMES AND ASSIGNING REFEREES IN TURKISH FOOTBALL LEAGUE

OLGU PELİN HÜSEYİNOĞLU

IŞIK UNIVERSITY

2014

A GENETIC ALGORITHM FOR SIMULTANEOUSLY SCHEDULING GAMES AND ASSIGNING REFEREES IN TURKISH FOOTBALL LEAGUE

OLGU PELİN HÜSEYİNOĞLU

B.S., Industrial Engineering, Işık University, 2010

Submitted to the Graduate School of Science and Engineering

in partial fulfillment of the requirments for the degree of

Master of Science

in

Industrial Engineering

IŞIK UNIVERSITY

2014

ISIK UNIVERSITY

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

A GENETIC ALGORITHM FOR SIMULTANEOUSLY SCHEDULING GAMES AND ASSIGNING REFEREES IN TURKISH FOOTBALL LEAGUE

OLGU PELİN HÜSEYİNOĞLU

APPROVED BY:

Assist. Prof. Tankut ATAN   Işık University ______________________

(Thesis Supervisor)

Assoc. Prof. S.Çağlar AKSEZER Işık University ______________________

Assoc. Prof. Olcay T. YILDIZ Işık University ______________________

APPROVAL DATE: ..../..../....

A GENETIC ALGORITHM FOR SIMULTANEOUSLY SCHEDULING GAMES AND ASSIGNING REFEREES IN TURKISH FOOTBALL LEAGUE

Abstract

Scheduling games in a league and referee assignment are important planning tasks of professional football federations. Previous research studied game scheduling and referee assignment problems separately. This thesis presents an integer linear model for simultaneously scheduling Turkish Football League (Süper Lig) matches and assigning referees to games. Due to the difficulty in obtaining an exact solution, we also develop a genetic algorithm for solving the problem approximately. In solving the simultaneous problem we consider several constraints used by Turkish Football Federation (TFF) along with additional constraints that are important for a good schedule. We use Turkish league data from 2010-2013 in our analysis of the developed genetic algorithm. Our heuristic approach provides a general framework that can be used for other leagues as well.

TÜRKİYE FUTBOL LİGİNDE SİMULTANE OYUN PROGRAMLAMA VE HAKEM ATAMA İÇİN GENETİK ALGORİTMA

Özet

Profesyonel futbol liglerinde oyunların çizelgelenmesi ve hakem ataması önemli planlama işidir. Daha önce yapılan araştıma çalışmalarında lig fikstürü oluşturulması ve hakem ataması ayrı problemler olarak ele alınmıştı. Bu tez çalışmasında eş zamanlı olarak Türkiye Futbol Fedarasyonu'nda lig fikstürü oluşturma ve hakem atama problemi için tam sayılı doğrusal bir model sunulmaktadır. Optimal bir sonuç elde etmede yaşanan zorluklar nedeni ile problemin çözümü için bir genetik algoritma geliştirilmiştir. Eş zamanlı problem çözümünde Türkiye Futbol Federasyonu'nun kullandığı çeşitli kısıtlar ve ek olarak iyi bir planlama için gerekli diğer önemli kısıtlar düşünülmüştür. Geliştirilen genetik algoritma analizlerinde 2010-2013 Türkiye Futbol Federasyonu verileri kullanılmıştır. Geliştirilen buluşsal yaklaşım genel bir çatı olup başka liglerde de kullanılanbilir.

Acknowledgements

I would like to thank to my supervisor, Assist. Prof. Tankut Atan for his guidance. I am very grateful to my family and my friends for their encouragements during my graduate studies.

To my family …

Table of Contents

List of Tables

List of Figures

List of Abbreviations

BJK	Beşiktaş
FB	Fenerbahçe
FIFA	Fédération Internationale de Football Association
GA	Genetic Algorithm
GAMS	General Algebraic Modeling System
GFS	Game Fixture Scheduling
GS	Galatasaray
IP	Integer Programming
MHK	National Referee Committee
RAP	Referee Assignment Problem
TFF	Turkish Football Federation

Chapter 1

Introduction

1.1 Motivation

Professional sports have great economic value for the players and the sports clubs involved all over the world. Football is one of the most, if not the most, popular sports worldwide.

Due to the increasing economic value of football, it is no more just sports but also an industry to work in. In Turkey, football has become a major business for many stakeholders, such as teams, fans and players. Furthermore, not only sportsmen but also operations researchers are interested in football academically.

The successful arrangement of league fixture table for any sports is complicated business, and is an art not easily acquired. One aim of fixture making in football is to ensure that each club in the league plays all the others in the same league, during the season both at home and away; this is called as a double round-robin tournament. League fixture scheduling has many changing constraints and rules across different leagues for different countries so a common model cannot be applicable all over the world. The task of scheduling matches in Turkey before each regular season taking into account different factors and to ensure that the resulting game calendar is simultaneously fair to the teams, and also economically beneficial and attractive to sports fans would be nearly impossible to arrange manually. In Turkish Football League (Spor Toto Süper Lig), Turkish Football Federation (TFF) determines scheduling of league fixture.

Second important issue in planning the schedule is assignment of referees to games. These assignments are also criticized by many people for several reasons as referee decisions affect the match score in some cases. We generally see that team players and team managers argue with the referees' decisions during the game. This situation creates tension between referee and the team. The number of referees assigned is changing depending on the sport and the tournament. For instance, football games require 4 to 6 referees, basketball games require 3 referees. There are a number of rules and objectives that should be taken into account when referees are assigned to a game. Also, in some amateur children leagues, some of the referees can be players or relatives, which is really unfair. In Turkey, Central Arbitration Committee (MHK) makes referee assignments for Süper Lig. The referee assignments are done weekly as the league progresses. Those assignments may ignore some constraints in the fair referee assignment problems such as skill level of the referee or overloaded referee assignments. Since we decided to solve game scheduling and referee assignment in GAMS as separate problems. We found that GAMS is not sufficient for solving fixture scheduling and referee assignment simultaneously. In order to construct a league schedule and to assign referees fairly simultaneously within Turkish League's constraints we developed a novel genetic algorithm.

1.2 Contributions

This thesis integrates two problems solved sequentially in practice, namely fixture building and referee assignment, and solves them simultaneously. First a linear integer model is given that incorporates several scheduling rules used by TFF as well. Then the problem is solved approximately with the help of a genetic algorithm. The genetic algorithm starts with a set of solutions obtained by combining random assignments to game and referee templates. Such templates are typical in league scheduling. We also introduce the concept of referee templates which are obtained by solving an optimization problem only consisting of referee scheduling constraints. Data from 2010-2013 Süper Lig seasons are used for numerical experiments on the genetic algorithm.

1.3 Thesis Organization

The remainder of the thesis report is organized as follows. Next we provide a the literature review related to the general concept of the problem. Chapter 2 introduces the mathematical model, and later Chapter 3 provides details of the referee assignment problem and describes the steps of the developed genetic algorithm. Finally, in Chapter 4, we present results of numerical experiments and conclusions drawn from this research.

1.4 Literature Review

There are a wide variety of heuristic approaches for solving the fixture scheduling and referee assignment problem which I have researched.

There are many academic researches on scheduling games for football leagues such as in Austria and Germany (Bartsch et al. (2006)), Belgium (Goossens and Spieksma (2009)), Brasil (Riberio and Urrutia (2012)), Canada (Kostuk and Willoughby (2012)), Chile (Duran et al. (2007), Dur an et al. (2012)), Denmark (Rasmussen (2008)), and Honduras (Fiallos et al. (2010)). Croce and Oliveri (2006) report computational studies for Italy.

Silva et al. (2002) improved a simulation technique for fixture scheduling in order to predict the number of points needed to qualify for play-off elimination in Brazilian league.

Ribeiro and Urrutia (2006) developed an optimization approach to play-off elimination, which solves the integer programming problems sequentially, and guesses which team will qualify for the play- offs.

Duarte et al. (2006) discussed the problem of assigning referees to tournament games when more than one game is played at the location on the same day and the referee can officiate more than one game. The skill level is a performance-based component of the model.

Gil-Lafuente (2003) discussed the Spanish football league. In Spanish football league referee assignments are made by computer. Gill-Lafuante compared those assignments to his proposed model which solves an assignment problem based on expert judgements of referees and criticality of games..

Süper Lig referee assignments are also made weekly. In their paper "Fair Referee Assignments For Professional Football Leagues" Yavuz et al. (2008) focused on assigning referees fairly. They developed a mathematical formulation for fair assignments for all season. Also, they developed a constructive heuristic and local search procedure.

Kendal et al. (2010) implies that key aspect of sporting events is the ability to generate schedules that optimize logistic issues and that are seen as fair to all those who have an interest. This is not just restricted to generating the fixtures, but also to other areas such as assigning officials to the games in the competitions.

Ribeiro (2012) discussed fundamental problems in sports scheduling and their formulations, and surveyed applications of optimization methods to scheduling problems in professional leagues of different sport disciplines such as football, baseball, basketball, cricket, and hockey.

Trick et al. (2012) discussed about baseball referee assignment and traveling referee problem in major baseball league in the United States. They used network optimization to schedule referees to baseball league. To develop this approach, they created the traveling umpire problem, which includes the major umpire scheduling issues and also provides a test for alternative techniques.

Alarcon et al. (2012) mentioned about integer linear programming for referee assignment problem in the Chilean professional football league. Their approach considers balance in the number of matches each referee must officiate, the frequency of each referee being assigned to a given team, the distance each referee must travel over the course of a season, and the appropriate pairings of referee experience or skill category with the importance of the matches.

Kendall (2008) discussed about the English football fixtures over holiday periods in order to minimize total distance traveled by all teams. Also search and local search algorithms are used in the paper. He found that schedules he created have better objective functions than the real fixture scheduling.

Goossens and Spieksma (2009) describe scheduling in the Belgian soccer league. They described how they automated and improved the development of the 2006–2007 season schedule, and explain how they achieved additional improvement by dividing the scheduling problem into two problems. And they compared the results with the manually created fixtures and found automated fixture has better results.

Chapter 2

Mathematical Model

2.1 Game Fixture Scheduling in TFF

Turkish Football Süper Lig consists of 18 teams. 18 teams play home and away games through the 34 weeks during the football season. Each week 18 teams can play 9 games. Total number of games through the season will be 34 x 9 = 306. Each game can be played with two teams. Each game should have (one home and one away team with each opponent). Each team should play a home followed by an away game on the consecutive weeks.

When scheduling the games, Turkish Football Federation (TFF) uses a template where teams are numbered from 1 to 18. This template has been taken from the British Football Association. After randomly assigning numbers in that template to teams a schedule is obtained.. Also this template minimizes the number of breaks, i.e. consecutive home or away games.

Fixture scheduling constraints can be divided into hard constraints and soft constraints. Hard constraints cannot be violated during the construction of the fixture. For example, each team playing one game each week is a hard constraint which cannot be violated. On the other hand there are soft constraints which are not affecting the schedules as much as hard constraints. But soft constraint violations are also tried to be minimized since it affects the objective function. For example, assigning no or two home games during the same week to the teams from the same city is a soft constraint, which can be violated but tried to be minimized.

When solving a fixture scheduling problem we must ensure that hard constraints are not violated and soft constraint costs are minimized. The quality of fixture scheduling is measured by the degree it violates the soft constraints and by the degree of hard constraints which are fixture conflicts for teams.

Constraints of the Fixture Scheduling Problem (GFS) used by TFF are given below:

- Each team should play only one game each week.

- Each team should play only once against every other team during the first and second halves of the season.
- Top games that are played between top teams (BJK, FB, GS) should be played intermittently which is decided by TFF. So the derby matches are played on the decided weeks.
- Teams from the same city cannot all play home games in the same week.
- Each team should play consecutive home games or away games as little as possible.

2.2 Referee Assignment by MHK

In Turkish Football Süper Lig each week 9 games can be played. Each game should be managed by one center referee, two linesmen and a fourth referee. Starting with the 2013 season, there are two additional referees for observing the goal areas. The center referee is the authority of the game, which is played between one home team and one away team. Two assistant referees are known as linesmen who are watching the lines and offside players also calling fouls when the center referee doesn't notice the foul. The fourth referee records yellow and red cards. Also fourth referee is the subsidiary referee when the center referee cannot be able to manage the game. In 2012-2013, there were 28 center referees available to manage the games. Each referee has a grade, which is decided depending on whether they are FIFA referee or upper classifying referee. FIFA referees have higher grades than upper classifying referees.

Referee assignment are made weekly according to MHK's decisions based on referee's grades and judgements by the committee. Since referees have a great impact on the game and can change the result of the game according to their decisions it's important to assign the right referee to the right game. Also FIFA referees may be managing more games than upper classified referees because of their grading's. This may cause that top games are mostly managed by the same FIFA referees which is unequal. The referees with lower ratings may have less chance to be assigned to an important match for gaining experience. Hence, they can not increase their ratings also which is unfair to upper classified referees.

Constraints of the Referee Assignment Problem used in our model for the Turkish Football League are given below:

- Each game between one home team and one away team should have one center referee.
- Referee score should be greater than the game rating score.
- Each referee cannot officiate more than a maximum number of games.
- A referee should call more than a minimum number of games.
- Each referee can officiate at most one match each week.
- Each referee cannot officiate the same game in the first and second half of the season.
- A referee cannot call two consecutive games of the same team.
- A referee cannot call more than a maximum number of games of the same team.
- Each referee should have at least one week of rest during a period of four consecutive weeks.

2.3 Mathematical Model - Simultaneous GFS and RAP (SGRSP)

We will refer to the combined problem as Simultaneous Game and Referee Scheduling Problem (SGRSP) in the remainder of the thesis. First, we introduce the notation used in the model.

