



KADIR HAS UNIVERSITY  
SCHOOL OF GRADUATE STUDIES  
DEPARTMENT OF INDUSTRIAL ENGINEERING

**THE REST DIFFERENCE PROBLEM IN ROUND-ROBIN  
TOURNAMENTS**

TASBIH TUFFAHA

DOCTOR OF PHILOSOPHY THESIS

ISTANBUL, JANUARY, 2022

Tasbih Tuffaha

Ph.D. Thesis

2022



# THE REST DIFFERENCE PROBLEM IN ROUND-ROBIN TOURNAMENTS



TASBIH TUFFAHA

A thesis submitted to  
the School of Graduate Studies of Kadir Has University  
in partial fulfilment of the requirements for the degree of  
Doctor of Philosophy in  
Industrial Engineering

İstanbul, January, 2022

## APPROVAL

This thesis titled THE REST DIFFERENCE PROBLEM IN ROUND-ROBIN TOURNAMENTS submitted by TASBIH TUFFAHA, in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Industrial Engineering is approved by

Assist. Prof. Burak avdarođlu (Advisor) .....  
Kadir Has University

Assoc. Prof. S. Tankut Atan (Co-Advisor) .....  
Baheşehir University

Prof. Dr. Ahmet Yucekaya .....  
Kadir Has University

Assist. Prof. Dilek Gnne Damıř .....  
zyeđin University

Assist. Prof. Mehmet nal .....  
zyeđin University

Assist. Prof. Esra Ađca Aktun .....  
Kadir Has University

I confirm that the signatures above belong to the aforementioned faculty members.

.....  
Prof. Dr. Mehmet Timur Aydemir  
Dean of School of Graduate Studies  
Date of Approval: 14.01.2022

**DECLARATION ON RESEARCH ETHICS AND  
PUBLISHING METHODS**

I, TASBIH TUFFAHA; hereby declare

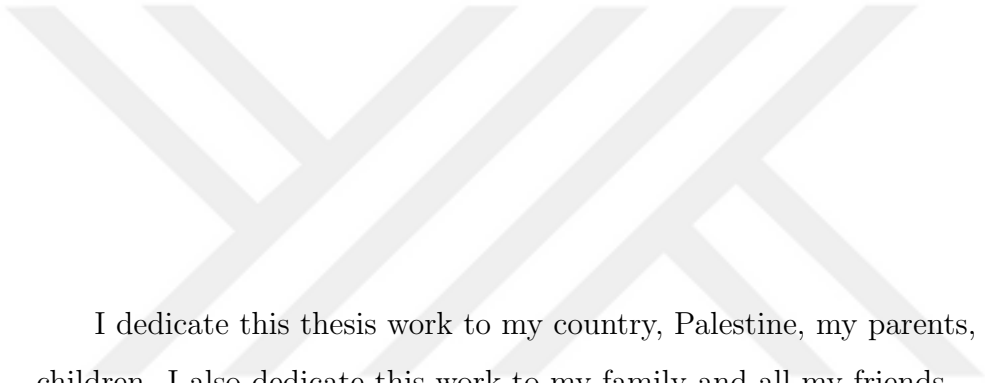
- that this Ph.D. Thesis that I have submitted is entirely my own work and I have cited and referenced all material and results that are not my own in accordance with the rules;
- that this Ph.D. Thesis does not contain any material from any research submitted or accepted to obtain a degree or diploma at another educational institution;
- and that I commit and undertake to follow the “Kadir Has University Academic Codes and Conduct” prepared in accordance with the “Higher Education Council Codes of Conduct”.

In addition, I acknowledge that any claim of irregularity that may arise in relation to this work will result in a disciplinary action in accordance with university legislation.

TASBIH TUFFAHA

.....

14.01.2022



I dedicate this thesis work to my country, Palestine, my parents, husband, and children. I also dedicate this work to my family and all my friends.

## ACKNOWLEDGMENT

First of all, I am deeply thankful to my advisor Dr. Burak Çavdarođlu and my co-advisor Dr. S. Tankut Atan for their appreciated advice, enduring support, and patience throughout my research and writing the thesis. Without their invaluable guidance, this study would have never been accomplished. Their immense knowledge and plentiful experience have encouraged me in all the time of my research.

Many thanks to all staff members in the School of Graduate Studies in Kadir Has University for their kind help and support that have made my study in Turkey a wonderful time.

Last but not least, I would like to express my deepest gratitude to my parents, husband, and children. Without their tremendous understanding and encouragement in the past few years, it would be impossible for me to complete my study. I also appreciate all the support and encouragement I received from the rest of my family and friends.

## ABSTRACT

Fairness is a key consideration in designing tournament schedules. When two teams play against each other, it is only fair to let them rest the same amount of time before their game. In this study, we aim to reduce, if not eliminate, the difference between the rest durations of opposing teams in each game of a round-robin tournament. The rest difference problem proposed in this study constructs a timetable that determines both the round and the matchday of each game such that the total rest difference throughout the tournament is minimized. We provide a mixed-integer programming formulation and a matheuristic algorithm that solve the problem. Moreover, we develop a polynomial-time exact algorithm for some special cases of the problem. This algorithm finds optimal schedules with zero total rest difference when the number of teams is a positive-integer power of 2 and the number of games in each day is even. Some theoretical results regarding tournaments with one-game matchdays are also provided.

**Keywords:** sports scheduling, round-robin tournaments, rest difference, heuristics



# LİG USULÜ TURNUVALARDA DİNLENME SÜRESİ FARKI PROBLEMİ

## ÖZET

Adillik, turnuva çizelgelerinin tasarlanmasında dikkat edilmesi gereken önemli bir husustur. İki takımın karşılaştığı bir maçta, müsabakadan önce takımların aynı süre dinlenmelerini sağlamak adil bir çizelge için gereklidir. Bu çalışmada lig usulü turnuvalarda her bir maçtaki takım çiftinin bir önceki maçlarından beri dinlendikleri süreler arasındaki farkı azaltan (mümkünse ortadan kaldıran) turnuva çizelgelerinin oluşturulması amaçlanmaktadır. Çalışmada önerilen dinlenme süresi farkı problemi, turnuva boyunca oluşan dinlenme süresi farklarının toplamını en aza indirecek şekilde her maçın hangi turda ve hangi maç gününde oynanacağını belirleyen bir maç takvimi oluşturmaktadır. Problemin çözümü için bir karma tamsayı programlama formülasyonu geliştirilmiş ve bir mat-sezgisel algoritması sunulmuştur. Ayrıca, problemin bazı özel durumları için polinom zamanlı bir kesin algoritma geliştirilmiştir. Eğer takım sayısı 2'nin pozitif tamsayı kuvvetiyse ve her maç günündeki oyun sayısı çift ise, bu algoritma toplam dinlenme süresi farkı sıfır olan maç takvimleri üretebilmektedir. Çalışmada ayrıca bazı maç günlerine tek maç atanması gereken problem örnekleri için teorik çıkarımlar verilmektedir.

**Anahtar Sözcükler:** spor çizelgeleme, lig usulü turnuvalar, dinlenme süresi farkı, sezgisel algoritmalar

# TABLE OF CONTENTS

ACKNOWLEDGMENT . . . . .	v
ABSTRACT . . . . .	vi
ÖZET . . . . .	vii
LIST OF FIGURES . . . . .	x
LIST OF TABLES . . . . .	xi
LIST OF ACRONYMS AND ABBREVIATIONS . . . . .	xiv
1. INTRODUCTION . . . . .	1
2. LITERATURE REVIEW . . . . .	6
2.1 Sports Scheduling . . . . .	6
2.2 Fairness Criteria . . . . .	9
2.2.1 Break minimization . . . . .	10
2.2.2 Balancing carryover effects . . . . .	13
2.2.3 Balancing strength groups . . . . .	16
2.2.4 Minimizing travel distances . . . . .	17
2.2.5 Minimizing rest differences . . . . .	18
2.3 Methods of Sports Scheduling . . . . .	22
2.3.1 Graphs . . . . .	22
2.3.2 Integer programming . . . . .	23
2.3.3 Constraint programming . . . . .	24
2.3.4 Heuristics . . . . .	25
3. REST DIFFERENCE PROBLEM . . . . .	30
3.1 Problem Definition . . . . .	30
3.2 Mixed-integer Programming (MIP-RDP) Model . . . . .	30
3.2.1 Sets . . . . .	30
3.2.2 Parameters . . . . .	31
3.2.3 Decision variables . . . . .	31
3.2.4 Mathematical model for the RDP (MIP-RDP) . . . . .	31
3.3 Theoretical Results for the Problem with One-game Match- days . . . . .	33

4. SOLUTION METHODOLOGIES . . . . .	38
4.1 A Polynomial-time Exact Method for Some Special Cases of the Problem . . . . .	38
4.1.1 <i>Step 1: Construct the groupings</i> . . . . .	39
4.1.2 <i>Step 2: Generate the games</i> . . . . .	40
4.1.3 <i>Step 3: Eliminate the rest differences</i> . . . . .	41
4.2 An ILS-based Matheuristic Algorithm for the Problem . .	47
4.2.1 <i>Step 1: Finding an initial schedule</i> . . . . .	48
4.2.2 <i>Step 2: Finding a schedule with an improved RD</i>	50
5. EXPERIMENTAL RESULTS . . . . .	51
5.1 Experimental Results of MIP-RDP . . . . .	51
5.2 Experimental Results of the Matheuristic . . . . .	52
5.3 Experimental Results of the Polynomial-time Exact Method	56
5.4 Results Comparison . . . . .	59
6. CONCLUSIONS . . . . .	62
References . . . . .	64
APPENDIX A: REFERENCE SOLUTIONS . . . . .	71
Curriculum Vitae . . . . .	91

## LIST OF FIGURES

Figure 2.1	A visual representation of the circle method . . . . .	9
Figure 2.2	Group-changing and group-balanced schedules . . . . .	17
Figure 2.3	A tournament with $n = 4$ represented by one-factorization of $K_4$	22
Figure 2.4	Examples of neighborhood structures . . . . .	26
Figure 2.5	The main two phases of TARS approach . . . . .	28
Figure 5.1	Best $RD$ values for each $\lambda_{crm}$ , $\lambda_{vm}$ and $\lambda_{tars}$ in the instances (a) $RDP(16, 8 1, 1, 1, 1, 1, 1, 1, 1)$ , (b) $RDP(18, 6 2, 2, 2, 1, 1, 1)$ and (c) $RDP(20, 5 2, 2, 2, 2, 2)$ . . . . .	55



## LIST OF TABLES

Table 1.1	An example of an <i>RDP</i> with 10 teams and 3 matchdays . . . . .	3
Table 1.2	Rest durations (in days) for the timetable given in Table 1.1 . . . . .	4
Table 2.1	Examples of symmetry structures used in DRRTs . . . . .	8
Table 2.2	An opponent schedule of a tournament with 6 teams . . . . .	10
Table 2.3	HAP for the opponent schedule shown in Table 2.2 . . . . .	11
Table 2.4	The <i>coe</i> values for the opponent schedule shown in Table 2.2 . . . . .	14
Table 2.5	The best <i>coe</i> values in the literature for several numbers of teams . . . . .	15
Table 3.1	A schedule for the $RDP(8, 3 2, 1, 1)$ . . . . .	33
Table 3.2	A schedule for the $RDP(10, 4 2, 1, 1, 1)$ . . . . .	35
Table 4.1	Team groupings for $n = 16$ obtained via swap operations . . . . .	40
Table 4.2	Generating the games for the first seven rounds of the $RDP(16, 4 2, 2, 2, 2)$ . . . . .	41
Table 4.3	An optimal schedule of the $RDP(16, 4 2, 2, 2, 2)$ . . . . .	42
Table 4.4	Eliminating rest differences of a swap operation . . . . .	43
Table 4.5	Eliminating rest differences of a swap operation . . . . .	43
Table 4.6	Team groupings for $n = 8$ . . . . .	44
Table 4.7	Finished schedule for the $RDP(8, 2 2, 2)$ . . . . .	44
Table 4.8	Building a schedule's team groupings . . . . .	46
Table 4.9	An optimal schedule ( $RD = 56$ ) of the $RDP(16, 6 2, 2, 1, 1, 1, 1)$ . . . . .	47
Table 5.1	Team numbers in the top-tier sports leagues of five European countries . . . . .	51
Table 5.2	Results of MIP-RDP . . . . .	53
Table 5.3	Results of the matheuristic . . . . .	57
Table 5.4	A sample of instances solved by different neighborhood structures . . . . .	58
Table 5.5	A sample of instances solved by TARS with two different approaches . . . . .	58
Table 5.6	Results of the exact method . . . . .	58
Table 5.7	Results comparison . . . . .	60
Table 5.8	Proven optimal solutions . . . . .	61

Table A.1	An optimal schedule ( $RD = 0$ ) of the $RDP(8, 2 2, 2)$ . . . . .	71
Table A.2	An optimal schedule ( $RD = 12$ ) of the $RDP(8, 3 2, 1, 1)$ . . . . .	71
Table A.3	An optimal schedule ( $RD = 24$ ) of the $RDP(8, 4 1, 1, 1, 1)$ . . . . .	72
Table A.4	An optimal schedule ( $RD = 0$ ) of the $RDP(10, 2 3, 2)$ . . . . .	72
Table A.5	An optimal schedule ( $RD = 16$ ) of the $RDP(10, 3 2, 2, 1)$ . . . . .	72
Table A.6	An optimal schedule ( $RD = 32$ ) of the $RDP(10, 4 2, 1, 1, 1)$ . . . . .	73
Table A.7	An optimal schedule ( $RD = 48$ ) of the $RDP(10, 5 1, 1, 1, 1, 1)$ . . . . .	73
Table A.8	An optimal schedule ( $RD = 0$ ) of the $RDP(12, 2 3, 3)$ . . . . .	73
Table A.9	An optimal schedule ( $RD = 0$ ) of the $RDP(12, 3 2, 2, 2)$ . . . . .	74
Table A.10	An optimal schedule ( $RD = 20$ ) of the $RDP(12, 4 2, 2, 1, 1)$ . . . . .	74
Table A.11	An optimal schedule ( $RD = 40$ ) of the $RDP(12, 5 2, 1, 1, 1, 1)$ . . . . .	74
Table A.12	An optimal schedule ( $RD = 60$ ) of the $RDP(12, 6 1, 1, 1, 1, 1, 1)$ . . . . .	75
Table A.13	The best schedule ( $RD = 2$ ) of the $RDP(14, 2 4, 3)$ . . . . .	75
Table A.14	The best schedule ( $RD = 12$ ) of the $RDP(14, 3 3, 2, 2)$ . . . . .	76
Table A.15	The best schedule ( $RD = 40$ ) of the $RDP(14, 4 2, 2, 2, 1)$ . . . . .	76
Table A.16	The best schedule ( $RD = 62$ ) of the $RDP(14, 5 2, 2, 1, 1, 1)$ . . . . .	77
Table A.17	The best schedule ( $RD = 86$ ) of the $RDP(14, 6 2, 1, 1, 1, 1, 1)$ . . . . .	77
Table A.18	The best schedule ( $RD = 112$ ) of the $RDP(14, 7 1, 1, 1, 1, 1, 1, 1)$ . . . . .	78
Table A.19	An optimal schedule ( $RD = 0$ ) of the $RDP(16, 2 4, 4)$ . . . . .	78
Table A.20	The best schedule ( $RD = 18$ ) of the $RDP(16, 3 3, 3, 2)$ . . . . .	79
Table A.21	An optimal schedule ( $RD = 0$ ) of the $RDP(16, 4 2, 2, 2, 2)$ . . . . .	79
Table A.22	An optimal schedule ( $RD = 28$ ) of the $RDP(16, 5 2, 2, 2, 1, 1)$ . . . . .	80
Table A.23	An optimal schedule ( $RD = 56$ ) of the $RDP(16, 6 2, 2, 1, 1, 1, 1)$ . . . . .	80
Table A.24	An optimal schedule ( $RD = 84$ ) of the $RDP(16, 7 2, 1, 1, 1, 1, 1, 1)$ . . . . .	81
Table A.25	An optimal schedule ( $RD = 112$ ) of the $RDP(16, 8 1, 1, 1, 1, 1, 1, 1,$ 1) . . . . .	81
Table A.26	The best schedule ( $RD = 8$ ) of the $RDP(18, 2 5, 4)$ . . . . .	82
Table A.27	The best schedule ( $RD = 30$ ) of the $RDP(18, 3 3, 3, 3)$ . . . . .	82
Table A.28	The best schedule ( $RD = 44$ ) of the $RDP(18, 4 3, 2, 2, 2)$ . . . . .	83
Table A.29	The best schedule ( $RD = 82$ ) of the $RDP(18, 5 2, 2, 2, 2, 1)$ . . . . .	83
Table A.30	The best schedule ( $RD = 114$ ) of the $RDP(18, 6 2, 2, 2, 1, 1, 1, 1)$ . . . . .	84

Table A.31	The best schedule ( $RD = 144$ ) of the $RDP(18, 7 2, 2, 1, 1, 1, 1, 1)$	84
Table A.32	The best schedule ( $RD = 176$ ) of the $RDP(18, 8 2, 1, 1, 1, 1, 1, 1)$	85
Table A.33	The best schedule ( $RD = 206$ ) of the $RDP(18, 9 1, 1, 1, 1, 1, 1, 1,$ 1) . . . . .	85
Table A.34	The best schedule ( $RD = 10$ ) of the $RDP(20, 2 5, 5)$ . . . . .	86
Table A.35	The best schedule ( $RD = 32$ ) of the $RDP(20, 3 4, 3, 3)$ . . . . .	86
Table A.36	The best schedule ( $RD = 58$ ) of the $RDP(20, 4 3, 3, 2, 2)$ . . . . .	87
Table A.37	The best schedule ( $RD = 96$ ) of the $RDP(20, 5 2, 2, 2, 2, 2)$ . . . . .	87
Table A.38	The best schedule ( $RD = 134$ ) of the $RDP(20, 6 2, 2, 2, 2, 1, 1)$ . . . . .	88
Table A.39	The best schedule ( $RD = 162$ ) of the $RDP(20, 7 2, 2, 2, 1, 1, 1, 1)$	88
Table A.40	The best schedule ( $RD = 194$ ) of the $RDP(20, 8 2, 2, 1, 1, 1, 1, 1, 1)$	89
Table A.41	The best schedule ( $RD = 232$ ) of the $RDP(20, 9 2, 1, 1, 1, 1, 1, 1, 1,$ 1) . . . . .	89
Table A.42	The best schedule ( $RD = 262$ ) of the $RDP(20, 10 1, 1, 1, 1, 1, 1, 1, 1,$ 1, 1) . . . . .	90

## LIST OF ACRONYMS AND ABBREVIATIONS

A	Away game
AA	Two consecutive away games
BMP	Break Minimization Problem
CP	Constraint Programming
DRRT	Double Round-Robin Tournament
FBTS	First Break Then Schedule
FSTB	First Schedule Then Break
GOP	Given Opponent Schedule
H	Home game
HAB	Home Away Bye
HAP	Home Away Pattern
HH	Two consecutive home games
IP	Integer Programming
LB	Lower Bound
MIP	Mixed Integer Programming
PRS	Partial Round Swap
PTS	Partial Team Swap
RD	Total Rest Difference
RDP	Rest Difference Problem
RMP	Rest Mismatch Problem
RRT	Round-Robin Tournament
RS	Round Swap
SRRT	Single Round-Robin Tournament
TARS	Teams and Rounds Swap
TC-BMP	Timetable Constrained Break Minimization Problem
TS	Team Swap
TTP	Traveling Tournament Problem



## 1. INTRODUCTION

Sports scheduling (timetabling) has always been a challenging problem for tournament organizers and a popular research topic for the practitioners of operations research. In particular, round-robin is a popular format of sports competitions. Round-robin scheduling focuses on scheduling problems to construct fair timetables for round-robin tournaments compatible with several restrictions imposed by the organizers. In round-robin tournaments, every pair of teams plays each other a fixed number of times during the competition. In single round-robin tournaments (SRRTs), each team plays each other team exactly once, while in double round-robin tournaments (DRRTs), each team plays each other twice.

