2009

Time Relaxed Round Robin Tournament and the NBA Scheduling Problem

Renjun Bao
Cleveland State University

TIME RELAXED ROUND ROBIN TOURNAMENT AND THE NBA SCHEDULING PROBLEM

RENJUN BAO

Bachelor of Engineering in Mechanical Engineering

HeFei University of Technology

June, 1998

Master of Science in Industrial Engineering

Cleveland State University

May, 2006

Submitted in partial fulfillment of requirement for the degree
DOCTOR OF ENGINEERING IN INDUSTRIAL ENGINEERING

At the

CLEVELAND STATE UNIVERSITY
DECEMBER, 2009

This dissertation has been approved

for the Department of Mechanical Engineering

and the College of Graduate Studies by

_____

Dissertation Chairman Dr. L. Kenneth Keys, Mechanical Engineering Date

_____

Dr. John L. Frater, Mechanical Engineering Date

_____

Dr. Wenbing Zhao, Electrical and Computer Engineering Date

_____

Dr. Hanz Richter, Mechanical Engineering Date

_____

Dr. Walter O. Rom, Operations and Supply Chain Management Date

_____

Dr. Joseph A. Svestka, Mechanical Engineering Date

ACKNOWLEDGEMENTS

First, I thank my advisors Dr. Joseph A. Svestka and Dr. L. Kenneth Keys. After I finished my master, I was struggling to find a doctoral research topic. There were moments I thought I would have to give up my pursuit of my doctoral degree. It was Dr. Svestka that inspired me of the idea of working on sports scheduling problem after he learned I was doing some work with the NBA, which eventually became my doctoral research topic.

Dr .Keys generously agreed to serve as the academic advisor for my graduate study. Without their help, it was impossible for me to start this doctoral research. Throughout my study, Dr. Keys and Dr. Svestka were always there whenever I needed any help. Dr. Svestka showed me how to be a good scholar, good teacher, and good person. Dr. Keys used his expertise in project management to help me with controlling the dissertation timeline. I cannot imagine how I could finish the work without their guidance, understanding, patience, and friendship.

Besides my advisors, I would like to thank the rest of my dissertation committee: Dr. John L. Frater, Dr. Wenbing Zhao, Dr. Walter O. Rom, and Dr. Hanz Richter. Their accessibility, encouragement, and insightful suggestions are invaluable. Dr. Saini Yang resigned from CSU just before I finished the dissertation but I still appreciate her help. Dr. Hanz Richter generously agreed to serve on my committee with short notice.

I would like to thank the Department of the Industrial and Manufacturing Engineering, now Mechanical Engineering at Cleveland State Universities. I met some great faculty

members and friends here, especially Dr. Nayfeh and Dr. Thomas. All of you have made my life here much easier and enjoyable.

My thanks go out to Dr. Michael Trick from Carnegie Mellon University. He kindly educated me on the sports scheduling. Meanwhile he gave me great suggestions on my dissertation research. His passion for sports scheduling and operation research inspired me a lot.

My sincere gratitude goes to Mr. Matt Winick, who is in charge of the NBA scheduling. He was kind to share all the NBA scheduling requirements with me. Also I will use this opportunity to thank Mr. Todd Harris from the NBA, who introduced me to Mr. Matt Winick. And I would like to thank Cleveland Cavaliers general manager Danny Ferry and his staff who provided insightful thoughts on the evaluation of the scheduling.

Last but not least, I will thank my family. My beautiful wife Qiao Zheng supports me all the time. Her support, encouragement, and understanding love is the ground my life has been built on in the last few years. I also have to thank my lovely three year old daughter Deja Bao, who is good at making me busier by asking me to play puzzles with her all the time. Her laughing, crying, and smiling have made my life more enjoyable during my stressful graduate study. I also thank my mother, who lives in a small town in China. She gave me my life to this world and went through tough time for me. My father passed away when I was an undergraduate freshman. I knew he wanted me to finish my dissertation and get my doctoral degree, that's why he named me "Liu Yang", which means "oversea education", when I was born in a small village in Northern China. I thank my father inspired me to go to the United States. His spirit helped me to fight through the hard time in the last fifteen years.

TIME RELAXED ROUND ROBIN TOURNAMENT AND THE NATIONAL BASKETBALL ASSOCIATION SCHEDULING PROBLEM

RENJUN BAO

ABSTRACT

This dissertation study was inspired by the National Basketball Association regular reason scheduling problem. NBA uses the time-relaxed round robin tournament format, which has drawn less research attention compared to the other scheduling formats. Besides NBA, the National Hockey League and many amateur leagues use the time-relaxed round robin tournament as well.

This dissertation study is the first ever to examine the properties of general time-relaxed round robin tournaments. Single round, double round and multiple round time-relaxed round robin tournaments are defined. The integer programming and constraint programming models for those tournaments scheduling are developed and presented. Because of the complexity of this problem, several decomposition methods are presented as well.

Traveling distance is an important factor in the tournament scheduling. Traveling tournament problem defined in the time constrained conditions has been well studied. This dissertation defines the novel problem of time-relaxed traveling tournament problem. Three algorithms has been developed and compared to address this problem.

In addition, this dissertation study presents all major constraints for the NBA regular season scheduling. These constraints are grouped into three categories: structural, external and fairness. Both integer programming and constraint programming are used to model these constraints and the computation studies are presented.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF MODELS

LIST OF DEFINITIONS

CHAPTER I

INTRODUCTION

1.1 The problem

In many countries, sports provide income besides entertainment. Maximizing the revenues for a sports league is in the best interest of not only the teams, but also the local communities. There are many factors that could affect league revenues, one of them is the schedule.

Most of the professional sports leagues let the central administrative offices make the schedules. There are some fundamental features of a schedule. The foremost is the structure. In modern days, most leagues use a round robin tournament schedule format which means each team play every other team for a fixed number of times during a time span, which is called a round. The round robin tournament schedules can be divided into two broad types: time constrained schedules and time relaxed schedules. In time constrained schedules, the number of available game slots is equal to necessary game slots. The time constrained schedules are used by many leagues, including most college basketball conferences, professional soccer leagues in the Europe and the South America. In the time relaxed schedules, the time of available game slots is bigger than the necessary number. It are used by a few leagues, the National Basketball Association

(NBA) and the National Hockey League (NHL) in North America are two examples. The NBA is the most popular professional basketball league in North America. This dissertation work was inspired by the scheduling problem for the NBA regular season.

Although this dissertation work is inspired by the NBA scheduling, the time relaxed schedule is widely used in many other leagues, especially amateur sports leagues. As a result of the limited arena and player availability, time relaxed round robin tournament format is the only choice for most amateur leagues.

To schedule a time relaxed round robin tournament is not a trivia task. Besides the basic structure requirement, there are many other constraints required for the scheduling. It is not uncommon to have conflicting requirements from individual teams because each team has its own interest. On top of that, there are some external factors that play an important role in the scheduling. For example, NBA's television network partner turner wants to exclusively reserve popular games for the prime time. However, local television networks want to broadcast local team's game during prime time as well. It is not an easy task to balance the conflicting interests.

Although there have been many research studies done on the sports scheduling in recent years, few of them focuses on the time relaxed round robin tournament. The research on the time relaxed round robin tournament will be an important building block for the future research. For example, an understanding of the feasibility of pattern sets and computation studies could lead to a better constructive method in the future.

To reduce the fatigue and save traveling time, it is important to minimize the total traveling distance for the league. It is even becoming more important for the NBA teams

because of the economic condition. This dissertation research defines and provides algorithms to tackle the traveling tournament problem in the time relaxed context.

It is a daunting task to schedule the regular season games for the NBA because of all those involved constraints. For the time being, most of the work is done manually and the use of the information technology is limited. Compared to other leagues, there are many constraints are unique in the NBA scheduling problem, which is unknown to the academic world. This dissertation study will outline most major constraints for the NBA scheduling problem.

1.2 Research objectives and academic contribution

The research objectives are threefold:

First, is to examine the properties of the time relaxed round robin tournament problem structure. Except few studies on the practical applications, there has no study been done on the general time relaxed round robin tournament problem to our best knowledge. The understanding of the properties, especially the decomposition methods, will be important building block for both practical application and future research. Besides that, this dissertation work will present the computation study of the integer programming and constraint programming model formulation for different kinds of time relaxed round robin tournaments.

The second, is to define and tackle the time relaxed traveling tournament problem. To reduce the traveling distance is in best interest for all teams. Traveling Tournament Problem (TTP) draw a lot of research interest in the last ten years. This problem posts new challenges if put in the time relaxed context. This dissertation study will define the

Time Relaxed Round Robin Tournament problem (TRTTP) and several benchmark problems are given. Although it is unrealistic to develop an algorithm to get an optimal solution for a league of 30 teams, it is still important to get the optimal solution for league of small size. For example, a league of 5 teams could be modeled as a division. In this dissertation study, an algorithm to solve the problems of 6 team cases to optimality will be presented.

The third objective is to model all the essential constraints that arise from the NBA scheduling problem. Although the previous studies on sports league scheduling have considered many constraints, there are still many constraints faced by the NBA have never been mentioned in other literatures. This dissertation will be the first ever to model these constraints.

1.3 Basic Terminology

The terminology used in the other sports scheduling literature is not consistent. We will use the terminology stated in the following through the paper unless marked explicitly.

We assume there are n, n even, teams in a sports league. The assumption about even will not lost any generality because we can add a dummy team if n is odd. A team play against the dummy team will have a *bye* at that time slot. We use letter T to represent the set of teams.

In this work we focus on the *round robin tournament* formulation, where all teams meet all other teams a fixed k number of times. If k is one, we call it a *single round robin tournament*; if k is two, we call it a *double round robin tournament*; if k is three, we

call it a *triple round robin tournament.* In the *double round robin tournament*, if every team has the same opponent in the second round as those in the first round but the venue is reversed, we call it *mirrored round robin.*

Competition between one team against another team is called a *game*. A game is carried out in exactly one time period *p* out of set *P.* In this work we use day as the time period. We use letter *D* to represent the set of days. If there are more available time slots than the games, it is a *time relaxed round robin tournament.* On the other hand, in a *time constrained round robin tournament* the number of the available time slots is equal to the games plus necessary byes.

We assume each team has its own venue. A game has to be carried out in exactly one of the two competitors' venue. A team plays a *home game* if the game is played in its own venue, an *away game* if the game is played in the opponent's venue. Sometimes team venue will be referred to as team arena. A team has a *bye*, or *off day* if it does not have any game on that day. When two teams play together, one will play home game and the other will play away game. We use *H*, *A*, *O* to note home games, away games, and off days respectively. *Home away pattern (HAP)* refers to a sequence of home games, away games, and byes during the tournament for a team. We call a set of HAP patterns as *HAP pattern sets* if each team is associated with a HAP. If a HAP has home game and away game only, we can use number 1 and 0 to represent them respectively. Sometimes two teams share the same venue, it is necessary for them not to play home games at the same time. A *complementary* HAP sets might be needed in this case. *Game Off pattern (GOP)* refers to a sequence of games, off days during a time relaxed tournament for a team. We call a set of GOP patters as *GOP pattern sets* if each team is associated with a GOP.

Timetable refers to the allocation of games into periods. Each row of the table corresponds to a team while the column corresponds to time periods. If a timetable combined with an according pattern sets, we call it a *schedule.* A schedule is *feasible* if it meets all the constraints, otherwise it is *infeasible.*

There are many basic requirements for a schedule. Normally there is a requirement that the deviation of the number of games played for all teams should be less than a fixed number. Meanwhile for a individual team the deviation of the number of home games played and away games played should be less than a fixed number at any specific time as well. Such a schedule is termed *balanced*. Because of the consideration of the attendances, no team wants to play consecutive games. If this occurs, it is called a *break.* If a team plays consecutive home games, it has *a home stand.* In the time relaxed tournament, if a team play two consecutive games on consecutive days, it has *back-to-back games.*

1.4 Outline

The work is organized as follows. All the prior related research is reviewed in Chapter II. It includes the topic on general sports scheduling, time relaxed tournament scheduling and basketball tournament scheduling.

Chapter III focuses on basic time relaxed round robin tournament scheduling problems. Single round, double round, and multiple round time relaxed round robin tournament optimization problems are defined. Because of the complexity of this problem, several decomposition methods are presented.

Time Relaxed Traveling Round Robin problem (TRTTP) is examined in Chapter IV. It is a time relaxed version of the well known traveling tournament problem (TTP). Besides the definition, several methods to tackle the TRTTP are described in this chapter.

In Chapter V, the NBA scheduling problem is presented. All the major constraints needed to make a feasible schedule are presented by means of both integer programming model formulation and constraint programming model formulation. Finally, Chapter VI gives conclusion and an outlook on future research.

CHAPTER II

LITERATURE REVIEW

This dissertation work focuses on a topic belongs to the sports scheduling, which normally is a combinational optimization problem. In this section, first we will review the research on the general sports scheduling. Then we will focus on the category of the time relaxed scheduling problems. The research studies related to the basketball scheduling are reviewed as well.

2.1 Sports scheduling

According to Nemhauser and Trick (1998), sports scheduling can be divided into two categories: temporally relaxed scheduling and temporally constrained scheduling problems. They will be referred to as time relaxed scheduling and time constrained scheduling respectively in this dissertation study. For the time constrained scheduling, the number of the games is equal to the minimum required time slots. For the time relaxed scheduling, the number of the available time slots is larger than the minimum required time slots.

In the last thirty years, there have been a lot of researches done on the topic of sports scheduling. Most of the research focuses on the time constrained scheduling. The research objectives can be classified into two broad groups: break minimization and traveling distance minimization. If a team plays two consecutive home or away games, it has a break. The undesired consequence caused by the consecutive games could include the attendance declining, unfairness for participants, etc. A schedule with as few breaks as possible is desired by most leagues. Long traveling distance will cause fatigue for the players and coach, which will give advantage to its opponents. Therefore teams want to minimize the traveling distance to avoid the fatigue and save cost. This section will review the studies done on the break minimization, the literature on the traveling distance minimization will be reviewed in Chapter IV.

During the history of constructing schedules for both professional and amateur sports tournaments, some mathematical tools have been utilized and 1-factorization is one of them. As early as 1980, the relationship between 1-factorizations of graphs and tournaments has been exploited by de Werra. A compact single round robin tournament can be associated with a complete graph. Each node corresponds to a team and each edge corresponds to a game between the teams associated with two end nodes. A 1-factorization of a complete graph is to construct as many n-1 1-factors corresponding to a partitioning of the games into n-1 slots. The orientation of the edge can be used to represent the game venue. The following Figure 1 is a 1-factorization example when there are 6 teams in the tournament. Figure 2 is the schedule based on the results from Figure 1. The positive sign is used to represent a home game, and a negative sign is used to represent an away game.

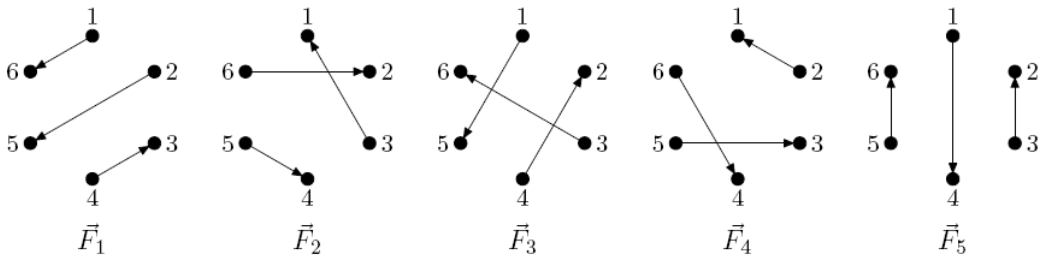

Figure 1 An example of oriented 1-factorization of K6

Slots	1	2	3	4	5
Team 1	−6	+3	−5	+2	−4
Team 2	−5	+6	+4	−1	+3
Team 3	+4	−1	−6	+5	−2
Team 4	−3	+5	−2	+6	+1
Team 5	+2	−4	+1	−3	−6
Team 6	+1	−2	+3	−4	+5

Figure 2 Schedule corresponding to the Figure-1

With the help of graph theory, researchers have developed many important theories. For example, the lower bound of the breaks for a single round robin schedule is n-2. This is easy to be proofed. If a team has no break, it has to alternate the game pattern every time after the first time slot. Because each two teams have to play at least once, no teams can have same pattern. There are only two no-break patterns are available: starting with a home game or starting with an away game. As a result at most two teams can have no-break patterns. All the other n-2 teams have at least one slot different, which cause a break. Originally the graph method was used to solve the problems without any other constraints. Although de Werra and other researchers developed some intricate solutions to solve the problem with constraints, the usage of the graph method is limited

if there are many constraints. We have to mention Rosa and Wallis (1982) work on the premature sets. They proved that premature sets of 1-factors exist under many circumstances, which means greedy construction of scheduling time slots one by one will not work.

Since decomposition is an effective method to tackle complicated problems, several studies on decomposition methods has been conducted. A sports scheduling problem can be decomposed into four steps: generating patterns, generating pattern sets, generating timetable, and generating complete schedules. Trick (2001) argued that the order should be arranged according to the difficulties of these steps. To get a schedule with minimum break, it is appealing to schedule the pattern and pattern sets at first. This brings the question to determine the feasibility of pattern sets. If an opponent schedule is scheduled first, integer programming and constraint programming among many other methods are used to tackle the break minimization problem.

Beginning in the 1990s, several metaheuristic approaches were applied to minimize the break in the practice problems. Willis and Terrill (1994) used simulated annealing and Wright (1995) used a tabu search for scheduling cricket tournaments.

2.2 Time relaxed scheduling

The majority of the professional sports leagues use time constrained schedules, such as the college basketball conferences in the United States, the professional soccer leagues in Europe, etc. However, there are two notable professional sports leagues in North America use time relaxed scheduling: National Hockey League (NHL) and

National Basketball Association (NBA). In the last thirty years, there has been couple tries to tackle the time relaxed scheduling problems.

The only study focus on the NBA scheduling problem was conducted by Bean and Birge (1980). The objective is to reduce the traveling distance. The authors state the problem cannot be solved by linear programming because the amount of the variables is too large. Therefore a two-phase heuristic method is proposed. The first phase is to make the traveling tours for each team, the second phase is to combine these tours according to the constraints. To make the tour for a team, the saving index based on the distance is calculated. The tour grows from one game to five, which is the maximum road trip games permitted. The tours are sorted by the traveling distance. The longest unscheduled tour is placed in the period of least home arena availability. A tour is divided into partial tours in case it cannot be placed. The authors report a result with 20% saving of the traveling distance compared to the official schedule.

There have been three researches focused on the NHL scheduling. Fraser (1982) developed a “road trip simulator ” model which has been proven inflexible to schedule all games. He made a statement that it was impossible to use the computer to generate a schedule without manually tuning or adjusting. Ferland and Fleurent (1991) proposed a decision support system to help users interactively make the schedule. There are two major components in the tool: road trip scheduler and exchanging tools. The first part is to schedule the required road trips for all teams. The road trips were classified as forced and free. A long period of unavailability of stadium will cause a forced road trip. It used a heuristics similar to the one used by Bean and Birge to construct the forced road trips. This method consists of four steps. The first step is to select the team with longest arena

unavailability. The second step schedules the two games with largest saving index in the middle of the period. The candidate games are those cannot be played in consecutive days because of the traveling distance. The third step is to fill in the games at the end of the road trip with the same logic used in the step 2. The last step is to fill in the games at the beginning of the road trip. The exchange procedure consists of two choices: single exchange and double exchange. In single exchange, the remaining game can be scheduled by changing one game. Similarly, the remaining game can be scheduled by changing two games in the double exchange. The tool can be used by expert in batch mode or try mode. The authors reported that this system produced promising results.

The latest research on the NHL scheduling was conducted by Costa (1995). A evolutionary tabu search algorithm was proposed. All the constraints are classified into two types: essential constraints and relaxed constraints. All the feasible schedules satisfy the essential constraints. The relaxed constraints are used to judge the solution quality. There are four steps in this algorithm. The first step is to generate initial solutions. Method used in the first step is similar to the one used by Ferland and Fleurent. The second step is to reproduce the solutions. The fitter a solution, the better chance it will be chosen for the reproduction. The third step is to crossover the solutions. It tries to put as many games as possible in a solution without changing the frame. All redundant games are removed according to the costs. The final step is to apply the tabu search. In each iteration, the games violating at least one relaxed constraint are chosen to change the game day. The last move is regarded as a tabu for the next certain iterations. Only a part of the constraints are used to construct the aspiration function. The author reported good computation results.