2.3.1 Index Sets

I	=	index set of teams (1…. 18).
J	=	index set of teams (1…. 18).
W	=	index set of weeks (1…17).
TOP	=	index set of top teams (1…. 3).
C	=	index set of cities.
$SAME_C$	=	index set of teams from city c.

RANGE	=	index set of weeks in the predefined range where derby matches should be played.
R	=	index set of referees.
W_1	=	index set of weeks in the first half of the season.

2.3.2 Parameters

ref_r	=	referee r's past performance rating.
$c_{i,j}$	=	minimum referee rating needed to call the game between i and j.
minWeeks	=	minimum number of total weeks for a referee to call games.
maxWeeks	=	maximum number of total weeks for a referee to call games.
maxTeamGames	=	maximum number of games for a referee to call for one team.
maxDerby	=	maximum number of derby's a referee can be assigned to.
penBreak	=	penalty of assigning consecutive home or away games for team i.
penRange	=	penalty of assigning top teams outside of the desired week range.
penSame	=	penalty of not having one home game for the teams from the same city.
penTopTwice	=	penalty of normal team playing against a top team on consecutive weeks.
penMinWeeks	=	penalty of assigning a referee to less than *minWeeks* games.
penMaxWeeks	=	penalty of assigning a referee to more than *maxWeeks* games.
penRating	=	penalty of assigning a referee with not an adequate rating to a game.

penGameTwice	=	penalty of a referee for calling the same game twice in the season.
penCallBacktoBack	=	penalty of a referee calling games for the same team consecutively.
penMaxTeamGames	=	penalty of assigning a referee to more than *maxTeamGames* times for the same team.
penMaxDerbys	=	penalty of assigning a referee more than *MaxDerby*
penRest	=	penalty for referee not resting at least one week during any four week stretch during the season.

2.3.3 Decision Variables

$x_{i,j,w}$	=	1 if team i plays a home game against team j in round (week) w; 0 otherwise.
$y_{i,w}$	=	1 if team i plays consecutive home games in week w and w+1 ; 0 otherwise.
$m_{i,j,w,r}$	=	1 if referee is assigned to game that team i plays a home game against team j in week w ; 0 otherwise.
$dBreak_i$	=	number of consecutive home games scheduled for team i.
dRange	=	number of top teams' games scheduled outside of the desired week range.
$dSamePlus_{w,c}$	=	1 if no home games are scheduled during week w for teams (a pair) from city c; 0 otherwise.
$dSameMinus_{w,c}$	=	1 if teams from city c play at home during week w, 0 otherwise.
$dMaxWeeks_r$	=	number of additional weeks a referee calls games on top of the desired *maxWeeks*.
$dRating_{i,j,w}$	=	additional rating needed for the assigned referee to call a home game against team j in week w.
$dMinWeeks_r$	=	number of missing weeks a referee calls games below the desired *minWeeks*.

dRest$_{w,r}$	=	1 if there is a four week stretch for referee r starting on week w; 0 otherwise.
dTopTwice$_{i,j,jj,w}$	=	number of missing weeks a referee calls games below the desired *minWeeks*.
dGameTwice$_{i,j,w,r}$	=	1 if there is a four week stretch for referee r starting on week w; 0 otherwise.
dMaxTeamGames$_{i,r}$	=	number of additional games referee r calls for team i on the top of the desired *maxTeamGames*
dCallBacktoBack$_{i,w,r}$	=	1 if referee r calls consecutive games for team i during weeks w and w+1 ; otherwise.
dMaxDerbys$_r$	=	number of additional derby matches assigned to referee r on top of the desired *maxDerbys.*

2.3.4 Formulation

Min z=

$$\sum_{i} 2penBreak\ dBreak_i + penRange\ dRange +$$

$$\sum_{w}\sum_{c} penSame(dSamePlus_{w,c} + dSameMinus_{w,c}) +$$

$$\sum_{r} penMinWeeks\ dMinWeeks_r + \sum_{r} penMaxWeeks\ dMaxWeeks_r +$$

$$\sum_{i}\sum_{j,j\neq i}\sum_{wj} penRating\ dRating_{i,j,w} + \sum_{r}\sum_{w} penRest\ dRest_{r,w} +$$

$$\sum_{i}\sum_{j}\sum_{jj}\sum_{w} 2\ penTopTwice\ dTopTwice_{i,j,jj,w} +$$

$$\sum_{i}\sum_{j}\sum_{w}\sum_{r} 2\ penGameTwice\ dGameTwice_{i,j,w,r} +$$

$$\sum_{i}\sum_{r} penMaxTeamGames\ dMaxTeamGames_{i,r} +$$

$$\sum_{i}\sum_{w}\sum_{r} penCallBack\ dCallBacktoBack\ Twice_{i,w,r} +$$

$$\sum_{r} penMaxDerbys\ dMaxDerbys_r \qquad (2.1)$$

Subject to

$$\sum_{w\in W_1} (x_{i,j,w} + x_{j,i,w}) = 1\ \ \forall i \in I, \forall j \in J,\ i < j \qquad (2.2)$$

$$\sum_{j,j\neq i} (x_{i,j,w} + x_{j,i,w}) = 1\ \ \forall i \in I, \forall w \in W_1 \qquad (2.3)$$

$$x_{i,j,w} = x_{j,i,w+|W_1|} \quad \forall i \in I, \forall j \in J, i \neq j \quad \forall w \in W_1 \qquad (2.4)$$

$$\sum_{j,j\neq i} x_{i,j,w} + x_{i,j,w+1} \leq 1 + y_{i,w} \quad \forall i \in I, \forall w < 33 \qquad (2.5)$$

$$x_{i,j,w} + x_{i,jj,w+1} + x_{j,i,w} + x_{jj,i,w+1} - dTopTwice_{i,j,jj,w} \leq 1 \; \forall i \notin TOP, \forall j \in TOP, \forall jj \in TOP \;\; j \neq jj, \forall w \in W_1 \quad (2.6)$$

$$\sum_{w} y_{i,w} - dBreak_i \leq 0 \;\; \forall i \in I \qquad (2.7)$$

$$\sum_{i,i\in TOP} \; \sum_{j,j\in TOP, j\neq i} \; \sum_{w,w\notin RANGE} x_{i,j,w} - dRange = 0 \qquad (2.8)$$

$$\sum_{i,i\in SAME_c} \; \sum_{j,j\in SAME_c} x_{i,j,w} + dSamePlus_{w,c} - dSameMinus_{w,c} = |SAME_c| - 1 \;\; \forall c \in C, \; \forall w \in W \;\; (2.9)$$

$$\sum_{r} m_{i,j,w,r} = x_{i,j,w} \quad \forall i \in I, \forall w \in W, \forall j \in J \; i \neq j \qquad (2.10)$$

$$\sum_{i} \sum_{j,j\neq i} m_{i,j,w,r} \leq 1 \;\; \forall r \in R, \forall w \in W, \; i \neq j \qquad (2.11)$$

$$\sum_{i} \; \sum_{j,j\neq i} \; \sum_{w} m_{i,j,w,r} - dminWeeks_r \geq minWeeks \quad \forall r \in R \qquad (2.12)$$

$$\sum_{i} \; \sum_{j,j\neq i} \; \sum_{w} m_{i,j,w,r} - dmaxWeeks_r \leq maxWeeks \quad \forall r \in R \qquad (2.13)$$

$$\sum_{r} ref_r \;\; m_{i,j,w,r} + dRating_{i,j,w} \geq x_{i,j,w} \; c_{i,j} \;\; \forall i \in I, \forall j \in J, \forall w \in W \; i \neq j \;\; (2.14)$$

$$\sum_{i} \; \sum_{ij,\neq i} \; \sum_{t\in W, w\leq t\leq w+3} m_{i,j,t,r} - dRest_{w,r} \leq 3 \;\; \forall w \in |W| - 3, \forall r \in R \qquad (2.15)$$

$$m_{i,j,w,r} + m_{j,i,w+|W_1|,r} - dGameTwice_{i,j,w,r} \leq 1 \;\; \forall r \in R, \forall i \in I, \forall j \in J, \forall w \in W_1 \;\; i \neq j (2.16)$$

$$\sum_{i,j\neq i} \sum_{w} (m_{i,j,w,r} + m_{j,i,w,r}) - dMaxTeamGames_{i,r} \leq maxTeamGames \;\; \forall r \in R, \forall i \in I \;\; (2.17)$$

$$\sum_{i,j\neq i} (m_{i,j,w,r} + m_{j,i,w,r} + m_{i,j,w+1,r} + m_{j,i,w+1,r}) - dCallBacktoBack_{i,w,r} \leq 1 \; \forall i \in I, \forall w \in |W|, \forall r \in R \;\; (2.18)$$

$$\sum_{i,i\in TOP} \; \sum_{j,j\neq i, j\in TOP} \; \sum_{w} m_{i,j,w,r} - dmaxDerbys_r \geq maxDerbys \quad \forall r \in R \qquad (2.19)$$

$$x_{i,j,w} \in \{0,1\} \; \forall i \in I, \; \forall j \in J, j \neq i, \; \forall w \in W \quad (2.20)$$

$$m_{i,j,w,r} \in \{0,1\} \; \forall i \in I, \; \forall j \in J, j \neq i, \; \forall w \in W, \; \forall r \in R \; (2.21)$$

$$dBreak_i \geq 0 \;\; \forall i \in I \;\; (2.22)$$

$$dRange \geq 0 \;\; (2.23)$$

$$dSamePlus_{w,c} \geq 0 \; \forall w \in W, \; \forall c \in C \; (2.24)$$

$$dSameMinus_{w,c} \geq 0 \; \forall w \in \; \forall c \in C \; (2.25)$$

$$dMaxWeeks_r \geq 0 \quad \forall r \in R \;\; (2.26)$$

$$dRating_{i,j,w} \geq 0 \; \forall i \in I, \; \forall j \in J, j \neq i, \; \forall w \in W \;\; (2.27)$$

$$dMinWeeks_r \geq 0 \quad \forall r \in R \;\; (2.28)$$

$$dRest_{w,r} \geq 0 \;\; \forall w \in W, \; \forall r \in R \;\; (2.29)$$

$$dTopTwice_{i,j,jj,w} \geq 0 \; \forall i \notin TOP, \forall j \in TOP, \forall jj \in TOP \;\; j \neq jj, \forall w \in W_1 \;\; (2.30)$$

$$dGameTwice_{i,j,w,r} \geq 0 \ \forall i \in I, \ \forall j \in J, j \neq i, \ \forall w \in W, \ \forall r \in R \ (2.31)$$

$$dMaxTeamGames_{i,r} \geq 0 \ \ \forall i \in I, \forall r \in R \quad (2.32)$$

$$dCallBacktoBack_{i,w,r} \geq 0 \ \ \forall i \in I, \ \forall w \in W, \forall r \in R \quad (2.33)$$

$$dMaxDerbys_{r} \geq 0 \ \ \forall r \in R \ (2.34)$$

The aim of the objective function is minimizing of total number of game and referee conflicts in the fixture scheduling and referee assignment. Eq. (2.2) ensures that each team must play one game every week through 17 weeks. Eq. (2.3) satisfies that each team must play with each other once a time through 17 weeks. Eq (2.4) satisfies first 17 weeks schedule for home and away games should be symmetric to the last 17 weeks. For instance; if team 1 is playing home game in 1^{st} week, team 1 must play away game on the 18^{th} week. Eq (2.5) satisfies that home and away team matching game cannot be the same on consecutive week. For instance; if team 1 plays home game on week 1 with team 2; in week 2 team1 cannot play home game with team 2.

There are three top teams in the Turkish Football league, which are Beşiktaş, Fenerbahçe and Galatasaray. Games played between those top teams are called derbys. Eq (2.6) satisfies the condition; normal teams cannot play consecutive games with Top teams. Eq (2.7) each team cannot play consecutive home games. For instance; if team #1 plays home game on week 1; on the week 2, team #1 must play away game. Eq (2.8) satisfies that top team cannot play derby games out of the desired week range, which is decided by TFF. Eq (2.10) satisfies the condition that each home game should have one assigned center referee. Eq (2.11) satisfies the condition that each referee can manage at most one game each week. For instance; center referee #1 cannot officiate two games on the same week. Eq (2.12) tries to satisfy the condition that each referee should officiate games less than Min weeks. Eq (2.13) tries to satisfy the condition that each referee cannot officiate games more than Max weeks. For instance, if there are 28 available referees for season 2012-2013, and there are 306 games through the whole season. One referee can manage at most 11 games (306/28). This max week constraint is flexible since available referees are changing from season to season.

Each referee has a rating scale starting from 1 to 10. And each game has a rating changing according to team matching's. Eq (2.14) tries to minimize the gap between skill level and game rating. For instance; Referee #1 has skill level of 6, and the game he assigned will be a derby and game rating is 10. If the referee rating is not sufficient to manage game rating, then there will be a penalty, which will increase the objective function value. Eq (2.15) limits the condition that each referee should have a one-week rest after 4 consecutive weeks of match management. Eq (2.16) satisfies the condition that referee who managed a game played between team #1 and team #2 on the first week, cannot officiate the same symmetric game played between team #1 and team #2 on the 18th week. Eq (2.17) satisfies the condition that each referee cannot officiate one team's match more than $maxTeamGames$. Eq (2.18) satisfies the condition that each referee cannot officiate one team's match on the consecutive weeks. Eq (2.19) satisfies the condition that each referee cannot officiate derby matches more than $maxDerbys$. Simultaneous GFS and RAP problem is tried to solve by GAMS 23.1. We run the problem one day long, but since the size of the problem is huge GAMS couldn't solve problem because of memory constraints. In order to see the magnitude of the problem we gave the constraints and variables in Table 2.1. Also the discrete columns of variables are 281,520 in the SGRPS model for 2010-2011 and 2011-2012 season's. For 2013-2014 seaeson's data discrete columns of variables are 302,328.