A round is a set of games in which every team plays at most one game. In time-constrained (compact) round-robin tournaments, every team plays exactly one game in each round and the games are scheduled to the minimum number of rounds necessary for finishing all the games. In a compact SRRT, if the number of teams is  $n$ , this minimum number happens to be  $n - 1$ . In contrast, time-relaxed round-robin tournaments schedule the games to more than  $n - 1$  rounds due to reasons such as the unavailability of teams or venues at some specific rounds. The survey by Rasmussen and Trick (2008) provides a coherent explanation of the various notions and terminology on round-robin scheduling. The fairness of a compact round-robin tournament can be defined with respect to various criteria, such as minimizing breaks, travel distances or rest differences, and balancing carryover effects or strength groups.

One of the most researched criteria is the minimization of breaks. A break occurs when a team plays two consecutive home games or two consecutive away games. The carry-over effect is also a well-researched criterion in sports literature. Assume a team, say team  $x$ , plays two consecutive games against two opponents. Team  $x$  may be affected positively (or negatively) after playing against the first opponent

due to the level of challenge in the game and may (or may not) play with its full potential in the next game against the second opponent. In this situation, it is said that the second opponent receives a carry-over effect from the first one. Minimization of total distance traveled by teams is another well-known criterion. When a team plays several consecutive away games without returning home, minimizing the total distance traveled by a team becomes the top priority. Another criterion is related to the strength groups, each of which contains teams of similar strength. When teams are partitioned into several strength groups, it is preferable to schedule the teams in such a way that avoids a team playing against the teams of the same group (e.g., extremely weak or extremely strong teams) in consecutive rounds.

Another criterion recently studied for round-robin tournaments is related to the difference in rest durations between the pair of teams playing against each other in a game, which is also the focus of this dissertation. In round-robin tournaments, when two teams play against each other, it is only fair to let them rest the same amount of time before their game. If the opposing teams are not rested equally before the game against each other, the team with a longer rest time will have an advantage over the less rested one. This is mainly because the less rested team is likely to be more tired while the opposing team can play to its full potential. Therefore, tournament organizers should try to minimize the difference in rest durations between opposing teams in designing fair timetables. If a league timetable ignores this aspect, it is often criticized by team managers whose teams have fewer days to rest than their opponents.

As a recent example of this situation, Premier League team West Ham United FC's coach David Moyes made a complaint to the press just before their league game against Tottenham Hotspur FC (Whetstone, 2020). Moyes emphasized that they had rested one day less than Tottenham since their last league matches, and commented that this situation was against them. West Ham lost that game 2-0. In another example, Frank Lampard, the former manager of Chelsea FC, moaned about Liverpool FC having two extra days to rest for Super Cup game against them

in the press conference aftermath of the game, which they lost on the serial penalties (Sandford, 2019).

Durán (2021), a recent survey on sports scheduling, addresses the rest difference as ‘a topic that will no doubt arise in the future’. In our study, we aim to reduce, if not eliminate, the difference between the rest durations of the opposing teams in a game. To be more specific, we investigate the *rest difference problem (RDP)* of minimizing the sum of rest differences in the games of a compact SRRT, namely the *total rest difference (RD)*. In RDP, we determine the round and matchday of each game so that the total rest difference is minimized. Although the thesis assumes the tournament in consideration is a compact SRRT, the results can easily be generalized for  $l$ -tuple round-robin tournaments where each team plays each other team  $l$  times.

In order to explain the rest difference problem, we provide an illustrative example shown in Table 1.1. In this example, we consider an SRRT in which each one of the  $n = 10$  teams plays against all other teams exactly once. Thus, the tournament is composed of 9 rounds, and each team plays exactly one game in each round. The total number of games is  $n \cdot (n - 1) / 2 = 45$ . Assuming that each round is spread into 3 consecutive matchdays, the numbers of games distributed to the first, second, and third matchdays are given to be 2, 2, and 1, respectively. It can be assumed that each round immediately starts the next day after the last matchday of the previous round. Thus, the tournament lasts for 27 days.

**Table 1.1** An example of an *RDP* with 10 teams and 3 matchdays

Round	Day1		Day2		Day3
1	1-10	2-9	3-8	4-7	5-6
2	10-2	7-5	8-4	9-3	1-6
3	1-4	5-3	6-2	7-10	8-9
4	2-3	8-6	9-5	10-4	1-7
5	4-5	10-8	2-7	3-6	1-9
6	1-5	6-4	7-3	8-2	9-10
7	1-8	9-7	10-6	2-5	3-4
8	4-2	1-3	5-10	6-9	7-8
9	1-2	3-10	4-9	5-8	6-7

Table 1.1 provides a feasible timetable for this example. Table 1.2 illustrates the rest durations of opposing teams in each game. One can observe that the opposing teams do not rest for an equal number of days in most games. For example, before the game between team 6 and team 2 in round 3, the teams have rested for 2 and 4 days, respectively. The rest difference of 2 days between the teams in this game is considered as an unfairness that works against team 6. When we sum the rest differences over all the games, the total rest difference  $RD$  in this tournament is found to be 38. Indeed, it is possible to find an optimal timetable with  $RD = 16$  for this example as it will be explained later.

**Table 1.2** Rest durations (in days) for the timetable given in Table 1.1

Round	Day1		Day2		Day3
1	-	-	-	-	-
2	3-3	2-1	3-3	4-3	5-3
3	1-2	3-2	2-4	4-4	4-4
4	2-3	1-2	2-4	3-4	5-4
5	2-2	2-3	4-2	4-4	3-4
6	1-3	2-3	3-3	4-3	3-5
7	3-2	1-2	2-4	3-4	4-5
8	1-2	3-1	3-3	3-4	5-5
9	3-3	3-2	4-3	3-2	4-3

In the thesis, we provide a mixed-integer programming formulation and a matheuristic algorithm for solving the rest difference problem. Moreover, we develop a polynomial-time exact algorithm for some special cases of the problem. This algorithm finds optimal schedules with zero total rest difference when the number of teams is a positive-integer power of 2 and the number of games in each day is even. Furthermore, we present some theoretical results regarding tournaments with one-game matchdays.

The structure of the study is organized as follows. The literature review related to our topic is provided in Chapter 2. The formal description of the  $RDP$  and the mixed-integer programming formulation of the problem are presented in Chapter 3. In Chapter 3, we also provide some theoretical results regarding tournaments with one-game matchdays. The solution methodologies are explained in Chapter 4,

including the description of the polynomial-time exact method and the matheuristic algorithm. Our experimental results are reported in Chapter 5, followed by a conclusion in Chapter 6.



## 2. LITERATURE REVIEW

### 2.1 Sports Scheduling

Sports scheduling is the process of organizing games between multiple teams into what is called a tournament. A tournament consists of a series of games in which teams play against each other. A Round-Robin Tournament (RRT) is a popular form of organizing sports competitions. In round-robin tournaments, every pair of teams plays each other a fixed number of times during the competition. In Single Round-Robin Tournaments (SRRTs), every team plays every other team exactly once, while in Double Round-Robin Tournaments (DRRTs), every team plays every other team twice. Rasmussen and Trick (2008) provide the definitions of the various notions and terminology related to round-robin tournaments.

A round is a set of games in which every team plays at most one game. In time-constrained (compact) round-robin tournaments, games are scheduled in a minimum number of rounds necessary for finishing all the games. If the number of teams is  $n$ , this minimum number happens to be  $n - 1$  in SRRTs. Therefore, in a compact SRRT with  $n$  teams, the tournament consists of  $n \cdot (n - 1)/2$  games arranged into  $(n - 1)$  rounds. In contrast, a time-relaxed round-robin tournament is scheduled in more than the necessary minimum rounds (i.e., more than  $(n - 1)$  rounds in an SRRT) due to various reasons, such as the unavailability of teams or venues at some specific rounds. In this study, we assume that the tournament in consideration is a compact single round-robin tournament. In other words, the games should be assigned to rounds such that each team plays exactly a single game in each round.

In the thesis, we name the schedule only providing the opponent of each team in each round as the *opponent schedule*. The opponent schedule does not specify the venue (i.e., the home team) and the matchday of the games. The *home-away-pattern*

(*HAP*) for a team decides whether the team plays at home or away in its games given by the opponent schedule. The *HAP set*, the set consisting of each team's HAP, determines in which venue (home or away) each team plays in each round. The *league fixture* is the term we will use for the schedules that combines opponent schedule with a compatible HAP set. It specifies the games to be played in each round with the information of which team will be the home team in each game. In the thesis, the term *schedule* or *timetable* is used to describe the league schedule that combines the opponent schedule with the information of the matchdays when the games are played. Thus, the rest difference problem in this study decided a timetable (schedule) which minimizes the total rest difference (RD) value.

In DRRTs, the tournament is often split into two intervals. If each interval contains an SRRT of the same teams, the tournament is said to be *phased*. Thus, in a phased DRRT, no team plays against another team for the second time, before it has played at least one match against all the other teams. There are different symmetry structures used in constructing phased DRRTs. In the mirrored structure, the opponents in the second interval are identical to the opponents of the first interval. In the inverted structure, the second interval is played in the reversed order of the first interval. In the English structure, the opponents in the first time slot of the second interval are the same as in the last time slot of the first interval, and the opponents of the  $i^{th}$  time slot of the second interval correspond with opponents of the  $(i - 1)^{th}$  time slot of the first interval. In the French structure, the opponents in the last time slot of the second interval are the same as in the first time slot of the first interval. For all other time slots, the opponents of the  $i^{th}$  time slot in the second interval are the same as in the  $(i + 1)^{th}$  time slot of the first interval (Van Bulck, Goossens, Schönberger, & Guajardo, 2020). Finally, in the free-style structure, we do not follow any rules, so the order of opponents in the second interval are independent of the order of opponents in the first interval. Table 2.1 illustrates examples of these symmetry structures.

**Table 2.1** Examples of symmetry structures used in DRRTs

	First Interval					Second Interval				
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
Mirrored	(1,2)	(3,4)	(5,6)	(7,8)	(9,10)	(2,1)	(4,3)	(6,5)	(8,7)	(10,9)
Inverse	(1,2)	(3,4)	(5,6)	(7,8)	(9,10)	(10,9)	(8,7)	(6,5)	(4,3)	(2,1)
English	(1,2)	(3,4)	(5,6)	(7,8)	(9,10)	(10,9)	(2,1)	(4,3)	(6,5)	(8,7)
French	(1,2)	(3,4)	(5,6)	(7,8)	(9,10)	(4,3)	(6,5)	(8,7)	(10,9)	(2,1)
Free-style	(1,2)	(3,4)	(5,6)	(7,8)	(9,10)	(6,5)	(2,1)	(4,3)	(8,7)	(10,9)

A special type of opponent schedule for constructing round-robin tournaments is known as *canonical schedule*. In the canonical schedule of an SRRT with even  $n$  teams, the set of the games of round  $r$  is given by the following equation.

$$Games_r = \{[n, r]\} \cup \{[r + k, r - k]; k = 1, \dots, n/2 - 1\} \quad (2.1)$$

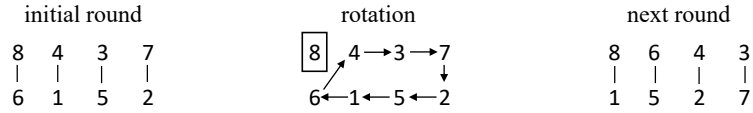
In this equation, terms  $r + k$  and  $r - k$  are expressed as  $mod(r + k, n - 1)$  and  $mod(r - k, n - 1)$ , respectively. In other words, they can only get one of the values in the set  $\{1, 2, \dots, n\}$ . For example, according to Equation 2.1, in a tournament with  $n = 8$ , the games of the third round ( $Games_3$ ) turn out to be  $\{[8, 3], [4, 2], [5, 1], [6, 7]\}$ . Canonical schedules are popular since they can guarantee a league fixture with the theoretical lower bound of the break minimization problem (which is  $n - 2$  for an even number of teams  $n$ ) if a proper HAP set is used (de Werra, 1981). This HAP set can be identified with the following two rules.

- For each game  $[n, r]$ , team  $r$  plays at home if  $r$  is even, and away if  $r$  is odd.
- For each game  $[r + k, r - k]$ , team  $r + k$  plays at home if  $k$  is even, and away if  $k$  is odd.

Another popular method for constructing canonical schedules is the *circle method* (Çavdaroglu & Atan, 2020; de Werra, 1981; Lambrechts, Ficker, Goossens, & Spieksma, 2018; Suksompong, 2016). The circle method works as follows. Suppose we apply the method with  $n$  teams. For the first round, we line the teams up in two rows and align each team in the first row with another team in the second. Thus, the games of the first round are composed of the pairs of aligned teams. To generate the games of the next round, the top-left team is fixed and the remaining teams are rotated



one step clockwise. If we apply the rotation  $n - 2$  times, an SRRT involving these  $n$  teams is constructed (Suksompong, 2016). Figure 2.1 shows the visual representation of two rounds generated using the circle method for  $n = 8$  teams. According to the figure, the games of the initial round are  $\{(8, 6), (4, 1), (3, 5), (7, 2)\}$ . After the rotation, the games of the next round are obtained as  $\{(8, 1), (6, 5), (4, 2), (3, 7)\}$ .



**Figure 2.1** A visual representation of the circle method

As of the year 2018, the professional football leagues of European countries such as Turkey, Russia and Portugal were scheduled using the canonical schedule. The canonical schedule was preferred in these countries because it can minimize the number of breaks. It may not have been preferred if additional criteria such as carryover effect or rest differences had been taken into account. For example, as of 2018, some countries such as England, Germany, France and Belgium were using customized schedules instead of the canonical schedule in their professional football leagues because of the criteria they considered beyond break minimization (D. Goossens, 2018).

## 2.2 Fairness Criteria

In sports scheduling, fairness issues are quite significant. Ensuring fairness between teams is the main consideration in designing sports schedules. Several fairness criteria are considered in developing sports schedules. For the compact round robin tournaments, the most popular fairness criteria are *(i)* minimizing breaks, *(ii)* balancing carryover effects, *(iii)* balancing strength groups, *(iv)* minimizing travel distances, and *(v)* minimizing rest differences. For a broad discussion of various fairness criteria in sports scheduling, one may refer to Kendall, Knust, Ribeiro, and Urrutia (2010) and Ribeiro (2012).

### 2.2.1 Break minimization

Break Minimization is one of the most researched criteria in sports scheduling. When a team plays a game in its venue, we say this team has a home game (H), and the opposite team has an away game (A). The home-away-pattern (HAP) is the sequence of home games and away games played by a single team. The break occurs when the team plays two consecutive home games (HH) or away games (AA).

In order to illustrate the occurrence of break in round-robin tournaments, Table 2.2 provides an illustrative opponent schedule for an SRRT with  $n = 6$  teams. Each row in the table shows the list of opponents of the corresponding team in 5 rounds (weeks). Table 2.3 shows a HAP set which is compatible with the opponent schedule presented in Table 2.2. For example, according to Table 2.3, team 1 has its first game at home (H) while team 4 plays its game away (A) at the third round. Since the consecutive home (HH) or consecutive away (AA) games in the HAP set ‘break’ the alternating home and away pattern, this phenomenon is named as break. In Table 2.3, the break occurrences are shown by underlined H’s or A’s. When the HAP set given in Table 2.3 is assigned to the opponent schedule of Table 2.2, a league fixture having 6 breaks is obtained.