Besides the professional leagues like the NBA and the NHL, some no-professional tournaments use the time relaxed scheduling as well. One of them received research attention is the amateur table tennis tournament in German. It was a double round robin tournament scheduling problem. The constraints considered in this research included off days, balance of the number of the games among teams, arena availabilities. Schonberger (2004) modeled this problem as a constraint satisfaction problem. The variables are games between each team, the values are the possible game dates for each game. The authors tried constraint programming and got poor results when the number of the teams getting large. Therefore a generic algorithm was proposed to solve the problem. The variable used in the CSP model is selected as the coding for the generic algorithm. The mutation operator is to swap the game days for two games. To drive the search toward the feasibility, the objective function penalized the violation of the constraints. A local improvement heuristic was implemented. The author reported good results based on the generic algorithm.

2.3 Basketball tournament scheduling

It is difficult to solve the basketball tournament scheduling by exact methods because of the enormous amount of the variables and values involved. Therefore heuristic algorithms were deployed for the basketball scheduling. The first tractable literature on the topic of basketball league scheduling in modern time was published in 1976. Campbell and Chen (1976) developed an algorithm to minimize the traveling distance for a college basketball league. This algorithm consists of two phases. The first phase is to find a schedule with minimum traveling distance for each team. To do that, teams are divided into pairs based on the distances. The optimal travel pattern for a team is to travel

to every other pair in one trip before return home, and use a round trip to visit the team paired with itself. In the second phase, all the other constraints are deployed to make a feasible solution. The authors reported a significant decrease in the traveling distance by using this algorithm.

Inspired by the work of Campbell and Chen, Bean and Birge (1980) tried to tackle the NBA scheduling problem by similar algorithm. It is reviewed in the previous section. Besides the algorithm used by Bean and Birge, other heuristic algorithms to make a feasible schedules were constructed. For example, Frencek (2001) used an algorithm based on the graph theory to make schedules for Czech national basketball league.

With the development of the computation technology, solving the basketball scheduling problem by exact method became a possibility. Nemhauser and Trick (1998) faced a problem to make the schedule for basketball games in the Atlantic Coast Conference (ACC). It was done in three steps. The first step was to find game patterns and pattern sets. At first all possible 38 game patterns for a team were generated. All the constraints related to the game pattern were considered in generating patterns, such as the number of home games and weekend games. Then all possible 17 pattern sets was generated. All the constraints related to the time slot were considered in generate pattern sets. The second step was to assign the teams to the pattern sets to generate the timetable. It took 10 minutes to generate 826 feasible timetables. The last step was to assign the team to timetables, which is most time-consuming. It took 24 hours to generate 17 feasible schedules.

If the objective is to find a feasible solution rather than an optimal value, constraint programming shows more strength compared to integer programming. Henz (2001) showed this result by solving Trick's problem with much less computation time. It produced similar results but reduced the overall computation time from 24 hours to 1 minute. The problem was decomposed into three phases, all of them were solved by constraint programming.

Like Trick, several other researchers used integer programming to tackle the basketball league scheduling problem. Voorhis (2005) developed an integer programming model to solve the basketball league scheduling problem for two college conferences. Later on he incorporated travel swings into his model.

Although exact methods are promising, heuristic algorithms are necessary in many cases. Wright (2006) used a metaheuristic algorithm to make schedule for the New Zealand basketball league. Instead of putting all requirements into constraints, the author used an objective function to model subjective requirements. A sub-cost guided simulated annealing algorithm was used to solve the problem to the minimum penalty.

CHAPTER III

TIME RELAXED ROUND ROBIN TOURNAMENT SCHEDULING

Time relaxed round robin tournaments have different characteristics compared to the time constrained counterparts. This chapter concerns the basic problems solely derived from time relaxed tournament structural requirements.

3.1 Time Relaxed Single Round Robin Tournament Problem (TRSRR)

Time relaxed single round robin tournaments have the general round robin tournament structure requirement: every team will meet every other team at a fixed number, which is one in this case. In a time constrained round robin tournament, the number of the available game days is equal to the minimum required days. Unlike its counterpart, time relaxed round robin tournaments have more game days than the minimum required. Therefore it is necessary to define the parameter of the number of available game days. In this dissertation work we assume the number of available time slots is double size of the number of the games. For instance, there are 2(n-1) time slots available if a team has to play n-1 games. Table 1 is an example schedule for a tournament with six teams.

Easton (2001) provided a formal definition of the time constrained basic single round robin problem. We will use the similar notation used in her definition to define the basic time relaxed single round robin tournament problem as the following:

Definition 1 The Basic Time relaxed single round robin tournament problem

Instance: a set of n teams $T= \{t_1, t_2, \ldots, t_n\}$

Question: Is there a mapping of the games in the set $G = \{g_{ij}: t_i,t_j \in T, i<j\}$ on the days in the set $D = \{D_k, k = 1,2,\ldots, 2(n-1)$ if n is even and $k = 1,2,\ldots, 2n$ if n is odd$\}$ such that no more than 1 game including t_i is mapped on any given day for all $t_i \in T$?

Table 1 Time Relaxed Single Round Robin Tournament for n = 6

	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10
Team 1	@3		vs 2	@6	vs5		vs 4			
Team 2			@1	vs 4		vs 5	@3			vs 6
Team 3	vs 1	@6			@4		vs2	@5		
Team 4	@6	@5		@2	vs3		@1			
Team 5		vs4			@1	@2		vs 3	@6	
Team 6	vs 4	vs 3		vs1					vs 5	@2

Like we mentioned in Chapter II, Werra and many other researchers did extensive research on the relationship between the graphics and the round robin tournament schedule construction. Although these researches are done in the time constrained context, some results are applicable to the time relaxed tournament as well. We use the single round robin tournament to demonstrate the relationship.

If there are 2n teams, a complete graph K_{2n} with 2n nodes is considered. Each node represents a team. Every node has an edge connecting to every other node, which represents a game. Therefore every node has 2n-1 edges connected to itself. We need 2n-1 color to distinguish all these edges. If we partition all the nonadjacent edges with same

color, we get so called 1-factors $F_1, \ldots, F_{2n-1,}$ corresponding to the games schedules in the rounds r = 1,2, …, 2n-1. The direction can be used to represent the venue information. To construct a schedule with minimum breaks, the so called canonical 1-factorization (F_1, F_2, …, F_n) was used. The factors F_i refers to the edge sets:

$$F_i = \{\{2n, i\}\} \cup \{\{i + k, i - k\} | k = 1, \ldots, n - 1\}$$

The numbers $i+k$ and $i-k$ are taken modulo $2n-1$ as one of the numbers 1,2, . . . , $2n-1$. If there are no constraints on the length of consecutive games, we can use the canonical 1-factoriztion to generate a schedule for the time relaxed round robin tournament, which happens to have minimum breaks.

The TRTTP can be models as both IP model and CP model, we will look at these two models in the following sections.

3.1.1 TRSRR IP model

TR-RRT can be modeled as an IP model with employing $2n(n-1)^2$ variables and $\frac{5n(n-1)}{2}$ values.

Model 1 TRSRR IP Model

$$\sum_{d \in D} (x_{ijd} + x_{jid}) = 1 \forall i, j \in T, i < j \tag{3.1}$$

$$\sum_{j \in \{T \setminus i\}} (x_{ijd} + x_{jid}) \leq 1 \forall i \in T, d \in D \tag{3.2}$$

$$x_{ijd} \in \{0,1\} \forall i, j \in T, i \neq j, d \in D \tag{3.3}$$

Binary variable $x_{i,j,d}$ represents team i will host team j on day d. It is equal to 1 if game (*i, j, d*) is carried out, and 0 otherwise. Constraint (3.1) restraints each team will play every other team once, constraint (3.2) restraints every team play at most one game on a given day. The above IP model performs poorly when the problem size grows to an extent. Therefore Trick (2003) suggested to add an extra constraint to the IP model for the time constrained SRR:

$$\sum_{i \in S,\, j \notin S} x_{ijt} \geq 1 \forall S \subset T, \left|S\right| odd \tag{3.4}$$

S is a set of teams. If all the sets are added, this odd-set constraint make sure each team will play exactly once in a time slot. Because every team is not required to play a game on every day, the above constraint cannot be applied to the time relaxed tournament.

3.1.2 TRSRR CP model

We first give an introduction to the constraint programming. Then we provide the CP model for the TRRRT problem.

3.1.2.1 Constraint Programming

The notation for the CSP is a triplet (X, D, C). X refers to a finite set of variables, D is the domain of variable represent all possible values for each variable, C is the set of the constraints. A solution to the CSP is a set of values assigned to each variable that all constraints are satisfied. The following is a brief description of the procedure to use constraint programming to solve the CSP. First we will encode the problem as P. Each constraint is associated with a filter algorithm aims to reduce the values in the domain which are not consistent with the constraint. To implement a constraint C, we will

generate two new problems by applying C' and negation –C' to P. The so-called consistency techniques are used during the process. If setting a variable from any value from its domain, all the other variables could choose some values to meet the constraint, that variable's domain is arc consistency. In this way we will get a binary search tree.

Regin (1999) stated a good CP model deal with four important problems: symmetries, implicit constraints, global constraints, pertinent and redundant constraints. The search space can be dramatically reduced by removing intrinsic symmetries caused by identical characteristics of variables. Sometimes the search can be speeded up by introducing implicit constraints. For example, the number of the breaks is always even. Adding this implicit constraint can reduce the search time dramatically. A global constraint normally involves a set of other constraints. For example, allDifferent is a powerful constraint to affect many variables simultaneously. Adding a symmetry or implicit, global constraint does not guarantee the computation reducing, which leads to the notation of pertinent constraint. If a constraint adding to the defined model cannot improve the performance, it is called redundant.

3.1.2.2 CP models

Henz (2003) did an extensive investigation on constraint programming formulations for single round robin tournament problem. The basic variables used in their model are *opponent[i, d]*, which represents team i's opponent on day d. The following is the CP model of the time constrained round robin tournament:

Model 2 Time Constrained Single Round Robin Tournament CP Model

$$all-different(O_{i1}, O_{i2,\ldots,} O_{in-1}) \forall i \in T \tag{3.5}$$

$$one-factor(O_{1d}, O_{2d}, ...O_{nd}) \forall d \in D \quad (3.6)$$

$$Oid \in \{1, 2, ...n\} \forall i \in T, d \in D \quad (3.7)$$

The constraint (3.7) defines the variable Oid's domain. The constraint (3.6) restraints every team will play exactly one game in every time slot. The constraint (3.5) restraints every team will play every other team exactly once. To understand how these two constraints are achieved, we will present formal explanation as following.

The definition for the all-different is:

$$all-different\{x_1, x_2, ..., x_m\} = \{(v_1, v_2, ...v_m) \in (D_{x1}, D_{x2}, ..., D_{xm} \mid \forall i, j, i \neq j, v_i \neq v_j)\} \quad (3.8)$$

x_i is the variable Di is the domain and Vi is the value of the variable which belongs to the domain. Two variables have to take on different values if both variables are non-zero.

The definition of the one-factor is:

$$One-factor(x_1, ..., x_m) = \{(v_1, ...v_m) \in Dx_1 \times ...Dx_m \mid \forall i, j, v_i \neq i, v_i = j \Leftrightarrow v_j = i\} \quad (3.9)$$

The definitions of x_i , Di and Vi are same as those used in (3.8). This constraint requires no variable can have value equals to itself. If variable i's value Vi equals j, then variable j's value Vj equals i as well.

For both all-different and one-factor constraint, there are several available propagation algorithms. The strengths of these algorithms vary depending on such factors as domain size, additional external constraints, and so on. Herz, Muller and Thief (2004)

examined several propagation approaches for the all-different and one-factor constraints in the constraint programming context. They convincingly argued that for the all-different constraint, the propagation from Reign (1999) should be used. For the one-factor constraint, the arc-consistence propagation will reduce the computation time under most circumstances.

In the time relaxed single round robin tournament, a team needs to play every other team once. Therefore the all-different opponent constraint is still valid. However, since a team will have same opponent (opponent 0 on off days) multiple times during the tournament, we have to alter the all-different opponent constraint accordingly. The one-factor constraint will be invalid in the time relaxed tournament, since teams will not play on every day. Given these considerations, we present the time relaxed single round robin tournament CP model as following. For consistence and comparison, similar variables used in the time constrained tournament models are used in our model. When team i does not play on day d, we assign value 0 to the variable *opponent[i, d]*.

Model 3 TRSRR CP Model

$$all-different\{O_{i1}, O_{i2}, \ldots, O_{id} \mid O_{id} \neq 0\} \forall i \in T \quad (3.10)$$

$$One-factor(O_{1d}, O_{2d}, \ldots, O_{nd} \mid O_{id} \neq 0) \forall i \in T, d \in D \quad (3.11)$$

$$O_{id} \in \{1, 2, \ldots n\} \forall i \in T, d \in D \quad (3.12)$$

The domain for the opponent variable is defined in (3.12). The first constraint (3.10) forces every team not to have duplicated opponents throughout the tournament except dummy 0. The second constraint (3.11) restrains teams not to have same opponent

on the same day. Like we stated early, a team will have dummy opponent 0 for more than once because the number of the time slots is greater than the number of the games. Consequently, we cannot simply copy the all-different algorithms from the time constrained tournament models. To use the research results from the time constrained context, we tried two methods. The first one is to filter the dummy opponent by adding one more variable to represent if a team play a game or not on a specific day. According to our basic computation test, this method leads to increasing computation time. Therefore we developed the second method, which is listed as following.

The requirement for every team to have different opponent throughout the tournament can be interpreted as play every other team exactly once. We get the all-different definition in the following equation:

$$\sum_{d \in D} (O_{id} = j) = 1 \forall i, j \in T, i \neq j \tag{3.13}$$

No team can play itself; no teams share same opponent; these two requirements in the constraint (3.9) are still valid:

$$O_{id} \neq i \forall i \in T, d \in D \tag{3.14}$$

$$O_{id} = j \Leftrightarrow O_{jd} = i \forall i, j \in T, d \in D \tag{3.15}$$

To clarify, the OPL code based on the above algorithms is listed.

```
int nbTeams  = …;
int nbTeamGames = nbTeams-1;
int nbDays = 2*nbTeamGames ;
range rngTeams = 1..nbTeams;
```

```
range rngDays = 1..nbDays;
dvar int opponent[rngTeams][rngDays] in 0..nbTeams;
subject to
{
        //All different
        forall(i,j in rngTeams: i!=j)
          sum(d in rngDays)(opponent[i][d] == j) == 1;
        //One-factor
        forall(i in rngTeams,d in rngDays)
                opponent[i][d]!=i;
        forall(d in rngDays,i,j in rngTeams:i!=j)
                (opponent[i][d] ==j) == (opponent[j][d] == i);
}
```

3.1.3 Completion of partial schedule

In the practical world, normally the schedulers face the problem of completing partial schedules, rather than starting from the scratch. The reason behind that is there are many requirements from external sources, such as TV networks. It is a common requirement from the TV networks to reserve high-profile games on specific dates. All these reserved games combining other requirements compose a partial schedule. Take the schedule in Table 1 as an example. If some but all games are schedules, we will get a partial schedule as listed in Table 2.

It is not a trivia task to complete a partial schedule. Actually Easton (2001) proved that there are no polynomial time algorithm to complete partial time constrained round robin tournament. Even for a partial tournament with only three periods are uncompleted and each team has at most three games unscheduled, Easton shows it is still a NP-hard problem. It can be reduced to the Latin square completion problem.

Table 2 Partial schedule for n = 6

	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10
Team 1			vs 2							
Team 2			@1			vs 5	@3			vs 6
Team 3		@6						@5		
Team 4	@6						@1			
Team 5						@2		vs 3		
Team 6	vs 4	vs 3								@2

The mapping between Latin square and time relaxed tournament is not valid because the number of time slots is greater than the number of the teams. Unfortunately we cannot provide formal proof the complexity of this completion problem, we conjecture it is NP-hard.

3.1.4 Optimization Problem

In the practical world, getting a feasible schedule is not the only requirement. It is not uncommon for a schedule to have one or some objectives besides many additional constraints. If we consider every game has a related cost, we get the time relaxed single round robin tournament optimization problem. As Briskorn (2007) defined the time constrained round robin tournament optimization, we will formally define the time relaxed single round tournament optimization as the following:

Definition 2 Time Relaxed Single Round Robin Tournament Optimization Problem

Given a set of teams T, |T| = n, and a set of available days D, |D| =2 (n−1), each triple (i, j,d) ∈ T × T × D, i ≠ j, represents a game of team i against team j at i's home on day d. Cost $c_{i,j,d}$ is given for each game. A feasible solution to the Time Relaxed Single Round Robin Tournament problem corresponds to a set of* $\frac{n(n-1)}{2}$ *triples such that*

(i) for each pair (i, j) ∈ T × T, i < j, exactly one triple of form (i, j, d) or (j, i, d) with d ∈D is chosen

(ii) for each pair (i, b) ∈ T × D, at most one triple of form (i, j, b) or (j, i, b) with j ∈ T\{i} is chosen.

The problem is to find a feasible solution having the minimum sum of chosen triples' cost.

Condition (i) forces each team to meet every other team once, condition (ii) restrains each team to play at most one game on a day. TRSRRT can be modeled as an IP model with employing $2n(n-1)^2$ variables and $\frac{5n(n-1)}{2}$ values.

Model 4 TRSRR Optimization problem IP model

$$\min \sum_{i \in T} \sum_{j \in \{T \setminus i\}} \sum_{d \in D} x_{ijd} C_{ijd} \tag{3.16}$$

$$\sum_{d \in D} (x_{ijd} + x_{jid}) = 1 \forall i, j \in T, i < j \tag{3.17}$$

$$\sum_{j \in \{T \setminus i\}} (x_{ijd} + x_{jid}) \leq 1 \forall i \in T, d \in D \tag{3.18}$$

$$x_{ijd} \in \{0,1\} \forall i, j \in T, i \neq j, d \in D \tag{3.19}$$

Binary variable $x_{i,j,d}$ is equal to 1 if game (i, j, b) is carried out, and 0 otherwise. Constraints (3.17) and (3.18) correspond to (i) and (ii) respectively, while (3.16) represents the goal of cost minimization.

Briskorn proofed the time constrained single round robin tournament optimization problem is NP-hard by using the reduction from the planar three index assignment problem (PTIAP). Although we cannot provide the formal proof here, we conjecture the time relaxed round robin tournament optimization problem is NP-hard as well.

3.2 Time Relaxed Double Round Robin Tournament (TRDRRT)

Teams play each other twice in a double round robin tournament, normally one at home, the other on road. In the time constrained schedules, the mirrored double round robin tournament is popular. A mirrored double round robin tournament consists of two rounds with identical timetables, and the venues in the second round are reversed to those in the first round. For example, it team i play at home with team j on day d ($d \leq \frac{D}{2}$), then team j will play at home with team i on day $d + \frac{D}{2}$. Because the game days in the time relaxed tournament are not fixed, we will not consider mirrored round robin tournaments in this dissertation study.

In the practical world, it is a common requirement for a league to have a round-based tournament if it has a multiple-round schedule. For example, a lot of leagues have all-star event after the half season. It makes sense to have a complete round before the

all-star event. In this section we will examine the round-based tournament at first; then we will look into the general tournament. The integer programming models are given for both formats.

3.2.1 Round-based time relaxed double round robin tournament

In the time constrained double round robin tournament, the number of the game days for the first round equals to the number for the second round by default. In the time relaxed double round robin tournament, we need to define the game days for each round. We use D_1 to represent the set of days in the first round, D_2 to represent the set of days in the second round , D to represent the set of days in two round, $D_1+D_2 = D$.

We will examine the optimization problem only. The definition is similar to the one of the time relaxed single round robin tournament. The following is the formal definition:

Definition 3 round-based time relaxed double round robin tournament

Given a set of teams T, $|T| = n$, and a set of available days D, $|D| = 4*(n-1)$. The days set is divided into two sub sets: D_1 and D_2. Each triple $(i, j, d) \in T \times T \times D$, $i \neq j$, represents a game of team i against team j at i's home on day d. Cost $c_{i,j,d}$ is given for each game.

A feasible solution to the Time Relaxed round-based double Round Robin Tournament problem corresponds to a set of n(n-1) triples such that*

(i) for each pair $(i, j) \in T \times T$, $i < j$, exactly one triple of form (i, j, d) or (j, i, d) with $d \in D_1$ is chosen

(ii) *for each pair (i, j) ∈ T × T, i ≠ j, exactly one triple of form (i, j, d) with d ∈ D is chosen*

(iii) *for each pair (i, d) ∈ T × D, at most one triple of form (i, j, d) or (j, i, d) with j ∈T\{i} is chosen.*

The problem is to find a feasible solution having the minimum sum of chosen triples' cost.