Table 2.1 Problem sizes of SGRSP

	2010-2011		2011-2012		2012-2013	
	Variables	Constraints	Variables	Constraints	Variables	Constraints
Initial	181,539	794,704	181,505	794,670	193,269	838,844
Reduced	165,913	429,872	165,879	429,838	177,643	462,342

Chapter 3

Genetic Algorithm (GA)

3.1 What is a Genetic Algorithm?

Genetic algorithms are search and optimization techniques based on Darwin's principle of natural selection. GA performs the search by solution recombination and belongs to the class of Evolutionary Algorithms (EA). Evolutionary Algorithm uses inheritance, natural selection, recombination and mutation to control the search process. The set of solutions is called population in Genetic Algorithm language. Single solution in the set of solutions is called chromosome or individual. Individuals are made of genes.

Parents in genetic algorithms can be recombined and the resulting solution is called offspring or child. The quality of each solution is determined by the help of the objective function. According to Genetic Algorithm literature objective function values are called fitness values. Every solution should have a fitness value in order to decide on selection or replacement process.

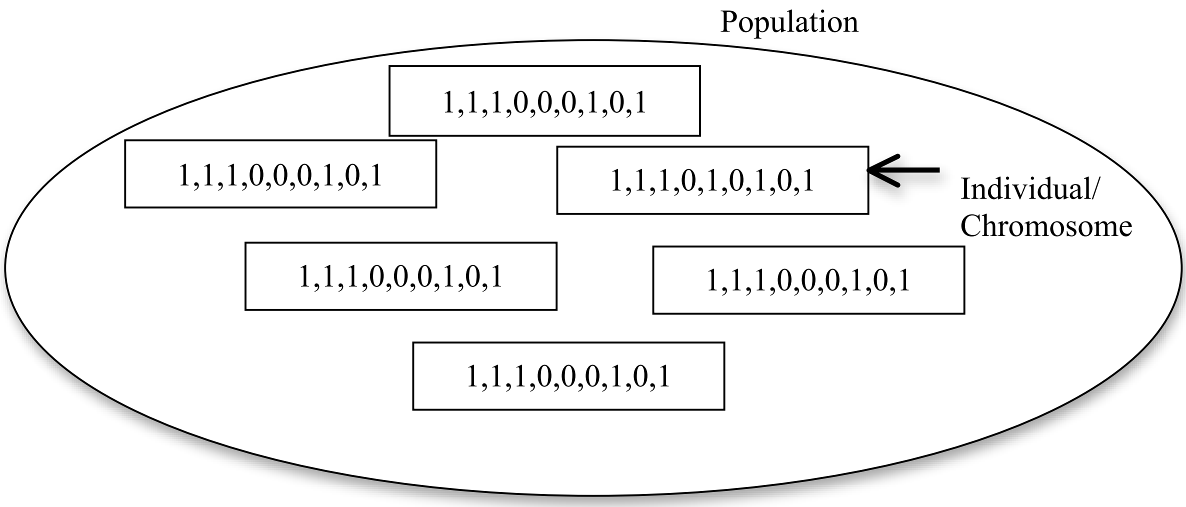

Figure 3.1 Genetic algorithm terminology

There are seven basic components of Genetic Algorithm, which are Population, Initialization, Evaluation, Selection, Recombination, and Mutation. Main components and their relations are illustrated in Figure 3.2.

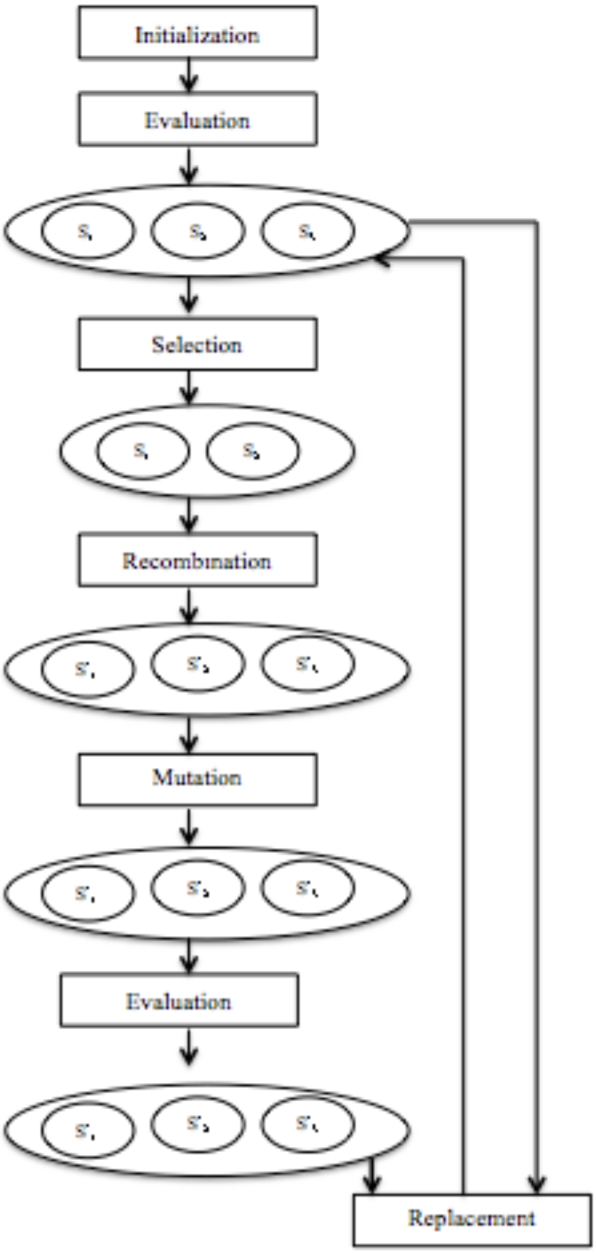

Figure 3.2 Genetic algorithm overview

3.2 Population

The population contains all candidate solutions. In the beginning the population should contain maximum amount of solutions for better results. As the search process continues, quality of the solutions will become better and converges to a final value. It is important to choose an appropriate population size. If the population size is too small, it would be difficult for the solution to converge to optimal. If the population size is too big then it would increase the computational effort.

3.3 Initialization

The initialization is needed to generate the solutions (chromosomes) in the first population. Generally, initialization scheme changes according to the problem at hand. Whatever method is used, initial population should consist of solutions as different as possible. Since it is better to produce diverse solutions at initialization, it is popular to use randomized sampling methods.

3.4 Evaluation

Evaluation determines the objective function of the solutions, which is called fitness in GA literature. The fitness is calculated in order to determine and replace bad solutions with better ones. Fitness of the solutions is generally needed for the selection and the replacement process of the Genetic Algorithm. There should be a quality measure during the replacement and selection process. The fitness of the solution steers the search process, so it should give information about the solutions' comparative quality.

3.5 Selection

The selection is needed in order to choose which solution will be used in recombination. This decision is based on the evaluation of the solutions. Generally the fitness value specifies the fair quality of the solutions. Better fitness-valued solution will be selected for recombination in the hope that it will produce good offspring's. The better solution will change according to problem type whether it is minimization or maximization problem. The problem tries to attain better solutions after recombination. Different types of methods are developed over past years, which will be explained detailed below.

- **Roulette-Wheel Selection:** Fitness proportionate selection, also known as roulette wheel selection, is a genetic operator used in genetic algorithms for selecting potentially useful solutions for recombination. Let's say that there are five solutions with below fitness values in a maximization problem. A higher fitness valued solution for the maximization problem is more likely to be selected. $f(s_1) = 15$, $f(s_2) = 25$, $f(s_3) = 35$, $f(s_4) = 45$, $f(s_5) = 50$. The probability of selection $p(s_i)$ can be calculated as follows:

$$p(s_i) = \frac{f(s_i)}{\sum_{k=1}^{n} f(s_k)}$$

  So the selection probabilities for example will be;
  $p\,(s_1) = \frac{15}{15+25+35+45+50} = \frac{15}{170}$, $p\,(s_2) = \frac{25}{170}$, $p\,(s_3) = \frac{35}{170}$, $p\,(s_4) = \frac{45}{170}$,
  $p(s_5) = \frac{50}{170}$.
  In this selection type highest fitness valued solution will have high probability to be selected, which will lead to premature convergence.

- **Linear-Rank Selection:** In order to solve the problems within the fitness proportionate selection in the roulette wheel selection, linear rank selection can be used. In this selection type fitness values are ordered descending or ascending according to problem type whether it is maximization or minimization. Let's consider the same solution set as above with the same fitness values: $f(s_1) = 15$, $f(s_2) = 25$, $f(s_3) = 35$, $f(s_4) = 45$, $f(s_5) = 50$. For maximization problem we should sort the values as follows: $f(s_1) > f(s_2) > f(s_3) > f(s_4) > f(s_5)$. According to formula below we can calculate the linear-rank selection probability:

$$s_i = \frac{r(s_i)}{\sum_{k=1}^{n} r(s_k)}$$

r will be the rank of the solution s_k. So the probabilities will be $p(s_1) = \frac{1}{15}$,

$p(s_2) = \frac{2}{15}, p(s_3) = \frac{3}{15}, p(s_4) = \frac{4}{15}, p(s_5) = \frac{5}{15}$: So the higher fitness value solutions are now less likely to be selected giving more chance to other solutions to be selected during selection step.

- **Tournament Selection:** Population is selected randomly and in this type of selection solutions are also selected according to their fitness values. For example: according to previous example selected solutions will be s_5 and s_6.

3.6 Recombination

Recombination follows evaluation and selection steps. Genes from two selected parents are chosen and swapped to hopefully become better fitness-valued solutions. In the Genetic Algorithm literature this process is called crossover. Crossover is a problem dependent operator. General idea for crossover is combining better parts of the two solutions to make high-qualified solution. The example below illustrates a crossover. Crossover point indicates where the crossover will start.

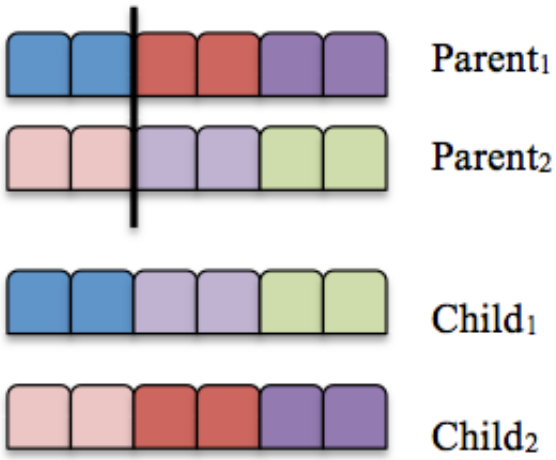

Figure 3.3: Example of a crossover

Since the recombined solutions use previous solutions they don't contain new or different information about the problems. For this reason, to introduce some diversity into the population the so-called mutation operator is used.

3.7 Mutation

Mutation is a genetic operator used to maintain genetic diversity from one generation of a population to the next. It is analogous to biological mutation. By the help of the mutation operator solution may change entirely from the beginning.

3.7 Replacement

After the processes; initialization, evaluation, selection, recombination and mutation, there will be two different populations. By the help of the replacement process some solutions may be discarded because of their low quality. Higher qualified solutions will survive. There are different kinds of replacement where two of the most popular are general replacement and steady state replacement Zäpfel (2010).

3.8 Referee Templates in GA

Before we give the details of the genetic algorithm developed for solving SGRSP, we will give the formulation of an optimization problem used in generating the initial population. We generate weekly refereeing templates for the whole season to be used in building the initial population. These templates obey certain referee work rules which are independent of the game schedule such as maximum number of games to call. This referee assignment problem (RAP) is solved optimally by the help of GAMS. Results coming from GAMS are used in population construction in the genetic algorithm. The optimal solution obeys work constraints for the referees. It does not specify names of the actual referees but only provides indices. Templates are obtained by randomly assigning actual referees to the generic solution (the indices) obtained optimally. Below we give the formulation for this model.

3.8.1 Index Sets

G = index set of games (1…. 9).

W = index set of weeks (1…34).

R = index set of referees.

3.8.2 Parameters

ref_r	=	referee r's past performance rating..
$c_{i,j}$	=	minimum referee rating needed to call the game between i and j.
minWeeks	=	minimum number of total weeks for a referee to call games.
maxWeeks	=	maximum number of total weeks for a referee to call games.
penMinWeeks	=	penalty of assigning a referee to less than *minWeeks* games.
penMaxWeeks	=	penalty of assigning a referee to more than *maxWeeks* games.
penRest	=	penalty for referee not resting at least one week during any four week stretch during the season.

3.8.3 Decision Variables

$m_{g,w,r}$	=	1 if referee is assigned to game g played in week w ; 0 otherwise.
$dMaxWeeks_r$	=	number of additional weeks a referee calls games on top of the desired *maxWeeks*.
$dMinWeeks_r$	=	number of weeks that the assigned number of referee r is above 4 on the consecutive weeks.
$dRest_{w,r}$	=	1 if there is a four week stretch for referee r starting on week w; 0 otherwise.