**Table 2.2** An opponent schedule of a tournament with 6 teams

		Rounds				
		<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
	<b>1</b>	2	3	4	5	6
	<b>2</b>	1	4	5	6	3
Teams	<b>3</b>	5	1	6	4	2
	<b>4</b>	6	2	1	3	5
	<b>5</b>	3	6	2	1	4
	<b>6</b>	4	5	3	2	1

An important point here is that the HAPs given in Table 2.3 has to be compatible with the opponent schedule of Table 2.2. For example, since there is a game between teams 3 and 5 in the first round, these teams cannot simultaneously play at home (H) or away (A) in this round. Additionally, since every team has to play against every other team in round-robin tournaments, it is impossible that two teams pos-

**Table 2.3** HAP for the opponent schedule shown in Table 2.2

		Rounds				
		<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
Teams	<b>1</b>	H	A	H	A	H
	<b>2</b>	A	H	A	H	A
	<b>3</b>	A	H	A	H	<u>H</u>
	<b>4</b>	H	A	<u>A</u>	<u>A</u>	<u>H</u>
	<b>5</b>	H	A	<u>H</u>	<u>H</u>	A
	<b>6</b>	A	H	<u>H</u>	A	<u>A</u>

sess the same HAP. Therefore, one cannot arbitrarily decide the HAP of a team. Namely, the HAP set of a league fixture has to be feasible with regard to the given opponent schedule. The problem of deciding the HAP for each team for a given opponent schedule (e.g., Table 2.2) in order to minimize the total breaks is known as the *timetable constrained break minimization problem (TC-BMP)* in the literature. Although the computational complexity of the TC-BMP is still an open research question, it is conjectured that the problem is NP-hard (Elf, Jünger, & Rinaldi, 2003). On the other hand, the *classic break minimization problem (BMP)* tries to minimize the total number of breaks under a series of hard constraints.

There exist numerous studies in the literature concerning break occurrences in round-robin tournaments. de Werra (1981) was the first who did much pioneering work concerning breaks. The study proves that for an even number of teams  $n$ , the minimum number of breaks is at least  $(n - 2)$ , while with an odd  $n$ , one can obtain a guaranteed schedule with no breaks. van 't Hof, Post, and Briskorn (2010) deal with a BMP which minimizes the number of breaks in an SRRT schedule with an even number of clubs with two teams each, and where the teams of the same club play against each other in the first round and never play at home simultaneously. Rasmussen and Trick (2007) investigate a BMP that tries to minimize the total number of breaks in a DRRT with place and separation constraints. Moreover, Urrutia and Ribeiro (2006) investigate the relation between breaks and travel distances in round-robin tournament scheduling problems.

In the literature, usually two different approaches have been used in designing sports schedules considering classic break minimization problem (BMP). In the “First

Schedule Then Break (FSTB)” approach, the games are determined first for each round, ignoring any home or away requirements, i.e., only determining the opponent schedule. Afterward, the corresponding home–away patterns are developed for each team, considering minimizing the number of breaks as much as possible. This approach is appropriate when there are critical requirements in the schedule that do not involve HAP, such as fixed-game assignments in the opponent schedule or balancing carry-over effects (which will be discussed later in this section). Thus, such requirements are met early in the schedule, and then the best HAP can be developed. Trick (2001) develops schedules in cases where a fixed-game assignment is the critical feature. He proposes a two-phase method, where the first phase is to schedule the teams considering the critical requirement and ignoring any home and away requirement, and the second phase involves assigning home and away teams.

The “First Break Then Schedule (FBTS)” approach is the opposite of the FSTB. It starts with developing feasible home–away patterns, often with a minimum number of breaks, then the opponent schedule compatible with the selected HAP set is determined. For example, Nemhauser and Trick (1998) develop an approach for constructing a schedule for a basketball competition where each school plays home and away games with each other. They propose an algorithm consisting of three main steps. First, they find a set of Home, Away, and Bye (HAB) patterns equal to the number of teams. Second, they assign games to the pattern set to come up with a league fixture. Finally, they find the final schedule by assigning the teams to the patterns. However, in both approaches; FBTS and FSTB, tournament organizers often aim to minimize the number of breaks as much as possible. In other words, they aim to alternate home games and away games for each team as much as possible.

Literature concerning the timetable constrained break minimization problem (TC-BMP) is also abundant. Régis (2001) proposes a constraint programming (CP) model that solves the instances of TC-BMP up to size 20. Trick (2001) proposes an integer programming (IP) model for TC-BMP extended with triangle valid inequalities. Elf et al. (2003) observe that an instance of TC-BMP can be transformed

into an equivalent maximum cut instance. Miyashiro and Matsui (2005) prove that finding a HAP set with exactly  $n - 2$  breaks for a given opponent schedule or showing that such a HAP set does not exist can be done in polynomial time. Miyashiro and Matsui (2006) formulate the TC-BMP as the MAX-RES-CUT problem (a variations of maximum cut problem in which a given pairs of vertices are forced to be on either the same side of the cut or on different sides of the cut) and the maximum 2-satisfiability problem. Applying Goemans and Williamson's approximation algorithm using semidefinite programming, they manage to generate solutions with good approximation ratios for the instances of TC-BMP up to 40 teams. Post and Woeginger (2006) show that the TC-BMP is NP-hard for a given partial opponent schedule with no less than three rounds.

### 2.2.2 Balancing carryover effects

The carry-over effect is another well-researched criterion in the literature. Since teams have different characteristics, some teams can benefit from the way the tournament is scheduled. For example, if team  $k$  plays against a strong team, say team  $i$ , in a game, it may be very tired after this game. This fatigue may affect team  $k$ 's performance in the next coming game. Suppose team  $k$  will play its next game with team  $j$ , in this case, we say team  $j$  receives a carry-over effect from team  $i$ , as team  $k$  may be exhausted and easier to defeat. If team  $i$  is a weak team, then team  $k$ 's morale may be boosted after its game against team  $i$  and it may result in a more confident game against team  $j$  in the next round. In that case, team  $j$  receives an adverse carry-over effect from team  $i$ . The number of times team  $j$  receives a carry-over effect from team  $i$  in a tournament is defined as  $c_{ij}$ . In the opponent schedules where the  $c_{ij}$  values are not evenly distributed among the team pairs, some  $c_{ij}$  values, say  $c_{AB}$ , will be high resulting in team  $B$  receiving too many carry-over effects from team  $A$ .

To provide a visual description of how carryover effects can be accumulated between some team pairs, Table 2.4 gives the  $c_{ij}$  values of the opponent schedule presented in

Table 2.2 for  $n = 6$ . According to Table 2.4, team 4 receives a carry-over effect from team 1 three times during the SRRT. The reason is that the teams playing against team 1 in rounds 1, 4 and 5 (3 times in total) play against team 4 in the next rounds. Note that the next round after round 5 is assumed to be round 1 due the circular definition of carry-over effects by K. G. Russell (1980). Since every team has to receive (send) a carry-over effect from (to) a team in each of its  $n - 1$  games, the sum of the values in each column (row) must be  $n - 1$  in Table 2.4. Since a team cannot receive a carry-over effect from itself, all the diagonal values are expected to be zero.

**Table 2.4** The *coe* values for the opponent schedule shown in Table 2.2

		Teams					
		1	2	3	4	5	6
Teams	1	0	0	1	3	0	1
	2	3	0	1	0	1	0
	3	1	1	0	1	1	1
	4	0	1	1	0	3	0
	5	1	0	1	0	0	3
	6	0	3	1	1	0	0

In a round-robin tournament, we measure how balanced the  $c_{ij}$  values are distributed among team pairs by calculating the carry-over effects value (*coe*) given by  $\sum_{ij} c_{ij}^2$  (K. G. Russell, 1980). For example, the *coe* value for the opponent schedule of Table 2.2 can be calculated as 60 by squaring and then summing the  $c_{ij}$  values in Table 2.4. If every team receives exactly one carry-over effect from every other team, *coe* can get the smallest possible value, which is  $n^2 - n$ . The opponent schedule with a *coe* value of  $n^2 - n$  is called a *balanced schedule* since it guarantees the most balanced distribution of  $c_{ij}$  values among team pairs. The *carry-over effects minimization problem* first introduced by K. G. Russell (1980) aims to distribute the carry-over effects among the team pairs in the most balanced way possible (i.e., it aims to minimize the *coe* value). A generally accepted polynomial-time solution method for the problem have not been found yet, and the computational complexity of the problem is still open.

K. G. Russell (1980) investigates a balanced distribution of the  $c_{ij}$  values among team pairs. The study shows that when the number of teams is a power of 2, a balanced schedule can always be achieved. The paper also reports *coe* values for the team numbers 6, 10, 12, 14, 18, and 20. Anderson (1999) finds balanced schedules for the first time for  $n = 20$  and 22. The procedure proposed by Anderson (1999) also finds the best solutions known to date, except for  $n = 12$ , which is obtained by Guedes and Ribeiro (2011). Table 2.5 shows the best *coe* values for several numbers of teams with the studies first discovering them. The asterisk symbol (\*) indicates that the schedule is balanced.

**Table 2.5** The best *coe* values in the literature for several numbers of teams

$n$	<i>coe</i>	Reference
6	60	K. G. Russell (1980)
8	56*	K. G. Russell (1980)
10	108	Anderson (1999)
12	160	Guedes and Ribeiro (2011)
14	234	Anderson (1999)
16	240*	K. G. Russell (1980)
18	340	Anderson (1999)
20	380*	Anderson (1999)

Günneç and Demir (2019) study the carry-over effects minimization problem with the restrictions on the maximum number of breaks per team. Lambrechts et al. (2018) prove that round-robin tournaments generated by the circle method (i.e., canonical schedules) have a maximum carry-over effects value. Kidd (2010) applies a tabu search to find starters that produce schedules with small carry-over effects values. On the other hand, Guedes and Ribeiro (2011) investigate a weighted version of the carry-over effects minimization problem, in which  $c_{ij}$  values have different weights. They provide a heuristic for minimizing the weighted carry-over effects value in round-robin tournaments.

### 2.2.3 Balancing strength groups

For round-robin tournaments, in which the teams can be partitioned into  $|S| \geq 2$  strength groups, the organizers might be interested in designing schedules that avoid teams playing against extremely weak or extremely strong opponents in consecutive rounds. Two problems have been defined with respect to the strength groups. In the first problem, so called the group-changing problem, the objective is to design an opponent schedule in which no team plays against two teams of the same strength group in two consecutive rounds. In the second problem, so called the group-balanced problem, the objective is to design an opponent schedule in which no team plays more than once against teams of the same strength group within  $|S|$  consecutive periods. Note that if a schedule is group-balanced, it must also be group-changing by definition.

Figure 2.2 provides an illustrative example with  $|S| = 4$  strength groups. SG1, SG2, SG3 and SG4 can be interpreted as the strength groups of extremely weak, weak, strong and extremely strong teams, respectively. The figure shows how the opponent list of a team should look like when the schedule is (a) group-changing, (b) group-balanced, and (c) neither group-changing nor group balancing. In Figure 2.2a, the team never plays against two teams of the same strength group in two consecutive rounds. In Figure 2.2b, the team never plays more than once against teams of the same strength group within 4 consecutive periods. Note that Figure 2.2a and 2.2b gives the opponent list of a single team. The aforementioned property in Figure 2.2a (2.2b) should hold for every team's opponent list in order to classify the schedule as group-changing (group-balanced). In Figure 2.2c, the team plays against teams of the same strength group in consecutive rounds (e.g., in rounds 1&2 and 4&5), so the schedule with such an opponent list for a team can neither be group-changing nor group balancing.

Briskorn (2009) is the first study introducing the concepts of group-changing and group-balanced schedules. For an even number of teams, the study discusses in



Rounds							
1	2	3	4	5	6	7	8
SG2	SG1	SG2	SG3	SG2	SG4	SG1	SG4
(a) group-changing							
SG3	SG1	SG4	SG2	SG3	SG1	SG4	SG2
(b) group-balancing							
SG1	SG1	SG4	SG2	SG2	SG3	SG3	SG1
(c) neither group-balancing nor group-changing							

**Figure 2.2** Group-changing and group-balanced schedules

which circumstances a group-changing schedule and a group-balanced schedule can be constructed and in which cases they do not exist. For example, it is proven that a group-balanced SRRT can be constructed if and only if  $|S|$  is even and  $n/|S|$  is even. Briskorn and Knust (2010) construct group-balanced schedules based on graph models for an odd number of teams. The study also constructs group-changing schedules for an even number of teams and  $|S| = 3$  groups. Additionally, they investigate generalized problems with arbitrary group sizes. Zeng and Mizuno (2013) blends the problems of strength groups with break minimization problem. They propose a constructive method to show the existence of group-balanced SRRTs with a minimum number of breaks, if  $|S|$  is a power of 2 with identical even group sizes. It is also shown that group-changing SRRTs with a minimum number of breaks can exist, if the number of teams  $n$  is a multiple of 4 and the union of some groups contains exactly half of the teams.

#### 2.2.4 Minimizing travel distances

Travel distances due to away games is a factor that impacts the teams' performance. Minimizing the travel distance is one of the fairness criteria frequently studied in the literature. In national leagues of countries with large areas (e.g., Brazil, Argentina, Chile, China) where the travel times are long, or in international tournaments where

long distances between countries need to be traveled, minimizing the total distance traveled has become an important goal in order to reduce travel costs and give players more time to rest and train between consecutive games. Therefore, when a team has to play two consecutive away games, this team usually travels directly from the first game site to that of the second, without returning home. Thus, minimizing the total distance traveled by all teams may positively impact their performance and reduce traveling cost. Bean and Birge (1980) studied basketball games over four seasons to analyze the effect of traveling on teams' performance and league cost. They found that the unnecessary number of airline miles traveled by teams resulted in teams fatigue and dollar losses to the owners. Their proposed heuristic algorithm succeeded in reducing the traveling distance while considering the league constraints, which leads to reducing the traveling cost by 20.4%.

The Traveling Tournament Problem (TTP) aims to schedule a double round-robin tournament, considering minimizing the total distance traveled by the teams. In TTP, no two teams play against each other in two consecutive rounds. Furthermore, the number of consecutive home games or consecutive away games in TTP has upper and lower bounds for each team.

The Traveling Tournament Problem was first established by Easton, Nemhauser, and Trick (2001). Kendall (2008) studies the minimization of total distance traveled by 92 football league clubs over two complete fixtures in England. Kim (2019) investigates the total travel distance in baseball league game scheduling, including certain requirements relevant to baseball leagues. Furthermore, Thielen and Westphal (2011) and Costa, Urrutia, and Ribeiro (2012) provide studies for minimizing the total travel distance of teams.

### **2.2.5 Minimizing rest differences**

The rest duration is the rest time between two consecutive games of a team. In sports literature, there is a strong body of evidence regarding the impact of rest

duration on teams' performance. Steenland and Deddens (1997) studied 8,495 basketball games over eight seasons to analyze the effects of travel and rest on a team's performance. They find that longer time between consecutive games of a team improved its performance, and the performance is affected negatively especially when the team has only one day to rest. In contrast, the team's peak performance occurs when it has at least three days of rest. They conclude that the negative effects of shorter duration between consecutive games occurred due to the lack of time for physical recovery. Some researchers show the relation between congested fixtures and high injury rates (Bengtsson, Ekstrand, & Häggglund, 2013; Scoppa, 2015).

Furthermore, Folgado, Duarte, Marques, and Sampaio (2015) investigate the effects of congested fixtures on teams' performance in football. They find that the tactical performance measured by players' movement has decreased during these games. They claim that this reduction was associated with fatigue and consequent adaptation strategies. Some researchers show the effect of fatigue on teams' performance in football (Sanchez-Sanchez et al., 2019; Watanabe, Wicker, & Yan, 2017). R. D. Goossens, Kempeneers, Koning, and Spieksma (2015) study the effect of fatigue resulting from the previous game on players' chances of winning the next game in tennis. A recent study by Esteves, Mikolajec, Schelling, and Sampaio (2021) finds that the likelihood of winning a game in basketball increases 38% from playing back-to-back games to one day rest between games.

Some researchers dealing with sports scheduling focus on the *individual rest duration* of a team between its consecutive games. In this approach, the researchers try to balance the rest duration over teams as much as possible. For example, Schönberger, Mattfeld, and Kopfer (2004), Knust (2010), and Van Bulck, Goossens, and Spieksma (2019) investigate the time-off between games involving the same team in time-relaxed non-professional tournaments considering the team and venue availability constraints. Van Bulck and Goossens (2020) propose two heuristics to handle the problem of rest time and the difference of games played in time-relaxed timetables. They developed a measure called aggregated rest time penalty which penalizes the

timetable each time a team has rest days between two consecutive games less than a certain number.

Knust (2008) focuses on minimizing the waiting times of teams playing in a single round-robin tournament with only a single venue available. For any odd number of teams, Knust constructs schedules that minimize the number of long waiting times and the total waiting time simultaneously. D. Goossens, Yi, and Van Bulck (2020) explore the trade-off between three fairness issues which are the consecutive home games, the carryover effect, and the number of rest days each team has between consecutive games. Van Bulck and Goossens (2020) examine whether a time-relaxed SRRT can be scheduled given a game-off-day pattern or home-away pattern sets. They aim to determine the minimal number of time slots the tournament can be scheduled given an equal number of prefixed absences per team.