The variables are same as those used in single round robin tournament problem.

Model 5 round based TRDRR – IP Model

$$\min \sum_{i \in T} \sum_{j \in T \setminus \{i\}} \sum_{d \in D} C_{ijd} x_{ijd} \tag{3.20}$$

$$\sum_{d \in D_1} (x_{ijd} + x_{jid}) = 1 \forall i, j \in T, j < i \tag{3.21}$$

$$\sum_{d \in D} x_{ijd} = 1 \forall i, j \in T, j \neq i \tag{3.22}$$

$$\sum_{j \in T \setminus \{i\}} (x_{ijd} + x_{jid}) \leq 1 \forall i \in T, d \in D \tag{3.23}$$

$$x_{ijd} \in 0,1 \forall i, j \in T, d \in D \tag{3.24}$$

Binary variable $x_{i,j,d}$ is equal to 1 if game (*i, j, d*) is carried out, and 0 otherwise. Constraints (3.21) restraint every team will play every other team once in the first round. Constraint (3.22) ensure every team will play every other team once at home during the

whole season. Constraint (3.23) requires every team will play at most one game on each day. The objective function (3.20) represents the goal of cost minimization.

It is obvious that the complexity of TRDRRT is decided by the complexity of TRSRRT. The TRDRRT can be solved in polynomial time only if TRSRRT can be solved in polynomial time. Since we conjecture TRSRRT is NP-hard, we conjecture TRDRRT is NP-hard as well.

3.2.2 General time relaxed double round robin tournament

The general time relaxed double round robin tournament is defined as:

Definition 4 General round-based time relaxed double round robin tournament

Given a set of teams T, $|T| = n$, and a set of available days D, $|D| = 4*(n-1)$. Each triple $(i, j, d) \in T \times T \times D$, $i \neq j$, represents a game of team i against team j at i's home on day d. Cost $c_{i,j,d}$ is given for each game.

A feasible solution to the Time Relaxed general double Round Robin Tournament problem corresponds to a set of n(n-1) triples such that*

(i) for each pair $(i, j) \in T \times T$, $i \neq j$, exactly one triple of form (i, j, d) with $d \in D$ is chosen

(ii) for each pair $(i, d) \in T \times D$, at most one triple of form (i, j, d) or (j, i, d) with $j \in T\backslash\{i\}$ is chosen.

The problem is to find a feasible solution having the minimum sum of chosen triples' cost.

The IP model is:

Model 6 general TRDRR – IP Model

$$\min \sum_{i \in T} \sum_{j \in T \setminus \{i\}} \sum_{d \in D} C_{ijd} x_{ijd} \tag{3.25}$$

$$\sum_{d \in D} x_{ijd} = 1 \forall i, j \in T, j \neq i \tag{3.26}$$

$$\sum_{j \in T \setminus \{i\}} (x_{ijd} + x_{jid}) \leq 1 \forall i \in T, d \in D \tag{3.27}$$

$$x_{ijd} \in 0,1 \forall i, j \in T, d \in D \tag{3.28}$$

Binary variable $x_{i,j,d}$ is equal to 1 if game (i, j, d) is carried out, and 0 otherwise. Constraints (3.26) restraint every team will play every other team once at home during the whole season. Constraint (3.27) restraint every team will play at most one game on each day. The objective function (3.25) represents the goal of cost minimization.

We conjecture the round-based double round robin tournament problem is NP-hard.

3.2.3 TRDRR CP Models

The venue factor is not considered in the TRSRR-CP model (Model 3). In the double round robin tournament, venue is usually a factor. Therefore we need to deal with the venues in the TRDRRT CP models accordingly. For demonstration, we choose the general TRDRRT as our modeling basis. We will compare three modeling approaches: home opponent based, road opponent based, and two variables based.

For the home opponent based approach, we change the variable *opponent[i][d]* in the Model 3 to *HO[i][d]*.

Model 7 Home opponent based CP model for TRDRRT

$$\sum_{d \in D} (HO_{id} = j) = 1 \forall i, j \in D, i \neq j \tag{3.29}$$

$$\sum_{j \in \{T \setminus i\}} (HO_{jd} = i) \leq 1 \forall i \in T, d \in D \tag{3.30}$$

$$HO_{id} = j \Rightarrow HO_{jd} = 0 \forall i, j \in T, d \in D \tag{3.31}$$

The variable HO_{id} represents team i's opponent on day D if i plays a home game, the domain are all teams in the tournament except itself, plus dummy 0 for off day. The constraint (3.29) restraints every team will play every other team once at home. The constraint (3.30) ensures every team play at most one game on each day. The constraint (3.31) requires one team stay at home and the other stay on road if they play a game on a specific day.

For the road based approach, the variable V_{id} is used to represent team i's opponent on day *d* when *i* plays on road.

Model 8 Road opponent based CP model for TRDRR

$$\sum_{d \in D} (V_{id} = j) = 1 \forall i, j \in D, i \neq j \tag{3.32}$$

$$\sum_{j \in \{T \setminus i\}} (V_{jd} = i) \leq 1 \forall i \in T, d \in D \tag{3.33}$$

$$V_{id} = j \Rightarrow V_{jd} = j \forall i, j \in T, d \in D \tag{3.34}$$

The variable V_{id} represents where team *i* play on day *d*. If the value is not *i*, it refers to team i's road opponent; if the venue is i, it means i either have a home game or off day. The constraint (3.32) force every team will play every other team exactly once on road. The constraint (3.33) restraints every team plays at most one game on each day. The constraint (3.34) ensures two opponents to stay at the same venue for a game.

Two variables O_{id} and *Vid* venue are used in the two variables based approach: one represents team i's opponent on day d, the other represents team i's venue on day d. The domains for these variables are same as those defined in the previous two models, which are the other teams in the tournament.

Model 9 Two variables based CP model for TRDRR

$$all-different(O_{1d}, O_{2d}, \ldots O_{nd} \mid V_{id} = 1) \forall i \in T, d \in D \tag{3.35}$$

$$all-different(O_{1d}, O_{2d}, \ldots O_{nd} \mid V_{id} = 0) \forall i \in T, d \in D \tag{3.36}$$

$$O_{id} = j \Rightarrow O_{jd} = i \forall i, j \in T, d \in D \tag{3.37}$$

$$O_{jd} = i => (V_{id} = 0 \,\&\, V_{jd} = 1) \,||\, (V_{id} = 1 \,\&\, V_{jd} = 0) \forall i, j \in T, d \in D \tag{3.38}$$

$$O_{id} \neq i \forall i \in T, d \in D \tag{3.39}$$

$$Oid \in \{T \setminus i\}, Vid \in \{0,1,2\} \forall i \in T, d \in D \tag{3.40}$$

The constraint (3.35) and (3.36) ensure every team plays every other team at home and on road exactly once respectively. The constraint (3.37) restrains two teams has each

other as opponent for a game. The constraint (3.38) enforces opponent has complementary venue. The constraint (3.39) ensures no team play itself.

The computation study for three CP models is carried out in ILOG OPL studio. All the default propagation setting is used. The results are listed in Table 3. "Home" refers to the Home-opponent variable based approach, "Road" refers to the Road-opponent variable based approach, "Two variables" refers to the two-variables based approach. N is the number of the teams, F is the failure time, T is the running time. We can tell Home-opponent variable based and Road-opponent variable based approaches have similar performance with road-opponent has a slight advantage. Both approaches are better than the two-variables based approach. It even takes more than 10 minutes to get a solution for N=12 cases, therefore we just run the three cases with N =6,8, and 10.

The result for N=18, road-opponent based approach is not typo. It might be interesting to investigate if some certain number has special structure advantages by using this method.

Table 3 CP models for unconstrained TRDRR

N	Home		Road		Two variables	
	F	T	F	T	F	T
6	200	0.03	104	0.02	205	0.02
8	200	0.05	100	0.05	2587	0.48
10	200	0.06	103	0.06	413861	81.89
12	200	0.09	200	0.08	?	?
14	200	0.17	200	0.14	?	?
16	200	0.31	200	0.27	?	?
18	200	0.45	112	0.28	?	?
20	200	0.61	200	0.53	?	?
22	200	0.83	200	0.66	?	?
24	200	1.16	200	0.89	?	?
26	200	1.56	200	1.19	?	?

28	200	2.06	200	1.64	?	?
30	200	2.67	200	2.31	?	?

3.3 Time Relaxed r Round Robin Tournament Problem (TR -rRRTP)

For the NBA scheduling, every team will play four times with the opponent from the same division. That makes a four-round robin tournament. Generally, we get a r-round robin tournament if each team will play every other team r times. Similar to the two round tournaments, we have round-based and general robin tournament. Both of them are similar to those problems defined in the double round robin tournament respectively. We assume r is even in this dissertation study. We will just list the general one here. The following is the formal definition of the time relaxed r-round robin tournament problem.

Definition 5 Time relaxed r-round robin tournament

Given a set of teams T, $|T| = n$, and a set of available days D, $|D| = 2r*(n-1)$. Each triple $(i, j, d) \in T \times T \times D$, $i \neq j$, represents a game of team i against team j at i's home on day d. Cost $c_{i,j,d}$ is given for each game.

A feasible solution to the Time Relaxed general r- Round Robin Tournament problem corresponds to a set of $\frac{n(n-1)*r}{2}$ *triples such that*

(iii) for each pair $(i, j) \in T \times T$, $i \neq j$, exactly $\frac{r}{2}$ *triple of form (i, j, d) with $d \in D$ is chosen*

(iv) for each pair $(i, d) \in T \times D$, at most one triple of form (i, j, d) or (j, i, d) with $j \in T\backslash\{i\}$ is chosen.

The problem is to find a feasible solution having the minimum sum of chosen triples' cost.

The following is the IP model:

Model 10 TR –rRRT – IP Model

$$\min \sum_{i \in T} \sum_{j \in T \setminus \{i\}} \sum_{d \in D} C_{ijd} x_{ijd} \tag{3.41}$$

$$\sum_{d \in D} x_{ijd} = \frac{r}{2} \forall i, j \in T, j \neq i \tag{3.42}$$

$$\sum_{j \in T \setminus \{i\}} (x_{ijd} + x_{jid}) \leq 1 \forall i \in T, d \in D \tag{3.43}$$

$$x_{ijd} \in 0,1 \forall i, j \in T, d \in D \tag{3.44}$$

Binary variable $x_{i,j,d}$ is equal to 1 if game (i, j, d) is carried out, and 0 otherwise. Constraints (3.42) restraint every team will play every other team $\frac{r}{2}$ *times* at home during the whole season. Constraint (3.43) restraint every team will play at most one game on each day. The objective function (3.41) represents the goal of cost minimization.

If the single round robin tournament can be solved in polynomial time, the r-round robin tournament can be solved in polynomial time as well. Therefore we conjecture the r-round robin tournament is NP-hard.

3.4 Break Minimization Problem

In the time constrained tournament scheduling, a break is defined as a team play two consecutive games, both either at home or on road. In the time relaxed instances, it is

common that there are some off days between two games for a team. In real world, some teams do not want to play consecutive home games or road games because of the impact on the revenue, regardless of the off days between. Therefore it is still meaningful to consider two consecutive games with off days between as a break.

Definition 6 Break in time relaxed schedule

A break occurs when a team plays two consecutive away games or two consecutive home games, even there are off days between.

If there are no limitations on the consecutive games or consecutive off days, it is obvious that there exists a schedule with zero breaks for every time relaxed schedule. Froncek (2003) stated that there exists an unique schedule without break even if each team has just one bye. With the constraints on the length of consecutive games, it is not trivia to find a schedule with zero breaks. We will use a CP model to demonstrate.

We will explain the variables used in the model first. O_{id} refers to team i's opponent on day d, P_{id} refers to the game type on day d. The domains for these two variables are same as those in Model 9. Variable brk_{id} equals 1 if team i has a break on day d, otherwise 0. To calculate brk_{id} , we introduce an aiding variable $V_{id..}$ The binary variable $V_{id.}$ is defined as the following: the constraint (3.52) defines $V_{id.}$ equals P_{id} when a team plays a game on day d; the constraint (3.53) set $V_{id.}$ to be previous day value if team i is off on day d. The dummy variable V_{i0} is zero for all teams. With the help of $V_{id,,}$ we can give the definition of brk_{id} in constraint (3.56). The model is listed as the following:

Model 11 Break minimization CP model

$$\min \sum_{i \in T} \sum_{d \in \{2,\ldots,D\}} brk_{id} \tag{3.45}$$

$$\sum_{d \in D} (O_{id} = j \,||\, P_{id} = 1) = 1 \forall i, j \in T, i < j \tag{3.46}$$

$$\sum_{d \in D} (O_{id} = j \,||\, P_{id} = 0) = 1 \forall i, j \in T, i < j \tag{3.47}$$

$$\sum_{k=\{0,1\}} (P_{i(d+k)} = 0) < 2 \forall i \in T, d \in \{1, \ldots D-1\} \tag{3.48}$$

$$\sum_{k=\{0,1,2\}} (P_{i(d+k)} != 2) < 3 \forall i \in T, d \in \{1, \ldots D-2\} \tag{3.49}$$

$$O_{id} = j \Rightarrow O_{jd} = i \forall i, j \in T, d \in D \tag{3.50}$$

$$O_{id} \in \{T \setminus i\}, P_{id} \in \{0,1,2\} \forall i \in T, d \in D \tag{3.51}$$

$$P_{id} \neq 2 \Rightarrow V_{id} = P_{id} \forall i \in T, d \in D \tag{3.52}$$

$$P_{id} = 2 \Rightarrow V_{id} = V_{i(d-1)} \forall i \in T, d \in D \tag{3.53}$$

$$V_{i0} = 0 \forall i \in T \tag{3.54}$$

$$V_{id} \in \{0,1\} \forall i \in T, d \in D \tag{3.55}$$

$$br_{id} = (V_{id} = V_{i(d-1)}) * (O_{id} \neq 0) \forall i \in T, d \in \{2, \ldots D\} \tag{3.56}$$

The objective (3.45) is to minimize the total breaks for all teams. The constraint (3.46) and (3.47) ensures a double round robin tournament. The constraint (3.48) refrains every team to have consecutive home games. The constraint (3.49) makes sure no more than three games in a row. The other constraints provide the definition for breaks or the relationships among variables. This model can use the size of 8 teams, but it cannot provide the optimal solution for 10 teams in

30 minutes. Some approaches to break the symmetry as Regan did could be used to improve the model, we believe the work is not a trivia task.

3.5 Decomposition Scheme

Because of the complexity of the problems, decomposition methods are widely used for the sports league scheduling. For the time constrained tournaments, generally there are two things needed to be decided regarding a game: game opponent and game type (home game or road game). Games should be carried out on which day is not a concern in the time constrained, because all game days are predefined. However, it is a decision should be made for the time relaxed schedules. All the decision problems can be decomposed as sub-problems of the original scheduling problem. We break down the schedule problems into four sub- problems. First of all we provide the definitions of some terms used in this section.

Definition 7 GOP pattern

A GOP is a string of length equals to game numbers containing 1 on day d if the specific team has a game on that day, 0 otherwise

Definition 8 GOP pattern set

A GOP pattern set is a collections of GOP and exactly each GOP is assigned to a distinct team in the tournament.

Definition 9 HAP pattern

A HAP is a string of length equals to game numbers containing 0 on day d if the specific team plays a home game on that day, 1 if the specific team plays a road game, 2 if the specific team has an off day.

Definition 10 HAP pattern set

A HAP pattern set is a collections of GOP and exactly each HAP is assigned to a distinct team in the tournament.

The four steps to make a time relaxed schedule corresponding to four sub-problems are listed as the following:

1) Find GOP set. For 8 teams, a possible GOP set would be

    1. G G O G O O G O O G G O G O
    2. G O G G O O G O G G O G O O
    3. O G G O O G G O G O G G O O
    4. O O O G G O G O G G O G G O
    5. G O G O G G O O G G O G O O
    6. G O G O O G G O O G G O G O
    7. G G O G O G O O G O G G O O
    8. G G O O O O G O G G O G G O

2) Assign games consistent with the GOP set to get an opponent schedule. If team i has j as the opponent on day d, j should has i as opponent on day d as well. For off days, team's opponent will be designated to 0. A possible opponent schedule for the above GOP is:

    1. 8 3 0 2 0 0 4 0 0 5 7 0 6 0
    2. 5 0 6 1 0 0 3 0 4 8 0 7 0 0
    3. 0 1 5 0 0 7 2 0 8 0 6 4 0 0
    4. 0 0 0 7 5 0 1 0 2 6 0 3 8 0
    5. 2 0 3 0 4 6 0 0 7 1 0 8 0 0
    6. 7 0 2 0 0 5 8 0 0 4 3 0 1 0
    7. 6 8 0 4 0 3 0 0 5 0 1 2 0 0

8. 1 7 0 0 0 0 6 0 3 2 0 5 4 0

3) Assign HAP set consistent with the opponent schedule to get a timetable. If a team plays at home on day d, its opponent will play on road on the same day. For the above opponent schedule, a possible timetable would be

1. +8 -3 0 +2 0 0 -4 0 0 +5 -7 0 +6 0
2. -5 0 +6 -1 0 0 +3 0 -4 +8 0 -7 0 0
3. 0 +1 -5 0 0 +7 -2 0 +8 0 +6 -4 0 0
4. 0 0 0 +7 -5 0 +1 0 +2 -6 0 +3 -8 0
5. +2 0 +3 0 +4 -6 0 0 +7 -1 0 +8 0 0
6. -7 0 -2 0 0 +5 -8 0 0 +4 -3 0 -1 0
7. +6 -8 0 -4 0 -3 0 0 -5 0 +1 +2 0 0
8. -1 +7 0 0 0 0 0 +6 0 -3 -2 0 -5 +4 0

4) Assign teams to the timetable to get the complete schedule. If the teams are A,B,C,D,E,F,G,H, we can assign them to the 1,2,3,4,5,6,7,8 respectively to get the complete schedule:

A. +H -C 0 +B 0 0 -D 0 0 +E -G 0 +F 0
B. -E 0 +F -A 0 0 +C 0 -D +H 0 -G 0 0
C. 0 +A -E 0 0 +G -B 0 +H 0 +F -D 0 0
D. 0 0 0 +G -E 0 +A 0 +B -F 0 +C -H 0
E. +B 0 +C 0 +D -F 0 0 +G -A 0 +H 0 0
F. -G 0 -B 0 0 +E -H 0 0 +D -C 0 -A 0
G. +F -H 0 -D 0 -C 0 0 -E 0 +A +B 0 0
H. -A +G 0 0 0 0 +F 0 -C -B 0 -E +D 0

Assigning teams to timetables is league dependent. For that season, this step is not considered in this dissertation study. The decomposition methods can be classified according to the orders of the first three steps Two steps could be combined to form a new step since there are three decisions in the time relaxed tournaments. Theoretically there are more than dozen decomposition methods by switching the decision orders. For example, the following are four possible decomposition methods by staring the game day decision first:

1. First GOP, then HAP, last opponent
2. First GOP, then opponent, then HAP
3. First GOP and HAP, then opponent
4. First GOP and opponent, then HAP

Actually some decomposition methods are essentially equivalent. For example, if the opponent for each day is decided, then the GOP is decided automatically. Therefore "First GOP, then opponent, last HAP" is same as "First GOP and opponent, last HAP". We will focus on the problems with objective to minimize the breaks. In the time constrained tournaments, there are two well known decomposition methods if the objective is to minimize breaks: one is First-Break-Then-Schedule, the other is First-Schedule-Then-Break. We will examine how these two decomposition methods work in the time relaxed context. Besides the two well-known decompositions, we will single out the GOP problem.

3.5.1 First-Schedule-Then-Break

Trick (2003) argues that the steps should be ordered such that the most critical aspects of the schedule are considered early in the solution process. If the opponent is more important than venue in the scheduling, it should be considered at first. Additionally, sometimes the opponent could be decided by the external factors in the early scheduling stage. If the objective is to minimize the total breaks, we get a method called "First-Schedule-Then-Break". This decomposition approach first generates an opponent schedule, then finds a feasible HAP sets with minimum breaks for the opponent schedule.

Post and Woeginger (2006) defined opponent schedule as a timetable determines every pair (i, d), i ∈ T, d∈ D, the opponent team i on day d. The table 7 is an example for the league of 6 teams playing a time relaxed single round robin tournament consisting of 10 game days.