Min z =

$$\sum_r penMinWeeks\ dMinWeeks_r + \sum_r penMaxWeeks\ dMaxWeeks_r + \sum_r \sum_w penRest\ dRest_{r,w} \quad (3.1)$$

Subject to

$$\sum_{r} m_{g,w,r} = 1 \quad \forall w \in W, \forall g \in G \qquad (3.2)$$

$$\sum_{g} m_{g,w,r} \leq 1 \ \forall r \in R, \forall w \in W \qquad (3.3)$$

$$\sum_{g}\sum_{w} m_{g,w,r} - dminWeeks_{r} \geq minWeeks \quad \forall r \in R \qquad (3.4)$$

$$\sum_{g}\sum_{w} m_{g,w,r} - dmaxWeeks_{r} \leq maxWeeks \quad \forall r \in R \qquad (3.5)$$

$$\sum_{g} \sum_{t \in W, w \leq t \leq w+3} m_{i,j,t,r} - dRest_{w,r} \leq 3 \quad \forall w \in |W| - 3, \forall r \in R \ (3.6)$$

$$m_{g,w,r} \in \{0,1\} \ \forall g \in G, \ \forall w \in W, \ \forall r \in R \ (3.8)$$

$$dMinWeeks_{r} \geq 0 \quad \forall r \in R \ \ (3.9)$$

$$dRest_{w,r} \geq 0 \ \ \forall w \in W, \ \forall r \in R \ \ (3.10)$$

$$dMaxWeeks_{r} \geq 0 \quad \forall r \in R \ \ (3.11)$$

3.8.4 GAMS Results

RAP is relatively simple to solve, and GAMS obtains an optimal solution within seconds. Also the discrete columns of variables are 7,956 in the RAP model for 2010-2011 and 2011-2012's season data. For 2012-2013's data discrete columns of variables are 8598.

Table 3.1 Problem sizes of RAP

	2010-2011		2011-2012		2012-2013	
	Variables	Constraints	Variables	Constraints	Variables	Constraints
Initial	2,049	8,893	2,049	8,893	2,183	9,577
Reduced	2,048	8,814	2,048	8,814	2,182	9,492

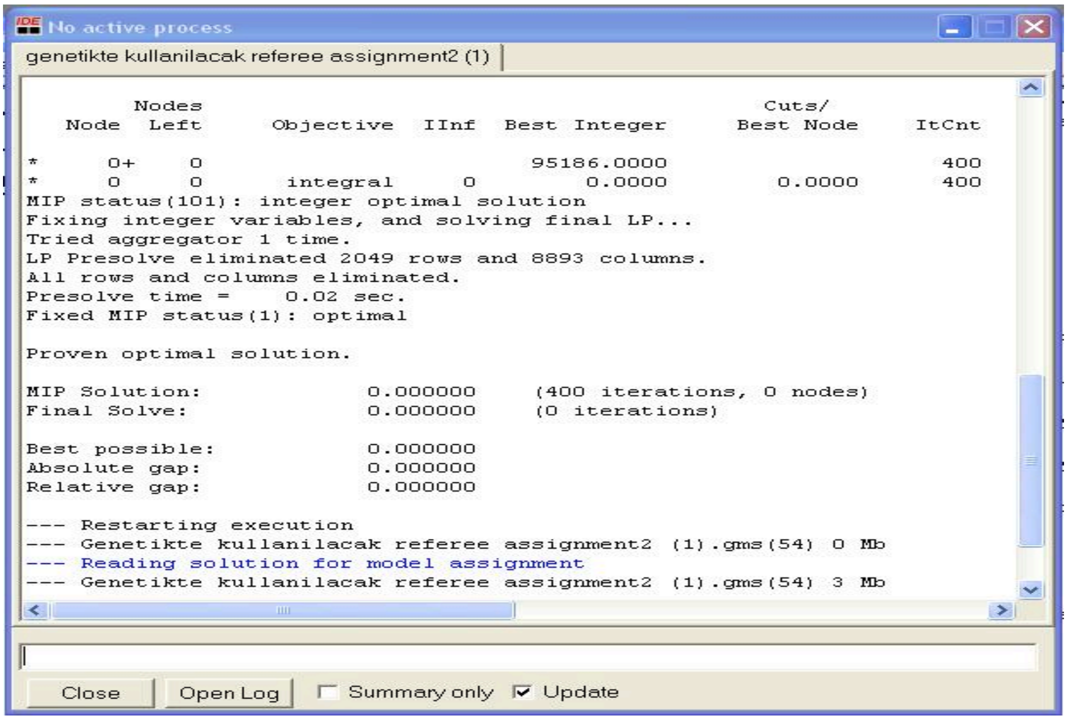

Figure 3.4: Output of GAMS for RAP

3.9 GA for SGRSP

GAMS couldn't find any solution for the simultaneous problem for games scheduling and referee assignment in Süper Lig. Therefore, an approximate solution procedure was needed. Inspired by the use of templates by TFF to schedule games, we first generated independent refereeing templates. These templates did not consider any game related properties of a good referee assignment such as a referee should not be assigned to both games of two teams in the season. We decided to use a genetic algorithm as it suited combining and improving these templates efficiently.

3.9.1 Population

The population size (nSoln) can be critical for the performance of GA: If nSoln is too small, it would be difficult for the solution to converge to optimal. If nSoln is too big, then it would increase the processing time of the program. After many experiments we decided that the nSoln is performing well with 250, and we continued our experiments with nSoln = 250.

3.9.2 Initialization

We used 2010-2013 fixture schedules. We used GAMS referee assignment solution as a template in the problem. We randomly assigned numbers in the RAP solution to referees in order to create different referee assignment templates. We also randomly assigned numbers in the match template of TFF to teams in order to create different game schedule templates. Randomly generated schedules and refereeing templates are randomly combined in order to create an initial population. nSoln will determine how many schedules will be produced randomly.

3.9.3 Evaluation

To measure the quality of the solutions fitness values are calculated for each randomized schedule. Since we are solving a minimization problem solutions are sorted in increasing order. Fitness value is calculated as the sum of penalties for fixture scheduling and referee assignment. Fitness will help us in the selection and replacement process.

3.9.4 Selection

After the calculation of fitness values and we sort fitness values in the increasing order. We used two different methods for selection: linear rank selection and roulette-wheel selection method. We calculated the probabilities of selection by the help of linear rank or roulette wheel selection. In roulette wheel selection probabilities' are calculated for each iteration since fitness values are affected from the probabilities. However, linear rank probabilities do not need to be calculated at each iteration of the GA.

3.9.5 Crossover

Crossover parents are also selected via linear-rank or roulette-wheel probabilities. New children are created from the selected parent solutions. Crossover process can be seen below for our Genetic Algorithm.

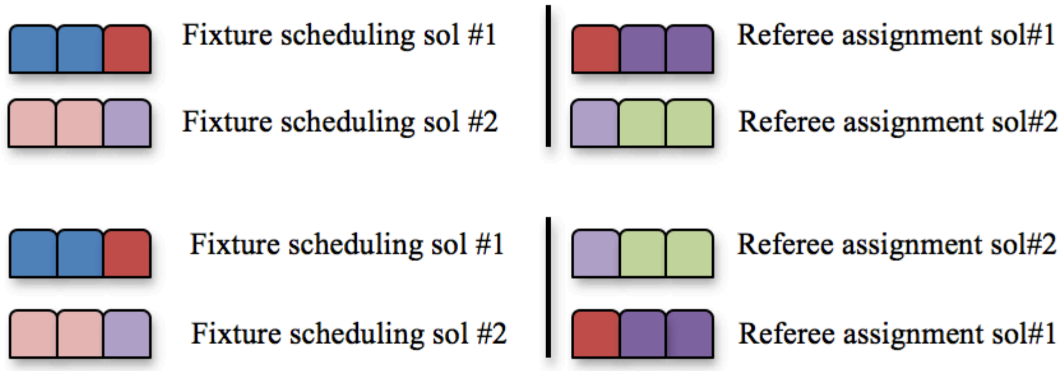

Figure 3.5: Example of a crossover in our GA

Crossover is applied with different crossover rates (0.2, 0.4, 0.6, 0.8, 1), in order to find the best performing crossover rate. Crossover rate determines the number of solutions to be used for crossover. In an elitist approach, a certain number of solutions are set aside and they are not used for crossover.

3.9.6 Mutation Operators

After Crossover process mutation is applied with different mutation rates (0.2, 0.4, 0.6, 0.8, 1) in order to find the best performing mutation rate. Mutation operators are probabilistically applied based on the chosen mutation rate. We used three types of mutation operators, which are mutationSameWeek, mutationDiffWeek and mutationSwapRefsWeek. MutationSameWeek function changes two randomly chosen referees in the same week. MutationDiffWeek function changes two random referees in two random weeks. MutationSwapRefsWeek function swaps all referees in two randomly chosen weeks.

	Ref1	Ref2	Ref3	Ref4	Ref5	Ref6	Ref7	Ref8	Ref 9
Week 1	3	5	8	9	15	20	10	2	13
Week 2	7	1	21	11	19	22	4	12	6

	Ref1	Ref2	Ref3	Ref4	Ref5	Ref6	Ref7	Ref8	Ref 9
Week 1	3	13	8	9	15	20	10	2	5
Week 2	7	1	21	11	19	22	4	12	6

Figure 3.6: Applied mutation same week operator

	Ref1	Ref2	Ref3	Ref4	Ref5	Ref6	Ref7	Ref8	Ref 9
Week 1	3	5	8	9	15	20	10	2	13
Week 2	7	1	21	11	19	22	4	12	6

	Ref1	Ref2	Ref3	Ref4	Ref5	Ref6	Ref7	Ref8	Ref 9
Week 1	3	5	8	9	15	20	10	2	13
Week 2	7	13	21	11	19	22	4	12	6

Figure 3.7: Applied mutation different week operator

	Ref1	Ref2	Ref3	Ref4	Ref5	Ref6	Ref7	Ref8	Ref 9
Week 1	3	5	8	9	15	20	10	2	5
Week 2	7	1	21	11	19	22	4	12	6

	Ref1	Ref2	Ref3	Ref4	Ref5	Ref6	Ref7	Ref8	Ref 9
Week 1	7	11	21	11	19	22	4	12	6
Week 2	3	5	8	9	15	20	10	2	5

Figure 3.8: Applied mutation swap week operator

The process is repeated and stops when the termination criterion is met (in this work the termination criterion is iteration size). Fitness calculation of each chromosome involves adding up penalties for soft constraints in the SGRSP.

Genetic Algorithm process of the program can be seen from the flow chart on the next page.

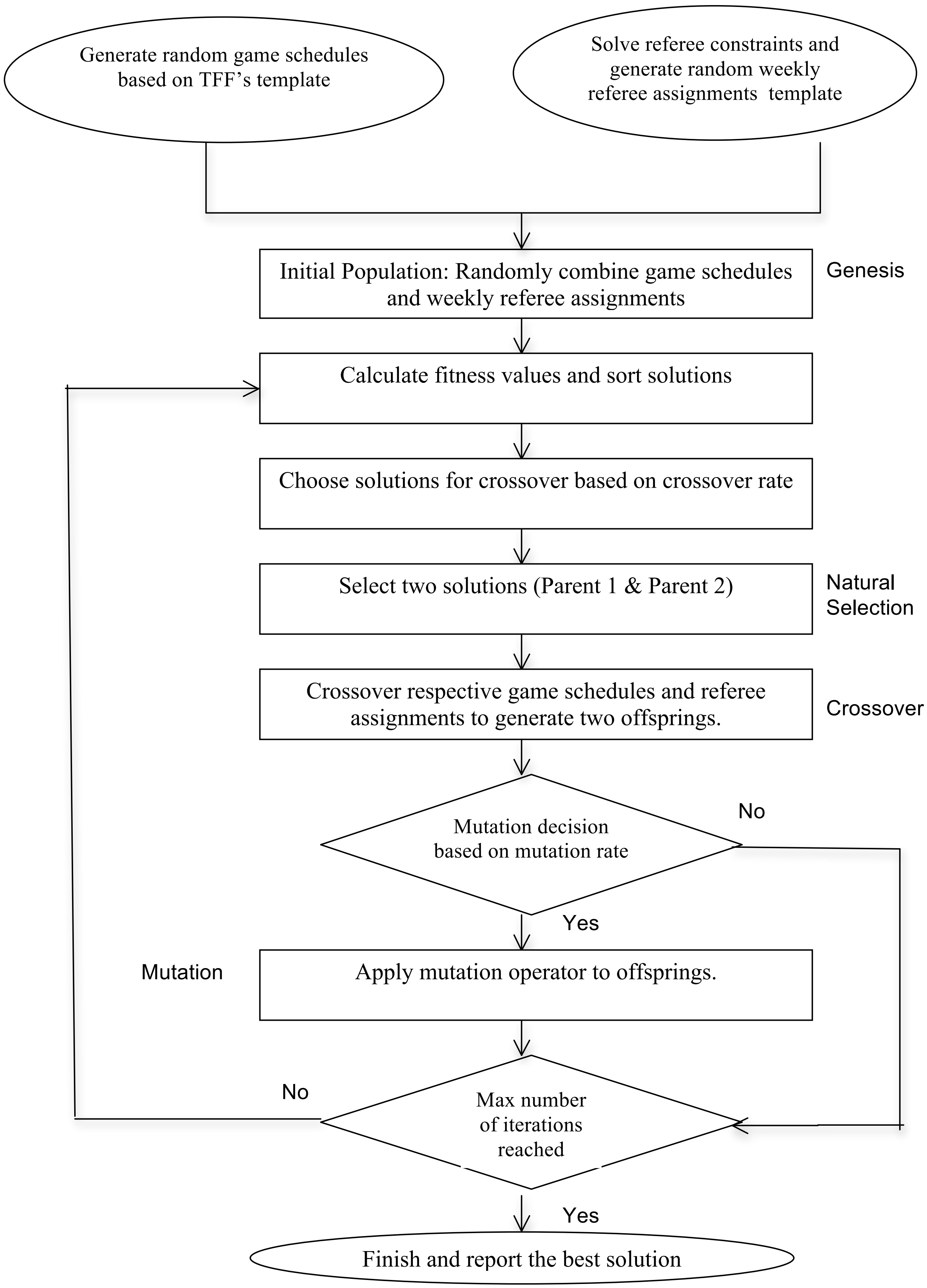

Figure 3.9: Flowchart of genetic algorithm for SGRSP

```
Read fixed TFF game template and referee assignment template
Calculate the fitness of the actual schedule and assignment.
for 1 to population size nSoln
        Generate random fixtures and referee assignments based on templates
and combine them
 end
for 1 to  iteration limit
        Calculate fitness values and sort fitness values increasingly.
        Select %xx of the population by using roulette wheel probabilities.
        for 1 to size of  crossover population
                Choose parents for crossover, generate new offspring by
                crossover
                Apply mutation operator(s) to new offsprings.
                Find fitness values for the mutated offsprings.
                Store offsprings in the  population.
        end
   end
   Report best solution
```

Figure 3.10: Pseudocode of genetic algorithm for SGRSP

Chapter 4

Experiments

The optimization model was solved with GAMS 23.2 using CPLEX. GA code was written with JAVA language and the experimental runs were performed on an Intel Core i7, 2.9 GHz, 8GB Ram computer.