On the other hand, few researchers have focused on *relative rest difference*. In this approach, the researchers minimize the difference in rest durations between the opposing teams of a game to promote a fair tournament. For example, Suksompong (2016) investigates the rest time in asynchronous round-robin tournaments. In these tournaments, all games are played at different times, i.e., there is only one game in each period (round). Furthermore, Suksompong (2016) investigates the rest time through three different measures. These measures include the guaranteed rest time, the games played difference index, and the rest difference index.

The rest difference index in Suksompong (2016) measures the maximum difference in rest durations of two opposing teams in a schedule. The rest difference index differentiates from the total rest difference (RD) considered in the thesis. Suksompong (2016) looks at minimizing the maximum of rest differences, whereas we are concerned with minimizing the sum of the rest differences of all games in this study. Furthermore, Suksompong (2016) only considers time-relaxed asynchronous round-robin tournaments, i.e. the RRTs where there is exactly one game in each round.

In our study, however, we consider various problem instances of compact RRTs in which the number of games in each round are usually more than one.

The minimization of rest mismatches is a related fairness criterion introduced by Atan and Çavdaroğlu (2018). A rest mismatch occurs when the opposing teams have different rest times before their game. The *rest mismatch problem (RMP)* involves assigning games of each round to  $p$  consecutive periods, such as days or time slots of a day, in a way that rest mismatches in the tournament are avoided as much as possible. Atan and Çavdaroğlu (2018) develop a heuristic algorithm for the case of two matchdays in an SRRT. In this special case, the algorithm is capable of finding an optimal schedule with zero mismatches when the number of teams is a multiple of 8, and a schedule with four mismatches when the number of teams is a multiple of 4 but not 8. It is worth mentioning that *rest mismatch* considered in Atan and Çavdaroğlu (2018) and *rest difference* considered in this study differ because the latter deals with not just the occurrence of a mismatch in rests but the magnitude of the difference between them. Moreover, the algorithm presented in Atan and Çavdaroğlu (2018) found solutions for only the cases where  $p = 2$ , while in the *RDP* proposed here, we are interested in all possible values of  $p$ .

Last but not least, Çavdaroğlu and Atan (2020) investigate the rest difference problem for given opponent schedules, i.e. schedules in which games have already been assigned to rounds. Therefore, their problem decides only the matchday of each game. The study shows that the rest difference problem of a given schedule is decomposable into optimizing the rounds separately, and that each decomposed problem is an instance of the quadratic assignment problem. It also provides a polynomial-time exact algorithm minimizing the total rest difference for opponent schedules constructed by the circle method. Çavdaroğlu and Atan (2020) solve a variation of *RDP* in which opponent schedule is given, while our study solves a general *RDP* by constructing a timetable that determines both the round and the matchday of each game.

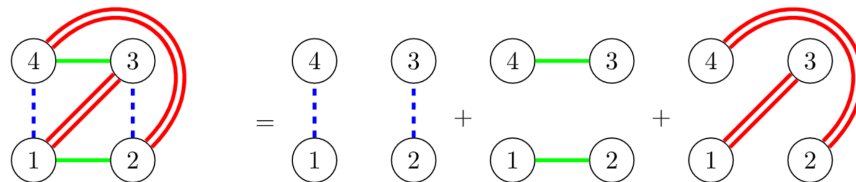
## 2.3 Methods of Sports Scheduling

Researchers follow different methods in designing sports schedules to ensure the quality and fairness of the schedules. In this section, we discuss some of the most widely used methods in sports scheduling. These methods are graphs, integer programming, constraint programming, and heuristics.

### 2.3.1 Graphs

Round-robin tournaments have a strong relationship with graph theory. Graph theoretical approaches are used by researchers in designing sports schedules. According to de Werra (1981), the relationship between graphs and tournaments can be presented by a complete graph  $K_n$  where  $n$  is an even number of teams. In such graphs, each node denotes a team, and an edge connects any two nodes denotes the game played by the two teams represented by the linked nodes.

Decomposition of a graph into factors (subgraphs) is called a one-factorization, as each one-factor (subgraph) is a collection of non-adjacent edges. In round-robin tournaments, these factors (subgraphs) correspond to rounds, and each subgraph represents a different round. The oriented coloring is another topic related to scheduling by graphs. In oriented coloring, the edges are colored by different colors to indicate in which round each game will be played. Moreover, the orientation can be given to each edge to represent the place of the game to be home or away. Figure 2.3 is an example of a tournament represented by one-factorization of  $K_4$ .



**Figure 2.3** A tournament with  $n = 4$  represented by one-factorization of  $K_4$   
Source: Januario et al. (2016)

One of the graphical methods used in constructing round-robin tournaments is the *Vizing method*. This method presents a framework for the construction of an ar-

bitrary edge coloring of a complete graph  $K_n$  with  $n - 1$  colors, where each one of  $n$  teams corresponds to a vertex, each game between teams  $i$  and  $j$  to an edge  $(i, j)$  of  $K_n$ , and each color to a distinct round (Januario, Urrutia, & Gerais-Brazil, 2012). Thus, edges with the same color are the games played during the same round, and each arbitrary edge coloring of a complete graph  $K_n$  represents an opponent schedule for an SRRT with  $n$  teams.

In sports literature, a rich line of researches exploits the link between graph theory and sports scheduling in developing sports schedules. de Werra (1981) defines the break in graphical perspective as two edges adjacent to the same node have the same direction in two consecutive rounds, either out or into this node. de Werra (1981) proves that in any oriented coloring, the minimum number of breaks is at least  $n - 2$ .

Januario and Urrutia (2016) present their neighborhood structure in graph theory terms and prove how this method increases the connectivity of the solution space. They evaluate its performance using the weighted carry-over effects minimization problem and the traveling tournament problem with predefined venues. Lewis and Thompson (2011) propose three algorithms to show how heuristic-based graph coloring methods can construct valid compact round-robin schedules, in a way to achieve the aim of producing initial feasible solutions to some scheduling problems in short amounts of time.

### **2.3.2 Integer programming**

Integer Programming (IP) is one of the most widely used methods in sports scheduling. It is used to model and solve sports scheduling problems. Integer Programming involves formulating an objective function and a set of constraints. The Objective function could be minimization or maximization of some criteria, such as minimization of breaks, coe, or rest mismatches. To solve sports scheduling problems, usually, a solver is used. The solver is optimized to solve the variables based on the objective function and constraints of the corresponding problem.

Some RRT problems can be solved directly by applying IP solver to the model. Still, in most cases, the problem is decomposed into stages to handle each stage of the problem by IP or by other methods, such as heuristics or constraint programming (Kendall et al., 2010). Briskorn, Drexl, and Spieksma (2010) provide integer programming models where the problem, such as break minimization, is decomposed into two subproblems to be solved sequentially. Furthermore, a rich line of researchers have used the IP method in solving scheduling problems. Atan and Cavdaroglu (2018) developed an IP model for minimizing the number of rest mismatches in SRRTs. Briskorn (2008), Rasmussen (2008), and D. Goossens and Spieksma (2009) apply IP method for solving scheduling problems in football leagues. Moreover, Irnich (2010), Easton, Nemhauser, and Trick (2003), and Rasmussen and Trick (2006) apply the IP in traveling tournament problem. Briskorn and Drexl (2009) provide IP models with constraints that consider external requirements, such as security aspects and legal requirements, and constraints that consider some fairness issues, such as break minimization. On the other hand, integer programming is also a useful tool for assignment problems. For example, Duarte, Ribeiro, Urrutia, and Haeusler (2006) apply IP in assigning referees to sports leagues.

### **2.3.3 Constraint programming**

Constraint Programming (CP) is another type of modeling used in sports scheduling. It differs from integer programming in that constraint programming is modeled by sets of variables and constraints rather than objective function, and often it is faster. In constraint programming, new constraints are derived, and variable values are fixed by manipulating the original sets of constraints and variable domains (Kendall et al., 2010).

Examples of applying CP in sports scheduling problems can be found in Régim (2001), who investigates the minimization of breaks in sports scheduling using constraint programming. He proposes a CP model to prove that the minimum number of breaks in sports schedules is  $n-2$ . Atan and Cavdaroglu (2018) developed a CP model



for minimizing the number of rest mismatches in SRRTs. In addition, Rasmussen and Trick (2009) apply constraint programming with other modeling approaches for minimizing total travel distance. Furthermore, constraint programming is used for sports scheduling problems in R. A. Russell and Urban (2006), Easton et al. (2003), and Rasmussen and Trick (2006).

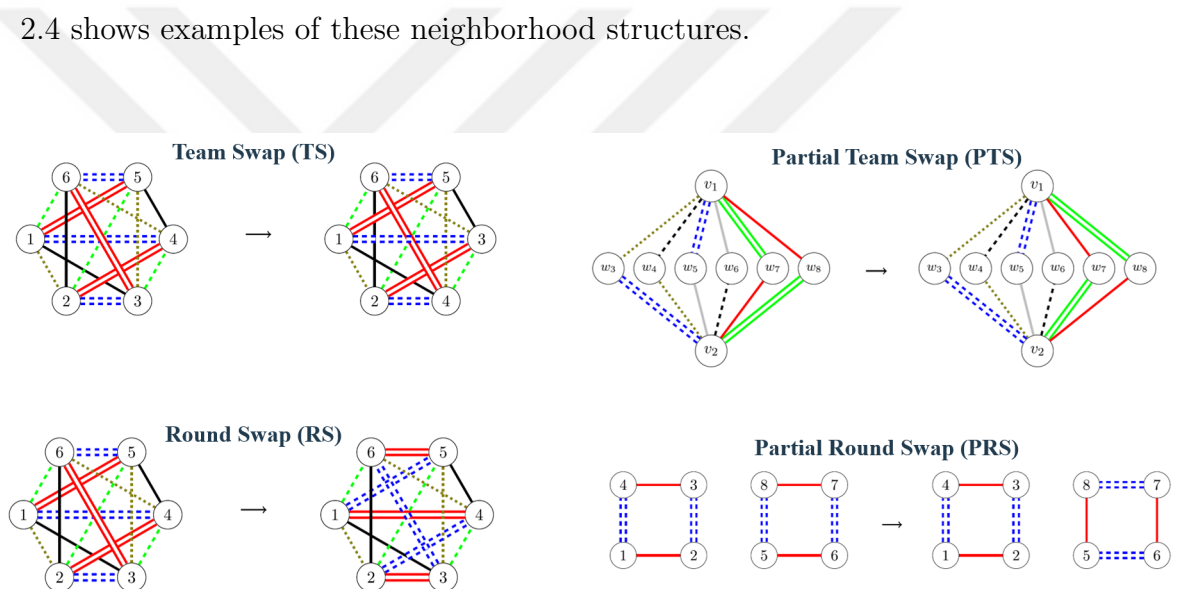
### 2.3.4 Heuristics

Most scheduling problems in sports are very hard in computational terms, such that in some cases, there is no computation time known for their solution. Therefore, the heuristic approach (or approximate algorithm) that provides sub-optimal solutions in reasonable computation time is often used in sports scheduling. Examples of using heuristics in sports scheduling problems can be found in Costa et al. (2012), Burke, De Werra, Silva, and Raess (2004), and Atan and Çavdaroğlu (2018).

Metaheuristics are a high-level approach that utilizes simple heuristics to find approximate solutions to computationally hard optimization problems. For example, tabu search (Van Bulck et al., 2019), simulated annealing (Anagnostopoulos, Michel, Van Hentenryck, & Vergados, 2006), genetic algorithms (Schönberger et al., 2004), and ant colonies (Uthus, Riddle, & Guesgen, 2009) are metaheuristics used in sports scheduling.

Matheuristics combine the characteristics of metaheuristics and mathematical programming techniques. Matheuristics are model-based heuristics that take advantage of the mathematical programming solvers to tackle hard optimization problems. In sports scheduling problems, matheuristics can exploit the features of the mathematical model to optimize some parts of the problem while the remaining parts of the problem are fixed. An example of employing matheuristics in sports scheduling can be found in Chandrasekharan, Toffolo, and Wauters (2019). Chandrasekharan et al. (2019) investigate the assignment of umpires to the games of a fixed double round-robin tournament.

Local search heuristics apply the concept of neighborhoods to move from one schedule to a neighbor schedule. It involves iteratively investigating a solution neighborhood, replacing the current solution with a better one within its neighborhood, and terminating when no better solution can be found. The commonly used neighborhood structures in round-robin tournaments are based on either swapping teams or swapping rounds in the schedule. In sports literature, different neighborhood structures have been used in scheduling round-robin tournaments. These neighborhood structures include Team Swap (TS), Round Swap (RS), Partial Team Swap (PTS), and Partial Round Swap (PRS) (Anagnostopoulos et al., 2006; Costa et al., 2012; Di Gaspero & Schaerf, 2007; Januario et al., 2016; Ribeiro & Urrutia, 2007). Figure 2.4 shows examples of these neighborhood structures.



**Figure 2.4** Examples of neighborhood structures  
Source: Januario and Urrutia (2016)

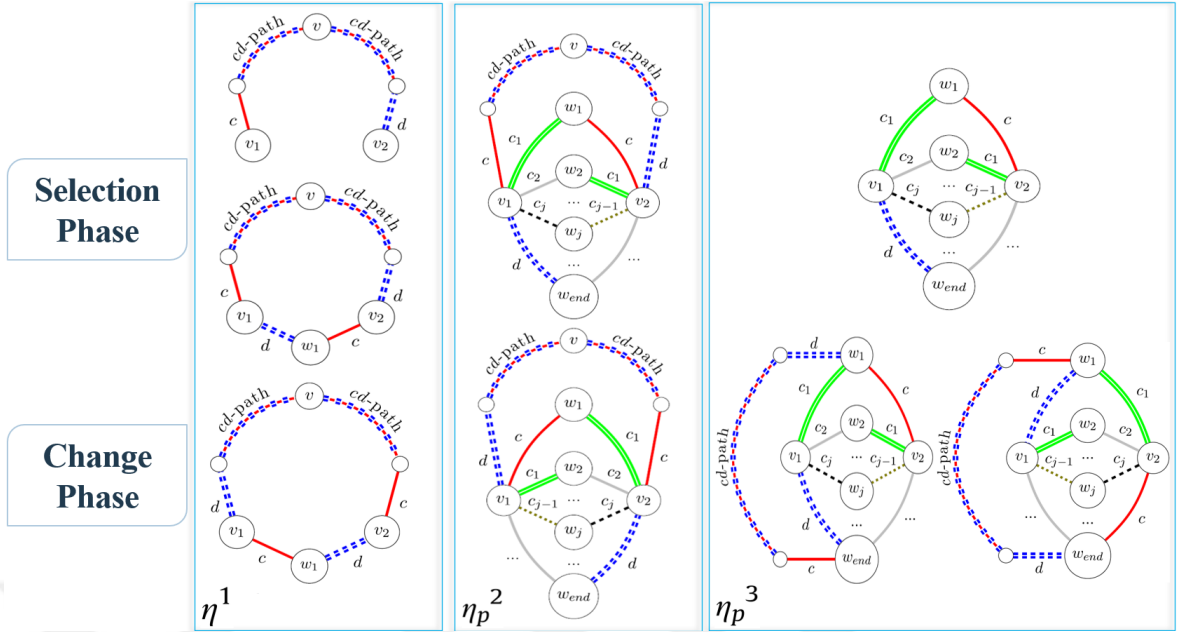
Figure 2.4 describes these neighborhood structures from a graphical perspective. Team Swap (TS) is obtained by swapping two distinct vertices in the graph. Round Swap (RS) is obtained by swapping two distinct colors in the graph. Partial Team Swap (PTS) is obtained by swapping the colors of edges adjacent to two distinct vertices  $v_1$  and  $v_2$  for all connected vertices  $w$ . Finally, Partial Round Swap (PRS) is obtained by swapping two distinct colors in any cycle in the subgraph induced by the edges colored with those colors.

However, most of these neighborhood structures are limited in covering the solution space. Recently, Januario and Urrutia (2016) introduced a new neighborhood structure called **Teams and Rounds Swap (TARS)** that increases the connectivity of the solution space. A neighborhood is connected whenever there exists a path in the transition graph that connects any two candidate solutions. TARS combines the characteristics of PTS and PRS in order to generate neighbor schedules that cannot be obtained using either of them. As in PTS, TARS considers vertices and paths between them, but in TARS, a single path of length larger than two vertices is constructed.

A move in TARS consists of two phases:

- **Selection Phase (Decomposition of the graph):** From a given schedule ( $S$ ), two distinct colors ( $c$ ) and ( $d$ ) and one vertex ( $v$ ) are determined as move parameters. The selection phase determines a set of subgraphs of  $S$  divided into three classes  $\eta^1$  and  $\eta_p^t$ , for  $t \in \{2,3\}$ , and  $1 \leq p \leq p_{max}$ , where  $p_{max} \leq n/2 - 1$ , and it depends on the maximum length of the *cd-path*. The *cd-path* is a path of length  $2p$  that is alternatively colored with  $c$  and  $d$  colors. The superscript  $t$  is used to identify the class of the subgraph which is used to determine the changes in the color assignment of  $S$  during the change phase. See Figure 2.5.
- **Change Phase (Recombination):** The color assignment of each subgraph identified in the previous phase is modified according to its class. The change phase occurs immediately after the selection phase. It returns a neighbor solution with the best cost function for the problem under consideration among those obtained through modifications introduced to the color assignment of the subgraphs selected in the selection phase.