Table 4 An example of opponent schedule – 6 teams

	1	2	3	4	5	6	7	8	9	10
Team 1	4		3	5				2		6
Team 2	5	6	4	3				1		
Team 3	6	5	1	2						4
Team 4	1		2	6		5				3
Team 5	2	3	6	1		4				
Team 6	3	2	5	4						1

There are two types of approaches to generate an opponent schedule: generating opponent schedule without constraints or completion of a partial opponent schedule. As

we notice the opponent schedule contains the GOP information. For example, the GOP pattern for the team 1 in the above table is GOGGOOOGOG.

In the opponent schedule generating phase, if the objective is to generate a regular schedule, we can use the following CP model:

Model 12 Generate Schedule Phase CP Model

$$\sum_{d \in D} (O_{id} = j) = 1 \forall i, j \in T, i < j \tag{3.57}$$

$$O_{id} = j \Leftrightarrow O_{jd} = i \forall i, j \in T, d \in D \tag{3.58}$$

$$O_{id} \neq i \forall i \in T, d \in D \tag{3.59}$$

$$\sum_{k \in \{0,1,2\}} (O_{i(d+k)} \neq 0) < 3 \forall i \in T, d \in \{1, \ldots D-2\} \tag{3.60}$$

The variable O_{id} refers to team i's opponent on day *d*. The constraint (3.57) ensures every team play each other exactly once. The constraint (3.58) is the typical opponent constraint. (3.59) forces each team not to play itself. The constraint (3.60) set the length of consecutive games to be less than three for each team.

After getting an opponent schedule, the next step is to find a feasible HAP with minimum breaks. We give the formal definition of this problem as the following:

Definition 11 Time relaxed break minimization problem

Given a timetable for a time relaxed tournament, the break minimization problem consists of finding a feasible pattern set which minimizes the number of breaks.

Régin (2001) presents a CP approach to address the time constrained break minimization problem. A problem of 16 teams can be solved by his approach in one minute by adding some constraints

to break symmery for improvement. Trick (2001) used IP model to solve the problem size of 20. Elf (2003) transferred the break minimization problem into a maximum cut problem. They solved problem of 26 teams in reasonable time. A similar approach is used by Miyashiro and Matsui (2006). They developed an approximate method to solve the problem of 40 teams.

We can use CP model to address the break minimization problem. The variables P_{id} and brk_{id} are similar to those used in Model 11.

Model 13 Break Minimization Phase CP-model

$$\min \sum_{i \in T} \sum_{d \in \{2,...D\}} brk_{id} \tag{3.61}$$

$$br_{id} = (P_{id} = P_{i(d-1)}) * (O_{id} \neq 0) \forall i \in T, d \in \{2,...D\} \tag{3.62}$$

$$O_{id} \neq 0 \Rightarrow (Pid = 0 \,\&\, P_{Oidd} = 1) \,||\, (Pid = 1 \,\&\, P_{Oidd} = 0) \forall i \in T, d \in D \tag{3.63}$$

$$O_{id} = 0 \Rightarrow P_{id} = P_{i(d-1)} \forall i \in T, d \in \{2,...,D\} \tag{3.64}$$

$$\sum_{k \in \{0,1\}} (O_{i(d+k)} \neq 0) * P_{i(d+k)} < 2 \forall i \in T, d \in \{1,...,D-1\} \tag{3.65}$$

The constraint (3.62) defines a break to be 1 if a team plays two consecutive home games or away games. The constraint (3.63) ensures two opponents have opposite venue when they play a game. The constraint (3.64) assigns a team to its previous venue if it has an off day. The constraint (3.65) makes sure no team can play consecutive home games. The objective (3.61) is to minimize the total breaks.

We run Model 12 to get the random schedule, then we use Model 13 to assign the game venues. We can get optimal solution for problem of 10 teams in seconds, but it take more than 10

minutes to solve the problem of 12 teams. We believe some improvement methods such as those used by Régin to break symmetries could be helpful to decrease the computation time, but it is not a trivia task. Another observation is we get some shcedules with minimum break closing to the team number. For example, there is a schedule for 8 teams with minimum break at 6.

For the comparation, we developed IP model for break minimization phase. The varialbes definiton are similar to those from Trick (2001). The following is the model:

Model 14 Break Minimization Phase IP-model

$$\min \sum_{i \in T} \sum_{d \in \{2,\ldots D\}} brk_{id} \tag{3.66}$$

$$start_{id} + \sum_{t<d} (to\_home_{id} - to\_away_{id}) = at\_home_{id} \forall i \in T, d \in D \tag{3.67}$$

$$brk_{id} = (at\_home_{id} = at\,home_{id} - 1) * (O_{id} \neq 0) \forall i \in T, d \in D \tag{3.68}$$

$$O_{id} = 0 \Rightarrow at\_home_{id} = at\_home_{id-1} \forall i \in T, d \in D \tag{3.69}$$

$$O_{id} = j \Rightarrow (at\_home_{jd} + at\_home_{id}) = 1 \forall i, j \in T, d \in D \tag{3.70}$$

$$to\_home + to\_away \leq 1 \forall i \in T, d \in D \tag{3.71}$$

$$at\_home + to\_home \leq 1 \forall i \in T, d \in D \tag{3.72}$$

$$\sum_{k \in \{0,1\}} at\_home_{i(d+k)} * (O_{i(d+k)} \neq 0) < 2 \forall i \in T, d \in \{1,..D-1\} \tag{3.73}$$

The definitions are variables are explained as the following: to_home_{id} and to_away_{id} equal 1 if a team goes home or go to plat a road trip after day d respectively, 0

otherwise. brk_{id} is defined as a break when a team play two consecutive home games or road games. The constraint (3.67) and (3.68) defines the variable *at_home* and *brk* respectively. The constraint (3.69) claims a team to stay at the previous venue if it has an off day. The constraint (3.70) makes sure one team at home and the other on road if they play a game. The constraint (3.73) forces a team not to play consecutive home games. The constraint (3.71) ensures a team not to go home and go on road at the same time on a specific day. The constraint (3.72) is added to improve the computation. The objective (3.66) is to minimize the total breaks for the whole tournament.

We use Model 12 to generate schedules then we run a computation studies on the IP model and CP model. The following table presents results:

Table 5 Break Minimization Phases

Team	IP			CP		
	Node	Time	Optimal	Fail	Time	Best
6	0	0.12	2	168	0.08	2
8	3	1.14	8	2225	0.31	8
10	0	1.82	8	1210723	97.43	8
12	0	3.79	16	5899790	600	16
14	3	14.25	27	5053636	600	27
16	52	36.68	33	3673985	600	34
18	14	62.77	42	2851148	600	46
20	47	126.09	50	2329830	600	56

We set the computation time to 600 seconds for both models. CP model has slight advantage for problem of eight teams or less, but it cannot finish the searching for problem of twelve teams or more. IP model can get the optimal solution for problem of 20 teams in 126 sections.

3.5.2 First-Break-Then-Schedule

In this section we look at the method consisting of two parts: first generate a schedule with minimum break without considering opponent, then generate the opponents in the second phase. A lot of researches have been done on the problem to minimize breaks in the time constrained tournament schedules. Because of the difference in the structures, only some of the results are applicable for the time relaxed problems. In the following we will list some properties related to the time relaxed tournament.

The limitation of the length of consecutive games is an important parameter for a time relaxed tournament. FronEek (2003) stated that a round robin tournament with one bye in each round has a unique schedule without break. If there is no limitation on the length of the consecutive games, every time relaxed tournament has at least one schedule without break.

Theory 3.1 If neither consecutive games nor consecutive off days are allowed, a tournament consisting of 2n teams will have at least 2n-4 breaks in a 2*(2n-1) time slot relaxed tournament.

Proof. We will use the 6 team instance as a example to demonstrate. Here 2n = 6, total time slots is 2*(2n-1) = 10. Because neither consecutive games nor off days are allowed, therefore all the games are distributed every other day. Like the time constrained tournament, there are only two possible game patterns have no breaks: HAHAH or AHAHA. For the first game, it can be carried out either on the first day or the second day.

Without lost of generality, we can assume team 1 and Team 3 have the HAHAH pattern, starting the game on the first day and second day respectively. Team 2 and team 4 have the AHAHA pattern, starting the game on the first day and second day respectively. All these 4 teams have no break as can tell. Two identical HAP patterns cannot be assigned to two teams, otherwise these two teams cannot play each other. Therefore every one of the rest 2n-4 team will have at least one slot different with one of the four teams, which will bring a break. As a result, the total break is at least 2n-4.

Table 6 An example for 6 teams

	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10
Team 1	H		A		H		A		H	
Team 2		H		A		H		A		H
Team 3	A		H		A		H		A	
Team 4		A		H		A		H		A
Team 5										
Team 6										

The first step for this method is to generate a HAP set. A common question is to decide whether a HAP set is feasible. The following is the formal definition of this problem.

Definition 12 HAP feasibility problem

Input: A HAP set $h \in \{0, 1, 2\}^{n\times(2n-2)}$

Question: Is h feasible?

The HAP set feasibility problem exists in the time constrained tournament scheduling as well. The HAP set feasibility problem is still open for the time constrained scheduling. Miyashiro et al.(2003) proposes necessary condition (3.74) for HAP sets to be

feasible. T' is a subset teams belong to T, p is any time period. C_0(T', p) and C_1(T', p) represent the number of zeros and ones respectively:

$$\sum_{p\in P} \min\{C_0(T', p), C_1(T', P)\} - \frac{|T'||T'-1|}{2} \geq 0 \forall T' \subset T \tag{3.74}$$

In each time slot, the maximum possible games among teams in subset T' is restricted to the minimum number of home or road games respectively. The summary of these minimum numbers should be equal or greater than the required games among teams in the subset, which is $\frac{|T'||T'-1|}{2}$. This condition can be checked in polynomial time. Miyashiro conjectured that this condition is sufficient. Miyashiro et al. also show that, for pattern sets with a minimum number of breaks and no more than 26 teams the condition (3.74) is both necessary and sufficient. However, the sufficiency is still not proven for other instances. Actually the constraint is applicable to the time relaxed schedules as well.

Briskorn (2007) provides an IP model stated that it is tighter than the condition given by Miyashiro. Similarly we present the following IP model to check the feasibility of the HAP:

Model 15 HAP feasibility problem IP model

$$\max z_h = \sum_{i\in T} \sum_{j\in T, j<i} \sum_{d\in D} x_{ijd} \tag{3.75}$$

$$\sum_{d\in D} x_{ijd} \leq 1 \forall i, j \in T, j < i \tag{3.76}$$

$$\sum_{j\in\{T\setminus i\}} x_{jid} \le 1 \forall i \in T, d \in D \tag{3.77}$$

$$\sum_{j\in T} x_{ijd} \le (h_{id} < 2) \forall i \in T, d \in D \tag{3.78}$$

$$x_{ijd} \le \left| h_{id} - h_{jd} \right| \forall i, j \in T, j < i, d \in D \tag{3.79}$$

The objective function (3.75) represents the goal to maximize the number of matches. The constraint (3.76) forces each pair of teams to meet at most once while constraint (3.77)restricts the number of matches per team and day to be less than or equal to one. These two constraints ensure a single RRT. Symbol h_{ip} is the entry of HAP set corresponding to team *i* and day *d*. The constraint (3.78) and (3.79) ensures two teams can meet on a day only if one play at home and the other play on the road. If the HAP sets are feasible, the objective number should be equal to $\frac{n(n-1)}{2}$. Briskorn proved the IP model for time constrained model is tighter than the condition given by Miyashiro. The sufficiency of this condition is not proven either.

3.5.3 GOP problem

As previously stated, we assume the total available days are double size of the games for all tournament schedules. There is a decision to pick the game days. We can formally define the game day decision problem for a single round robin tournament as the following:

Definition 13 GOP problem

Given a set T, |T| = n, a set D, |D| = 2n – 2, each double (i, d) ∈ T x D equals 1 if team i plays a game on day D, and cost C_{id} for each (i, d) ∈ (T × D). A feasible solution to the GOP y problem corresponds to a set of n(n−1) doubles such that*

(i) for each team, exactly n-1 doubles are chosen

(ii) for each team, no more than two doubles within a consecutive fixed number are selected

(iii) the selected doubles can be used to generate a feasible timetable

The objective of the game day problem is to find a feasible solution having the minimum cost of chosen doubles cost.

The first two conditions can be easily modeled. However, the third condition is not straightforward. One obvious characteristics of feasible GOP is the number of the teams playing games on a specific day should be even. If we employ this requirement only for iii condition, we get the following IP model:

Model 16 GOP generating model

$$(\sum_{i \in T} x_{id} \%) 2 = 0 \forall d \in D \tag{3.80}$$

$$\sum_{d \in D} x_{id} = |T| - 1 \forall i \in T \tag{3.81}$$

$$\sum_{k=\{0,1,2\}} x_{i(d+k)} \leq 2 \forall i \in T, d \in \{1, \ldots, D-2\} \tag{3.82}$$

$$x_{id} \in 0,1 \forall i \in T, d \in D \tag{3.83}$$

Binary variable x_{id} is equal to 1 if team i plays a game on day d, otherwise 0. Constraint (3.81) is to make sure the total number of games for each team is equal to n-1,

constraint (3.80) is to ensure the total number of teams will play game on any day is not odd, which is impossible for a tournament. Constraint (3.82) represents the requirement every team will play no more than two consecutive games.

The above model provides necessary conditions. Are these conditions enough to provide a feasible schedule for a tournament? The answer is no. Here we will give a contradiction example.

Table 5 is a game day assignment generated by Model 16.We can verify that this GOP pattern has no feasible schedule. If there are only two teams playing on a day, we know these two teams are opponents to each other on that day. Therefore we can decided the opponent schedule on day 3, day 4, day 5, day 7 and day 7. After this step, we get a partial timetable in table 6. Now we check the schedule of team 5. On day 9, there are three possible opponents for team 5: team 4, team 5, or team 6. However, all of these three teams have been assigned already. Therefore there is no opponent available for team 5 on day 9. This example proofed that Model 16 cannot guarantee a feasible schedule.

Table 7 An GOP generated by Model 16

	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10
Team 1	1	1	0	0	0	1	0	1	0	1
Team 2	1	1	0	0	0	1	0	1	0	1
Team 3	1	0	0	1	0	1	0	0	1	1
Team 4	0	1	1	0	1	0	0	0	1	1
Team 5	0	0	1	1	0	1	1	0	1	0
Team 6	1	1	0	0	1	0	1	0	1	0

Table 8 Partial schedule based on Table 7

	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10
Team 1								2		

Team 2								1		
Team 3				5						
Team 4			5		6					
Team 5			4	3			6		?	
Team 6					4		5			

Now it is natural to ask the question: what are the characteristics of a feasible GOP? We formally define the GOP feasibility problem as follows:

Definition 14 GOP feasibility problem

Input: A GOP set $g \in \{0, 1\}^{n\times(2n-2)}$

Question: Is g feasible?

The GOP feasibility problem is important. At the early stage of the NBA scheduling, the first step is to decide the game days based on the available days provided by individual team. The solution for the feasibility problem provides a good way to detect the infeasibility caused by the conflicting requirements from the teams.

As we noticed, the necessary conditions embraced in Model 16 are not sufficient. Therefore we need to find more characteristics of GOP feasibility. There are similarities between GOP and HAP.

Similar to the HAP set feasibility, we propose the following necessary condition for the GOP feasibility:

$$\sum_{p\in P}(\sum_{i\in T'} x_{ip}) / 2 \geq \frac{|T'||T'-1|}{2} \forall T' \subset T \tag{3.84}$$

T' is a subset of teams T. The maximum possible games among the subset T' on a specific day is restricted to the total number of the non-zero variables divided by 2. For

example, if the subset contains 5 teams and all the GOP entries are 1 on day 1, then there should be equal to or less than 2 games among the teams from this subset on day 1. If the sum total of these numbers are strictly less than $\frac{|T'||T'-1|}{2}$, this GOP is infeasible.

This condition is necessary, but its sufficiency is hard to proven. The IP model for the feasibility of GOP pattern set is listed as the following:.

Model 17 GOP feasibility Model

$$\max z_g = \sum_{i \in T} \sum_{j \in T, j<i} \sum_{d \in D} x'_{ijd} \tag{3.85}$$

$$\sum_{d \in D} x'_{ijd} = 1 \forall i, j \in T, j < i \tag{3.86}$$

$$\sum_{j \in \{T \setminus i\}} x'_{jid} \leq 1 \forall i \in T, d \in D \tag{3.87}$$

$$x'_{ijd} \leq G_{jd} \forall i, j \in T, j \neq i, d \in D \tag{3.88}$$

$$x'_{ijd} \in \{0,1\} \forall i, j \in T, j \neq i, d \in D \tag{3.89}$$

The objective function (3.85) represents the goal to maximize the number of games. The constraint (3.86) forces each pair of teams to meet exactly once while constraint (3.87) restricts the number of games per team play on a specific day to be less than or equal to one. These two constraints ensure a single RRT. G_{id} is the entry of GOP set corresponding to team *i* and day *d*. The constraint (3.88) ensures only teams with entry 1 can meet for a game on a specific day. If the GOP sets are feasible, the

objective number should be equal to $\frac{n(n-1)}{2}$. This IP model can be checked in polynomial time. For example, the GOP pattern listed in the table 5 returns the optimal number of 9, which is less than $\frac{6*(6-1)}{2}$ = 15. Therefore that GOP is not feasible. We conjecture that this condition is sufficient for the GOP feasibility. Unfortunately we cannot provide the formal proof. We can run some test for this model. We tested the instances of 6,8,10, 14 teams. It detects all the infeasible models.

CHAPTER IV

TIME RELAXED TRAVELING TOURNAMENT PROBLEM

Traveling distance is a major concern in the tournament scheduling problems. Every team wants to minimize the travel distance as much as they can. Long traveling distance means long traveling time which is highly unwanted during the tightly scheduled season. Besides that, long traveling distance has some other undesired results, such as players fatigue, high traveling cost, etc. On the other side, a certain amount of traveling distance is necessary. The administrative agency has to make sure the traveling distance is fair for all teams when it attend to minimize the total traveling distance for the whole league.

This chapter will look at the traveling distance factor in the time relaxed tournament scheduling problem. First we will review the related work. Then the problem of Time Relaxed Traveling Tournament Problem (TRTTP) is presented. In the third section we present algorithms to tackle the TRTTP.

4.1 Minimizing traveling distance in practical applications

If a team is not required to return home after every game, it is possible to combine consecutive away games into one trip to save traveling distance. During the middle 70's, the surging oil price urged teams to find a way to save traveling cost. As a scheduling

constraint, traveling distance has been studied in several practical sports scheduling problems since then. Since several problems are basketball league related, they have been reviewed in Chapter II. In this section we will focus on the literatures regarding the minimizing traveling distance in practical applications.

The problem of minimizing traveling distance can be models as an IP problem. However it is beyond the current computer computation capability to solve these IP problems because the size is too large. Therefore some effective heuristic algorithms are developed. Campbell and Chen published the first paper with minimizing traveling cost as an objective in 1976. They faced the problem of minimizing the traveling distance for a baseball league. They developed a heuristic approach based on the pairing. The major idea in their approach is to pair teams, visitors will reduce the traveling distance by playing the pair teams in one trip rather than in two separate trips. Many algorithms resembling this two-phase heuristic approach were used for some other applications in the following years. To name a few, Ball and Webster (1977) solved a college basketball conference scheduling problem; Bean and Birge (1977) tackled the NBA scheduling problem.

With the development in the computation capabilities, metaheuristic algorithms are used to address the minimizing traveling distance problem. Costa(1995) developed a generic algorithm to tackle the NHL scheduling problem with minimizing traveling distance as an objective. Wright (2006) uses a type of simulated annulling to solve the scheduling problem for the New Zealand basketball league, traveling distance is embodies in the objective functions.

Besides the practical problems, there is a theoretical problem with minimizing traveling distance as the objective has drawn a lot of research attentions since it was introduced. It is called the traveling distance problem (TTP), which is the main topic in the following section.

4.2 Traveling Tournament Problem (TTP)

Traveling tournament problem (TTP) was originally introduced in 2001 by Easton, Nemhauser, and Trick. It is motivated by the Major League Baseball (MLB) scheduling problem. The TTP captures the silent features of the MLB scheduling problem, which combines the traveling distance minimization and game pattern feasibility.

4.2.1 The definition of TTP

The formal definition of the TTP is replicated as following:

Definition 15 Traveling tournament problem (TTP)

Input: n, the number of teams; D an n by n integer distance matrix; L, U integer parameters.

Output: A double round robin tournament on the n teams such that

– The number of consecutive home games and consecutive away games are between L and U inclusive, and

– The total distance travelled by the teams is minimized.