4.1 Problem Instances

In our experiments, we used 2010-2013 fixture schedules and referee assignments in order to understand how soft constraints affect our solution. Also we would like to see how the three data sets change objective function values. The parameters used in the algorithm are chosen after different experiments as shown in Table 4.1. The population size is also chosen after many trials in order to find the best performing population size.

4.2 Preliminary Experiments

Different experiments are done with the 2010-2013 seasons' data in order to find the best performing population size, iteration, mutation rate and crossover rates. Parameters shown on Table 4.1 are experimented to find to best solutions. Datasets of the problem are given in Table 4.1 according to season.

Table 4.1 Parameters of the genetic algorithm

Selection Type	**Population Size**	**Season**	**Mutation Type**	**Iteration trials**	**Mutation Rate**	**Cross over Rate**
Roulette	250	2010-2011	Same Week	1500	1	1
Linear		2011-2012	Diff. Week	1000	0.8	0.8
		2011-2013	Swap Referee	750	0.6	0.6
				500	0.4	0.4
					0.2	0.2

As we look at the experiment results on our three season's template 250 initial solutions and 1,500 iterations give better results compared to other experiments. As iteration size increased; objective function values decreased incrementally as we see from the tables. We also saw that improvements nearly stopped after 1,500 iterations.

Table 4.2 Datasets

Season	**2010-2011**	**2011-2012**	**2012-2013**
# of weeks	34	34	34
# of games	306	306	306
# of referees	26	26	28
# of teams	18	18	18

Table 4.3 Name and value of penalties

Name of Penalty	Value of Penalty
penRefConsec	4
penSymmetry	8
penRatingScore	4
penRefMaxWeeks	2
penRefMinWeeks	2
penMaxTeamGames	**4**
penConsecTop	8
penConsecHome	8
penRange	8
penSame	8
penTopAll	8
penRefMaxDerby	8
penRefConsecTeam	8

Table 4.4 Name and value of parameters

Name of Parameter	Value of Parameter
nRefMaxConsec	3
nRefMaxWeeks	11 or 12 (306/# of referees)
nRefMinWeeks	1
nMaxTeamGames	5
nRefMaxDerby	2

Referee ratings are given according to the referee's skill level. There are two types of referees that are upper classified referee and FIFA referee. FIFA referees are graded as 8 whereas upper classified referees are graded as 5.

4.2 Numerical Experiments

All reported numbers are averages of 10 runs of each experiment for the same parameter values. All experiments are done by the help of Java.

Table 4.5 2010-2011 season, sameweek mutation, roulette wheel objective values

SameWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	536	533	519	521	523
	0.4	516	530	522	512	513
	0.6	526	529	503	511	531
	0.8	517	526	538	514	487
	1	508	516	514	522	518

Table 4.6 2010-2011 season, diffweek mutation, roulette wheel objective values

DiffWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	539	514	524	540	524
	0.4	522	528	520	537	518
	0.6	517	510	537	527	504
	0.8	524	520	521	518	508
	1	520	508	514	520	515

Table 4.7 2010-2011 season, swapweek mutation, roulette wheel objective values

SwapWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	514	512	506	520	520
	0.4	525	505	506	509	513
	0.6	504	504	502	490	506
	0.8	491	489	520	503	499
	1	510	496	518	498	488

Table 4.8 2011-2012 season, sameweek mutation, roulette wheel objective values

SameWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	471	463	481	462	459
	0.4	473	477	465	460	470
	0.6	454	464	463	470	459
	0.8	460	459	468	469	459
	1	464	455	465	470	462

Table 4.9 2011-2012 season, diffweek mutation, roulette wheel objective values

DiffWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	480	481	481	474	465
	0.4	469	465	476	474	481
	0.6	475	462	457	470	475
	0.8	471	440	464	462	474
	1	466	457	465	461	464

Table 4.10 2011-2012 season, swapweek mutation, roulette wheel objective values

SwapWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	464	464	472	462	458
	0.4	461	452	464	463	447
	0.6	464	458	447	456	450
	0.8	457	460	448	456	441
	1	450	451	445	444	444

Table 4.11 2012-2013 season, sameweek mutation, roulette wheel objective values

SameWeek Mutation-Roulette		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	451	446	436	445	452
	0.4	436	441	446	440	432
	0.6	438	442	442	436	436
	0.8	445	429	442	438	445
	1	448	434	441	446	438

Table 4.12 2012-2013 season, diffweek mutation, roulette wheel objective values

DiffWeek Mutation-Roulette		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	441	448	449	447	443
	0.4	442	446	439	440	444
	0.6	448	449	442	452	450
	0.8	444	424	444	438	439
	1	449	439	440	450	434

Table 4.13 2012-2013 season, swapweek mutation, roulette wheel objective values

SwapWeek Mutation-Roulette		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	433	443	438	430	435
	0.4	428	435	441	428	427
	0.6	439	433	434	424	424
	0.8	433	425	437	427	418
	1	422	432	429	429	425

Table 4.14 2010-2011 season, sameweek mutation, linear rank objective values

SameWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	532	530	535	527	543
	0.4	538	535	528	492	520
	0.6	522	512	528	528	534
	0.8	528	522	500	512	501
	1	495	507	513	526	491

Table 4.15 2010-2011 season, diffweek mutation, linear rank objective values

DiffWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	539	528	536	522	528
	0.4	536	534	540	512	540
	0.6	528	519	527	511	514
	0.8	515	515	505	523	525
	1	512	506	518	527	523

Table 4.16 2010-2011 season, swapweek mutation, linear rank objective values

SwapWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	512	518	515	526	517
	0.4	518	518	528	502	504
	0.6	502	522	525	496	514
	0.8	510	524	517	488	510
	1	525	504	505	519	506

Table 4.17 2011-2012 season, sameweek mutation, linear rank objective values

SameWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	469	468	484	470	463
	0.4	476	468	468	472	470
	0.6	472	466	462	467	476
	0.8	470	464	476	466	452
	1	462	453	463	451	462

Table 4.18 2011-2012 season, diffweek mutation, linear rank objective values

DiffWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	480	474	474	471	477
	0.4	467	470	469	466	479
	0.6	466	473	473	473	482
	0.8	477	475	468	480	468
	1	476	470	468	459	464

Table 4.19 2011-2012 season,swapweek mutation, linear rank objective values

SwapWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	463	458	466	462	462
	0.4	472	456	460	460	454
	0.6	466	461	449	447	458
	0.8	460	456	454	445	448
	1	454	456	448	452	439

Table 4.20 2012-2013 season, sameweek mutation, linear rank objective values

SameWeek Mutation-Linear		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	445	445	443	445	442
	0.4	445	440	439	439	441
	0.6	458	461	434	446	440
	0.8	454	442	441	431	440
	1	449	454	442	433	441

Table 4.21 2012-2013 season, diffweek mutation, linear rank objective values

DiffWeek Mutation-Linear		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	450	455	450	447	450
	0.4	443	441	455	457	444
	0.6	444	442	445	448	445
	0.8	446	440	452	452	444
	1	451	440	442	437	436

Table 4.22 2012-2013 season, swapweek mutation, linear rank objective values

SwapWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	444	443	440	433	446
	0.4	445	442	437	428	437
	0.6	431	434	441	424	427
	0.8	434	430	426	429	426
	1	439	426	425	418	426

Table 4.23 2010-2011 season, sameweek mutation, roulette wheel best initial

SameWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	627	620	634	625	612
	0.4	620	621	634	605	621
	0.6	624	621	609	630	635
	0.8	616	626	631	613	632
	1	608	617	632	599	620

Table 4.24 2010-2011 season, diffweek mutation, roulette wheel best initial

DiffWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	612	610	605	629	616
	0.4	608	630	612	614	606
	0.6	625	608	632	615	606
	0.8	639	638	646	606	623
	1	621	605	602	621	611

Table 4.25 2010-2011 season, swapweek mutation, roulette wheel best initial

SwapWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	612	621	602	616	624
	0.4	607	616	610	602	604
	0.6	618	632	614	644	606
	0.8	631	614	604	623	627
	1	617	624	613	611	601

Table 4.26 2011-2012 season, sameweek mutation, roulette wheel best intial

SameWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	551	543	529	539	544
	0.4	528	528	535	523	530
	0.6	553	542	522	547	546
	0.8	541	536	542	538	540
	1	533	551	543	550	540

Table 4.27 2011-2012 season, diffweek mutation, roulette wheel best initial

DiffWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	533	542	562	538	536
	0.4	542	529	530	542	546
	0.6	556	546	529	520	567
	0.8	547	541	543	532	539
	1	534	546	540	545	544

Table 4.28 2011-2012 season, swapweek mutation, roulette wheel best initial

SwapWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	546	540	530	533	533
	0.4	536	538	544	553	546
	0.6	557	551	523	543	539
	0.8	550	548	538	534	532
	1	544	535	534	529	546

Table 4.29 2012-2013 season, sameweek mutation, roulette wheel best initial

SameWeek Mutation-Roulette		Mutation Rate				
2012-2013	Obj.Value	0.2	0.4	0.6	0.8	1
Cross over Rate	0.2	518	512	499	516	526
	0.4	500	504	530	519	523
	0.6	508	505	514	517	517
	0.8	504	506	507	511	516
	1	522	524	504	511	513

Table 4.30 2012-2013 season, diffweek mutation, roulette wheel best initial

DiffWeek Mutation-Roulette		Mutation Rate				
2012-2013	Obj.Value	0.2	0.4	0.6	0.8	1
Cross over Rate	0.2	510	522	500	512	519
	0.4	517	510	514	524	512
	0.6	521	510	509	521	498
	0.8	526	518	517	521	518
	1	512	512	524	518	512

Table 4.31 2012-2013 season, swapweek mutation, roulette wheel best initial

SwapWeek Mutation-Roulette		Mutation Rate				
2012-2013	Obj.Value	0.2	0.4	0.6	0.8	1
Cross over Rate	0.2	510	511	511	521	500
	0.4	520	509	511	511	520
	0.6	512	528	523	528	517
	0.8	508	520	507	511	519
	1	508	505	516	513	519

Table 4.32 2010-2011 season, sameweek mutation, linear rank best initial

SameWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	615	626	613	617	614
	0.4	615	627	610	618	604
	0.6	600	609	615	593	584
	0.8	611	609	626	592	606
	1	621	614	609	608	607

Table 4.33 2010-2011 season, diffweek mutation, linear rank best. initial

DiffWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	636	624	601	595	614
	0.4	605	622	610	621	592
	0.6	630	627	600	602	626
	0.8	624	608	585	628	624
	1	616	628	619	620	607

Table 4.34 2010-2011 season, swapweek mutation, linear rank best initial

SwapWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	598	606	618	628	644
	0.4	612	596	638	618	616
	0.6	616	626	621	627	620
	0.8	647	617	608	606	636
	1	620	615	630	614	614

Table 4.35 2011-2012 season, sameweek mutation, linear rank best initial

SameWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	545	546	532	551	540
	0.4	546	539	541	528	527
	0.6	538	545	546	533	536
	0.8	529	540	529	551	546
	1	535	544	558	542	549

Table 4.36 2011-2012 season, diffweek mutation, linear rank best initial

DiffWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	539	535	526	543	555
	0.4	533	532	550	538	550
	0.6	554	526	536	534	547
	0.8	538	541	538	549	548
	1	549	541	543	542	536

Table 4.37 2011-2012 season,swapweek mutation, linear rank best initial

SwapWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	549	548	530	542	529
	0.4	545	544	556	543	554
	0.6	531	554	540	538	535
	0.8	523	540	533	550	538
	1	543	539	539	556	531

Table 4.38 2012-2013 season, sameweek mutation, linear rank best initial

SameWeek Mutation-Linear		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	518	503	525	517	513
	0.4	516	514	496	510	517
	0.6	518	522	506	516	522
	0.8	525	501	514	519	524
	1	510	524	525	513	525