In Figure 2.5, the selection phase starts with creating the *cd-path*. The *cd-path* starts from  $v_1$  ends with  $v_2$ , and alternatively colored with  $c$  and  $d$  colors, starts with  $c$ , and



**Figure 2.5** The main two phases of TARS approach  
Source: Januario and Urrutia (2016)

ends with  $d$ . We also set the vertex  $v$  in the middle of the  $cd$ -path. After creating the  $cd$ -path, we start adding other vertices  $w_j$ . We start with  $w_1$ . If  $w_1 = w_{end}$ , then the subgraph  $\eta^1$  is complete and ready to introduce the change phase. Otherwise, we complete adding  $w_j$  until we reach  $w_{end}$ . After adding all vertices, we obtain the subgraph  $\eta_p^2$ . In  $\eta_p^2$ , we color the edges adjacent to  $v_1$  and  $v_2$  with  $c_j$  and  $c_{j-1}$ , respectively. After adding all  $w_j$ , we remove the  $cd$ -path and obtain the subgraph  $\eta_p^3$ . After this last step, both  $\eta_p^2$  and  $\eta_p^3$  are ready to be processed by the change phase.

In the change phase,  $c$  and  $d$  colors of  $\eta^1$  are exchanged along the edges in order to obtain a subgraph equivalent to the one in Figure 2.5. Similarly, in  $\eta_p^2$ , colors are exchanged along the  $cd$ -path starting from  $v_1$  and ending at  $v_2$ . We also exchange the colors adjacent to  $(w_j, v_1)$  and  $(w_j, v_2)$ . In  $\eta_p^3$ , colors exchanging starts from  $w_{end}$  ends with  $w_1$ . Next, in  $\eta_p^3$ , we rotate  $fan(v_1)$  and  $fan(v_2)$  by shifting  $fan(v_1)$  backward and shifting  $fan(v_2)$  forward, where  $fan(v_1)$  and  $fan(v_2)$  are the edges adjacent to  $v_1$  and  $v_2$ , respectively. See Figure 2.5.

In order to perform a move in the TARS neighborhood structure, every subgraph  $\eta^1$ ,  $\eta_p^2$ , and  $\eta_p^3$  obtained in the selection phase is used as an input in the change phase. Note that all neighbor solutions created in the change phase are evaluated, and the one with the best cost function value of the associated problem is returned.

In brief, TARS approach involves generating new schedules from a given schedule ( $S$ ). These new schedules are generated in an iterated manner. At each iteration, TARS requires a team ( $v$ ) and two distinct rounds ( $c$ ) and ( $d$ ). For each possible combination of  $v, c$ , and  $d$ , new schedules are generated. After applying the TARS operations for all  $(v, c, d)$  combinations, it is possible to further reduce the objective function value because TARS operation can visit new neighborhood structures with a higher level of connectivity in the solution space. For further details about TARS, you can refer to Januario and Urrutia (2016).

### 3. REST DIFFERENCE PROBLEM

#### 3.1 Problem Definition

Suppose there is an SRRT with  $n$  (even) teams. Such a tournament consists of  $n - 1$  rounds with  $n/2$  games in each round. These games are played in  $p$  consecutive matchdays. Let  $\gamma_d$  denote the number of games in matchday  $d \in \{1, 2, \dots, p\}$ . The rest difference is defined as the difference between rest periods the opposing teams had before their game against each other. The rest difference problem  $RDP(n, p | \gamma_1, \dots, \gamma_p)$  constructs a timetable that determines both the round and matchday of each game such that the total rest difference  $RD$  throughout the tournament is minimized. In this study, we occasionally use the notation  $RDP(n, p)$  to denote the problem instances with  $n$  teams,  $p$  matchdays and any number of games in the matchdays.

#### 3.2 Mixed-integer Programming (MIP-RDP) Model

In this section, we provide a mixed-integer programming (MIP-RDP) model that formulates the  $RDP(n, p | \gamma_1, \dots, \gamma_p)$ . The MIP-RDP model concurrently decides in which round and matchday each game needs to be scheduled such that the total rest difference is minimized. Below, the description of sets, parameters, and decision variables are provided, followed by the formulation.

##### 3.2.1 Sets

$T$  : Teams, indexed by  $i, j \in \{1, \dots, n\}$ .

$G$  : Games, indexed by  $g, g', g'' \in \{1, \dots, n \cdot (n - 1)/2\}$ .

$R$  : Rounds, indexed by  $r \in \{1, \dots, n - 1\}$ .

$D$  : Matchdays, indexed by  $d \in \{1, \dots, p\}$ .

### 3.2.2 Parameters

$play_{g,i}$  : 1 if team  $i$  plays in game  $g$ ; 0 otherwise.

$\gamma_d$  : Number of games to be played in matchday  $d$  of each round.

$M$  : A sufficiently large number. It should be at least equal to the maximum possible difference in rest periods between two teams (i.e.  $M \geq p - 1$ ).

### 3.2.3 Decision variables

$z$  : Total rest difference.

$x_{g,r,d}$  : Game assignment variable. 1 if game  $g$  is assigned to matchday  $d$  of round  $r$ ; 0 otherwise.

$y_{g,r}$  : Round indicator variable. 1 if game  $g$  is played in round  $r$ ; 0 otherwise.

$f_g$  : Rest difference variable. The number of days the first team in game  $g$  rested less than its opponent.

$s_g$  : Rest difference variable. The number of days the second team in game  $g$  rested less than its opponent.

### 3.2.4 Mathematical model for the RDP (MIP-RDP)

$$\min \quad z = \sum_{g \in G} (f_g + s_g) \quad (3.1)$$

subject to:

$$\sum_{r \in R} \sum_{d \in D} x_{g,r,d} = 1 \quad \forall g \in G \quad (3.2)$$

$$\sum_{g \in G: \text{play}_{g,i}=1} \sum_{d \in D} x_{g,r,d} = 1 \quad \forall i \in T, \forall r \in R \quad (3.3)$$

$$\sum_{g \in G} x_{g,r,d} = \gamma_d \quad \forall r \in R, \forall d \in D \quad (3.4)$$

$$\sum_{d \in D} x_{g,r,d} = y_{g,r} \quad \forall g \in G, \forall r \in R \quad (3.5)$$

$$\sum_{d \in D} d \cdot x_{g',r-1,d} - \sum_{d \in D} d \cdot x_{g'',r-1,d} + s_g - f_g \leq M \cdot (1 - y_{g,r}) \quad \forall \{i, j \in T, \\ g, g', g'' \in G : \text{play}_{g,i} = \text{play}_{g,j} = \text{play}_{g',i} = \text{play}_{g'',j} = 1\}, \forall r \in R \setminus \{1\} \quad (3.6)$$

$$x_{g,r,d} \in \{0, 1\} \quad \forall g \in G, \forall r \in R, \forall d \in D \quad (3.7)$$

$$y_{g,r} \in \{0, 1\} \quad \forall g \in G, \forall r \in R \quad (3.8)$$

$$f_{g,r}, s_{g,r} \geq 0 \quad \forall g \in G, \forall r \in R \quad (3.9)$$

The objective function minimizes the  $RD$  in the schedule. Constraints 3.2 declare that each game must be assigned to a matchday of a round. Constraints 3.3 ensure that each team plays exactly once in each round. Since the number of games in each matchday is determined a priori, Constraints 3.4 set the number of games to be played in each matchday. Constraints 3.5 determine in which round a game was set to be played by the model. If game  $g$  (between teams  $i$  and  $j$ ) is assigned to round  $r$ , Constraints 3.6 identify the matchdays of team  $i$  and team  $j$  in the previous



round  $r - 1$  and find the time difference of these matchdays. This difference is equal to the rest difference between  $i$  and  $j$  in the corresponding game  $g$ . Constraints 3.7 through Constraints 3.9 give the types of decision variables.

### 3.3 Theoretical Results for the Problem with One-game Matchdays

In this section, we give some theoretical results regarding tournaments with one-game matchdays (i.e., matchdays  $d$  where  $\gamma_d = 1$ ) by first constructing a lower bound ( $LB$ ) for the total rest difference  $RD$  in such tournaments. Then, the  $LB$  is used to show the optimality of certain schedules derived from other optimal schedules.

Let  $p_1, p_1 \geq 0$ , denote the number of matchdays with only one game. As an illustrative example, let us consider the  $RDP(8, 3|2, 1, 1)$  presented in Table 3.1 where  $p_1 = 2$ . In the schedule, day 2 and day 3 are the “one-game” matchdays as each includes only one game throughout the tournament. Consider game (1-4) played in a one-game matchday (day 3) of round 3. Both teams 1 and 4 must play with different opponents in round 4. There will be rest differences in both of these games (games (2-4) and (1-3)) since teams 2 and 4 (and teams 1 and 3) have played in different days of round 3. Thus, when we have one-game matchday(s) (i.e., when  $p_1 > 0$ ), we can not avoid the rest difference (i.e.,  $RD > 0$ ).

**Table 3.1** A schedule for the  $RDP(8, 3|2, 1, 1)$

Round	Day 1	Day 2	Day 3
1	3-6 4-5	2-8	1-7
2	2-7 3-5	1-8	4-6
3	5-7 6-8	2-3	1-4
4	2-4 6-7	1-3	5-8
5	3-8 4-7	2-6	1-5
6	1-6 2-5	7-8	3-4
7	3-7 4-8	1-2	5-6

**Theorem 1.** *The rest difference in each round is at least  $p_1 + \text{mod}(p_1, 2)$ . Therefore,  $(n - 2) \cdot [p_1 + \text{mod}(p_1, 2)]$  is a lower bound for the  $RD$  value of an  $SRRT$ .*

*Proof.* Let us first consider the case where  $p_1$  is even. When  $p_1 = 0$ , we have a trivial lower bound of 0. Assume that  $p_1 > 0$ . Since the teams in a one-game matchday (say day  $d$ ) of a round (say round  $r$ ) must play against two other teams (say  $t$  and  $t'$ ) in two other games (say in games  $g$  and  $g'$ ) of the next round, the occurrence of a rest difference is unavoidable in both  $g$  and  $g'$ . The magnitude of the rest difference for  $g$  and  $g'$  will be equal to 1, which is the smallest possible value, if  $t$  and  $t'$  played against each other in another one-game matchday (say day  $d^*$ ) of round  $r$  and if  $d$  and  $d^*$  are consecutive matchdays. Thus, each pair of one-game matchdays in each round causes an increase by at least 2 in the  $RD$  value.  $p_1/2$  pairs of one-game matchdays in each round will then lead to an increase in the  $RD$  value by at least  $p_1$ . As the rest difference can occur in all the rounds except the first round, the total rest difference in the tournament has to be at least  $(n - 2) \cdot p_1$  when  $p_1$  is even.

Now let us consider the case where  $p_1$  is odd.  $(p_1 - 1)/2$  pairs of one-game matchdays lead to an increase in the  $RD$  value by at least  $p_1 - 1$ . The remaining two teams in the one-game matchdays either play each other (say in game  $g^*$ ) or play against two other teams (say  $t$  and  $t'$ ) in two other games (say in games  $g$  and  $g'$ ) in the next round. Since these remaining teams can not be paired with teams of consecutive one-game matchdays,  $g^*$  must have a rest difference of at least 2, or both  $g$  and  $g'$  must have a rest difference of at least 1. Thus,  $p_1$  one-game matchdays in each round lead to an increase in the  $RD$  value by at least  $p_1 - 1 + 2 = p_1 + 1$ . As the rest difference can occur in all the rounds except the first round, the total rest difference in the tournament has to be at least  $(n - 2) \cdot (p_1 + 1)$  when  $p_1$  is odd. As a result, when we have an  $RDP(n, p)$  with  $p_1$  one-game matchdays, the lower bound of the total rest difference is  $LB = (n - 2) \cdot [p_1 + \text{mod}(p_1, 2)]$ .  $\square$

The  $RDP(8, 3|2, 1, 1)$  given in Table 3.1 is an example where  $p_1$  is even. Table 3.2 provides a schedule for the  $RDP(10, 4|2, 1, 1, 1)$  where  $p_1$  is odd. The schedules given in both tables have achieved an  $RD$  value equal to the lower bound presented in Theorem 1.

**Table 3.2** A schedule for the  $RDP(10, 4|2, 1, 1, 1)$ 

Round	Day 1	Day 2	Day 3	Day 4	
1	2-7	6-8	3-10	1-9	4-5
2	1-5	7-8	2-6	4-10	3-9
3	1-8	3-4	2-10	6-9	5-7
4	2-9	4-8	5-6	7-10	1-3
5	2-4	8-9	3-5	6-7	1-10
6	1-7	4-9	2-8	3-6	5-10
7	2-5	3-8	1-4	7-9	6-10
8	9-10	5-8	1-6	2-3	4-7
9	1-2	8-10	3-7	4-6	5-9

**Special Case:** In asynchronous round-robin tournaments, all matchdays in the tournament schedule are one-game matchdays. Examples of this type of tournaments are the  $RDP(10, 5|1, 1, 1, 1, 1)$ ,  $RDP(12, 6|1, 1, 1, 1, 1, 1)$ , and  $RDP(16, 8|1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$ . In these tournaments, since  $p_1 = n/2$ , the lower bound is  $LB = (n - 2) \cdot \lceil n/2 + \text{mod}(n/2, 2) \rceil$ .

**Theorem 2.** Assume that we have an optimal schedule for an  $RDP(n, p)$  with  $RD = LB$ . Further assume  $p_1$  is even and there exists a matchday  $d$  with  $\gamma_d > 2$  in the  $RDP(n, p)$ . If matchday  $d$  is split into two consecutive days with  $\gamma_d - 1$  and 1 games without changing the order of games, this new schedule remains optimal for the  $RDP(n, p + 1)$ .

*Proof.* After splitting matchday  $d$  into two consecutive matchdays, assume matchdays  $\bar{d}$  and  $d'$  with  $\gamma_{\bar{d}} - 1$  and 1 games, respectively, are generated with new  $RD$  value  $z'$ . Suppose that there is a game  $(i', j')$  placed at one-game matchday  $d'$  in round  $r - 1$ , and there exist games  $(i', j)$  and  $(i, j')$  in the next round  $r$ . Thus, teams  $i$  and  $j$  played on a day other than  $d'$  in round  $r - 1$  before the split, and games  $(i', j)$  and  $(i, j')$  had some rest difference that was optimal (zero or otherwise). The only new rest differences that can occur are due to the newly generated one-game day  $d'$ . Since both  $i'$  and  $j'$  playing in  $d'$  of round  $r - 1$  will have to play teams from other days in round  $r$ , there can be new rest differences arising from the games  $(i', j)$  and  $(i, j')$ . This will increase the rest difference of games  $(i', j)$  and  $(i, j')$  by at most 1, resulting in an increase of rest difference by

at most 2 in the next round  $r$ . Thus, the new optimal solution  $z'$  can be at most  $LB+2\cdot(n-2)$ . Since splitting matchday  $d$  into two consecutive matchdays with  $\gamma_d-1$  and 1 games increases  $p_1$  by 1 and  $p_1+1$  is odd, the new lower bound is calculated as  $LB' = (n-2)\cdot[p_1+1+\text{mod}(p_1+1, 2)] = (n-2)\cdot[p_1+2] = LB+2\cdot(n-2)$ . Since the new schedule's  $RD$  value cannot be less than  $LB'$  (i.e.,  $z' \geq LB+2\cdot(n-2)$ ), and the additional rest differences due to the split is at most  $2\cdot(n-2)$  (i.e.,  $z' \leq LB+2\cdot(n-2)$ ), the new schedule should be optimal.  $\square$

For example, we can use Theorem 2 to obtain the optimal schedule for the  $RDP(16, 3|4, 3, 1)$  by splitting the second matchday in the optimal schedule of the  $RDP(16, 2|4, 4)$  into two days with 3 and 1 games.

**Theorem 3.** *Assume that we have an optimal schedule for an  $RDP(n, p)$  with  $RD = LB$ . Further assume there exists a matchday  $d$  with  $\gamma_d = 2$  games. If matchday  $d$  is split into two consecutive one-game days without changing the order of games, this new schedule remains optimal for the  $RDP(n, p+1)$ .*

*Proof.* When the optimal  $RD$  before the split was zero, all teams in the matches of round  $r$  ( $r > 1$ ) must have played on the same day in round  $r-1$ . This means that all teams which play on day  $d$  before the split also played among each other throughout the tournament. Thus, when day  $d$  is split, the two games among these teams will generate a rest difference of 2 per round, and the  $RD$  will increase by  $2\cdot(n-2)$ . Since this is equal to the  $LB$  for a schedule with two one-game matchdays ( $p_1 = 2$ ) per Theorem 1, the schedule is optimal. In the case when  $p_1 > 0$  and the optimal  $RD$  is positive (i.e. not zero and equal to the  $LB$ ), then all the teams playing in days with more than one-game in them must again have played on the same day in round  $r-1$  since all the existing rest differences must be due to the one-game matchdays.

When day  $d$  with two games is split into consecutive one-game matchdays, games in those days will generate an additional rest difference of  $2\cdot(n-2)$  bringing the

$RD$  value to  $LB + 2 \cdot (n - 2)$ . This value is the lower bound for the new schedule with  $p_1 + 2$  one-game matchdays, and therefore, is also optimal.  $\square$

In Chapter 5, we report several results where Theorem 3 is used to obtain the optimal schedules for relevant instances of  $n = 8, 10, 12$ , and  $16$ . For example, the optimal schedule for the  $RDP(10, 4|2, 1, 1, 1)$  can be obtained by splitting the second matchday in the optimal schedule of the  $RDP(10, 3|2, 2, 1)$  into two consecutive one-game days.



## 4. SOLUTION METHODOLOGIES

In this chapter, we discuss the methodologies which are used for the solution of the *RDP*. We provide two methods for solving the rest difference problem. First, a polynomial-time exact method that finds the optimal schedules for some special instances of the *RDP* is described. Second, we present a matheuristic algorithm that solves the general rest difference problem.