There are two optional constraints: the games between same opponents cannot happen in consecutive time slots, which is called no repeater constraint; the second round

is mirrored to the first found, which is called mirroring constraint. Notice these two requirements cannot be relevant at the same time, because the mirrored schedule guarantees no repeater.

After the introduction, TTP draw a lot of research interest from both operations research and constraint programming communities. Two instances of the TTP were listed in the original paper. The first one is called circle instances. All teams are regarded as nodes on a circle. The distance for all the adjacent nodes is constant. The second is National League instance. The distances for teams are based on the National League teams in the Major League Baseball. Different instances can be gained by varying the number of teams in the tournament, such as NL4, NL6, NL8, etc. Several additional instances were added afterward. For examples, the constant instances in which all the distances among teams are same, the super instances which are based on a rugby league, the NFL instances which are based on the National Football League from the United states, and so on.

4.2.2 Research progress

After the TTP was presented, a lot of algorithms has been attempted to solve the problem. All the instance classes together with the current best upper and lower bounds can be checked at Trick's "Traveling tournament problem challenge" page. So far, finding the optimal solution for 10 teams is still open.

Easton et al (2002) present a method based on the independent lower bound (ILB) in their TTP introduction paper. First of all they calculate the traveling distance lower bound for each team without consideration of the other teams. Later on the trips

generated in the first step was combined according to the timetable constraints. This method solves the problem of NL4 and NL 6 to optimality.

Because of the enormous amount of the variables, this method cannot solve the problem of eight teams. Easton et al (2002) present a method based on the column generation techniques for the eight team instance. The master problem uses IP to assign tours to teams. The pricing problem uses CP to find all the tours with negative reduced cost for teams. They report the program generated a solution for NL8 after 24 hours of parallel computation on 20 CPU, but they cannot prove the optimality. Sirnich (2009) proved that the solution indeed is the optimal solution.

The instances of ten teams and above are still unsolved. Besides methods to reach optimality, various methods employing heuristic algorithms to find approximate solutions have been developed. Anagnostopoulos et al (2006) used a method called TTSA to get some best-to-date solutions for many instances. TTSA is a variation of the typical simulated annealing algorithms. The searching neighborhood is constructed by five types of moves: *SwapHomes, SwapRounds, SwapTeams, Partial SwapRounds and PartialSwapTeams*. An ejection chain is used to restore the structure after the moves. All the constraints except the no repeaters can be accomplished by this procedure. The no repeaters constraint is modeled in the objective function. This algorithm generated some best known solutions to many instances.

In addition to the simulated annealing, some other heuristic algorithms have been used to tackle the TTP as well. Schaerf (2007) used tabu search, Lim et al (2006) used an algorithm combining simulated annealing and hill-climbing, etc. All these effective

algorithms are proved to be promising method if the objective is to get a good feasible solution, rather than to attain the optimal solution.

4.3 Time Relaxed Traveling Tournament Problem (TRTTP)

If we put the traveling tournament problem in the time relaxed context, we get a new problem called Time Relaxed Traveling Tournament Problem (TRTTP). TRTTP shares most characteristics with the TTP meanwhile carries some unique properties. In this section we present the formal definition first. Secondly, we will look at the complexity of this problem. Then we will present algorithms used to tackle different instances.

4.3.1 Definition

The formal definition of TRTTP is listed as the following:

Definition 16 Time relaxed traveling tournament problem:

Input: n, the number of teams; D, an n by n integer distance matrix; L, U, B, O integer parameters.

Output: A double round robin tournament on the n teams such that

(i) The number of consecutive home games and consecutive away games are between L and U inclusive;

(ii) The consecutive games without off days is less than B;

(iii) The consecutive off days is less than O;

(iv) The total available game slots are 4(n-1);

(v) The total distance travelled by the teams is minimized.

There are three new constraints compared to the time constraint TTP. If a team play games on two consecutive days, it has a back-to-back game on the second time slot. Normally teams want to avoid such situation, so we introduce the new parameter B to limit the length of the consecutive games. Analogously, we restrict the length of consecutive off days. The third new constraint is the number of the total available time slots. We define the total available game slots to be 2 times the number of the games for each team. Because each team has to play 2(n-1) games in a double round tournament, the total available time slots are 4(n-1).

4.3.2 Complexity

For comparison, we will look into the complexity of the TTP before examine the complexity of TRTTP.

In extreme cases, the complexity of the TTP is obvious. For examples, when L and U are both equal to one, the optimal solution is a constant. When U is equal to two, a polynomial time algorithm analogical to find the minimum weight for a complete graph can be attained. When U is equal to n-1, this is a traveling salesman problem (TSP), which has been proven to be a NP-complete problem. Actually the analog to the TSP stands in many cases. The round robin tournament problem is analogical to the multiple-salesman traveling problem to some extent, but it is more difficult because of the one-factor constraint. By the time of this dissertation writing, the complexity of the TTP is still an open problem when L and U equal to a value between 3 and n. Easton (2001) conjectured that TTP is a NP-hard problem except some extreme instances. She made a statement that even single team problem is strong NP-complete when L and U take

certain values. This problem can be reduced by the partition into isomorphic sub-graphs restricted to path problem (PISP), which is proven as NP-complete.

Similarly, we can easily get the complexity of TRTTP when the parameters take extreme values. If U is 1, every team has to travel round trip for each opponent. Therefore the traveling distance for each team is a constant. A polynomial algorithm for TRTTP is trivia.

When U equals 2, we can find a polynomial time algorithm to solve TRTTP. We assume the distance matrix satisfies the reflexive properties and triangle inequality. It is obvious that a team will have a shorter traveling distance if it has a consecutive away trip to visit two teams together, rather than two round trips to each team separately. Based on this observation, we can tell easily if a team wants to travel as short as possible, it has to have as many consecutive away trips as possible. The away opponents should be paired to minimize the traveling distance. This is the pairing theory used in Campbell and Chen (1977). Figure 3 illustrate an example when team number is eight. Team i will take three separate away trips to visit pair teams, then it will take a round trip to the team paired with itself. No matter what the designated team i is, the pairing results is the same. The following model can be used to find the pairs:

Model 18 Pairing Models

$$\min \sum_{j\in\{i+1,T\}} p_{ij} * d_{ij} \forall i \in T \tag{4.1}$$

$$\sum_{j=\{1,..,i-1\}} p_{ji} + \sum_{j=\{i+1,...,T\}} p_{ij} = 1 \forall i \in T \tag{4.2}$$

$$p_{ij} = \{0,1\} \forall i, j \in T, i < j \tag{4.3}$$

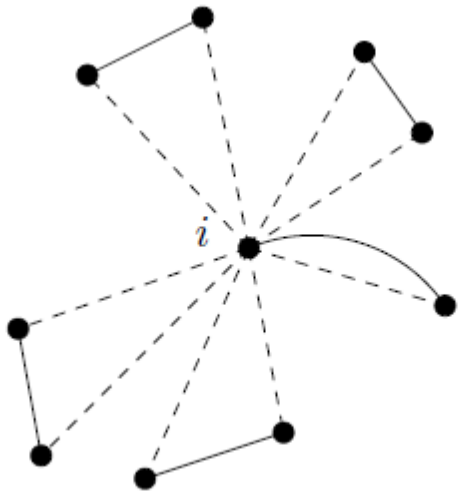

Figure 3 Optimal travel pattern for U=2

The Binary variable P_{ij} equals 1 when team i is paired with team j. The objective function (4.1) is to minimize the traveling distance for the pairs, which can be considered as the minimum weight matching. The equation (4.2) restrains each team to be exactly in one pair. After the pairs are generated, we can make a timetable based on the pairs. This algorithm is polynomial.

When U equals to a value between three and n-1, we cannot provide a formal proof the complexity. We conjecture the problem is NP-hard because it is analogical to the TSP.

4.3.3 Independent Lower Bound

If we make the schedule with minimum traveling distance for one team without considering the other teams, it is a strong lower bound for that team. If we sum up the lower bounds for all teams in the tournament, we have a lower bound which is called

independent lower bound (ILB). We will introduce the single team problem first since it is the fundamental idea behind the ILB.

Definition 17 Time Relaxed Single Team Problem (TRSTP)

Instance: n, the number of teams; t_r, the designated team; D, an n by n integer distance matrix; L, U, B,O integer parameters, k is the integer threshold.

Questions: Does there exist a tour for t_r meeting the following constraints meanwhile the traveling distance is less than or equal to k?

(i) The number of consecutive home games and consecutive away games are between L and U inclusive;

(ii) The consecutive games is less than B;

(iii) The consecutive off days is less than O;

(iv) The total available game slots are 4(n-1);

The condition (i) restrains the length of consecutive home or away games. Meanwhile the length of the consecutive games regardless of the game type is restricted by condition (ii). The condition (iii) forces team to avoid too long off days. The last condition (iv) defines the number of available time slots. The objective is to find a tour with traveling distance is less than or equal to the threshold number. We will introduce the concept of tour in the following paragraph.

A tour is a vector of length of total available time slots. Each element in the vector represents the venue where the team visits on that day. It is used to descript the trip taken by the team during the tournament. We use the number of the team to notate the venue. For simplicity, if a team has consecutive away games, we assume the team will travel to the opponent's venue directly after the first away game rather than returning to home. If there is an off day between two consecutive away games, we assume the team

will return home. In real world, it is more complicated than this. For example, a team from the east coast will have a long trip to visit the west coast teams every season. Normally they will stay on the road between games rather than return home. However, sometimes teams prefer to stay at home during off day between two away games if it is possible. It is believed among players and coaches that they can have better rest at home. It is so called "home pillow" advantage among players. Taking into all these considerations will make our model too complicated. As a result we assume teams will return home on off days between two away games. We use number 0 to represent the off day. For example, in a instance with n = 4, L = 1, U = 3, B = 3, a possible tour for team 1 is {1,1,2,1,3,1,1,1,1,4,1,1}, note that each opponent venue appears exactly once, and home venue appears 9 times. Actually the return home assumption will not change the optimal solution for TRSTP. If we assume teams will stay on road on off days, only the travelling day will be changed, the travelling distance will remain the same.

If U equals to 1 or 2, the analysis of complexity for the TRTTP still stands for TRSTP. Now we examine the instances when U equals three. We can use both integer programming and constraint programming to find the ILB for a team. The constraint programming modeling is outlined here. It is similar to the method used by Easton (2001). The basic variables are the venues where the team will play on each day. The domains for these variables are all the teams in the tournament. Two dummy variables are added for the purpose of calculating traveling distance: day 0 and day n+1. Both of the dummy variables are set to the home team. Traveling distance for consecutive time slots are set as variables as well, but only used for calculating distance. The objective is to minimize the total traveling distance.

Additionally, there is a need for one more type variables in our model. In the time constrained model, every team has two options in a time slot: plays either a home game or a road game. In the time relaxed model, every team has three options: home game, road game and off day. Whether teams return home or stay on road will not change the optimal value. We assume the team will return home on off days during an away trip, and we assume teams only travel the day before the game day.

We list the CP model to calculate the ILB in the following Model 19. The variable V_d refers to the venue where team i will stay on day d, the domain for this variable are all teams. The binary variable H_d equals 1 if team i plays a home game on day d, otherwise 0. C_d are the traveling distance for team i on day d.

Model 19 ILB – CP model

$$\min \sum_{d \in \{0..D\}} C_d \tag{4.4}$$

$$\sum_{d \in D} (v_d = j) = 1 \forall j \in \{T \setminus i\} \tag{4.5}$$

$$\sum_{d \in D} (H_d = 1) = nbTeams - 1 \tag{4.6}$$

$$\sum_{k \in \{1..B\}} (V_{(d+k)} != i \,||\, H_{(d+k)} = 1) < B \forall d \in \{1, ..., D - B\} \tag{4.7}$$

$$\sum_{k \in \{1..O\}} (V_{(d+k)} = i \,\&\, H_{(d+k)} = 0) < O \forall d \in \{1, ..., D - O\} \tag{4.8}$$

$$v_0 = 0 \tag{4.9}$$

$$v_{D+1} = 0 \tag{4.10}$$

$$Cd = dis[v_d, v_{d+1}] \forall d \in D - 1 \qquad (4.11)$$

The objective (4.4) is to minimize the total travel distance for team i from the starting day to the finishing day. The constraint (4.5) ensure team i will visit every other team once. Then the constraint (4.6) forces team i to play the required number of home games. The following (4.7) and (4.8) set the limitation on the consecutive games and off days respectively. Then (4.9) and (4.10) declare team start and finish the tournament at home. The last equation (4.11) calculates the traveling distance for each day. The notation $dis[v_d, v_{d+1}]$ refers to the distance between two venues where team I stay on day D and the following day.

To clarify, we will list an example in the OPL language in the following. The parameters Ub and Od represent the variable U and O respectively.

```
using CP;
int nbTeams = ...;
int nbDays = ...;
int Ub = ...;
int Od = ...;
int home = ...;
int distance[1..nbTeams,1..nbTeams]=...;
dvar int venue[0..nbDays+1] in 1..nbTeams;
dvar int homegame[1..nbDays] in 0..1;
dvar int travel[0..nbDays] in 0..1380;
dexpr float totalcost= sum(d in 0..nbDays)travel[d];
```

```
subject to
{
//visit each opponent once
        forall(i in 1..nbTeams: i!=home)
                sum(d in 1..nbDays)(venue[d] == i) == 1;
//stay at home except road games
        sum(d in 1..nbDays)(venue[d]== home) == nbDays-nbTeams+1;
//nbTeams-1 home games
        sum(d in 1..nbDays)homegame[d] == nbTeams -1;
//can't have road game and home game together
        forall(d in 1..nbDays)
                venue[d]!= home =>homegame[d] == 0;
//No more than Ub consecutive games
        forall(d in 1..nbDays-ub)
                sum(j in 1..ub)(venue[d+j]!=home || homegame[d+j] ==1)<Ub;
//No more than ub consecutive off days
        forall(d in 1..nbDays-Ob)
                sum(j in 1..Ob)(venue[d+j]==home && homegame[d+j] == 0)<Od;
//start and finish the tour at home
        venue[0] == home;
        venue[nbDays+1] == home;
        forall(d in 0..nbDays)
                travel[d] == distance[venue[d],venue[d+1]];
```

}

We set the parameters used in the model as the following: L = 1; U=B = 3; O = 5. We run the above OPL code in ILOG Studio 6.0 for NL 6 and NL8 instances by using the default propagation setting. The program cannot finish the searching after thirty minutes. For both the NL 6 and NL 8 (A) instance, the program cannot get any improvement after the first minute. Table 9 and Table 10 provide the computation results respectively. The number 38670 for eight team instance is same as the result from Easton (2001). The number 22557 for six team instance is smaller than Easton's 22969. The difference can be caused by the non-repeater constraint in Easton's model, which is not included in our model. We will prove that these numbers are actual the lowest bound a team can get.

Table 9 ILB-CP computation results for NL 6 teams

Team	Distance
1	4147
2	3328
3	3200
4	3711
5	4953
6	3218
Total	**22557**

Table 10 ILB-CP computation results for NL 8 teams

Team	Distance
1	4772
2	4491
3	4340
4	5165
5	6654
6	4131
7	4278
8	4839
Total	38670

When U equals three, a team has three options for a trip: one-team trip, which means visiting one team and returning to home; two-team trip, which means visiting two teams then returning home, three-team trip, which means visiting three teams before returning to home. Similar to the pairing model, we will use an IP model to find the optimal trip pattern for a team when U equals three:

When U is equal to 2, the optimal travel pattern for a team is to have as many pair trips as possible. At first glance, the travel pattern with as many three-team trip as possible seems attractive for a team when U is equal to 3. Actually it is not the case. For one thing, teams cannot be divided into three-team groups under many circumstances. Figure 4 and Figure 5 are two examples. Additionally, sometimes it is better for a team to have a pair of two-team trips rather than a three-team trip plus a round trip to a single team. For example, the optimal travel pattern for team 6 in the instance NL 8(A) is to have two pair trips. It is illustrated in Figure 6. As a result, the IP model to find the optimal traveling pattern when U equals three should consider all these circumstances. The following Model 20 can be used to find the optimal travel pattern when U equals three.

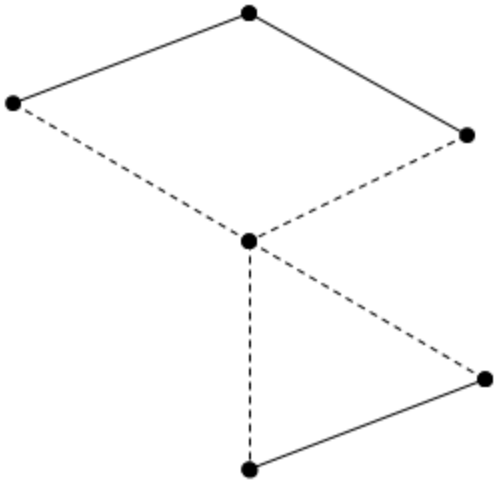

Figure 4 Optimal travel pattern when U = 3 for n = 6

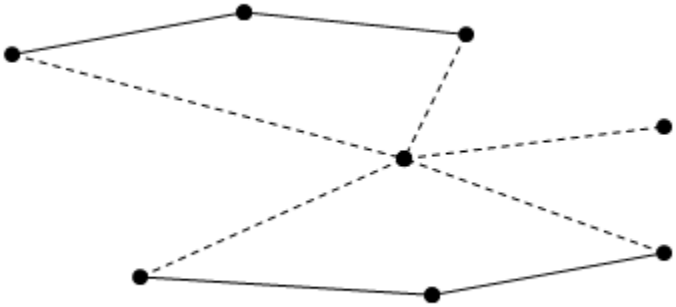

Figure 5 An optimal travel pattern when U = 3 for n = 8

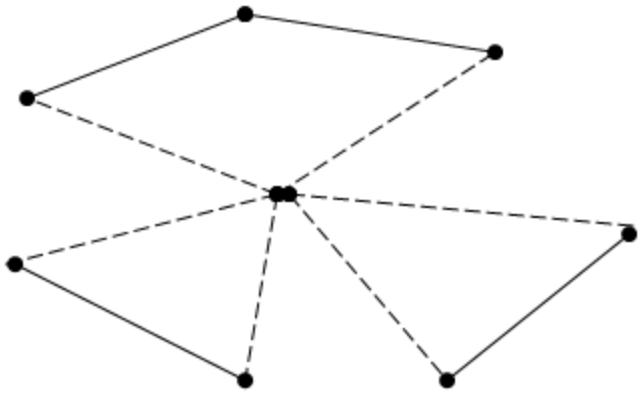

Figure 6 An optimal travel pattern with two pair trips when U =3 for n = 8

Model 20 Grouping model when U =3

$$\min(\sum_{i\in T}\sum_{j\in T}\sum_{k\in T} R_{ijk} * G_{ijk} + \sum_{i\in T}\sum_{j\in T} P_{ij} * T_{ij}) \tag{4.12}$$

$$\sum_{j\in T}\sum_{k\in T}(G_{ijk} + G_{jik} + G_{jki}) + \sum_{j\in\{1,\ldots,i)} P_{ji} + \sum_{j\in\{i+1,\ldots T\}} P_{ji} = 1 \forall i \in T \tag{4.13}$$

$$R_{ijk} = dis_{hi} + dis_{ij} + dis_{jk} + dis_{kh} \forall i, j, k \in T \tag{4.14}$$

$$T_{ij} = dis_{hi} + dis_{ij} + dis_{jh} \forall i, j \in T \tag{4.15}$$

$$G_{ijk} \in \{0,1\} \forall i, j, k \in T \tag{4.16}$$

$$P_{ij} \in \{0,1\} \forall i, j \in T \tag{4.17}$$

There are two binary variables used in this model. This first one is G_{ijk} , which is equal to 1 if a team visit i first, then go from i to j, j to k, finally return from k to home;

otherwise 0. The second one is P_{ij} , which is equal to 1 if a team visits team I first, then goes from i to j, and returns from j to home. The objective function (4.12) is to minimize the total traveling distance for all trips. The constraint (4.13)forces each team belongs to exactly one trip, either three-team or two-team. The constraint (4.14) and (4.15) defines the traveling distance for three-team trip and two-team trip respectively.

The optimal solution with traveling pattern can be gained in seconds. The computation results match those listed in Table 9 and Table 10. Therefore we know those results are optimal.

4.3.4 Solution Method

Three methods based on the ILB are outlined in this section. The first one is a CP model for teams return home on off days during the consecutive away games. The second one is a CP model for teams stay on road on off days during the consecutive away games. The last one is a decomposition method which is based on the concept of optimal travel pattern.