Table 4.39 2012-2013 season, diffweek mutation, linear rank best initial

DiffWeek Mutation-Linear		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	504	500	512	511	524
	0.4	508	501	519	485	501
	0.6	508	504	508	508	516
	0.8	510	518	529	522	514
	1	522	506	520	515	502

Table 4.40 2012-2013 season, swapweek mutation, linear rank best initial

SwapWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	522	515	522	524	520
	0.4	511	519	517	523	501
	0.6	513	511	514	519	517
	0.8	534	524	516	519	515
	1	528	518	518	507	512

Standard deviations in the experiments indicate that there may be a deviation of a few percentage points in the performance when running the algorithm

Table 4.41 2010-2011 season, sameweek mutation, roulette wheel standard deviation

SameWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	24	32	33	18	38
	0.4	32	34	25	42	29
	0.6	16	19	38	34	39
	0.8	25	30	40	17	23
	1	14	11	36	22	26

Table 4.42 2010-2011 season, diffweek mutation, roulette wheel standard deviation

DiffWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	34	40	23	37	40
	0.4	21	21	25	30	28
	0.6	37	41	23	38	31
	0.8	33	26	42	38	29
	1	37	18	27	22	40

Table 4.43 2010-2011 season, swapweek mutation, roulette wheel standard deviation

SwapWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	17	35	27	23	35
	0.4	33	37	17	39	25
	0.6	24	21	27	31	25
	0.8	33	32	26	16	21
	1	21	30	38	31	31

Table 4.44 2011-2012 season, sameweek mutation, roulette wheel standard deviation

SameWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	21	16	26	16	23
	0.4	9	16	19	10	16
	0.6	23	13	19	15	10
	0.8	12	13	18	19	13
	1	21	12	19	11	24

Table 4.45 2011-2012 season, diffweek mutation, roulette wheel standard deviation

DiffWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	20	18	14	12	16
	0.4	19	14	12	18	15
	0.6	15	22	17	18	18
	0.8	14	20	12	17	20
	1	13	22	15	22	11

Table 4.46 2011-2012 season, swapweek mutation, roulette wheel standard deviation

SwapWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	12	22	14	15	13
	0.4	20	19	9	18	12
	0.6	13	19	20	16	14
	0.8	14	15	17	18	12
	1	11	18	13	14	13

Table 4.47 2012-2013 season, sameweek mutation, roulette wheel standard deviation

SameWeek Mutation-Roulette		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	14	22	19	18	16
	0.4	9	13	15	26	20
	0.6	17	10	19	12	6
	0.8	16	18	23	10	15
	1	21	11	17	20	13

Table 4.48 2012-2013 season, diffweek mutation, roulette wheel standard deviation

DiffWeek Mutation-Roulette		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	16	16	16	22	14
	0.4	18	14	9	13	18
	0.6	14	9	11	10	20
	0.8	8	15	15	14	14
	1	16	12	21	6	17

Table 4.49 2012-2013 season, swapweek mutation, roulette wheel standard deviation

SwapWeek Mutation-Roulette		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over R C ate	**0.2**	18	20	14	10	18
	0.4	22	14	15	9	15
	0.6	15	13	15	14	12
	0.8	16	14	12	16	8
	1	13	20	10	13	10

Table 4.50 2010-2011 season, sameweek mutation, linear rank standard deviation

SameWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	29	25	30	26	35
	0.4	33	28	19	19	34
	0.6	25	15	37	20	32
	0.8	27	45	19	38	28
	1	20	32	19	25	30

Table 4.51 2010-2011 season, diffweek mutation, linear rank standard deviation

DiffWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	18	20	27	32	29
	0.4	23	42	33	36	33
	0.6	31	33	25	21	39
	0.8	32	26	20	22	34
	1	26	35	29	36	28

Table 4.52 2010-2011 season, swapweek mutation, linear rank standard deviation

SwapWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	29	39	30	39	31
	0.4	28	38	25	18	23
	0.6	32	25	38	22	25
	0.8	24	24	19	31	15
	1	22	30	29	28	34

Table 4.53 2011-2012 season, sameweek mutation, linear rank standard deviation

SameWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	25	19	19	22	16
	0.4	12	13	18	15	17
	0.6	23	19	21	17	21
	0.8	19	20	18	24	17
	1	19	23	10	13	18

Table 4.54 2011-2012 season, diffweek mutation, linear rank standard deviation

DiffWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	28	18	12	19	9
	0.4	25	11	11	17	23
	0.6	19	7	19	26	19
	0.8	14	19	17	12	16
	1	13	12	16	23	14

Table 4.55 2011-2012 season,swapweek mutation, linear rank standard deviation

SwapWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	7	21	16	12	18
	0.4	21	27	18	17	17
	0.6	12	12	10	17	17
	0.8	17	14	17	14	15
	1	16	18	11	15	13

Table 4.56 2012-2013 season, sameweek mutation, linear rank standard deviation

SameWeek Mutation-Linear		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	29	15	17	17	11
	0.4	17	18	13	12	18
	0.6	16	16	18	17	20
	0.8	10	22	14	18	26
	1	18	14	13	17	18

Table 4.57 2012-2013 season, diffweek mutation, linear rank standard deviation

DiffWeek Mutation-Linear		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	20	11	18	20	15
	0.4	25	20	14	11	18
	0.6	17	20	18	10	15
	0.8	15	12	11	18	17
	1	11	13	12	13	14

Table 4.58 2012-2013 season, swapweek mutation, linear rank standard deviation

SwapWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	16	11	15	16	9
	0.4	13	8	12	15	21
	0.6	15	17	14	17	10
	0.8	23	18	16	16	13
	1	17	13	14	19	16

Table 4.59 2010-2011 season,sameweek mutation, roulette wheel min objective values

SameWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	508	472	476	496	484
	0.4	468	484	472	456	460
	0.6	460	480	448	464	460
	0.8	500	472	464	488	456
	1	484	500	448	496	480

Table 4.60 2010-2011 season, diffweek mutation, roulette wheel min objective values

DiffWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	516	472	496	476	468
	0.4	496	488	476	496	480
	0.6	460	444	476	464	472
	0.8	476	480	456	464	456
	1	468	492	460	480	472

Table 4.61 2010-2011 season,swapweek mutation, roulette wheel min objective values

SwapWeek Mutation-Roulette		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	484	464	440	480	440
	0.4	484	444	472	472	476
	0.6	480	476	468	424	460
	0.8	416	432	464	480	472
	1	484	456	476	464	436

Table 4.62 2011-2012 season, sameweek mutation, roulette wheel min objective values

SameWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	436	424	432	448	412
	0.4	444	452	428	436	440
	0.6	404	440	428	448	440
	0.8	444	436	432	440	436
	1	448	428	432	440	412

Table 4.63 2011-2012 season, diffweek mutation, roulette wheel min objective values

DiffWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	456	456	420	456	432
	0.4	428	448	456	448	444
	0.6	452	432	436	436	440
	0.8	448	432	448	424	444
	1	432	432	436	432	452

Table 4.64 2011-2012 season,swapweek mutation,roulette min wheel objective values

SwapWeek Mutation-Roulette		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	448	420	436	412	436
	0.4	440	404	448	424	428
	0.6	448	428	416	420	432
	0.8	432	432	424	420	420
	1	420	424	420	428	424

Table 4.65 2012-2013 season,sameweek mutation, roulette wheel min objective values

SameWeek Mutation-Roulette		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	420	400	412	412	416
	0.4	432	416	412	368	388
	0.6	408	412	404	412	424
	0.8	428	400	392	420	428
	1	404	416	412	416	416

Table 4.66 2012-2013 season, diffweek mutation, roulette wheel min objective values

DiffWeek Mutation-Roulette		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	412	424	420	392	424
	0.4	420	420	428	420	404
	0.6	412	436	424	436	416
	0.8	432	408	432	416	420
	1	420	416	400	440	400

Table 4.67 2012-2013 season,swapweek mutation, roulette wheel min objective values

SwapWeek Mutation-Roulette		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	412	392	416	416	408
	0.4	380	408	408	416	408
	0.6	412	408	412	392	404
	0.8	416	400	416	392	408
	1	412	404	416	408	400

Table 4.68 2010-2011 season, sameweek mutation, linear rank min objective values

SameWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	472	496	460	488	496
	0.4	472	488	508	460	468
	0.6	488	484	488	484	508
	0.8	480	472	472	476	468
	1	464	476	480	472	440

Table 4.69 2010-2011 season, diffweek mutation, linear rank min objective values

DiffWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	520	488	460	460	468
	0.4	500	460	480	444	452
	0.6	484	460	500	476	460
	0.8	464	476	472	480	472
	1	456	472	480	480	492

Table 4.70 2010-2011 season, swapweek mutation, linear rank min objective values

SwapWeek Mutation-Linear		**Mutation Rate**				
2010-2011	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	476	448	464	456	476
	0.4	464	444	496	464	476
	0.6	468	496	472	460	472
	0.8	484	488	484	460	484
	1	492	468	440	468	468

Table 4.71 2011-2012 season, sameweek mutation, linear rank min objective values

SameWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	420	436	456	444	436
	0.4	452	448	440	456	444
	0.6	420	432	428	436	432
	0.8	452	424	444	404	424
	1	440	396	448	436	432

Table 4.72 2011-2012 season, diffweek mutation, linear rank min objective values

DiffWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	444	440	456	440	456
	0.4	424	452	456	440	440
	0.6	420	460	436	424	452
	0.8	456	440	444	452	444
	1	448	452	436	420	444

Table 4.73 2011-2012 season,swapweek mutation, linear rank min objective values

SwapWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	452	412	428	444	428
	0.4	424	404	432	428	432
	0.6	452	444	432	424	432
	0.8	436	440	420	424	416
	1	432	432	428	428	420

Table 4.74 2012-2013 season, sameweek mutation, linear rank min objective values

SameWeek Mutation-Linear		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	376	416	428	424	440
	0.4	424	416	436	424	424
	0.6	428	408	404	404	408
	0.8	428	384	424	400	420
	1	408	412	412	400	364

Table 4.75 2012-2013 season, diffweek mutation, linear rank min objective values

DiffWeek Mutation-Linear		**Mutation Rate**				
2012-2013	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	420	440	440	416	424
	0.4	396	404	404	448	408
	0.6	404	392	416	428	424
	0.8	424	428	428	408	416
	1	432	424	416	412	416

Table 4.76 2012-2013 season, swapweek mutation, linear rank min objective values

SwapWeek Mutation-Linear		**Mutation Rate**				
2011-2012	**Obj.Value**	**0.2**	**0.4**	**0.6**	**0.8**	**1**
Cross over Rate	**0.2**	424	428	416	396	404
	0.4	420	428	420	404	404
	0.6	412	404	420	396	404
	0.8	380	404	420	404	404
	1	404	404	384	388	404

4.3 Results

Table 4.62 summarizes our results. As can be seen from the results, actual scheduling results can be improved by using an analytical approach. Standard deviations in the experiments indicate that there may be a deviation of a few percentage points in the performance when running the algorithm. Furthermore, the results are sensitive to the chosen penalty coefficients.

Table 4.77 Min. objective function values of 2010-2013 seasons with roulette wheel

	Roulette		
Minimum Obj. Function	**2010-2011**	**2011-2012**	**2012-2013**
SameWeekMutation	487	454	429
DiffWeekMutation	504	440	424
SwapWeekMutation	488	441	418
Best Initial	599	520	498
Real Schedule	690	594	548
GAMS	682	607	556

Table 4.78 Min. objective function values of 2010-2013 seasons with linear rank

	Linear		
Minimum Obj. Function	**2010-2011**	**2011-2012**	**2012-2013**
SameWeekMutation	520	451	431
DiffWeekMutation	505	459	436
SwapWeekMutation	488	439	418
Best Initial	584	523	485
Real Schedule	690	594	548
GAMS	682	607	556

Table 4.79 Summary of min. objective function values of 2010-2013 seasons

Minimum Obj. Function	**2010-2011**	**2011-2012**	**2011-2012**
SameWeekMutation	487	451	429
DiffWeekMutation	504	440	424
SwapWeekMutation	488	439	418
Best Initial	584	520	485
Real Schedule	690	594	548
Genetic	487	439	418

Table 4.80 Improvement % with real schedule

	Roulette			**Linear**		
Improvement % with Real Schedule	**2010-2011**	**2011-2012**	**2012-2013**	**2010-2011**	**2011-2012**	**2012-2013**
SameWeekMutation	29%	24%	22%	25%	24%	21%
DiffWeekMutation	27%	26%	23%	27%	23%	20%
SwapWeekMutation	29%	26%	24%	29%	26%	24%

Table 4.81 Improvement % with best initial

	Roulette			**Linear**		
Improvement % Best Initial	**2010-2011**	**2011-2012**	**2012-2013**	**2010-2011**	**2011-2012**	**2012-2013**
SameWeekMutation	19%	13%	14%	11%	14%	11%
DiffWeekMutation	16%	15%	15%	14%	12%	10%
SwapWeekMutation	19%	15%	16%	16%	16%	14%

After many different trials, results show that crossover rate 1.0 and mutation rate 1 gave better results on the experiments. Further trials may be done via these metrics.
3 mutation types used in order to find the best performing one and concluded that MutationSwapWeek gave better results than MutationSameWeek. Moreover trials done via MutationSameWeek gave better results than MutationDiffWeek.

2010-2013 seasons' data were used in the experiments. Real data and our genetic algorithm results were compared and concluded that 25--30 percent improvement was observed.