### 4.1 A Polynomial-time Exact Method for Some Special Cases of the Problem

In this section, we propose a polynomial-time exact method which can be applied to the  $RDP(n, p | \gamma_1, \dots, \gamma_p)$  for  $n = 2^k$  ( $n \geq 8$ ) where  $k \in \mathbb{Z}^+$ , and the number of games in each period is divisible by 2, i.e.  $\text{mod}(\gamma_d, 2) = 0 \forall d \in D$ . The schedule generated by this method guarantees that the total rest difference in the whole tournament equals zero. Schedules generated by this algorithm for  $n = 2^k$  ( $n \geq 8$ ) where  $k \in \mathbb{Z}^+$  also provide optimal schedules for cases with one-game matchdays where the number of one-game matchdays are even due to Theorem 2 and Theorem 3.

The proposed method will be referred as Algorithm 1, and is conducted in three steps. In the first step, we construct groupings of four teams distributed over  $n/4$  days, which contain all possible combinations of four teams necessary to have an SRRT where each team has to play with each other team. Consecutive groupings are formed by swapping two teams from two different days with each other. Later, we will show that the rest differences that occur after such a swap operation can be eliminated. Second, we let the teams of each grouping play each other using the circle method to generate the games of the next rounds before moving on to

generating the games for the next set of groupings. In the third step, we eliminate all rest differences by applying necessary game swaps between days of certain rounds.

---

**Algorithm 1** Pseudocode of the polynomial-time exact method

---

```

1: function SWAP(Set  $T$  of team indices, List groupingBlockInfo)
2:   if cardinality( $T$ ) == 4 then
3:     return  $T$ , groupingBlockInfo
4:   else
5:     Generate team sets of size  $n/2$  as illustrated in Table 4.8:
6:      $T_1 = \{T[1], \dots, T[n/2]\}$ 
7:      $T_2 = \{T[n/2 + 1], \dots, T[n]\}$ 
8:      $T_3 = \{T[1], \dots, T[n/4], T[n/2 + 1], \dots, T[3n/4]\}$ 
9:      $T_4 = \{T[n/4 + 1], \dots, T[n/2], T[3n/4 + 1], \dots, T[n]\}$ 
10:     $T_5 = \{T[1], \dots, T[n/4], T[3n/4 + 1], \dots, T[n]\}$ 
11:     $T_6 = \{T[n/4 + 1], \dots, T[n/2], T[n/2 + 1], \dots, T[3n/4]\}$ 
12:   end if
13:   for  $block \in \{1, \dots, 6\}$  do
14:     Append  $block$  to groupingBlockInfo
15:     Call SWAP( $T_{block}$ , groupingBlockInfo)
16:   end for
17: end function

18: function MAIN(null)
19:   Input the number of teams  $n = 2^k$  ( $n \geq 8$ ) where  $k \in \mathbb{Z}^+$ 
20:   Set  $T = \{1 \rightarrow n\}$  and  $p = n/4$ 
21:   Set groupingBlockInfo to an empty list
22:   Call SWAP( $T$ , groupingBlockInfo) and generate all groupings necessary for an SRRT
23:   Using groupingBlockInfo, place the team groupings to their correct round ( $r_a, r_b$ , etc.)
24:   Generate all the games within each team grouping (day) using the circle method, and build an SRRT
25:   Eliminate the rest differences in the tournament by swapping the proper games between pairs of groupings that have exchanged teams via swap operations in their respective rounds
26:   return 0
27: end function

```

---

#### 4.1.1 Step 1: Construct the groupings

We construct the groupings as follows. Starting from the first round, split the teams into groupings of four teams with consecutive teams in each one of the  $n/4$

groupings. Then assign the grouping (1,2,3,4) to the first day, the grouping (5,6,7,8) to the second day, etc. until we reach the  $n/4$ th day, which includes the grouping  $(n-3, n-2, n-1, n)$ . When the tournament schedule is constructed, we will let all games of these groupings to be played before moving on to the games of the second set of groupings (in this case in round 4). To generate the next set of groupings, take two different days from the current set of groupings, and swap two teams from each day with each other so that all pairs of teams will be grouped together with different pairs of teams compared to the previous set of groupings.

In the same way, the next set of groupings is generated by team swaps applied to the second set of groupings. Each time a new set of groupings is formed, there will be  $n/8$  such swap operations. Keep generating new groupings via team swaps between two different days until you exhaust all possible combinations adding up to  $n/2 - 1$  sets of groupings. Table 4.1 shows the grouping construction for the  $RDP(16, 4|2, 2, 2, 2)$  where the colored team pairs indicate the swap operations. Note that these groupings are just combinations of four teams and not the actual games played.

**Table 4.1** Team groupings for  $n = 16$  obtained via swap operations

Groupings	Day 1		Day 2		Day 3		Day 4	
$r_a$	1 2	3 4	5 6	7 8	9 10	11 12	13 14	15 16
$r_b$	1 2	5 6	3 4	7 8	9 10	13 14	11 12	15 16
$r_c$	1 2	7 8	3 4	5 6	9 10	15 16	11 12	13 14
$r_d$	1 2	9 10	3 4	11 12	5 6	13 14	7 8	15 16
$r_e$	1 2	11 12	3 4	9 10	5 6	15 16	7 8	13 14
$r_f$	1 2	13 14	3 4	15 16	5 6	9 10	7 8	11 12
$r_g$	1 2	15 16	3 4	13 14	5 6	11 12	7 8	9 10

#### 4.1.2 Step 2: Generate the games

All games necessary to have an SRRT are generated in this step. Using the circle method, we generate the games by letting the teams of the same grouping play against each other on the same day until they finish all possible games among each other. In the first round, in which teams have not faced each other before, one can generate  $\binom{4}{2} = 6$  games from each grouping. Considering two games per day,



the games of the first three rounds can be constructed. Similarly, in the remaining rounds, we let the teams of each grouping play each other on the same day until they finish all possible games among each other using the circle method. Note that two games of each grouping (games between the swapped teams) were played before. The remaining four games of each grouping can be played in two rounds (an even round and the next (odd) round). Applying the circle method, keep generating the games for each couple of rounds by letting the teams of each grouping finish all the games among each other. Table 4.2 shows the generated games for the first seven rounds of the  $RDP(16, 4|2, 2, 2, 2)$  where the colored cells are the games that need to be swapped to eliminate the rest differences.

**Table 4.2** Generating the games for the first seven rounds of the  $RDP(16, 4|2, 2, 2, 2)$

	Round	Day 1	Day 2	Day 3	Day 4
$r_a$	1	1-3 2-4	5-7 6-8	9-11 10-12	13-15 14-16
	2	1-4 3-2	5-8 7-6	9-12 11-10	13-16 15-14
	3	1-2 3-4	5-6 8-7	9-10 12-11	13-14 16-15
$r_b$	4	1-5 2-6	3-7 4-8	9-13 10-14	11-15 12-16
	5	1-6 2-5	3-8 4-7	9-14 10-13	11-16 12-15
$r_c$	6	1-7 2-8	3-5 4-6	9-15 10-16	11-13 12-14
	7	1-8 2-7	3-6 4-5	9-16 10-15	11-14 12-13

### 4.1.3 Step 3: Eliminate the rest differences

Due to the swap operations performed in constructing the groupings, teams that played on different days in the odd rounds (except round 1) will have to play each other in the even rounds (starting from round 4) which will result in rest differences. Observe that one can swap two games played in those odd rounds involving the swapped teams when the groupings were formed (colored games in Table 4.2) to get rid of these rest differences. For example, in Table 4.2, one can swap games between teams 3-4 and 5-6, and games between teams 12-11 and 13-14 in round  $r_b - 1$  to get rid of the rest difference in round  $r_b$ . The resulting  $RDP(16, 4|2, 2, 2, 2)$  schedule after eliminating all the rest differences is shown in Table 4.3.

**Table 4.3** An optimal schedule of the  $RDP(16, 4|2, 2, 2, 2)$ 

	Round	Day 1		Day 2		Day 3		Day 4	
$r_a$	1	1-3	2-4	5-7	6-8	9-11	10-12	13-15	14-16
	2	1-4	3-2	5-8	7-6	9-12	11-10	13-16	15-14
	3	1-2	5-6	3-4	8-7	9-10	13-14	12-11	16-15
$r_b$	4	1-5	2-6	3-7	4-8	9-13	10-14	11-15	12-16
	5	1-6	4-7	3-8	2-5	9-14	12-15	11-16	10-13
$r_c$	6	1-7	2-8	3-5	4-6	9-15	10-16	11-13	12-14
	7	1-8	9-16	3-6	11-14	2-7	10-15	4-5	12-13
$r_d$	8	1-9	2-10	3-11	4-12	5-13	6-14	7-15	8-16
	9	1-10	4-11	3-12	2-9	5-14	8-15	7-16	6-13
$r_e$	10	1-11	2-12	3-9	4-10	5-15	6-16	7-13	8-14
	11	1-12	8-13	3-10	6-15	5-16	4-9	7-14	2-11
$r_f$	12	1-13	2-14	3-15	4-16	5-9	6-10	7-11	8-12
	13	1-14	4-15	3-16	2-13	5-10	8-11	7-12	6-9
$r_g$	14	1-15	2-16	3-13	4-14	5-11	6-12	7-9	8-10
	15	1-16	2-15	3-14	4-13	5-12	6-11	7-10	8-9

In Algorithm 1, the computational step having the highest complexity is given in line 25. This step performs  $n/8$  swaps in  $n/2 - 2$  rounds. Therefore, the running time complexity of Algorithm 1 is  $O(n^2)$ , which means the algorithm is polynomial time.

**Theorem 4.** *The method given by Algorithm 1 results in a schedule with zero RD when  $n$  is a positive-integer power of 2 where  $n \geq 8$ , and when the number of games on each day is even.*

*Proof.* Let's say we have a tournament where there are only two games (with four teams) on each day. Let those teams playing on the same day finish all the possible games among each other. We now define certain swap operations that will help constructing a schedule with zero total rest difference. Consider two different days, and swap two teams from each day with each other so that they will face different teams in the next round, say  $r_b$ . Due to the swap operation, teams that played on different days in round  $r_b - 1$  will have to play each other in round  $r_b$ . Observe that one can swap two games played in round  $r_b - 1$  involving the swapped teams (colored grey in Table 4.4) to get rid of this rest difference (see Table 4.4). Notice that this operation will work any time when teams from two different days are swapped

between two different days in a round. In Table 4.4, the game between teams 3 and 4 should be swapped with the game between teams 5 and 6 to get rid of the rest difference that will otherwise occur in round  $r_b$ . In Table 4.4, it was assumed that the teams playing in round  $r_a$  have never faced each other before. Table 4.5 shows a situation where the teams playing in a round that were not swapped have already faced each other before a team swap operation. In Table 4.5, games between team 1 and team 2, team 3 and team 4, team 5 and team 6, and team 7 and team 8 were assumed to have been played. Still, with a game swap in round  $r_b - 1$  (among the greyed games) the rest difference in round  $r_b$  can be eliminated.

**Table 4.4** Eliminating rest differences of a swap operation

Round	Day 1		Day 2	
$r_a$	1-3	2-4	5-7	6-8
	1-4	3-2	5-8	7-6
$r_b - 1$	1-2	3-4	5-6	8-7
$r_b$	1-5	2-6	3-7	4-8

**Table 4.5** Eliminating rest differences of a swap operation

Round	Day 1		Day 2	
$r_a$	1-3	2-4	5-7	6-8
$r_b - 1$	1-4	3-2	5-8	7-6
$r_b$	1-5	2-6	3-7	4-8

Assume that  $n = 8$  with two days where there are only two games on each day. When  $n$  is less than 8, we cannot have a tournament with more than one day where all days have two games each. Let's build our tournament so that the whole tournament consists of intermediate rounds (not consecutive) where the groupings (teams that will play among each other) in those rounds have been determined by team swap operations between two days only (such as in rounds  $r_a$  and  $r_b$  in Tables 4.4 and 4.5). The resulting groupings (not the actual games to be played) with 8 teams is shown in Table 4.6. Teams playing on the same day will actually finish all the games among each other in a fashion as explained in Tables 4.4 and 4.5. Thus, the whole tournament can be built performing team swap operations between two different days, and we already know that if that is possible then we have a way of getting rid of the rest differences. In round  $r_b$ , teams 3 and 4 are swapped with teams 5 and 6;

and in round  $r_c$ , teams 5 and 6 are swapped with teams 7 and 8. Those three (if the number of teams is equal to  $n$  then the number of different groupings is  $n/2 - 1$ ) sets of groupings constitute all the possible team combinations necessary to have a round-robin tournament with  $n = 8$  teams where each team should play all other teams once. The resulting schedule shown in Table 4.7 demonstrates that there will be no rest differences using the prescribed swap operations when  $n = 8$  after all games are played.

**Table 4.6** Team groupings for  $n = 8$

Groupings	Day 1	Day 2
$r_a$	1 2 3 4	5 6 7 8
$r_b$	1 2 5 6	3 4 7 8
$r_c$	1 2 7 8	3 4 5 6

**Table 4.7** Finished schedule for the  $RDP(8, 2|2, 2)$

Round	Day 1	Day 2
$r_a$	1-3 2-4	5-7 6-8
	1-4 3-2	5-8 7-6
	1-2 5-6	3-4 8-7
$r_b$	1-5 2-6	3-7 4-8
	1-6 4-7	3-8 2-5
$r_c$	1-7 2-8	3-5 4-6
	1-8 2-7	3-6 4-5

Let's now double  $n$  to 16. We are interested in obtaining a round-robin tournament schedule with  $n/4$  days in each round with two games on each day, i.e. there will be four teams playing on each day. Teams are numbered from 1 to  $n$ . We can build the team groupings so that initially teams from 1 to  $n/2$ , and  $n/2 + 1$  to  $n$  are grouped together forming two separate groups of size  $n/2$ . Let these sets of  $n/2$  teams finish the games among each other using swap operations. Then, let teams 1 to  $n/4$  match with teams  $n/2 + 1$  to  $3n/4$ , and teams  $n/4 + 1$  to  $n/2$  match with teams  $3n/4 + 1$  to  $n$  forming two different sets of  $n/2$  teams. This grouping can be achieved via swap operations. In the round just before the matching, teams  $n/2 + 1$  and  $n/2 + 2$  are grouped together either with teams  $3n/4 + 1$  and  $3n/4 + 2$ , or with  $n - 1$  and  $n$ . On the other hand,  $n/2 + 3$  and  $3n/4$  are grouped with the other pair among teams from  $3n/4 + 1$  through  $n$  teams  $n/2 + 1$  and  $n/2 + 2$  are not grouped with. In the other half's grouping among teams from 1 to  $n/2$ , teams 1 and 2 are grouped either

with  $n/4 + 1$  and  $n/4 + 2$ , or with  $n/2 - 1$  and  $n/2$ ; also teams 3 and 4 are grouped together with the pair from  $n/4 + 1$  to  $n/2$  that teams 1 and 2 are not grouped with. Thus, one can swap  $n/4 + 1$  and  $n/4 + 2$  with either  $n/2 + 1$  and  $n/2 + 2$ , or with  $n/2 + 3$  and  $3n/4$ ; and also  $n/2 - 1$  and  $n/2$  with the the other pair from  $n/2 + 1$  to  $3n/4$ . After the matching, let the matched teams finish their remaining games among each other (the way these teams are grouped is again determined via swap operations).

Finally, let teams 1 to  $n/4$  match with teams  $3n/4 + 1$  to  $n$ , and teams  $n/4 + 1$  to  $n/2$  match with teams  $n/2 + 1$  to  $3n/4$ . This matching can also be achieved via swap operations in a manner similar -albeit with different team indices- to the swap operations done when teams 1 to  $n/4$  were matched with teams  $n/2 + 1$  to  $3n/4$ , and teams  $n/4 + 1$  to  $n/2$  were matched with teams  $3n/4 + 1$  to  $n$ . Once these groups, each of size  $n/2$ , also finish their remaining games among each other (again via swap operations) the tournament is complete because all teams have already played against every other team. Table 4.1 shows the groupings when  $n = 16$  whereas Table 4.3 shows the resulting optimal schedule when finished.

In a similar fashion, when  $n \geq 32$ , groupings of teams can be obtained by creating different sets of  $n/2$  teams in three stages as shown in Table 4.8. The first column in the table indicates the number of groupings that will result from the matchings in the table. In the second and third stages there will be less groupings because some teams were already grouped together (thus played against each other) before. Since we already know how to form consecutive groups via swap operations for  $n/2$  teams, and how then to schedule their games with zero rest difference, we can also find a schedule with no rest differences whenever  $n$  is further doubled.

Above discussion shows that tournaments of zero  $RD$  with two games on each day where the number of teams is a power of 2 can be generated. This also indicates that rest differences in those schedules can be eliminated whenever the number of games in all days is a multiple of 2 such as in the case of the  $RDP(16, 2|4, 4)$ ,

**Table 4.8** Building a schedule's team groupings

1	1	$n/4$	$n/4 + 1$	$n/2$	$n/2 + 1$	$3n/4$	$3n/4 + 1$	$n$
$n/4 - 1$	BLOCK 1 Groupings of $\{1, \dots, n/2\}$				BLOCK 2 Groupings of $\{n/2 + 1, \dots, n\}$			
$+n/8$	BLOCK 3 Groupings of $\{1, \dots, n/4, n/2 + 1, \dots, 3n/4\}$				BLOCK 4 Groupings of $\{n/4 + 1, \dots, n/2, 3n/4 + 1, \dots, n\}$			
$+n/8$	BLOCK 5 Groupings of $\{1, \dots, n/4, 3n/4 + 1, \dots, n\}$				BLOCK 6 Groupings of $\{n/4 + 1, \dots, n/2, n/2 + 1, \dots, 3n/4\}$			

$RDP(16, 2|6, 2)$ ,  $RDP(16, 3|4, 2, 2)$  and  $RDP(16, 3|2, 4, 2)$ . In tournaments with  $n$  teams where  $n$  is a power of 2 and where the number of games on each day is a multiple of 2, one could first build a schedule with a zero  $RD$  for the  $RDP(n, n/4)$  with two games on each day. Then, when days in that schedule are combined there will be no rest differences in the resulting schedule as well because the games which were part of the combined days had no rest differences before.