The computation results from Model 19 can serve as a strong lower bound for TRTTP. We can modify it for calculating the minimum traveling distance for all teams. We set the parameters as the following: the maximum consecutive off days O is 4, maximum consecutive games U=B= 3. The flowing is the model for the whole tournament.

Model 21 TRTTP – Return Home on off days Model

$$\min \sum_{i \in T} \sum_{d \in \{0,\ldots D\}} C_{id} \tag{4.18}$$

$$v_{i(d+k)} \neq i + \sum_{j \in \{T \setminus i\}} (v_{j(d+k)} = i) \leq 1 \forall i \in T, d \in D \tag{4.19}$$

$$\sum_{k \in \{0,\ldots,4\}} (v_{i(d+k)} \neq i) + \sum_{k \in \{0,\ldots,4\}} \sum_{j \in \{T \setminus i\}} (v_{j(d+k)} = i) \geq 1 \forall i \in T, d \in \{1,\ldots,D-4\} \tag{4.20}$$

$$\sum_{k \in \{0,\ldots,3\}} (v_{i(d+k)} \neq i) + \sum_{k \in \{0,\ldots,3\}} \sum_{j \in \{T \setminus i\}} (v_{j(d+k)} = i) \leq 3 \forall i \in T, d \in \{1,\ldots,D-3\} \tag{4.21}$$

$$\sum_{k \in \{0,\ldots,1\}} \sum_{j \in \{T \setminus i\}} (v_{j(d+k)} = i) \leq 1 \forall i \in T, d \in \{1,\ldots,D-1\} \tag{4.22}$$

$$\sum_{d \in D} (v_{jd} = i) = 1 \forall i, j \in T, i \neq j \tag{4.23}$$

The objective (4.18) is to minimize the traveling distance for all teams. The constraint (4.19) ensures every team will play at most one game on any day. The constraint (4.20) and (4.21) limit the length of consecutive games and off days respectively. The constraint (4.22) makes sure every team will not play consecutive home games. The constraint (4.23)forces each team will visit every other team exactly once. The definition for the cost is same as those in Model 19, which is not listed in the model. To clarify, we list the OPL for this model as the following:

```
using CP;
int nbTeams = ...;
int nbDays = 4*(nbTeams-1);
int distance[1..nbTeams,1..nbTeams]=...;
dvar int venue[1..nbTeams][0..nbDays+1] in 1..nbTeams;
dvar int travel[1..nbTeams][0..nbDays] in 0..1380;
```

```
dexpr float totalcost= sum(i in 1..nbTeams,d in 0..nbDays)travel[i][d];

minimize totalcost;

subject to

{

//visit each opponent once

        forall(i,j in 1..nbTeams:i!=j)

                sum(d in 1..nbDays)(venue[i][d] == j) == 1;

//stay at home except road games

        forall(i in 1..nbTeams)

                sum(d in 1..nbDays)(venue[i][d]== i) == nbDays-nbTeams+1;

 //No more than 3 consecutive games

        forall(i in 1..nbTeams,d in 1..nbDays-3)

                sum(k in 0..3)(venue[i][d+k]!=i)+sum(j in 1..nbTeams:j!=i, k in

0..3)(venue[j][d+k]==i)<=3;

//No consecutive home games

        forall(i in 1..nbTeams,d in 1..nbDays-1)

                sum(j in 1..nbTeams:j!=i, k in 0..1)(venue[j][d+k]==i) <=1;

//No more than 4 consecutive off days

        forall(i in 1..nbTeams,d in 1..nbDays-4)

                sum(k in 0..4)(venue[i][d+k]!=i)+sum(j in 1..nbTeams:j!=i, k in

0..4)(venue[j][d+k]==i)>=1;

//No more than 1 game on each day

        forall(i in 1..nbTeams, d in 1..nbDays)
```

```
        {
            (venue[i][d]!=i)+sum(j in 1..nbTeams:j!=i)(venue[j][d] == i)<=1;
            forall(j in 1..nbTeams:j!=i)
                venue[j][d] == i =>venue[i][d]== i;
        }
    //Start and finish at home
        forall(i in 1..nbTeams)
        {
            venue[i][0] == i;
            venue[i][nbDays+1] == i;
        }
        forall(i in 1..nbTeams,d in 0..nbDays)
            travel[i][d] == distance[venue[i][d],venue[i][d+1]];
    }
```

The one-dimensional variables in the ILB model are changed to two-dimensional variables accordingly in the above model. Variable *venue[i][d]* refers to the venue where team i will play on day d. The domains for these variables are all teams in the tournament. Two dummy variables for day 0 and day n+1 are added for each team. Both of the dummy variables are set to the team itself. Traveling distance on consecutive days are set as variables as well, but only used for calculating distance. We assume the team will return home if it has an off day.

For the comparison, the CP model for teams not to return home on off days during the away games is developed as well.

Model 22 TRTTP – Stay on road on off days Model

$$\min \sum_{i \in T} \sum_{d \in \{0,\ldots,D\}} C_{id} \tag{4.24}$$

$$\sum_{k \in \{0,1\}} (G_{i(d+k)} = 0) \leq 1 \forall i \in T, d \in \{1,\ldots,D-1\} \tag{4.25}$$

$$\sum_{k \in \{0,\ldots,3\}} (G_{i(d+k)} \neq 2) \leq 3 \forall i \in T, d \in \{1,\ldots,D-3\} \tag{4.26}$$

$$\sum_{k \in \{0,\ldots,4\}} (G_{i(d+k)} = 2) \geq 1 \forall i \in T, d \in \{1,\ldots,D-4\} \tag{4.27}$$

$$\sum_{m \in \{0,\ldots,k\}} (G_{i(d+m)} = 0) = 0 \Rightarrow \sum_{m \in \{0,\ldots,k\}} (G_{i(d+m)} = 1) \leq 3 \forall i \in T, k \in \{0,\ldots,D\}, d \in \{1,\ldots,D-k\} \tag{4.28}$$

$$\sum_{d \in D} (G_{id} = 1 \& O_{id} = j) = 1 \forall i, j \in T, i \neq j \tag{4.29}$$

$$\sum_{d \in D} (G_{id} = 0 \& O_{id} = j) = 1 \forall i, j \in T, i \neq j \tag{4.30}$$

$$G_{id} = 0 \Rightarrow V_{id} = i \forall i \in T, d \in D \tag{4.31}$$

$$G_{id} = 1 \Rightarrow V_{id} = O_{id} \forall i \in T, d \in D \tag{4.32}$$

$$G_{id} = 2 \Rightarrow V_{id} = V_{i(d-1)} \forall i \in T, d \in D \tag{4.33}$$

Three types of variables are used in this model: G_{id}, O_{id} and V_{id}. G_{id} equals 0 if team i has a home game on day d, equals 1 if team i has a road game on day d, and 2 if team i has an off day on day d. O_{id} refers to team i's opponent on day d, the domain is all teams. V_{id} refers to where team i stays on day d, the domain is all teams as well. The objective (4.24) is to minimize the total cost for all teams. The constraint (4.25) ensures

no team play consecutive home games. The lengths of home games and off days are defined by (4.26) and (4.27) respectively. The constraint (4.28) makes sure the length of away games will be equal or less than three. The constraints (4.29) and (4.30) makes sure a double round robin tournament. A team will stay at home on home game day, stay at opponent venue on road game day, and stay at the previous venue on off days, where are defined by (4.31), (4.32), and (4.33) respectively.

The decomposition method is based on the optimal tour. The first step is to generate optimal tours with minimum traveling distance for every team. We force the traveling distance for each team equals to the result from Model 19. The second step is to select tours for a feasible tournament. The following model is used to select tours:

Model 23 Tour selection model

$$\sum_{i \in Ut} i * x_i = 1 \forall t \in T \tag{4.34}$$

$$\sum_{i \in Ut} (v_{id} \neq i) * x_i + \sum_{i \notin Ut} (v_{id} = i) * x_i \leq 1 \forall t \in T, d \in D \tag{4.35}$$

$$\sum_{m \in \{0,\ldots,3\}} \sum_{i \in Ut} (v_{i(d+m)} \neq i) * x_i + \sum_{m \in \{0,\ldots,3\}} \sum_{i \notin Ut} (v_{i(d+m)} = i) * x_i \leq 3 \forall t \in T, d \in D - 3 \tag{4.36}$$

$$\sum_{m \in \{0,\ldots,4\}} \sum_{i \in Ut} (v_{i(d+m)} = i) * x_i + \sum_{m \in \{0,\ldots,4\}} \sum_{i \notin Ut} (v_{i(d+m)} \neq i) * x_i \leq 4 \forall t \in T, d \in D - 4 \tag{4.37}$$

$$\sum_{m \in \{0,1\}} \sum_{i \notin Ut} (v_{i(d+m)} = i) * x_i \leq 1 \forall t \in T, d \in \{1,\ldots,D-1\} \tag{4.38}$$

$$x_i \in \{0,1\} \tag{4.39}$$

$$v_{td} \in T \forall t \in T, d \in D \tag{4.40}$$

The binary variable x_i is equal to 1 if tour i is selected, otherwise 0. The variable v_{td} represents the venue where team t stays on day d. T is the set of the teams in the tournament. U is the set of tours. Ut represents the tours taken by team t. The set of constraints (4.34) forces each team to have on tour selected. The set of constraints (4.35) restrict a team not to play a home game and a road game on the same day. The set of constraints (4.36) restrains the length of consecutive games taken by team T, either home games or away games. The set of constraints (4.37) restrict the length of off days for every team. The constraint (4.38) makes sure every team will not play home games on consecutive days. The computation results are listed in

Table 11. All the computation studies were carried out on a personal computer running windows XP with 2.6GHZ CPU, 3G RAM. For the decomposition method, we set the initial solution pool for each team to be 200. We can get the optimal solution for problem of four teams in about two minutes. For the problem of six or eight teams, we cannot get the optimal solution. For the "return home" and "Stay on road" instances, we set the computation time to 1000 for four team instance, 10000 for six team instance and eight instance. The "Stay on road" model takes longer compared to the "Return home" model because of more variables. That could be the reason for the solution value is bigger.

Table 11 TRTTP computation results

Teams	Return home			Stay on road			Decomposition	
	F	T	Value	F	T	Value	T	value
4	8754920	1000	8044	4263785	1000	8044	121	8044
6	46492244	10000	22710	11217105	10000	22711		/
8	23615225	10000	40443	1977763	10000	44668		/

CHAPTER V

NBA SCHEDULING PROBLEM

The National Basketball Association (NBA) is a professional basketball league in North America. Currently there are 30 teams in the league, grouped into the eastern conference and western conference, each conference has 15 teams. Each conference is divided into three divisions. Each division has five teams. A NBA season consists of three parts: preseason, regular season and playoffs. This dissertation study focuses on the regular season scheduling problem. The NBA regular season starts at the late October or beginning of November and ends in the middle of April. Totally there are about 165 days for whole season. Each team will play 82 games during this time span , 41 at home and 41 on the road. There are 1,230 games in total.

Because too many constraints are involved, the NBA regular season scheduling (referred to the NBA scheduling in this dissertation study) is far beyond the current computation facility capability. There is no practical algorithm available to generate a complete schedule automatically. As a result, most of the work is done manually. Currently a company from Denver, Colorado makes the schedule for the NBA. The scheduling procedure starts every February. Each team will provide about 60 available dates for the 41 home games. The scheduler will make a draft schedule based on information from all teams and some other sources. The draft schedule will be modified

according to team feedback in the next phase. The NBA will release the official schedule at the beginning of July. The whole process is very time consuming which takes about five months.

On the other hand, the demand for the automatic scheduling approach is obvious for many reasons. For example, TV networks want to feature high-profile games. They want to see all possible scenarios to arrange those games under different circumstance, which is extremely difficult for manually scheduling approach. In the sports world, unexpected events happen all the time and those events could affect the scheduling. For instance, some trades could affect the distribution of attractive games dramatically. In 2007, Kevin Garnet was traded from the Minnesota Timberwolves to the Boston Celtics. The league had to delay releasing the schedule because of this trade. The decision makers would like to compare different scheduling before they make the choice. Normally manual scheduling approach cannot produce multiple schedules simultaneously because of the time limitation. However, the automatic scheduling approach can produce many solutions at the same time by changing parameters.

Besides reducing the scheduling time, the automatic scheduling approach could provide an optimal solution in terms of some objectives. Although the scheduling company does an excellent job, there is much room for the current official schedule to be improved. For example, the Cleveland Cavaliers will meet Chicago Bulls four times in the regular season. In the 2008 official schedule, two games between Cavaliers and Bulls are scheduled in March and there are only four days between these two games. The other two games are scheduled in April, and there is only one week between these two games.

The ideal distribution of these four games should be evenly throughout the whole season given these two teams are rivals from the same division.

Although it is difficult to design an algorithm to produce a complete schedule automatically under current computation capability, the problem definition and properties study will provide building blocks for the future research in this area. This dissertation will list all the major constraints in the NBA scheduling problem. All the constraints are grouped into three parts: basic structure constraints, external and team specific constraints and fairness constraints.

Integer programming and constraint programming are successfully used for the time constrained round robin tournament scheduling problems. Trick (2003) did research to compare the strengths of these two methods. Inspired by his work, we will model the constraints by both integer programming and constraint programming. In most cases we only show the IP modeling definition of the constraints.

We run a series of computation studies to compare these modes. The test machine is a personal computer running Windows XP with 2.2GHz Duo Core CPU and 1GBs of Ram. For the IP model, the search engine is ILOG CPLEX 11.2 and the default branching strategy was used unless explicitly marked. Running time (RT) and branch nodes (N) are used to indicate the performance. The time unit for the running time is second by default. If the model cannot terminate searching after 30 minutes or other explicitly marked time period, we will give either the best solution or mark the question as "/", which means no solution available. For the CP model, the search engine is ILOG CP Optimizer 2.1 and the default propagation setting was used unless explicitly marked. Running time (RT) and

failed branches (F) are used to indicate the performance. The time unit for the running time is second by default. Testing instances are generated by changing the size of tournament, which is decided by the number of teams. The computation study details are given accordingly. If the program cannot terminate the searching after 30 minutes or other explicitly marked time period, we use "/" to indicate there is no solution available. All the values for both IP model and CP model are average number of three runs under the same configuration.

5.1 Basic structure constraint

In essence the NBA scheduling is a time relaxed round robin tournament scheduling problem. In Chapter III we have studied some basic properties of this kind scheduling problems. In this section we will present the basic structure requirements for a feasible NBA schedule.

5.1.1 Basic problems

In this section we will present both IP and CP models with computation study results for the basic problems. Although we list the integer models in the Chapter III already, we will repeat here again to make this section complete. All the round robin tournaments can be decomposed to either one or multiple single round robin tournaments. Therefore we choose the single round robin tournament for demonstration. We first look at the basic model without any constraints.

Model 24 Basic NBA scheduling IP Model

$$\sum_{d \in D} (x_{ijd} + x_{jid}) = 1 \forall i, j \in T, i < j \tag{5.1}$$

$$\sum_{j\in\{T\setminus i\}} (x_{ijd} + x_{jid}) \leq 1 \forall i \in T, d \in D \tag{5.2}$$

$$x_{ijd} \in \{0,1\} \forall i, j \in T, i \neq j, d \in D \tag{5.3}$$

If team *i* plays team team *j* on day *d*, the binary variable x_{ijd} equal 1, otherwise 0. To clarify, the following is the OPL code for **Model 24.**

```
int nbTeams = …;
int nbDays = 2*(nbTeams-1);
dvar int play[1..nbTeams][1..nbTeams][1..nbDays] in 0..1;
subject to
{
        //Play each other once
        forall(ordered i,j in 1..nbTeams)
                sum(d in 1..nbDays)(play[i][j][d]+ play[j][i][d])  ==1;
        //At most one game each day
        forall(i in 1..nbTeams, d in 1..nbDays)
                sum(j in 1..nbTeams)(play[i][j][d]+play[j][i][d]) <=1;
}
```

The basic variable used in CP model presented in the Chapter III is *opponent*, which represents team's opponent on a specific day. If we change the variable to *day*, which represents the day on which a specific game holds, we get another approach for the CP modeling.

Model 25 CP Model based on Day

$$allDifferent(\sum_{j\in\{T\setminus i\}} d_{ij})\forall i \in T \tag{5.4}$$

$$d_{ij} = d_{ji}\forall i, j \in T \tag{5.5}$$

$$d_{ii} = 0\forall i \in T \tag{5.6}$$

$$d_{ij} \in \{0,1,...,D\}\forall i, j \in T \tag{5.7}$$

The variable d_{ij} represents the day on which team i plays against team j, its domains are all available game days plus zero. Constraint (5.4) forces every team will play different games on different day. Since this model is based on the single round robin tournament, we use constraint (5.5) to reduce domain searching. Constraint (5.6) forces no team can play itself. To clarify, we present the CP model based on *day* in OPL as following:

```
using CP;
int nbTeams = …;
int nbDays = 2*(nbTeams-1);
range rngTeams = 1..nbTeams;
dvar int day[rngTeams][ rngTeams] in 0..nbDays;
subject to
{
        //all different constraint
        forall(i in rngTeams){
```

```
            allDifferent(all(j in rngTeams)day[i][j]);
        }
        //one-factor constraint
        forall(i,j in rngTeams){
            day[i][j] == day[j][i];
            day[i][i] == 0;
        }
    }
```

Table 12 shows the computation results for basic problem. Computation time is listed in seconds. We can tell CP model based on *day* variable has best performance. The searching domain is much smaller in this model compared to the other two models, which could be the major contribution to the decrease in the computation time. We also notice IP model performs better than the CP model based on the *opponent* variable.

Table 12 Computation results – Basic instance

Teams	CP-day model	CP-opponent model	IP Model
6	0	0.02	0.01
10	0	0.02	0.01
14	0.03	0.11	0.09
18	0.03	0.27	0.15
22	0.03	0.73	0.31
26	0.03	1.36	0.56
30	0.03	2.56	0.76

5.1.2 Conferential/Divisional Games

All 30 NBA teams are grouped into conferences and divisions according to the geographical locations. How to arrange the conferential or divisional games is a major

concern for the NBA scheduling problem. We will abstract major requirements regarding this aspect in this section.

There are two conferences in the current NBA league: the eastern conference and the western conference. To reduce the travel cost, each team will play a team from the other conference only twice, one home game and one away game. It is a typical double round robin tournament. To demonstrate, we list the constraint clause in IP model only. The explanation of variable and other basic constraints are same as those in Model 24. We use Ca as the notation for one conference, Ca' as the other. Constraint (5.8) forces every team to play one home game against every team from the other conference. For the CP models, we just use the model based on *opponent* variable because the model based on day variable is not suitable for the conferential games. We use *homeopponent* as the variable as listed in Model 7. The computation results are listed in Table 13.

$$\sum_{d \in D} x_{jid} = 1 \forall i \in C_a, \; j \in C'_a \tag{5.8}$$

Table 13 Conference games

Team	IP		CP	
	N	RT	F	RT
6	0	0	200	0
10	0	0.06	515	0
14	0	0.33	4550	0.33
18	0	1.08	31749	3.75
22	0	7.32	1798834	221.2
26	0	12.52	/	/
30	0	22.0		

The IP model performs better than the CP model. The CP model cannot find a feasible solution for instance of 26 teams in 30 minutes, while the IP model can get one in only 12.52 seconds.

In each conference, there are three divisions. Totally there are six divisions in the NBA league, each division consists of five teams. Every team will play four times with a team from the same division, two home games and two away games. It is a four round robin tournament. We use T_a to notate the set of teams from the same division. The following constraint (5.9) forces every team to play every other team from the same division two home games throughout the whole tournament.

$$\sum_{d \in D} x_{ijd} = 2 \forall i, j \in T_a, i \neq j \tag{5.9}$$

Table 14 Division games

Teams	IP		CP	
	N	RT	F	RT
4	0	0.03	127	0.02
6	0	0.03	200	0.03
8	0	0.09	200	0.09
10	0	0.17	200	0.16

It is really rare for a division to have more ten teams, therefore we limited our instances to ten teams in the computation study. The results are showed in Table 14. We can tell IP models performances resembling CP performances.

Each team will play the remaining 36 games with the other 10 teams from the same conference. Among these teams, 6 teams have 4 games (2 home games and 2 away games) and 4 teams have 3 games. This is based on a 5-year rotation. Because it is not a

complete double round robin, how to assign the game venue for the odd game is an issue. In the practical scheduling, NBA chooses venues for such games based on the previous schedules. For example, Cleveland Cavaliers plays totally three games with New Jersey Nets in 2008-2009 season, two out of three played at home. During the last time these two teams play three times at a season, New Jersey Nets played two home games against Cleveland Cavaliers. Taking into account the previous schedules will make the model too complicated. For simplicity, we just designate certain teams to play two home games, one away game against some specific opponents.

$$\sum_{d \in D} x_{ijd} = 1 \forall i \in T,\ j \in T_i' \tag{5.10}$$

$$\sum_{d \in D} x_{ijd} = 2 \forall i \in T,\ j \in T_i'' \tag{5.11}$$

Because the requirements resemble those in the divisional game models, the computation study results should be similar as well. Therefore there is no need to do the computation study for this model.