Appendix 1: GAMS File for SGRSP (2010-2011)

Set i index sets of teams
/BJK,FB,GS,TRAB,KYS,BRS,IBB,BCS,KRB,ESK,GAZ,GNCB,KSMP,ANT,MIY,AKG,SVS,KNS/;

alias (i,j);

alias(i,jj);

Set w index set of week /1*34/;

alias(w,t);

Set TOP(i) index set of top teams /BJK,FB,GS/;

Set c index set of
cities/ISTANBUL1,ISTANBUL2,ANKARA,TRABZON,BURSA,GANTEP,KAYSERI,ESKISEHIR,KARABUK,MANISA,ANTALYA,SIVAS,KONYA,IZMIR/;

Set SAME(c,i) index set of cities-teams with more than one team /

ANKARA.(AKG,GNCB)
ISTANBUL1.(FB,BJK,GS)
ISTANBUL2.(IBB,KSMP)

/;

Set r index set of referees /

bulentyildirim,huseyingocek,aytekindurmaz,cuneytcakir,barissimsek,ozguryankaya,
halisozkahya,suleymanabay,mustafakamilabitoglu,suatarslanboga,tolgaozkalfa,
kuddusimuftuoglu,ilkermeral,serkancinar,bunyamingezer,denizcoban,firataydinus,
abdullahyilmaz,huseyinsabanci,yunusyildirim,koraygencerler,mustafailkercoskun,
hakanceylan,metekalkavan,cagataysahan,mustafaogretmenoglu

/

Set RANGE(w) index set of weeks in the predefined range where derby matches should be played /5,9,14/;
Parameter numTeams(c) /

ANKARA = 2
ISTANBUL1 = 3
ISTANBUL2 = 2

/;

Parameter PenBreak(i) penalty for consecutive home games of team i /

BJK=8
FB=8
GS=8
TRAB=8
KYS=8
BRS=8
IBB=8
BCS=8
KRB=8
ESK=8
GAZ=8
GNCB=8
KSMP=8
ANT=8
MIY=8
AKG=8
SVS=8
KNS=8
/;

Parameter ref(r) referee r's performance rating /

bulentyildirim=10
huseyingocek=10
aytekindurmaz=8
cuneytcakir=10
barissimsek=10
ozguryankaya=8
halisozkahya=10
suleymanabay=8
mustafakamilabitoglu=8
suatarslanboga=8
tolgaozkalfa=8
kuddusimuftuoglu=8
ilkermeral=8
serkancinar=8
bunyamingezer=8
denizcoban=8
firataydinus=10
abdullahyilmaz=8
huseyinsabanci=8
yunusyildirim=8
koraygencerler=8
mustafailkercoskun=8
hakanceylan=8
metekalkavan=10
cagataysahan=8
mustafaogretmenoglu=8

/;

Parameter rr(i,j) minimum referee rating needed to call home game of team i against team j /

FB.ANT=5
KRB.MIY=2
IBB.KYS=2
AKG.TRAB=5
BCS.BJK=6
GAZ.KSMP=5
ESK.GNCB=2
BRS.KNS=5
SVS.GS=5
MIY.AKG=2
KNS.ESK=3
GNCB.GAZ=4
KSMP.BCS=4
KYS.KRB=2
TRAB.FB=8
ANT.SVS=2
BJK.IBB=5
GS.BRS=7
SVS.BRS=4
AKG.KYS=2
BCS.GNCB=3
FB.MIY=5
KRB.BJK=5
IBB.KSMP=3
ESK.GS=5
ANT.TRAB=5
GAZ.KNS=5
KSMP.KRB=3
KYS.FB=5
BRS.ESK=4
GNCB.IBB=2
BJK.AKG=5
MIY.ANT=2
TRAB.SVS=5
KNS.BCS=4
GS.GAZ=7
FB.BJK=8
TRAB.MIY=5
ANT.KYS=2
SVS.ESK=2
AKG.KSMP=3
KRB.GNCB=2
IBB.KNS=3
BCS.GS=6
GAZ.BRS=6
KYS.TRAB=5
MIY.SVS=2
GS.IBB=5

GNCB.AKG=2
KSMP.FB=6
BJK.ANT=5
ESK.GAZ=4
BRS.BCS=5
KNS.KRB=3
FB.GNCB=5
IBB.BRS=4
MIY.KYS=2
AKG.KNS=3
KRB.GS=5
BCS.ESK=3
TRAB.BJK=8
ANT.KSMP=3
SVS.GAZ=4
GS.AKG=5
KSMP.TRAB=6
BRS.KRB=4
GNCB.ANT=2
BJK.MIY=5
KYS.SVS=2
GAZ.BCS=5
ESK.IBB=2
KNS.FB=6
KRB.ESK=2
KYS.BJK=5
ANT.KNS=3
FB.GS=8
AKG.BRS=4
IBB.GAZ=4
MIY.KSMP=3
TRAB.GNCB=5
SVS.BCS=3
ESK.AKG=2
KNS.TRAB=6
GS.ANT=5
GNCB.MIY=2
KSMP.KYS=3
BJK.SVS=5
BCS.IBB=3
GAZ.KRB=4
BRS.FB=7
KYS.GNCB=2
MIY.KNS=3
SVS.IBB=2
FB.ESK=5
AKG.GAZ=4
KRB.BCS=3
BJK.KSMP=6
TRAB.GS=8
ANT.BRS=4

BCS.AKG=3
ESK.ANT=2
KNS.KYS=3
GS.MIY=5
GNCB.BJK=5
KSMP.SVS=3
IBB.KRB=2
GAZ.FB=7
BRS.TRAB=7
FB.BCS=6
MIY.BRS=4
TRAB.ESK=5
AKG.IBB=2
KSMP.GNCB=3
BJK.KNS=6
KYS.GS=5
ANT.GAZ=4
SVS.KRB=2
KRB.AKG=2
IBB.FB=5
KNS.KSMP=4
GS.BJK=8
GNCB.SVS=2
BCS.ANT=3
GAZ.TRAB=7
ESK.MIY=2
BRS.KYS=4
FB.KRB=5
BJK.BRS=7
TRAB.BCS=6
AKG.SVS=2
KSMP.GS=6
KYS.ESK=2
MIY.GAZ=4
ANT.IBB=2
KNS.GNCB=3
IBB.TRAB=5
ESK.BJK=5
SVS.KNS=3
GS.GNCB=5
AKG.FB=5
KRB.ANT=2
BCS.MIY=3
GAZ.KYS=4
BRS.KSMP=5
FB.SVS=5
KYS.BCS=3
ANT.AKG=2
GNCB.BRS=4
KSMP.ESK=3
BJK.GAZ=7

MIY.IBB=2
TRAB.KRB=5
KNS.GS=6
ANT.FB=5
MIY.KRB=2
KYS.IBB=2
TRAB.AKG=5
BJK.BCS=6
KSMP.GAZ=5
GNCB.ESK=2
KNS.BRS=5
GS.SVS=5
AKG.MIY=2
ESK.KNS=3
GAZ.GNCB=4
BCS.KSMP=4
KRB.KYS=2
FB.TRAB=8
SVS.ANT=2
IBB.BJK=5
BRS.GS=7
BRS.SVS=4
KYS.AKG=2
GNCB.BCS=3
MIY.FB=5
BJK.KRB=5
KSMP.IBB=3
GS.ESK=5
TRAB.ANT=5
KNS.GAZ=5
KRB.KSMP=3
FB.KYS=5
ESK.BRS=4
IBB.GNCB=2
AKG.BJK=5
ANT.MIY=2
SVS.TRAB=5
BCS.KNS=4
GAZ.GS=7
BJK.FB=8
MIY.TRAB=5
KYS.ANT=2
ESK.SVS=2
KSMP.AKG=3
GNCB.KRB=2
KNS.IBB=3
GS.BCS=6
BRS.GAZ=6
TRAB.KYS=5
SVS.MIY=2
IBB.GS=5

AKG.GNCB=2
FB.KSMP=6
ANT.BJK=5
GAZ.ESK=4
BCS.BRS=5
KRB.KNS=3
GNCB.FB=5
BRS.IBB=4
KYS.MIY=2
KNS.AKG=3
GS.KRB=5
ESK.BCS=3
BJK.TRAB=8
KSMP.ANT=3
GAZ.SVS=4
AKG.GS=5
TRAB.KSMP=6
KRB.BRS=4
ANT.GNCB=2
MIY.BJK=5
SVS.KYS=2
BCS.GAZ=5
IBB.ESK=2
FB.KNS=6
ESK.KRB=2
BJK.KYS=5
KNS.ANT=3
GS.FB=8
BRS.AKG=4
GAZ.IBB=4
KSMP.MIY=3
GNCB.TRAB=5
BCS.SVS=3
AKG.ESK=2
TRAB.KNS=6
ANT.GS=5
MIY.GNCB=2
KYS.KSMP=3
SVS.BJK=5
IBB.BCS=3
KRB.GAZ=4
FB.BRS=7
GNCB.KYS=2
KNS.MIY=3
IBB.SVS=2
ESK.FB=5
GAZ.AKG=4
BCS.KRB=3
KSMP.BJK=6
GS.TRAB=8
BRS.ANT=4

AKG.BCS=3
ANT.ESK=2
KYS.KNS=3
MIY.GS=5
BJK.GNCB=5
SVS.KSMP=3
KRB.IBB=2
FB.GAZ=7
TRAB.BRS=7
BCS.FB=6
BRS.MIY=4
ESK.TRAB=5
IBB.AKG=2
GNCB.KSMP=3
KNS.BJK=6
GS.KYS=5
GAZ.ANT=4
KRB.SVS=2
AKG.KRB=2
FB.IBB=5
KSMP.KNS=4
BJK.GS=8
SVS.GNCB=2
ANT.BCS=3
TRAB.GAZ=7
MIY.ESK=2
KYS.BRS=4
KRB.FB=5
BRS.BJK=7
BCS.TRAB=6
SVS.AKG=2
GS.KSMP=6
ESK.KYS=2
GAZ.MIY=4
IBB.ANT=2
GNCB.KNS=3
TRAB.IBB=5
BJK.ESK=5
KNS.SVS=3
GNCB.GS=5
FB.AKG=5
ANT.KRB=2
MIY.BCS=3
KYS.GAZ=4
KSMP.BRS=5
SVS.FB=5
BCS.KYS=3
AKG.ANT=2
BRS.GNCB=4
ESK.KSMP=3
GAZ.BJK=7

IBB.MIY=2
KRB.TRAB=5
GS.KNS=6

/;

Parameter MaxDerbys max number of derbys any referee can call /2/;

Parameter MinWeeks minimum number of total weeks for any referee to call games /2/;

Parameter MaxWeeks maximum number of total weeks for any referee to call games /12/;

Parameter MaxTeamGames maximum number of games for any referee to call for one team /4/;

Parameter PenRange penalty of assigning top teams outside of the desired week range /8/;

Parameter PenSame penalty of assigning no or two home games during the same week to the teams from the same city /8/;

Parameter PenMaxWeeks penalty of assigning a referee to more than MaxWeeks games /2/;

Parameter PenMinWeeks penalty of assigning a referee to less than MinWeeks games /2/;

Parameter PenRating penalty of assigning referee to a game with not adequate rating /4/;

Parameter PenTopTwice penalty of playing top teams consecutively /8/;
Parameter PenGameTwice penalty of calling the same game twice /8/;

Parameter PenMaxTeamGames penalty of calling more than a certain number of games for the same team/ 8/;

Parameter PenCallBacktoBack penalty for calling same team's matches consecutively /8/;

Parameter PenMaxDerbys penalty for calling too many derbys /4/;

Parameter PenRest(r,w) penalty for having a 4-week stretch starting at week w;

PenRest(r,w) = 3;

Binary Variable x(i,j,w) 1 if team i plays home against j in week w 0 otherwise;

Binary Variable y(i,w) 1 if team i plays a consecutive home game against team j in round(week) w 0 otherwise;

Binary Variable m(i,j,w,r) 1 if referee is assigned to game that team i plays a home game against team j in week w 0 otherwise;

Positive Variable dBreak(i) number of consecutive home games scheduled for team i;

Positive Variable dRange number of top games scheduled outside of the desired week range;

Positive Variable dSamePlus(w,c) 1 if no home games are scheduled during week w for teams(a pair) from city c;

Positive Variable dSameMinus(w,c) 1 if teams from city c play at home during week w;

Positive Variable dMaxWeeks(r) number of weeks referee r calls less than the desired maximum number of weeks;

Positive Variable dMinWeeks(r) number of weeks referee r calls more than the desired minimum number of weeks;

Positive Variable dRating (w,i,j) additional rating needed for the assigned referee to call a home game against team j in week w;

Positive Variable dRest(r,w) 4-week stretch for referee r starting at week w - 1 or 0;

Positive Variable dTopTwice(i,j,jj,w) 1 if team i plays TOP teams j and jj consecutively starting week w - 1 or 0;

Positive Variable dGameTwice(i,j,r,w) 1 if referee r calls same teams' matches in both halves of the season;

Positive Variable dMaxTeamGames(i,r) number of games referee r calls for team i in addition to the max matches;

Positive Variable dCallBacktoBack(i,r,w) 1 if referee r calls same team's matches consecutively in week w and w+1;

Positive Variable dMaxDerbys(r) number of derbys called by the same referee above max derbys;

Variable z Objective function;

Equation OF minimize penalties;

Equation eachteamshouldplayagainsteachother(i,j);

Equation eachteamshouldplayonegameeachweek(i,w);