Thus, the algorithm will work for any instance of the  $RDP(n, p|\gamma_1, \dots, \gamma_p)$  where the team number can be expressed as a power 2, i.e.  $n = 2^k$  ( $n \geq 8$ ) where  $k \in \mathbb{Z}^+$ , and the number of games in each period is divisible by 2, i.e.  $\gamma_d \bmod 2 = 0 \forall d \in D$ .  $\square$

Using the procedure discussed here, we can find the optimal solution for the instances such as the  $RDP(16, 4)$  and  $RDP(32, 8)$ . With the help of Theorem 2 and Theorem 3, these solutions can easily be used to derive optimal schedules for the instances such as the  $RDP(16, 5 \cdots 8)$  and  $RDP(32, 8 \cdots 16)$ . As an example, the optimal schedule for the  $RDP(16, 6|2, 2, 1, 1, 1, 1)$  is provided in Table 4.9.

It can be noted that the order of games in this schedule is exactly the same with the order of games in the schedule of Table 4.3 for the  $RDP(16, 4|2, 2, 2, 2)$ . The schedule is obtained by splitting day 3 and day 4 into two days in Table 4.3.

**Table 4.9** An optimal schedule ( $RD = 56$ ) of the  $RDP(16, 6|2, 2, 1, 1, 1, 1)$ 

Round	Day 1		Day 2		Day 3	Day 4	Day 5	Day 6
1	1-3	2-4	5-7	6-8	9-11	10-12	13-15	14-16
2	1-4	3-2	5-8	7-6	9-12	11-10	13-16	14-15
3	1-2	5-6	3-4	8-7	9-10	13-14	12-11	15-16
4	1-5	2-6	3-7	4-8	9-13	10-14	11-15	12-16
5	1-6	4-7	3-8	2-5	9-14	12-15	11-16	10-13
6	1-7	2-8	3-5	4-6	9-15	10-16	11-13	12-14
7	1-8	9-16	3-6	11-14	2-7	10-15	4-5	12-13
8	1-9	2-10	3-11	4-12	5-13	6-14	7-15	8-16
9	1-10	4-11	3-12	2-9	5-14	8-15	7-16	6-13
10	1-11	2-12	3-9	4-10	5-15	6-16	7-13	8-14
11	1-12	8-13	3-10	6-15	5-16	4-9	7-14	2-11
12	1-13	2-14	3-15	4-16	5-9	6-10	7-11	8-12
13	1-14	4-15	3-16	2-13	5-10	8-11	7-12	6-9
14	1-15	2-16	3-13	4-14	5-11	6-12	7-9	8-10
15	1-16	2-15	3-14	4-13	5-12	6-11	7-10	8-9

## 4.2 An ILS-based Matheuristic Algorithm for the Problem

In this section, we propose a matheuristic algorithm that solves the  $RDP(n, p|\gamma_1, \dots, \gamma_p)$ . In our matheuristic algorithm, we utilize some local neighborhood search techniques to generate random *opponent schedules* in which the round of each game is decided but the matchday is not. To improve the rest difference value for an  $RDP(n, p|\gamma_1, \dots, \gamma_p)$  instance, we apply these local searches iteratively and find neighbor opponent schedules that can hopefully produce better rest difference values. For each generated opponent schedule, we optimally assign the games to the matchdays using a MIP model. Finally, the schedule with the best  $RD$  value is reported. In other words, the proposed matheuristic iteratively employ a MIP model to optimize a subproblem while the remainder of the problem is fixed. The ILS-based matheuristic is summarized in Algorithm 2. The algorithm relies on finding an initial schedule with a ‘good’  $RD$  (Step 1), and then further improving the  $RD$  value using a recently introduced neighborhood structure (Step 2).

---

**Algorithm 2** Pseudocode of the ILS-based matheuristic algorithm

---

```
1: function FINDSCHEDULE( $n, p, \gamma_1 \cdots \gamma_p, \lambda_{crm}, \lambda_{vm}, \lambda_{tars}$ )
2:   Generate a round-robin schedule  $S$  for  $n$  teams using the circle method
3:   for  $i \in \{1, \dots, \lambda_{crm}\}$  do
4:     Randomize the rounds of  $S$  to obtain opponent schedule  $S_i^{crm}$ 
5:     Find the minimum  $RD$  value ( $z_i^{crm}$ ) for  $S_i^{crm}$  using MIP-GOS
6:   end for
7:   for  $j \in \{1, \dots, \lambda_{vm}\}$  do
8:     Generate a random edge coloring for  $n$  teams using the Vizing
       method to obtain opponent schedule  $S_j^{vm}$ 
9:     Find the minimum  $RD$  value ( $z_j^{vm}$ ) for  $S_j^{vm}$  using MIP-GOS
10:  end for
11:  Select the opponent schedule  $S_{best}$  having  $z = \min_{\forall i, \forall j} \{z_i^{crm}, z_j^{vm}\}$ 
12:  for  $k \in \{1, \dots, \lambda_{tars}\}$  do
13:    Select an arbitrary combination of  $(v, c, d)$ 
14:    Apply the TARS neighborhood search on  $S_{best}$  using team  $v$  and
       rounds  $c$  and  $d$  to obtain opponent schedule  $S_k$ 
15:    Find the minimum  $RD$  value ( $z_k$ ) for  $S_k$  using MIP-GOS
16:  end for
17:  Select the schedule  $S^*$  having  $z^* = \min_{\forall k} \{z_k\}$ 
18:  return  $z^*, S^*$ 
19: end function
```

---

#### 4.2.1 Step 1: Finding an initial schedule

Algorithm 2 starts by generating an initial opponent schedule  $S$  for an SRRT with  $n$  teams using the circle method (line 2). In this initial opponent schedule, the games of each round are identified without assigning the games into matchdays. In the next step, the rounds of  $S$  are randomized  $\lambda_{crm}$  times to generate neighbor schedules  $S_i^{crm}$ . Since each opponent schedule is obtained by randomizing the rounds of  $S$  generated by circle method, we call this approach *circle-and-randomize*. Next, for each  $S_i^{crm}$ , the games are assigned into matchdays using a mixed-integer programming model (MIP-GOS) (lines 3-6). MIP-GOS is a modified version of MIP-RDP presented in Section 3.2.4 and solves the  $RDP(n, p | \gamma_1, \dots, \gamma_p)$  for a given opponent schedule (GOS) (i.e., a given round assignment of games). In MIP-GOS, the main decision variable is  $x_{g,d}$  which indicates whether game  $g$  is assigned to matchday  $d$ . The variables  $f_g$  and  $s_g$  are the rest difference variables and have the same description given in Section 3.2.3. The parameter  $play_{g,r,i}$  is equal to 1 if game  $g$  is assigned to round  $r$  and



team  $i$  plays in the game, 0 otherwise. The formulation of MIP-GOS is provided with equations 4.1-4.6.

$$\min \quad z = \sum_{g \in G} (f_g + s_g) \quad (4.1)$$

subject to:

$$\sum_{d \in D} x_{g,d} = 1 \quad \forall g \in G \quad (4.2)$$

$$\sum_{g \in G} x_{g,d} = \gamma_d \quad \forall d \in D \quad (4.3)$$

$$\sum_{d \in D} d \cdot x_{g',d} - \sum_{d \in D} d \cdot x_{g'',d} + f_g - s_g = 0 \quad \forall \{i, j \in T, g, g', g'' \in G \forall r \in R \setminus \{1\}\} \\ : \text{play}_{g,r,i} = \text{play}_{g,r,j} = \text{play}_{g',i,r-1} = \text{play}_{g'',j,r-1} = 1 \} \quad (4.4)$$

$$x_{g,d} \in \{0, 1\} \quad \forall g \in G, \forall d \in D \quad (4.5)$$

$$f_g, s_g \geq 0 \quad \forall g \in G \quad (4.6)$$

In the formulation, the objective function 4.1 minimizes the total rest difference for the given opponent schedule. Constraints 4.2 ensure every game is assigned to exactly one matchday in the schedule. Constraints 4.3 set the number of games to be played in each matchday. Constraints 4.4 define the value of variable  $f_g$  ( $s_g$ ) as the number of days the first (second) team in game  $g$  rested less than the second (first) team. Constraints 4.5 and 4.6 give the types of decision variables. Unlike MIP-RDP,

MIP-GOS does not decide which round each game should be assigned to. Therefore,  $x$  variable does not possess an  $r$  index, and  $y$  variable is not needed anymore. Moreover, due to the same reason, MIP-GOS does not require any constraints similar to Constraints 3.3, 3.5 and 3.8 of MIP-RDP. Since MIP-GOS is only concerned with the matchday assignment, it runs much faster than MIP-GOS in commercial solvers as it will be demonstrated later in Chapter 5.

Januario et al. (2016) show that it is not possible to reach all opponent schedules by only shuffling the rounds (i.e., circle-and-randomize approach) as the corresponding edge colorings are still isomorphic to each other. Moreover, they show that any arbitrary opponent schedule can be reached by using Vizing's method. Therefore, an iterative approach similar to the one described in lines 3-6 is also applied for a set of opponent schedules generated by the Vizing method. Algorithm 2 generates a total of  $\lambda_{vm}$  opponent schedules using the Vizing method, and solves MIP-GOS model for each one of them (lines 7-10). The best schedule  $S_{best}$  with the lowest  $RD$  value resulting from both methods is selected to be used in Step 2 (line 11).

#### **4.2.2 Step 2: Finding a schedule with an improved $RD$**

We apply TARS operations presented in Section 2.3.4 to the best initial schedule resulting from the first step (found by either the circle-and-randomize method or the Vizing method) to further reduce the  $RD$  value. Inside TARS operations, we use MIP-GOS to find the best neighbor schedule (lines 12-16). MIP-GOS assigns games into matchdays to minimize the total rest difference value of each schedule. Finally, TARS returns the schedule with the lowest total rest difference (lines 17-18).

## 5. EXPERIMENTAL RESULTS

In our experimental analysis, we consider  $RDP(n, p | \gamma_1, \dots, \gamma_p)$  instances that have even  $n$  teams ranging from 8 to 20, and  $p$  matchdays ranging from 2 to  $n/2$ . The reason for not exceeding 20 teams in our experiments is that the professional leagues usually comprises no more than 20 teams, as can be seen in Table 5.1, which shows the number of teams (as of the year 2021) in the top-tier football and basketball leagues of five European countries.

**Table 5.1** Team numbers in the top-tier sports leagues of five European countries

Country	Football League	Basketball League
England	20	13
Spain	20	18
Germany	18	18
Italy	20	17
France	20	16

For the sake of brevity, instead of considering all possible allocations of  $(\gamma_1, \dots, \gamma_p)$ , we assume that the games are allocated to  $p$  matchdays evenly. If an even allocation with equal number of games in each matchday is not possible, then the numbers of games in the matchdays are assumed to be in descending order with minimal deviation among matchdays (i.e.  $0 \leq \gamma_{d-1} - \gamma_d \leq 1 \forall d$ ). For example, in the  $RDP(12, 4)$ , the number of games in the matchdays would be  $(\gamma_1, \gamma_2, \gamma_3, \gamma_4) = (2, 2, 1, 1)$ . One can find all the instances considered in the first column of Table 5.2.

### 5.1 Experimental Results of MIP-RDP

All instances under consideration were run by MIP-RDP presented in Section 3.2. MIP Solutions were obtained with GAMS using either Gurobi 9.1 solver (Gurobi Optimization Inc., 2021) or CPLEX solver (IBM ILOG CPLEX, 2017). Note that

these instances were solved by both solvers, but only the best solutions were reported. All experimental runs were executed on an Intel Core i7-7600U CPU 2.9 GHz computer with 8GB of RAM. A time limit of 10 hours (36,000 seconds) was used. The best MIP solutions found within the time limit are listed in the second column of Table 5.2.

The gap percentages between the best MIP solutions and the LB are listed in the third column of Table 5.2. Note that the solver could not find a lower bound greater than zero for most instances in which there are no one-game matchdays. Therefore, the percentages of the gap in those instances are 100%. For the instances with one-game matchdays, i.e., where  $p_1 > 0$ , the best MIP solutions were compared with the LB presented in Section 3.3. The percentages of these instances are colored with grey in Table 5.2.

The values marked with an asterisk (\*) in Table 5.2 are guaranteed optimal values. In all these instances, the results found equal exactly the *LB* presented in Section 3.3. According to the table, MIP-RDP has found optimal solutions with zero rest difference in few instances particularly when  $p$  values equal 2 such as in the  $RDP(8, 2|2, 2)$ ,  $RDP(10, 2|3, 2)$ ,  $RDP(12, 2|3, 3)$ , and  $RDP(16, 2|4, 4)$ . On the other hand, in some instances with relatively large  $n$  and  $p$  such as the  $RDP(18, 8|2, 1, 1, 1, 1, 1, 1)$  and  $RDP(20, 6|2, 2, 2, 2, 1, 1)$ , MIP-RDP running on commercial solvers could not even find a feasible solution within the time limit.

## 5.2 Experimental Results of the Matheuristic

The algorithm of the matheuristic (Algorithm 2) was coded using Python. Each one of the  $\lambda_{crm}$  ( $\lambda_{vm}, \lambda_{tars}$ ) opponent schedules is solved by the MIP-GOS model. Gurobi 9.1 solver was employed in running MIP-GOS. MIP-GOS finds the optimal result in less than 10 seconds for each given opponent schedule of the  $RDP(n, p|\gamma_1, \dots, \gamma_p)$ . When implementing Algorithm 2, we generate  $\lambda_{crm} = 1,000$  random permutations of the rounds of the initial opponent schedule that is constructed using the circle

**Table 5.2** Results of MIP-RDP

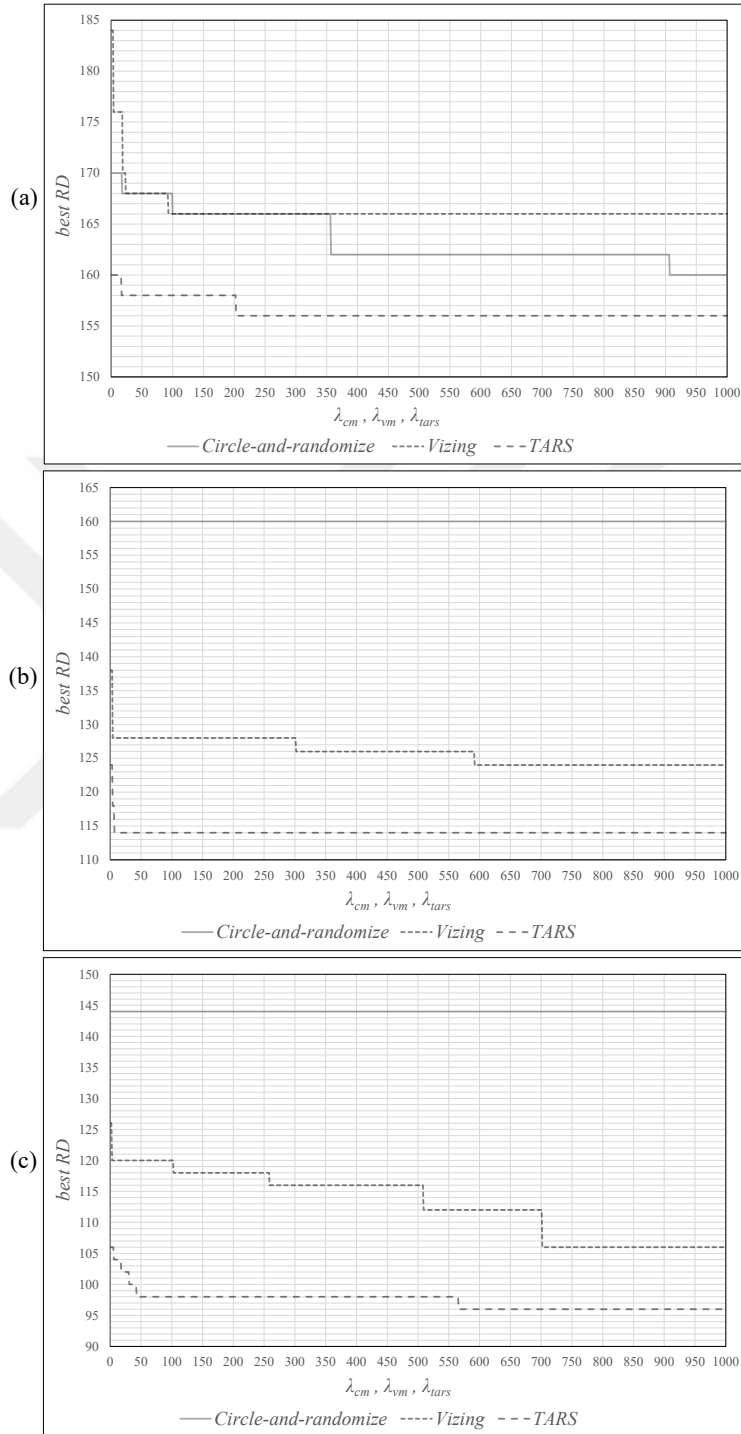
$RDP(n, p   \gamma_1, \dots, \gamma_p)$	Best MIP solution (sec.)	%Gap
(8, 2 2, 2)	0* (0.391)	0
(8, 3 2, 1, 1)	12* (36000)	0
(8, 4 1, 1, 1, 1)	24* (36000)	0
(10, 2 3, 2)	0* (238)	0
(10, 3 2, 2, 1)	20 (36000)	20
(10, 4 2, 1, 1, 1)	32* (36000)	0
(10, 5 1, 1, 1, 1, 1)	48* (36000)	0
(12, 2 3, 3)	0* (9183)	0
(12, 3 2, 2, 2)	6 (36000)	100
(12, 4 2, 2, 1, 1)	32 (36000)	38
(12, 5 2, 1, 1, 1, 1)	56 (36000)	29
(12, 6 1, 1, 1, 1, 1, 1)	80 (36000)	25
(14, 2 4, 3)	4 (36000)	100
(14, 3 3, 2, 2)	18 (36000)	100
(14, 4 2, 2, 2, 1)	52 (36000)	54
(14, 5 2, 2, 1, 1, 1)	80 (36000)	40
(14, 6 2, 1, 1, 1, 1, 1)	90 (36000)	20
(14, 7 1, 1, 1, 1, 1, 1, 1)	128 (36000)	25
(16, 2 4, 4)	0* (66.75)	0
(16, 3 3, 3, 2)	28 (36000)	100
(16, 4 2, 2, 2, 2)	44 (36000)	100
(16, 5 2, 2, 2, 1, 1)	84 (36000)	67
(16, 6 2, 2, 1, 1, 1, 1)	128 (36000)	56
(16, 7 2, 1, 1, 1, 1, 1, 1)	152 (36000)	45
(16, 8 1, 1, 1, 1, 1, 1, 1, 1)	180 (36000)	38
(18, 2 5, 4)	16 (36000)	100
(18, 3 3, 3, 3)	36 (36000)	100
(18, 4 3, 2, 2, 2)	54 (36000)	100
(18, 5 2, 2, 2, 2, 1)	118 (36000)	73
(18, 6 2, 2, 2, 1, 1, 1)	176 (36000)	64
(18, 7 2, 2, 1, 1, 1, 1, 1)	172 (36000)	44
(18, 8 2, 1, 1, 1, 1, 1, 1, 1)	- (36000)	-
(18, 9 1, 1, 1, 1, 1, 1, 1, 1, 1)	- (36000)	-
(20, 2 5, 5)	14 (36000)	100
(20, 3 4, 3, 3)	70 (36000)	100
(20, 4 3, 3, 2, 2)	108 (36000)	100
(20, 5 2, 2, 2, 2, 2)	166 (36000)	100
(20, 6 2, 2, 2, 2, 1, 1)	168 (36000)	79
(20, 7 2, 2, 2, 1, 1, 1, 1)	260 (36000)	72
(20, 8 2, 2, 1, 1, 1, 1, 1, 1)	- (36000)	-
(20, 9 2, 1, 1, 1, 1, 1, 1, 1, 1)	- (36000)	-
(20, 10 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)	- (36000)	-