5.1.3 Consecutive games

In the time constrained schedules, the limitation on the consecutive games is usually set on the game types. For example, the length of consecutive home games or away games should fit in a range. In the time relaxed schedules, there are more limitations on the length of consecutive games. In this section we list major constraints regarding this requirement arose from the NBA scheduling problem.

The attendance will be affected if a team has two home games in a row on consecutive

days. Therefore such game pattern is unwanted by teams.

$$\sum_{k\in\{0,1\}}\sum_{j\in\{T\setminus i\}} x_{ij(d+k)} < 2 \forall i \in T, d \in \{1,\ldots,D-1\} \tag{5.12}$$

Besides the attendance, fatigue is another factor limiting the length of consecutive games. It is required that no team can play three games in a row.

$$\sum_{k\in\{0,1,2\}}\sum_{j\in\{T\setminus i\}} (x_{ij(d+k)} + x_{ji(d+k)}) < 3 \forall i \in T, d \in \{1,\ldots,D-2\} \tag{5.13}$$

Another basic requirement for the length of consecutive games is that no team can have six or more games in eight days:

$$\sum_{k\in\{0,\ldots,7\}}\sum_{j\in\{T\setminus i\}} (x_{ij(d+k)} + x_{ji(d+k)}) < 6 \forall i \in T, d \in \{1,\ldots,D-7\} \tag{5.14}$$

For the CP model, we need two variables instead of single *opponent*, because the game venue is involved in these constraints. We add another variable *venue* to specify the game location. The computation results are listed in Table 15. CP model performs better in every test instance. There are several notable aspects in the results. For example, the computation time for 18 teams under IP model is unusual compared to the other instances; the fail number for 22 teams under CP model is unusual compared to the other instances. Unfortunately we cannot find the systemic correlations here.

Table 15 Consecutive games

Teams	IP		CP	
	N	RT	F	RT
6	**0**	**0.05**	**16**	**0.03**
10	**0**	**0.21**	**181**	**0.05**
14	**0**	**1.28**	**103**	**0.13**
18	**0**	**53.17**	**107**	**0.20**

22	**0**	**23.96**	**410**	**0.98**
26	**0**	**57.05**	**221**	**1.16**
30	**0**	**134.24**	**102**	**2.37**

5.1.4 Consecutive off days

In the time constrained schedule, constraints on the *bye* time slot are usually not necessary since the number of games is decided by the predefined time slots. In the time relax schedule, each team could have an off day besides the home game and road game. In the practical NBA scheduling, there is a limitation on the length of the consecutive off days. Except the starting week and ending week, each team will play at least 2 games in a week.

$$\sum_{j\in\{T\setminus i\}}\sum_{t\in\{0..6\}}(x_{ij(d+t)}+x_{ji(d+t)})\geq 2\forall i\in T, d\in\{1,\ldots,D-6\} \tag{5.15}$$

Another constrains is limitation on the length of the consecutive off days. We model this constraint by setting the upper bound of off days to 4 days.

$$\sum_{j\in\{T\setminus i\}}\sum_{t\in\{0..4\}}(x_{ij(d+t)}+x_{ji(d+t)})\geq 1\forall i\in T, d\in\{1,\ldots,D-4\} \tag{5.16}$$

The computation study results of constraint on the length of consecutive off days are similar to those regarding the consecutive games. The CP models perform better than IP models for every test instance. In general, the fail number for CP model is much bigger than those in consecutive games instances.

Table 16 Consecutive off days

Teams	IP		CP	
	N	RT	F	RT
6	**0**	**0.03**	**0**	**0**

10	**0**	**0.13**	**104**	**0.09**
14	**0**	**1.16**	**517**	**0.33**
18	**0**	**14.28**	**2162**	**1.94**
22	**0**	**73.47**	**2982**	**4.88**
26	**0**	**126.63**	**3442**	**9.16**
30	**0**	**242.11**	**4870**	**19.61**

5.2 External and team specific Constraints

For the NBA scheduling, there are many constraints besides the basic structure requirements. Among them, the external constraints and requirements from the individual team are most important. We will list the major constraints regarding these two topics in this section.

5.2.1 Television Schedule

Like the other professional leagues, television networks are the major revenue sources for the NBA. NBA has contracts with Turner media group for the right to nationally televise games. Turner media group has several national TV networks, including ABC, TNT, and ESPN. Besides the national TV network, NBA also has contracts with local television networks where the teams located. Recently the NBA has signed multiple contracts with several international television outlets.

To get a better view rating, normally All TV networks want to air attractive games at prime time. Meanwhile it is possible for a TV network to reserve exclusive broadcasting right for a specific day. For example, TNT will air two games on Thursday night. They require that generally no more than three games can be scheduled on each Thursday. ESPN will broadcast two games on Wednesday night and Friday night respectively. ABC will broadcast either one or two games on Sunday afternoon during

non-football season. All these requirements make the scheduling problem difficult.

There are many ways to define the attractive games. Normally attractive is associated with certain teams. For example, games between teams having super stars are normally regarded as attractive games. However, there is not always the case. A game between two week teams could be an attractive game if they are rivalries. At the end of the season, a game between two non-elite teams which have potential to make playoff is regarded as attractive games as well. Although at the scheduling stage it could be difficult to predict which teams could make playoffs, it is always a good practice to put some games like this at the end of the season. Briskorn (2009) use a probability function to decide whether a game is attractive or not. To keep things simple, we will use a similar method. Normally there is one elite team in each division, therefore we use the probability 0.2 for all games to be attractive. To indicate whether a game between team i and j is attractive, we introduce a parameter a_{ij}. a_{ij} is equal to 1 if the game is attractive, otherwise it is equal to 0.

The objective is to distribute the attractive games according to the TV stations' requirements. It is important to balance the games among TV stations, including the national TV stations and local TV stations. For the national TV stations, we just limit the maximum attractive games in each time slot. Meanwhile we use D' as the set of special time slots including Thursdays and Sundays. Because local TV networks normally associated with one team, we just add constraint (5.15) for local TV stations to make sure each team will play at least twice a week.

$$\sum_{i \in T} \sum_{j \in \{T \setminus i\}} a_{ij} x_{ijd} < a_{\max} \forall d \in D \tag{5.17}$$

$$\sum_{i \in T} \sum_{j \in \{T \setminus i\}} a_{ij} x_{ijd} > a_{\min} \forall d \in D' \tag{5.18}$$

The computation results are listed in

Table 17. Both IP and CP models can solve the problem of 30 teams quickly. CP model has slight advantage compared to IP model. For comparison, we change the constraint (5.17) and (5.18) to let every Thursday and Sunday play exactly two attractive games, the computation results are similar to Table 17.

Table 17 TV schedules

Teams	IP		CP	
	N	RT	F	RT
6	**0**	**0**	**0**	**0**
10	**0**	**0.00**	**435**	**0.05**
14	**0**	**0.02**	**1615**	**0.39**
18	**0**	**0.06**	**512**	**0.25**
22	**0**	**0.22**	**212**	**0.34**
26	**0**	**0.25**	**866**	**1.75**
30	**0**	**0.41**	**202**	**0.91**

5.2.2. Home away pattern

Like we previously stated, games can be classified by game venues as home games and road games. The sequence of game types forms game patterns which referred to as HAP in this dissertation study. There are many important constraints on the game patterns. For instance, the constraint on avoiding consecutive home games and the constraint on length of consecutive games are typical constraints on HAP.

Normally every team has preference regarding the home game or away game on a specific day, although it is rare for a team to define the whole HAP during the scheduling stage. To demonstrate the constraint, we will generate a feasible HAP in the first phase, then we will try to find an opponent schedule based on the HAP in the second phase.

We examined the HAP feasibility problem in Chapter III. To make sure we have a feasible HAP, we just use a model to randomly generate a complete schedule, then we use the place information as the HAP for our second phase.

$$P_{id} = 0 \Rightarrow \sum_{j \in T} x_{ijd} = 1 \forall i \in T, d \in D \tag{5.19}$$

$$P_{id} = 1 \Rightarrow \sum_{j \in T} x_{jid} = 1 \forall i \in T, d \in D \tag{5.20}$$

$$P_{id} = 2 \Rightarrow \sum_{j \in T} (x_{ijd} + x_{jid}) = 0 \forall i \in T, d \in D \tag{5.21}$$

The computation results are listed in Table 18. When the problem size is less than 14, CP model has slight advantage over IP. When the problem size is equal to or larger than 14, it is obvious that IP model performs much better. CP model cannot find a feasible solution after 10 minutes computation, while IP model find one in just 3.38 seconds.

Table 18 HAP Patterns

Teams	IP		CP	
	N	RT	F	RT
6	**0**	**0.03**	**0**	**0**
10	**0**	**0.09**	**41**	**0.02**
14	**0**	**0.86**	**110**	**0.03**
18	**0**	**3.38**	/	/
22	**0**	**17.34**	/	/
26	**8**	**174.47**	/	/
30	**44**	**511.06**	/	/

5.2.3. Arena availability

For most of the NBA teams, they rent the arenas or stadiums which generally belong to the local government. These arenas will be used for other activities besides

basketball games, such as concerts, hockey games, etc. Consequently the arena is not always available, which post a very important restriction for the schedule.

We introduce a parameter v_{id} to indicate whether the arena for team i is available on day d.

$$x_{ijd} \leq v_{id} \forall i, j \in T, i \neq j, d \in D \tag{5.22}$$

Normally each team has only about from 50 to 60 days available for the 41 home games. Because there are roughly 180 days for the whole season, the probability for each day's availability is about 1/3. We use a random generator to simulate the probability for our test instances. It randomly assigns a number from 0 to 100 to the variable.

For the integer programming, we use the same model used in the HAP constraint. The following is the OPL function to model the arena availability constraint:

```
forall(i in 1..nbTeams, d in 1..nbDays)
    sum(j in 1..nbTeams)play[i][j][d]<=(available[i][d]<=33);
```

For the constraint programming, we change the variable *opponent* to the new variable *homeopponent*. The following is the OPL code:

```
//play each other exactly once
forall(ordered i,j in rngTeams)
    sum(d in rngDays)(homeopponent[i][d]== j||homeopponent[j][d] == i) ==
1;

//No team play itself
forall(i in rngTeams,d in rngDays)
    homeopponent[i][d]!=i;
```

```
//opponent constraint
forall(d in rngDays,i,j in rngTeams)
        (homeopponent[i][d] == j) => (homeopponent[j][d] == 0);
//At most one game each day
forall(i in rngTeams,d in rngDays)
        (homeopponent[i][d] !=0 )+ sum(j in rngTeams)(homeopponent[j][d] ==
i)<=1;
//Arena availability
forall(i,j in rngTeams, d in rngDays)
        (available[i][d]>=33) =>homeopponent[i][d] == 0 ;
```

The computation result is listed in

Table 19. It is obvious that the IP approach does better in this test instance. For the CP approach, one interesting observation is the computation time is not linearly increasing with the size growth of the test instances. The problem of 14 team instance uses less time than 10 team instance. The CP cannot finish searching for problem of 18 teams in 30 minutes. This could be caused by the change in sizes of the search domains.

Table 19 Arena availability

Teams	IP		CP	
	N	RT	F	RT
6	**0**	**0.04**	**202**	**0.14**
10	**0**	**0.18**	**740382**	**27.06**
14	**0**	**0.34**	**96949**	**6.41**
18	**0**	**0.43**	/	/
22	**0**	**0.79**	/	/
26	**0**	**1.36**	/	/
30	**0**	**2.28**	/	/

5.2.4. Team forbidden games

If a team requests not to have some certain games on a particular day, we call this type requirement as the team forbidden games. Almost each team has such requirements. For example, some teams don't want to play a home game if there is a football game in the same city on that day. The Utah Jazz will not play home games on Sunday because of religion reason.

Besides the requirements from individual teams, there are some other constraints can be treated as the team forbidden games.. For example, there are no games scheduled on super bowl night and the NBA All-star weekend. This can be considered as the forbidden games requirement for all teams.

We can treat this constraint similar as the arena availability. We set v_{ijd} equal to 0 if team i does not want to play team j on day d. We generate random numbers for v_{ijd}.

$$x_{ijd} \le V_{ijd} \forall i, j \in T, d \in D \tag{5.23}$$

The constraint for CP model is similar to the arena model:

```
//Forbidden games
    forall(i,j in rngTeams, d in rngDays)
        (forbidden[i][d][j]>=31) =>homeopponent[i][d] !=j ;
```

The computation results are listed in Table 20. IP approach has slight advantage compared to CP approach.

Table 20 Team forbidden

	IP	CP

	N	RT	F	RT
6	**0**	**0**	**12**	**0.03**
10	**0**	**0.02**	**102**	**0.03**
14	**0**	**0.05**	**103**	**0.06**
18	**0**	**0.11**	**106**	**0.14**
22	**0**	**0.22**	**131**	**0.31**
26	**0**	**0.45**	**200**	**0.66**
30	**0**	**0.63**	**200**	**1.08**

5.2.5. Complementary schedules

The LA Lakers and The LA Clippers share the same arena – the STAPLS center. The schedule for these two teams should be complementary, which means these two teams cannot play home games simultaneously. Because all the teams can be mutated in the schedule, we use i1 and i2 to represent the two teams respectively.

$$\sum_{j \in T} (x_{i_1 jd} + x_{i_2 jd}) <= 1 \forall d \in D \tag{5.24}$$

Table 21 Two teams have complementary schedules

	IP		CP	
	N	RT	F	RT
6	**0**	**0.025**	**139**	**0**
10	**0**	**0.02**	**200**	**0.97**
14	**0**	**0.015**	**200**	**0.12**
18	**0**	**0.02**	**200**	**0.25**
22	**0**	**0.03**	**200**	**0.41**
26	**0**	**0.02**	**200**	**0.69**
30	**0**	**0.03**	**200**	**1.17**

The computation results are listed in Table 21. We can tell IP does better than CP in this test case.

5.2.6. Maximum value schedules

The revenue for a game can be different if it is held on different date. One of the major scheduling objectives is to maximize the total revenue for the league. For the instance of team n, we randomly assign a value v_{ijd} between 0 and n^2 to each game on each day *game(i,j,d)*. This can be easily done in the ILOG OPL Studio by using the function rand(n^2).

The search strategy is very important for the constraint programming. We tried two search strategies. The first one is the default setting by the ILOG OPL Studio. The second one is to select the variables with the smallest domain size, and start searching with the largest value. It doesn't show big difference for these two searching strategies. The computation study result is listed in

Table 22. The CP model cannot even find the optimal solution for problem of 6 teams in 30 minutes. It's obvious that IP based approach does much better than the CP based approach.

$$\max \sum_{i \in T} \sum_{j \in T} \sum_{d \in D} x_{ijd} * v_{ijd} \tag{5.25}$$

Table 22 Maximum value

	IP		CP	
	N	RT	F	RT
6	**15**	**0.06**	/	/
10	**15**	**0.23**	/	/
14	**15**	**0.23**	/	/
18	**15**	**0.40**	/	/
22	**23**	**0.93**	/	/
26	**15**	**1.03**	/	/
30	**23**	**1.81**	/	/

5.3 Fairness constraint

One of the fundamental requirements of a schedule is to be impartial to all participants. Fairness can be measured by different objectives. There have been plenty research done on the fairness requirements in the time constrained scheduling. Because of the difference in the time structure, there are some fairness requirements are unique in the NBA scheduling problem. This dissertation will put emphasis on these constraints.

5.3.1 Back-to-back games

In the time constrained schedules, normally the interval between consecutive games is one week or couple days. The term "break" refers to consecutive home games or consecutive away games, which is unwanted pattern. The literature on this topic has been reviewed in Chapter II.

In the NBA schedule, the time unit of game is day. We stated in constraint (5.12) that no team play home games on two consecutive days. However, team might play road games on two consecutive days. Another possibility is a team play games on two consecutive days, one is on road and other one is at home. If a team plays games on two consecutive days, we call the second game as a *back-to-back* game. Normally teams prefer to avoid back-to-back games. The team playing an opponent who is on the second night of the back-to-back games has advantage over his opponent. As a result, it is important to keep the number of back-to-back games equal or close to equal among teams. If the back-to-back games are necessary, it should be evenly distributed throughout the whole season.

Since the basic model does not have variables for game venue, we add set of variables *place[i][d]* for the purpose of calculating the break. The domain for the variable are 0,1, or 2. If team i plays a home game on day *d place[i][d]* equals 0, an away game equals 1, and off days equals 2. Team i has a break on day d if the value *place[i][d]* is same *as place[i][d-1]*. The objective is to make a schedule without back-to-back games. It is obvious that it is possible if there is no other constrains required since we have more days available for games. We can prove this quickly by the CP model based on day. We use a new constraint `allMinDistance.` It is a constraint available in OPL language. This constraint requires each game day has to be at least one day apart from the other game day. This constraint is stronger than the combination of the two constraints we listed. This model can generate a schedule with back-to-back for problem of 30 teams in less than 3 seconds. The computation results are listed in Table 23.

Table 23 Schedule without back-to-back games by CP model based on day

Team	CP model based on	
	F	RT
6	0	0.05
10	1	0.02
14	0	0.02
18	21	0.03
22	10	0.05
26	4	0.06
30	3278	2.86

If some additional constraints are considered, the CP model based on day is not proper in this scenario. We add the limit on the consecutive home games, off days and arena availability into the model. The definition of back-to-back game variable is listed in (5.27).

$$\min \sum_{d\in\{2,\ldots,D\}} \sum_{i\in T} bb_{id} \tag{5.26}$$

$$bb_{id} = \Big(\sum_{k\in\{0,1\}} \sum_{j\in T} (x_{ij(d-k)} + x_{ji(d-k)}) = 2 \Big) \forall i \in T, d \in \{2,\ldots,D\} \tag{5.27}$$

The objective (5.26) is to generate a schedule with minimum back-to-back games. Integer programming can get the results for up to 10 teams in 30 minute computation time. After we get the minimum value, notated by Bmin, we use the following constraint to evenly distribute the back-to-back games to every team:

$$\sum_{d\in\{2,\ldots,D\}} bb_{id} = \left| \frac{B\min}{n} \right| \forall i \in T \tag{5.28}$$

We tune the parameters to make sure teach team will have at last one back-to-back game. The CP model cannot get the results for 10 teams in reasonable computation time.

Table 24 Back-to-back games

Teams	IP		CP (opponent)	
	N	RT	F	RT
6	0	1	5139	0.45
8	6	17.25	25874	341.62
10	19	33.07	4687378	833.08

5.3.2 Weekend games

Weekend games will bring more revenue to the home teams compared to the weekday games, therefore every team want to have as many weekend games as possible. Consequently we want to maximize the total weekend games in our modeling. The following model is used to calculate the maximum weekend games with some necessary

constraints.

Model 26 Weekend games IP model

$$\max \sum_{i \in T} \sum_{j \in T} \sum_{d \in Dwd} x_{ijd} \tag{5.29}$$

$$\sum_{j \in \{T \setminus i\}} (x_{ijd} + x_{jid}) \leq 1 \forall i \in T, d \in D \tag{5.30}$$

$$\sum_{d \in D} (x_{ijd} + x_{jid}) = 1 \forall i, j \in T, i < j \tag{5.31}$$

$$\sum_{j \in T} \sum_{k \in \{0,1\}} x_{ij(d+k)} < 2 \forall i \in T, d \in D \tag{5.32}$$

$$\sum_{j \in T} \sum_{k \in \{0,1,2\}} (x_{ij(d+k)} + x_{ji(d+k)}) < 3 \forall i \in T, d \in \{1, \ldots, D-2\} \tag{5.33}$$

$$\sum_{j \in T} \sum_{k \in \{0,\ldots,4\}} (x_{ij(d+k)} + x_{ji(d+k)}) > 1 \forall i \in T, d \in \{1, \ldots, D-4\} \tag{5.34}$$

The objective (5.29) is to maximize the weekend games. The constraint (5.30) ensures each team will play at most one game on each day. The constraint (5.31) forces every team will play the other team exactly once. The constraint (5.32) makes sure every team will not have consecutive home games. The constraint (5.33) and (5.34) forces the lengthy of consecutive games or consecutive off days to be less than 3 and 5 respectively. Normally all games scheduled on the Friday, Saturday and Sunday can be considered as weekend games. Of course there are some slightly difference between games on these days. For simplicity, we treat them same in this dissertation study. D_{wd} is used as the notation of weekend days. W_{max} refers to the maximum weekend games from Table 25.