Equation symmetryoftheleaguefixture(i,j,w);

```
Equation eachteamshouldnotplayconsecutivehomegames(i,w);

Equation nobreakforeachteamifpossible(i);

Equation normalteamscannotplaybacktobackwithtopteams(i,j,jj,w);

Equation allsamecityteamscannotplayhomegame(c,w);

Equation topteamsshouldnotplayoutsidedefinedrange;

*Equation topgamesshouldnotbeforthefirstfourweeks(i,j); redundant because of range


Equation arefereeshouldbeassignedtoeveryplayedgame(i,j,w);

Equation eachrefereeshouldbeassignedtoatmost1gameperweek (r,w);

Equation numberofmatchesassignedtoarefereecannotbeunderminweeks(r);

Equation numberofmatchesassignedtoarefereecannotexceedmaxweeks(r);

Equation eachrefereeshouldbeassignedaccordingtoitsskilllevel (i,j,w);

Equation refereecanbeassignedtomaximum3gamesin4consecutiveweeks(r,w);

Equation refereeshouldnotcallagametwice(i,j,r,w);

Equation refereecannotcalllotsofsameteamgames(i,r);

Equation refereecannotcallsameteamsgamesbacktoback(i,r,w);

Equation refereecannotcallmanyderbys(r);


OF..

z =e= sum((i),2*PenBreak(i)*dBreak(i)) +  PenRange*dRange +
sum((w,c),PenSame*(dSamePlus(w,c) + dSameMinus(w,c))) +
sum(r,PenMinWeeks*dMinWeeks(r)) + sum(r,PenMaxWeeks*dMaxWeeks(r)) +
sum((w,i,j),PenRating*dRating(w,i,j)) + sum((r,w),PenRest(r,w)*dRest(r,w)) +
sum((i,j,jj,w),2*PenTopTwice*dTopTwice(i,j,jj,w)) +
sum((i,j,r,w),PenGameTwice*dGameTwice(i,j,r,w)) +
sum((i,r),PenMaxTeamGames*dMaxTeamGames(i,r)) +
sum((i,r,w),PenCallBacktoBack*dCallBacktoBack(i,r,w)) +
sum((r),PenMaxDerbys*dMaxDerbys(r));
```

```
eachteamshouldplayagainsteachother(i,j)$(ord(i) < ord(j))..

sum(w$(ord(w) le 17),x(i,j,w) + x(j,i,w)) =e= 1;


eachteamshouldplayonegameeachweek(i,w)$((ord(w) le 17))..

sum(j$(ord(j) <> ord(i)),x(i,j,w) + x(j,i,w)) =e= 1;


symmetryoftheleaguefixture(i,j,w)$((ord(i) <> ord(j)) and (ord(w) le 17))..

x(i,j,w) =e=  x(j,i,w+17);



eachteamshouldnotplayconsecutivehomegames(i,w)$((ord(w) ne 17) and (ord(w) <
34))..
sum(j$(ord(j) ne ord(i)),x(i,j,w) + x(i,j,w+1)) - y(i,w) =l= 1;

nobreakforeachteamifpossible(i)..

sum(w,y(i,w)) - dBreak(i) =l= 0;



normalteamscannotplaybacktobackwithtopteams(i,j,jj,w)$((ord(i)<>ord(j)) and (ord(i)
<> ord(jj)) and (ord(j) <> ord(jj))

and (not TOP(i)) and (TOP(j) and TOP(jj)) and (ord(w) < 17))..

x(i,j,w) + x(i,jj,w+1) + x(j,i,w) + x(jj,i,w+1) - dTopTwice(i,j,jj,w) =l= 1;

allsamecityteamscannotplayhomegame(c,w)..

sum((i,j)$(SAME(c,i) and (not SAME(c,j)) and (ord(j) ne ord(i))),x(i,j,w)) -
dSameMinus(w,c) + dSamePlus(w,c) =e= numTeams(c) - 1;

topteamsshouldnotplayoutsidedefinedrange ..

sum(i$TOP(i),sum(j$(TOP(j) and ord(j) <> ord(i)),sum(w $(not
Range(w)),(x(i,j,w))))) - dRange =e= 0 ;

*topgamesshouldnotbeforthefirstfourweeks(i,j)$(ord(i) <= 3 and ord(j) <= 3 and ord(i)
<> ord(j))..

*sum(w$(ord(w)<=4),x(i,j,w)) =e= 0;


arefereeshouldbeassignedtoeveryplayedgame(i,j,w)$(ord(i)<>ord(j))..
```

```
sum(r,m(i,j,w,r)) =e= x(i,j,w);

eachrefereeshouldbeassignedtoatmost1gameperweek(r,w)..

sum((i,j)$(ord(i) <> ord(j)),m(i,j,w,r)) =l= 1;

numberofmatchesassignedtoarefereecannotbeunderminweeks(r)..

sum((i,j,w)$(ord(i) <> ord(j)),m(i,j,w,r)) +  dMinWeeks(r) =g= MinWeeks;

numberofmatchesassignedtoarefereecannotexceedmaxweeks(r)..

sum((i,j,w)$(ord(i) <> ord(j)),m(i,j,w,r)) -  dMaxWeeks(r) =l= MaxWeeks;

eachrefereeshouldbeassignedaccordingtoitsskilllevel (i,j,w)$(ord(i) <> ord(j))..

sum(r,ref(r)*m(i,j,w,r)) + dRating(w,i,j) - rr(i,j)*x(i,j,w) =g= 0;

refereecanbeassignedtomaximum3gamesin4consecutiveweeks(r,w)$(ord(w) le 31)..

sum((i,j,t)$((ord(t) ge ord(w)) and (ord(t) le ord(w)+3) and(ord(i) <> ord(j))),
m(i,j,t,r)) - dRest(r,w) =l= 3;


refereeshouldnotcallagametwice(i,j,r,w)$((ord(w) le 17) and ord(i) <> ord(j))..

m(i,j,w,r) + m(j,i,w+17,r) - dGameTwice(i,j,r,w) =l= 1;

refereecannotcalllotsofsameteamgames(i,r)..

sum((j,w)$(ord(j) <> ord(i)),m(i,j,w,r) + m(j,i,w,r)) - dMaxTeamGames(i,r) =l=
MaxTeamGames;

refereecannotcallsameteamsgamesbacktoback(i,r,w)$((ord(w) ne 17) and (ord(w) <
34))..

sum(j$(ord(j) <> ord(i)),m(i,j,w,r) + m(j,i,w,r) + m(i,j,w+1,r) + m(j,i,w+1,r)) -
dCallBacktoBack(i,r,w) =l= 1;

refereecannotcallmanyderbys(r)..

sum((i,j,w)$((ord(i) ne ord(j)) and TOP(i) and TOP(j)),m(i,j,w,r)) - dMaxDerbys(r)
=l= MaxDerbys;

model assignment /all/ ;

assignment.reslim=5;

assignment.iterlim=1;

assignment.optcr = 0.0;
```

```
assignment.limrow = 500;

option mip=cplex;

solve assignment using mip minimizing z;
```

Appendix 2: GAMS File for RAP

Sets

g games /1*9/
w weeks /1*34/
r referees /1*26/
alias(w,t);

Parameter PenMaxWeeks penalty of assigning a referee to more than MaxWeeks games /2/;
Parameter PenMinWeeks penalty of assigning a referee to less than MinWeeks games /2/;
Parameter MinWeeks minimum number of total weeks for any referee to call games /2/;
Parameter MaxWeeks maximum number of total weeks for any referee to call games /12/;
Parameter PenRest(r,w) penalty for having a 4-week stretch starting at week w;
PenRest(r,w) = 3;

Binary Variable m(g,w,r) 1 if referee is assigned to game g in week w 0 otherwise ;

Positive Variable dMaxWeeks(r) number of weeks referee r calls less than the desired maximum number of weeks;

Positive Variable dMinWeeks(r) number of weeks referee r calls more than the desired minimum number of weeks;

Positive Variable dRest(r,w) 4-week stretch for referee r starting at week w - 1 or 0;

Variable z Objective function;

Equation OF minimize penalties;

Equation arefereeshouldbeassignedtoeveryplayedgame(g,w);

Equation eachrefereeshouldbeassignedtoatmost1gameperweek (r,w);

Equation numberofmatchesassignedtoarefereecannotbeunderminweeks(r);

Equation numberofmatchesassignedtoarefereecannotexceedmaxweeks(r);

Equation refereecanbeassignedtomaximum3gamesin4consecutiveweeks(r,w);

```
OF..

z =e=sum(r,PenMinWeeks*dMinWeeks(r)) + sum(r,PenMaxWeeks*dMaxWeeks(r))
+ sum((r,w),PenRest(r,w)*dRest(r,w));

numberofmatchesassignedtoarefereecannotbeunderminweeks(r)..

sum((g,w),m(g,w,r)) +  dMinWeeks(r) =g= MinWeeks;

numberofmatchesassignedtoarefereecannotexceedmaxweeks(r)..

sum((g,w),m(g,w,r)) -  dMaxWeeks(r) =l= MaxWeeks;

refereecanbeassignedtomaximum3gamesin4consecutiveweeks(r,w)$(ord(w) le 31)..

sum((g,t)$((ord(t) ge ord(w)) and (ord(t) le ord(w)+3)), m(g,t,r)) - dRest(r,w) =l= 3;

arefereeshouldbeassignedtoeveryplayedgame(g,w)..

sum(r,m(g,w,r)) =e= 1;

eachrefereeshouldbeassignedtoatmost1gameperweek(r,w)..

sum(g,m(g,w,r)) =l= 1;

model assignment /all/ ;

assignment.reslim=50000;

assignment.iterlim=10000;

assignment.optcr = 0.0;

assignment.limrow = 5000;

option mip=cplex;

solve assignment using mip minimizing z;

file out /refassign.txt/; out.pc =5 ;

put out;

put 'Mgwr'/;

loop((g,w,r),put$(m.l(g,w,r)>0) g.tl,w.tl,r.tl,m.l(g,w,r)/);

put /;

putclose out;
```

References

[1] Alarcón, F., Durán, G. and Guajardo, M. 2013, *Referee assignment in the Chilean football league using integer programming and patterns*. International Transactions in Operational Research. doi: 10.1111/itor.12049.

[2] Bartsch, T., Drexl, A., Kroeger, S., 2006. *Scheduling the professional soccer leagues of Austria and Germany*. Computers and Operations Research 33 (7), 1907–1937.

[3] Ribeiro, C. C., 2012. *Sports scheduling: Problems and applications*. International Transactions in Operational Research (19) , 201–226.

[4] Croce, F. D., Oliveri, D., 2006. *Scheduling the Italian football league: An ILP-based approach.* Computers and Operations Research 33 (7), 1963– 1974.

[5] Goossens D., Spieksma, F., 2009. *Scheduling the Belgian Soccer League*, Interfaces 39:2, 109-118.

[6] Duarte A., Ribeiro C. C., Urrutia S., 2006. *Referee assignment in sports tournaments. In: Proceedings of the sixth international conference on the practice and theory of automated timetabling,* PATAT'06 Proceedings of the 6th international conference on Practice and theory of automated timetabling 5-7.

[7] Durán, G., Guajardo, M., Miranda, J., Sauré, D., Souyris, S., Weintraub, A., 2007. *Scheduling the Chilean soccer league by integer programming.* Interfaces 37 (6), 539–552.

[8] Durán, G., Guajardo, M., Wolf-Yadlin, R., 2012. *Operations Research Techniques for Scheduling Chile's Second Division Soccer League.* Interfaces 42 (3), 273–285.

[9] Fiallos, J., Pérez, J., Sabillón, F., Licona, M., 2010. *Scheduling soccer league of Honduras using integer programming.* In: Johnson, A., Miller, J. (Eds.), Proceedings of the 2010 Industrial Engineering Research Conference.

[10] Butenko, S., Gil-Lafuente, J. and Pardalos, P., 2003. *Economics, management and optimization in sports,* Springer, Berlin, 101–120.

[11]Kendall, G., 2008. *Scheduling English football fixtures over holiday periods,* Journal of the Operational Research Society 59, 743-755

[12] Kostuk, K., Willoughby, K., 2012. *A decision support system for scheduling the Canadian Football League*, Interfaces 42 (3), 286–295.

[13] Zäpfel, G., Braune, R., Bögl, M., 2010.*Metaheuristic Search Concepts.* Springer, Austria, ISBN 978-3-642-11342-0,121-143.

[14] Yavuz, M., Inan, H., Fığlalı, A.,2008. *Fair referee assignments for professional football leagues.* Computers & Operations Research 35 p. 2937 – 2951

[15] Rasmussen, F., 2008. *Scheduling a triple round-robin tournament for the best Danish soccer league.* European Journal of Operational Research 185 (2), 795–810.

[16] Rasmussen, F., Trick, M., 2008. Round robin scheduling - A survey. European Journal of Operational Research 188 (3), 617–636.

[17] Ribeiro C, Urrutia S. *Scheduling the Brazilian soccer championship. In: Proceedings of the sixth international conference on the practice and theory of automated timetabling,* PATAT 2006, Brno, Czech Republic, 2006, forthcoming. Umpires and the Traveling Umpire Problem , Informs 2012 , Interfaces 42(3), p. 232–244

[18] Silva, C., Garcia, E., Saliby, E., 2002. *General applications 1: soccer championship analysis using monte carlo simulation.* In: WSC '02: Proceedings of the 34th conference on Winter simulation, Winter Simulation Conference; p. 2011–16.

[19] *http://www.tff.org/default.aspx?pageID=545*

Curriculum Vitae

Olgu Pelin Hüseyinığlu was born in 13 July 1988, ın İstanbul. She received his B.S degree in Industrial Engineering in 2010 from Işık Univesity. Her research interest includes operations research, optimization, genetic algorithms and data mining.