Table 25 gives the computation results.

Since the available weekend games are limited, it is important to balance the weekend games among all teams. Normally all games scheduled on the Friday, Saturday and Sunday can be considered as weekend games. Of course there are some slightly difference between games on these days. For simplicity, we treat them same in this dissertation study. D_{wd} is used as the notation of weekend days. W_{max} refers to the maximum weekend games from Table 25.

Table 25 Maximum Weekend Games

Teams	Maximum Weekend games
6	6
8	16
10	20
12	36
14	49
16	64
18	90
20	100
22	132
24	144
26	182
28	210
30	240

$$\sum_{d \in D_{wd}} \sum_{j \in \{T \setminus i\}} x_{ijd} = \left| \frac{W_{Max}}{n} \right| \forall i \in T \tag{5.35}$$

The above constraint (5.35)forces each team will play exactly same amount of weekend games, notated by $\left| \frac{W_{Max}}{n} \right|$ which is equal to W_{max} divided by the total teams. If the number is not integer, we round off the number. The computation results are listed in Table 26.

Table 26 Weekend games

	IP		CP	
	N	RT	F	RT
6	0	0.05	5	0.02
8	0	0.31	30	0.04
10	0	0.84	1541	0.38
12	0	7.63	661553	143.75
14	0	11.31	4091	2.22
16	0	78.84	/	/
18	0	179.47	/	/
20	0	510.48	1541	0.38
22	0	1745.95	/	/
24	0	5173.07	1541	0.38
26	0	/	/	/
28	0	/	1541	0.38
30	0	/	/	/

5.3.3 Travel distance

Because the traveling can cause players fatigue and increase the cost, all teams want to minimize their travel distance. To be fair, the traveling distance for every team should be within a reasonable range. We assume each team starts and finishes the tournament at home. When they play consecutive road games, they will stay on the road without returning home. Actually it is not unusual for teams to return home during the road game span, especially if there are more than two days off between them. We will make this assumption to avoid too complicated model. The fairness requirement should be under the circumstance with minimizing the total traveling distance as the objective. If we use Tr_i to notate the traveling distance for team i, this constraint can be listed as the following. We will set minimum and maximum traveling distance for each team.

$$Tr_i \in \{Tr_{\min}, Tr_{\max}\} \forall i \in T \tag{5.36}$$

The IP model needs additional variables besides *play[i][j][d]* . The variables *location[j][i][d]* is 1 if team i plays at j's venue on day d, otherwise 0 . If team i has an

off day on day d, *location[j][i][d]* is equal to *location[j][i][d-1]*. Now we can use another set of variables *travel[i][j][k][d]* to calculate the traveling distance. This model performs poorly in CPLEX. We cannot get results for even 6 teams. Trick (2003) suggested to use another formulation for the traveling distance. It is based on all the possible trips a team could take: *trips[i][i1][d]*, *trips [i][i1][i2][d]*,and *trips[i][i1][i2][i3][d]*. This formulation has to be modified to be suitable for the time relaxed tournaments scheduling. We believe it is not a trivia ask.

Table 27 Travel distance

	IP		CP	
	N	RT	F	RT
4	/	/	4643	0.58
6	/	/	/	/
8	/	/	/	/
10	/	/	/	/

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

In the introduction to this dissertation, there academic contribution goals were set forth for this research: to examine the structural properties of the time relaxed round robin tournament, to define and tackle the time relaxed traveling tournament problem, to provide major modeling constraints for the NBA scheduling problem.

To date, there has no research done on the properties of general time relaxed round robin tournament problems. We present the formal definition of the single round, double round, and r-round tournament problems in the time relaxed context. The integer programming models are developed and presented. To improve the performance, some strengthening constraints are used for the time constrained tournaments. The strengthening constraints for time relaxed IP models could be a research topic in the future. Meanwhile the proof of the complexities of these problems is needed in the future research. The break minimization problem is defined in the time relaxed context. Currently we can only solve problem of 8 teams in reasonable time. We did not deploy symmetry breaking efforts in our model, therefore adding symmetry breaking constraints could be the first step to improve the results. Because of the complexity of the problem, decomposition is a promising approach. We examined two different decomposition methods to address the break minimization problem. Although we can solve the problem of thirty teams use the “”first-schedule-then-break” method, the minimum break is not zero. We addressed the HAP feasibility problem. We did not present the method to generate a HAP set with minimum breaks. It could be a research topic in the future. Additionally, we presented the GOP feasibility problem, which is

unique but important problem in the time relaxed scheduling. How to utility the GOP properties for other objectives, such as break minimization or travelling distance minimization, could be an interesting topic for the future research.

To address the traveling factor in the time relaxed round robin tournament, we introduced the TRTTP problem. In the formal definition of TRTTP, we use a few more parameters compared to the TTP. To examine the complexity of TRTTP, the single team problem TRSTP is defined. The complexity heavily depends on the parameter of consecutive away games U. A polynomial time algorithm is presented when U is two. When U equals to three or more, we conjecture the problem becomes NP-hard.

It is difficult to get the optimal solution for TRTTP instances of more than eight teams given the current computation capability. To evaluate the solutions, it is important to attain good lower bound and upper bound. As a strong lower bound, Independent Lower Bound (ILB) for TRTTP is defined and calculated in this dissertation. For the upper bound, we use a CP algorithm to generate feasible solutions. Although only U equals to 3 is considered in this dissertation, U equals to four or more is found in practical applications. It could be a research topic in the future. We made two cases for the TRTTP: one is teams staying on road on off days during the away trip the other is returning home instead. In real world this decision can be decided by the length of off days and distance between home and away teams. How to incorporate these constraints will be another research topic in the future. Additionally, we present the decomposition method based on the optimal tour. The other decomposition methods can be used for the future research, for instance, the branch-and-price method.

Parallel computation has been proven to be effective for solving TTP. We do not use parallel computation in our study because of limited computation facility. Additionally, several metraheuristic algorithms, including simulated annealing, tabu serach, generic algorithms have

been used to generate good solutions for the TTP problems. All of these algorithms could be used to address the TRTTP in the future.

We grouped major NBA scheduling requirements into three categories: structural, external and fairness constraints. Some constraints are not listed because of complexity or other reasons, we will list a few for the future research. We treat all back-to-back games as the same in our model, actually there are difference among the types of back-to-back games. If teams have to play a back-to-back game, normally they prefer to play the second game at home rather than on road. Since the objective to minimize back-to-back games are important for the NBA scheduling, it will be a research topic in the future how to minimize total number of the back-to-back games with more constraints considered besides those we employed. Although we did not include the general break minimization in the scheduling, it is a scheduling constraint as well. For example, teams do not want to play five consecutive five home games just to avoid one back-to-back game. No-repeater is a soft constraint in the scheduling, which is not included in our modeling. In the future the constraint to require there are at least fixed amount games separate games featuring same opponents.

The modeling method in this dissertation study performs poorly for the constraint of travelling fairness. Therefore new modeling methods are needed. For efficiency, it is better for teams to have a trip of at least three games if they travel to another coast. Another constraint we have not included in this dissertation is the overnight flight distance. Team cannot have a flight over certain amount distance overnight.

We run the computation studies for all constraints in both IP and CP models. It could be a good research topic to develop a model to test the integration groups combing different constraints. For example, how the IP model and CP model perform for the models combining structural and external requirements. Our research shows that IP models performs better for

problems of optimality and CP model performs better for feasibility problem, how to combine them for some specific problems could be a good research topic on the future .

BIBLIOGRAPHY

Anagnostopoulos, A., Michel.L, Van Hentenryck.P, Vergados.Y. (2006). A simulated annealing approach to the traveling tournament problem. *Journal of Scheduling , 9*, 177-193.

Andries E. Brouwer a, G. F. (2008). Tight bounds for break minimizationin in tournament scheduling. *Journal of Combinatorial Theory* , 1065–1068.

Ball B.C,Webster D.B. (1977). Optimal scheduling for even numbered team athletic conferences. *AIIE Transactions , 9*, 161–169.

Bartsch.T, Drexl.A, Kroger.S. (2006). Scheduling the professional soccer leagues of Austria and Germany. *Computers and Operations Research , 33* (7), 1907-1937.

Bean J.C, Birge, J.R. (1980). Reducing travelling costs and player fatigue in the national basketball association. *Interfaces , 10*, 98-102.

Briskorn, D. (2009). Combinatorial properties of strength groups in round robin tournaments. *European Journal of Operational Research , 192*, 744-754.

Briskorn, D. (2007). Sports Leagues Scheduling: Models, Combinatiorial Properties, and Optimization Algorithms. *Lecture notes in economics and mathematical systems. 603.* Berlin: Springer.

Campbell, R.T., Chen, D.S. (1976). A minimum distance basketball scheduling problem. (R. S.P.Ladany, Ed.) *Management Science and Systems,Studies in the Management Sciences , 4*, 15-26.

Colbourn, C. (1983). Embedding Partial Steiner Triple Systems is NP-Complete. *Journal of Combinatorial Theory,Series A , 35*, 100-105.

D, C. (1995). An evolutionary tabu search algorithm and the NHL scheduling problem. *INFOR , 33*, 161-178.

Dalibor, F. (2005). Round robin tournaments with one by no break in home-away pattern is unique. In E. B. G. Kendall (Ed.), *MultidisciplinaryScheduling: Theory and Applications* (pp. Part 9, 331–340). Berlin: Springer.

de Werra, D. (1980). Geography, games and graphs. *Discrete Applied Mathematics , 2*, 327-337.

de Werra, D. (1982). Minimizing irregularities in sports schedules using graph theory. *Discrete Applied Mathematics , 4*, 217-226.

de Werra, D., Ekim, T.; Raess, C. (2006). Construction of sports schedules with multiple venues. *Discrete Applied Mathematics , 154*, 47-58.

Drexl A, Knust S. (2007). Sports league scheduling: Graph- and resource-based models. *Omega , 35*, 465-471.

Easton K, Nemhauser G,Trick M. (2001). The traveling tournament problem:Description and benchmarks. In T.Walsh, *Principles and Practice of Constraint Programming - CP 2001,Lecture Notes in Computer Science* (Vol. 2239, pp. 580-585). Berlin/Heidelberg: Springer.

Easton. K,Nemhauser.G,Trick. M. (2003). Solving the traveling tournament problem: A combined integer programming and constraint programming approach. In P. E.Burke,

Practice and Theory of Automated Timetabling IV,Lecture Notes in Computer Science (Vol. 2740, pp. 100-109). Berlin/Heidelberg: Springer.

Easton.K. (2002). Using Integer Programming and Constraint Programming to Solve Sports Scheduling Problems. *Ph.D Thesis* . Georgia Institute of Technology.

Easton.K, T. N. (2004). Sports Scheduling. In J. Leung (Ed.), *Handbook of Scheduling* (pp. 52.1–52.19). CRC Press.

Elf .M,Junger.M, Rinaldi. G. (2003). Minimizing breaks by maximizing cuts. *Operations Research Letters , 31*, 343-349.

Ferland, J.A.; Fleurent, C. (1991). Computer aided scheduling for a sport league. *INFOR* , *29*, 14-25.

Fraser, W. (1982). The role of computer simulation in building the National Hockey League Schedule. Montreal, Quebec, Canada: IBM Canada Limited.

Frieze, A. (1983). Complexity of a 3–Dimensional Assignment Problem. *European Journal of Operational Research , 13*, 161-164.

Froncek, D. M. (2003). Round Robin Tournaments with one bye and no breaks in Home-away Patterns are unique. In G. E. Kendall (Ed.), *Multidisciplinary scheduling: theory and application* (pp. 331-340). Berlin: Springer.

Froncek, D. (2001). Scheduling the Czech national basketball league. *Congressus Numerantium , 153*, 5-24.

Garey.M.R, J. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: Freeman.

Henz, M. (1999). Constraint-based round robin tournament planning. In d. S. D (Ed.), *Proceedings of the international conference on logic programming* (pp. 545-557). Las Cruces, New Mexico: MIT Press.

Henz, M. (2004). Playing with constraint programming and large neighborhood search for traveling tournaments. In M. T. E. Burke (Ed.), *Proceedings PATAT 2004*, (pp. 23–32).

Henz, M. (2001). Scheduling a major college basketball conference – revisited. *Operations Research , 49*, 163–168.

Henz, M.; Müller, T.; Thiel, S. (2004). Global constraints for round robin tournament scheduling. *European Journal of Operational Research , 153*, 92–101.

ILOG. (2008). *ILOG OPL Development Studio IDE.*

K.Karlof, J. (2006). *Integer Programming: Theory and Practice.* New York: Taylor & Francis.

Kendalla.G, K. R. (2010). Scheduling in sports: An annotated bibliography. *Computer & Operations Research , 37*, 1-19.

Knust, S. (2009). *Classification of literature on sports scheduling*. Retrieved from http://www.inf.uos.de/knust/sportlit_class

Knust, S.; von Thaden, M. (2006). Balanced home–away assignments. *Discrete Optimization , 3*, 354–365.

Lim, A., Rodrigues, B., & Zhang, X. (2006). A simulated annealing and hill-climbing algorithm for the traveling tournament problem. *European Journal of Operational Research , 174* (3), 1459–1478.

Miyashiro, R.; Matsui, T. (2006). Semidefinite programming based approaches to the break minimization problem. *Computers & Operations Research , 33* (7), 1975–1982.

Miyashiro, R.; Matsui, T. (2005). A polynomial-time algorithm to find an equitable home–away assignment. *Operations Research Letters , 33*, 235–241.

Miyashiro.R, I. a. (2003). Characterizing Feasible Pattern Sets with a Minimum Number of Breaks. In E. B. Causmaecker (Ed.), *Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling* (pp. 78–99). Berlin: Springer.

Nemhauser, G.L.; Trick, M.A. (1998). Scheduling a major college basketball conference. *Operations Research , 1*, 1–8.

Pinedo, M. (2002). *Scheduling: Theory, Algorithms and Systems*. New Jersey: Prentice Hall.

Post, G., & Woeginger, G. (2006). Sports tournaments, home–away assignments, and the break minimization problem. *Discrete Optimization , 3* (2), 165–173.

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

Rasmussen, R.V., Trick , M.A. (2008). Round Robin scheduling - a survey. *European Journal of Operational Research , 188*, 617-636.

Rasmussen, R.V., Trick, M.A. (2007). A Benders approach for constrained minimum break problem. *European Journal of Operational Research , 177* (1), 198–213.

Régin, J.-C. (1994). A filtering algorithm for constraints of difference in CSPs. *AAAI-94,proceedings of the Twelve National Conference on Artificial Intelligence*, (pp. 362-367). Seattle, Washington.

Régin, J.-C. (2001). Minimization of the number of breaks in sports scheduling problems using constraint programming. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science , 57*, 115–130.

Ribeiro CC, Urrutia S. (2007). Scheduling the brizilian soccer tournament with fairness and broadcast objectives. *Lecture notes in computer science , 3867* (Practice and theory of automated timetabling VI), 147-157.

Ribeiro, C.C.; Urrutia, S. (2007). Heuristics for the mirrored traveling tournament problem. *European Journal of Operational Research , 179* (3), 775–787.

Rosa, A.; Wallis, W.D. (1982). Premature sets of 1-factors or how not to schedule round-robin tournaments. *Discrete Applied Mathematics , 4*, 291–297.

Russell, R.A.; Leung, J.M.Y. (1994). Devising a cost effective schedule for a baseball league. *Operations Research , 42* (4), 614–625.

Russell.K.G. (1980). Balancing Carry–Over Effects in Round Robin Tournaments. *Biometrika , 67*, 127–131.

Schaerf, A. (1999). Scheduling sport tournaments using constraint logic programming. *Constraints , 4*, 43–65.

Schaerf, A., Di Gaspero, L. (2007). A composite-neighborhood tabu search approach to the traveling tournament problem. *Journal of Heuristics , 13*, 189–207.

Schonberger. J,Mattfeld .D.C, Kopfer .H. (2004). Memetic algorithm timetabling fo rnon-commercial sportleagues. *European Journal of Operational Research , 153*, 102-116.

Schreuder, J. (1992). Combinatorial aspects of construction of competition dutch professional football leagues. *Discrete Applied Mathematics , 35*, 301–312.

Schreuder, J. (1980). Constructing timetables for sport competitions. *Mathematical Programming Study* , 58–67.

Sierksma, G. (2002). *Linear and Integer Programming.* New York: Marcel Dekker.

Stefan, I., Ulrich, S. (2009). *A new Branch-and-Price Algorithm for the Traveling Tournament Problem (TTP).* RWTH Aachen: OR-01-2009.

Trick, M. (2001). A schedule-then-break approach to sports timetabling. In E. B. Erben (Ed.), *Lecture Notes in Computer Science. 2079*, pp. 242–252. Berlin / Heidelberg: Springer.

Trick, M. (2009). *Challenge Traveling Tournament Instances*. Retrieved from http://mat.gsia.cmu.edu/TOURN/

Trick, M. (2003). Formulations and Reformulations in Integer Programming. Carnegie Mellon University.

Trick, M. (2003). Integer and Constraint Programming Approaches for round robin tournament Scheduling. (E. B. Causmaecker, Ed.) *Lecture Notes in Computer Science , 2740*.

Urban, T.L.; Russell, R.A. (2003). Scheduling sports competitions on multiple venues. *European Journal of Operational Research* , 302-311.

Voorhis, T. (2002). Highly constrained college basketball scheduling. *Journal of the Operational Research Society , 53*, 603-609.

Willis, R., & Terrill, B. (1994). Scheduling the Australian state cricket season using simulated annealing. *Journal of the Operational Research Society , 45* (3), 276-280.

Wright, M. (2006). Scheduling fixtures for basketball New Zealand. *Computers & Operations Research , 3*, 1875–1893.

Wright, M. (1995). Timetabling county cricket fixtures using a form of tabu search. *Journal of the Operational Research Society , 45* (7), 758–770.

APPENDIX - TRTTP INSTANCES

National League Instances

6 team instance

[0	745	665	929	605	521]
[745	0	80	337	1090	315]
[665	80	0	380	1020	257]
[929	337	380	0	1380	408]
[605	1090	1020	1380	0	1010]
[521	315	257	408	1010	0]

8 team instance

[0	745	665	929	605	521	370	587]
[745	0	80	337	1090	315	567	712]
[665	80	0	380	1020	257	501	664]
[929	337	380	0	1380	408	622	646]
[605	1090	1020	1380	0	1010	957	1190]
[521	315	257	408	1010	0	253	410]
[370	567	501	622	957	253	0	250]
[587	712	664	646	1190	410	250	0]

National Basketball Association Instances

Division instance

	1	2	3	4	5
1 Detroit MI		98	239	220	225
2 Cleveland OH	98		333	303	257
3 Milwaukee WI	239	333		82	246
4 Chicago IL	220	303	82		167
5 Indianapolis IN	225	257	246	167	

Conference instance

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1 Detroit MI		98	239	220	225	461	496	494	628	296	408	575	1157	956	506
2 Cleveland OH	98		333	303	257	369	409	407	554	243	310	536	1094	896	437
3 Milwaukee WI	239	333		82	246	700	734	732	856	516	640	660	1275	1070	662
4 Chicago IL	220	303	82		167	672	712	710	848	514	598	579	1195	990	589
5 Indianapolis IN	225	257	246	167		593	644	643	805	499	493	415	1028	824	427
6 Philadelphia PA	461	369	700	672	593		66	66	255	258	139	666	1047	882	473
7 New York NY	496	409	734	712	644	66		2	189	257	205	732	1104	944	540
8 Jersey City NJ	494	407	732	710	643	66	2		189	255	205	732	1105	944	540
9 Boston MA	628	554	856	848	805	255	189	189		340	394	921	1270	1119	728
10 Rochester NY	296	243	516	514	499	258	257	255	340		305	733	1224	1039	584
11 Washington DC	408	310	640	598	493	139	205	205	394	305		527	933	759	336
12 Atlanta GA	575	536	660	579	415	666	732	732	921	733	527		617	413	206
13 Miami FL	1157	1094	1275	1195	1028	1047	1104	1105	1270	1224	933	617		204	657
14 Orlando FL	956	896	1070	990	824	882	944	944	1119	1039	759	413	204		461
15 Charlotte NC	506	437	662	589	427	473	540	540	728	584	336	206	657	461	

League instance

Data will be available on a website to be launched soon.