method. We also generate  $\lambda_{vm} = 1,000$  arbitrary edge colorings using the Vizing method.

As noted in Algorithm 2, instead of executing TARS operations for all possible combinations of  $(v, c, d)$  where  $v \in T$  and  $c, d \in R$ , we prefer to run them for  $\lambda_{tars} = 1,000$  randomized combinations. We have selected three arbitrary instances of the rest difference problem to demonstrate why these values for  $\lambda$  parameters are preferred. In Figure 5.1, by selecting  $\lambda_{crm}$ ,  $\lambda_{vm}$  and  $\lambda_{tars}$  values up to 1,000, the best  $RD$  value is plotted after each iteration for the instances (a)  $(16, 8|1, 1, 1, 1, 1, 1, 1, 1)$ , (b)  $(18, 6|2, 2, 2, 1, 1, 1)$  and (c)  $(20, 5|2, 2, 2, 2, 2)$ . Indeed, at first, we ran all three instances by letting  $\lambda_{crm} = 10,000$  and  $\lambda_{vm} = 10,000$  and did not observe any ‘best  $RD$ ’ value better than the ones reported in the figure. Therefore, we have concluded that the selection of  $\lambda_{crm}$  and  $\lambda_{vm}$  as 1000 would be adequate. It can be noted that the initial best  $RD$  value at the start of the TARS operations is the lowest value achieved by either the circle-and-randomize or the Vizing method.

An interesting outcome in the figure is that the best  $RD$  value stays the same for all random permutations of the rounds in the circle-and-randomize method when  $n = 18$  and  $n = 20$ . On the other hand, an improvement can be achieved in the optimal  $RD$  value (from 170 to 160) by randomizing the rounds when  $n = 16$ . As a matter of fact, the circle-and-randomize method has resulted in an improvement in the optimal  $RD$  value for only the cases  $n = 10$  and  $n = 16$ . Randomizing the rounds of an opponent schedule generated by the circle method does not change the optimal  $RD$  value at all when  $n = 8, 12, 14, 18$  and  $20$ . TARS operations attain the best  $RD$  values of 156, 114, 96 for  $(16, 8|1, 1, 1, 1, 1, 1, 1, 1)$ ,  $(18, 6|2, 2, 2, 1, 1, 1)$  and  $(20, 5|2, 2, 2, 2, 2)$  in the 203rd, 7th and 566th iterations, respectively.

Moreover, we have executed TARS operations in these three instances for all possible combinations of  $(v, c, d)$  and found that the final best  $RD$  value occurs to be 154, 114 and 96. This means that even though we try all  $(v, c, d)$  combinations, the improvement in the  $RD$  value is tiny in  $(16, 8|1, 1, 1, 1, 1, 1, 1)$  and absent in



**Figure 5.1** Best  $RD$  values for each  $\lambda_{crm}$ ,  $\lambda_{vm}$  and  $\lambda_{tars}$  in the instances (a)  $RDP(16, 8|1, 1, 1, 1, 1, 1, 1, 1)$ , (b)  $RDP(18, 6|2, 2, 2, 1, 1, 1)$  and (c)  $RDP(20, 5|2, 2, 2, 2, 2)$

$(18, 6|2, 2, 2, 1, 1, 1)$  and  $(20, 5|2, 2, 2, 2, 2)$ . Therefore, we preferred to restrict the number of  $(v, c, d)$  combinations to 1,000 to also save computational time.

Table 5.3 reports the best  $RD$  values found by MIP-GOS for the schedules generated by the circle-and-randomize method, the Vizing method, and TARS operations in the second, third and fourth columns, respectively. It can be observed that the matheuristic method finds feasible solutions for all instances under consideration, even for those not solved by MIP-RDP. Moreover, It finds optimal solutions for all instances with  $n = 8$  and  $n = 10$ , and for the instance  $RDP(12, 2|3, 3)$ . The  $RD$  values in these instances equal exactly the  $LB$  presented in Section 3.3.

On the other hand, for comparison purposes, we select a sample of  $DRP$  instances to be solved by other neighborhood structures presented in Section 2.3.4. Table 5.4 reports the results of these neighborhood structures compared to TARS. It can be observed that TARS outperforms all other neighborhood structures in all instances. Moreover, instead of applying TARS operations on the best initial schedule generated, we applied TARS to all generated random opponent schedules for a sample of  $RDP$  instances. To save computational time, the games of these schedules were assigned into matchdays randomly, i.e., without using MIP-GOS. We also reduce the value of  $RD$  by swapping two games with the highest  $RD$  value inside some rounds. Table 5.5 reports the results of a sample of instances compared with matheuristic results. It can be observed that our matheuristic (TARS with MIP-GOS) outperforms TARS with random schedules.

### 5.3 Experimental Results of the Polynomial-time Exact Method

Table 5.6 summarizes the instances solved by our polynomial-time exact method (combined with Theorem 3). It can be noted that this method has found optimal schedules for all instances having  $n = 8$  and  $n = 16$  except the  $RDP(16, 3|3, 3, 2)$ . The reason why an optimal solution can not be attained for the  $RDP(16, 3|3, 3, 2)$  is that it is not possible to obtain the game distribution  $(3, 3, 2)$  by applying Theorem



**Table 5.3** Results of the matheuristic

$RDP(n, p   \gamma_1, \dots, \gamma_p)$	Circle-and-randomize	Vizing	TARS
(8, 2 2, 2)	12	0*	0*
(8, 3 2, 1, 1)	24	12*	12*
(8, 4 1, 1, 1, 1)	36	24*	24*
(10, 2 3, 2)	4	2	0*
(10, 3 2, 2, 1)	20	18	16*
(10, 4 2, 1, 1, 1)	36	34	32*
(10, 5 1, 1, 1, 1, 1)	52	50	48*
(12, 2 3, 3)	20	10	0*
(12, 3 2, 2, 2)	40	24	6
(12, 4 2, 2, 1, 1)	60	44	28
(12, 5 2, 1, 1, 1, 1)	80	60	54
(12, 6 1, 1, 1, 1, 1, 1)	100	82	74
(14, 2 4, 3)	24	4	2
(14, 3 3, 2, 2)	48	22	12
(14, 4 2, 2, 2, 1)	72	46	40
(14, 5 2, 2, 1, 1, 1)	96	68	62
(14, 6 2, 1, 1, 1, 1, 1)	120	92	86
(14, 7 1, 1, 1, 1, 1, 1, 1)	144	120	112
(16, 2 4, 4)	28	16	0
(16, 3 3, 3, 2)	22	30	18
(16, 4 2, 2, 2, 2)	72	60	36
(16, 5 2, 2, 2, 1, 1)	98	86	76
(16, 6 2, 2, 1, 1, 1, 1)	104	110	100
(16, 7 2, 1, 1, 1, 1, 1, 1)	136	138	128
(16, 8 1, 1, 1, 1, 1, 1, 1, 1)	160	166	154
(18, 2 5, 4)	32	14	8
(18, 3 3, 3, 3)	64	44	30
(18, 4 3, 2, 2, 2)	96	60	44
(18, 5 2, 2, 2, 2, 1)	128	90	82
(18, 6 2, 2, 2, 1, 1, 1)	160	124	114
(18, 7 2, 2, 1, 1, 1, 1, 1)	192	156	144
(18, 8 2, 1, 1, 1, 1, 1, 1, 1)	224	186	176
(18, 9 1, 1, 1, 1, 1, 1, 1, 1, 1)	256	218	206
(20, 2 5, 5)	36	24	10
(20, 3 4, 3, 3)	72	44	32
(20, 4 3, 3, 2, 2)	108	72	58
(20, 5 2, 2, 2, 2, 2)	144	106	96
(20, 6 2, 2, 2, 2, 1, 1)	180	148	134
(20, 7 2, 2, 2, 1, 1, 1, 1)	216	172	162
(20, 8 2, 2, 1, 1, 1, 1, 1, 1)	252	206	194
(20, 9 2, 1, 1, 1, 1, 1, 1, 1, 1)	288	244	232
(20, 10 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)	324	274	262

**Table 5.4** A sample of instances solved by different neighborhood structures

$RDP(n, p \gamma_1, \dots, \gamma_p)$	TARS	RS	PRS	PTS
(10, 3 2, 2, 1)	16 (604)	18(9)	18(9)	18 (93)
(12, 4 2, 2, 1, 1)	28 (2703)	32(34)	32(69)	32 (383)
(14, 5 2, 2, 1, 1, 1)	62 (8404)	66(93)	64(112)	64 (3135)
(16, 7 2, 1, 1, 1, 1, 1)	128 (59547)	130(397)	130(496)	130 (8185)
(18, 4 3, 2, 2, 2)	44 (38868)	48(249)	48(239)	48 (4515)
(20, 5 2, 2, 2, 2, 2)	96 (123653)	98(642)	98(598)	100 (17198)

**Table 5.5** A sample of instances solved by TARS with two different approaches

$RDP(n, p \gamma_1, \dots, \gamma_p)$	TARS	
	with random schedules (sec.)	with MIP-GOS (sec.)
(8, 2 2, 2)	4 (2889)	0 (39)
(10, 3 2, 2, 1)	22 (6792)	16 (604)
(12, 3 2, 2, 2)	38 (24385)	6 (2488)
(18, 5 2, 2, 2, 2, 1)	182 (42427)	82 (48413)

3 on any other optimal schedule. Table 5.6 also reports the optimal results for instances having  $n = 12$ . Atan and Çavdaroglu (2018) had obtained an optimal schedule with no rest mismatches for the rest mismatch problem  $RMP(12, 3|2, 2, 2)$  via constraint programming. Starting from this schedule, which obviously also has an  $RD$  of 0, optimal schedules for the  $RDP(12, 4|2, 2, 1, 1)$ ,  $RDP(12, 5|2, 1, 1, 1, 1)$ , and  $RDP(12, 6|1, 1, 1, 1, 1, 1)$  can be constructed by utilizing Theorem 3. An optimal solution for the starting schedule of the  $RDP(12, 3|2, 2, 2)$  is provided in Appendix A.

**Table 5.6** Results of the exact method

$RDP(n, p \gamma_1, \dots, \gamma_p)$	Exact Method
(8, 2 2, 2)	0*
(8, 3 2, 1, 1)	12*
(8, 4 1, 1, 1, 1)	24*
(12, 4 2, 2, 1, 1)	20*
(12, 5 2, 1, 1, 1, 1)	40*
(12, 6 1, 1, 1, 1, 1, 1)	60*
(16, 2 4, 4)	0*
(16, 4 2, 2, 2, 2)	0*
(16, 5 2, 2, 2, 1, 1)	28*
(16, 6 2, 2, 1, 1, 1, 1)	56*
(16, 7 2, 1, 1, 1, 1, 1, 1)	84*
(16, 8 1, 1, 1, 1, 1, 1, 1, 1)	112*

## 5.4 Results Comparison

Table 5.7 summarizes the  $RD$  values found by **MIP-RDP**, the polynomial-time exact method and the matheuristic algorithm in the third, fourth, and fifth columns, respectively. The lower bound found by Theorem 1 is also provided for each instance in the second column. The best solutions found for each instance are bolded, and the values marked with an asterisk (\*) are guaranteed optimal values. When we apply the matheuristic algorithm to all instances under consideration, we either could improve the results of **MIP-RDP** further or find feasible solutions for unsolved instances. Thus, the proposed matheuristic outperforms **MIP-RDP** running on commercial solvers. It can also be observed that the matheuristic method is more likely to come up with much better results than **MIP-RDP** with increased values of  $n$  and  $p$ . The polynomial-time exact method could find optimal solutions not found by other methods particularly in instances with  $n = 12$  and  $n = 16$ , such as  $(12, 4|2, 2, 1, 1)$  and  $(16, 4|2, 2, 2, 2)$ .

In Table 5.4, we list all optimal results found by our proposed methods. In all these instances, the results found equal exactly the  $LB$  presented in Section 3.3, i.e., they are proven to be optimal.

**Table 5.7** Results comparison

$RDP(n, p   \gamma_1, \dots, \gamma_p)$	LB	MIP-RDP (sec.)	Exact Method	Matheuristic
(8, 2 2, 2)	0	<b>0*</b> (0.391)	<b>0*</b>	<b>0*</b>
(8, 3 2, 1, 1)	12	<b>12*</b> (36000)	<b>12*</b>	<b>12*</b>
(8, 4 1, 1, 1, 1)	24	<b>24*</b> (36000)	<b>24*</b>	<b>24*</b>
(10, 2 3, 2)	0	<b>0*</b> (238)	–	<b>0*</b>
(10, 3 2, 2, 1)	16	20 (36000)	–	<b>16*</b>
(10, 4 2, 1, 1, 1)	32	<b>32*</b> (36000)	–	<b>32*</b>
(10, 5 1, 1, 1, 1, 1)	48	<b>48*</b> (36000)	–	<b>48*</b>
(12, 2 3, 3)	0	<b>0*</b> (9183)	–	<b>0*</b>
(12, 3 2, 2, 2)	0	<b>6</b> (36000)	–	<b>6</b>
(12, 4 2, 2, 1, 1)	20	32 (36000)	<b>20*</b>	28
(12, 5 2, 1, 1, 1, 1)	40	56 (36000)	<b>40*</b>	54
(12, 6 1, 1, 1, 1, 1, 1)	60	80 (36000)	<b>60*</b>	74
(14, 2 4, 3)	0	4 (36000)	–	<b>2</b>
(14, 3 3, 2, 2)	0	18 (36000)	–	<b>12</b>
(14, 4 2, 2, 2, 1)	24	52 (36000)	–	<b>40</b>
(14, 5 2, 2, 1, 1, 1)	48	80 (36000)	–	<b>62</b>
(14, 6 2, 1, 1, 1, 1, 1)	72	90 (36000)	–	<b>86</b>
(14, 7 1, 1, 1, 1, 1, 1, 1)	96	128 (36000)	–	<b>112</b>
(16, 2 4, 4)	0	<b>0*</b> (66.75)	<b>0*</b>	<b>0*</b>
(16, 3 3, 3, 2)	0	28 (36000)	–	<b>18</b>
(16, 4 2, 2, 2, 2)	0	44 (36000)	<b>0*</b>	36
(16, 5 2, 2, 2, 1, 1)	28	84 (36000)	<b>28*</b>	76
(16, 6 2, 2, 1, 1, 1, 1)	56	128 (36000)	<b>56*</b>	100
(16, 7 2, 1, 1, 1, 1, 1, 1)	84	152 (36000)	<b>84*</b>	128
(16, 8 1, 1, 1, 1, 1, 1, 1, 1)	112	180 (36000)	<b>112*</b>	154
(18, 2 5, 4)	0	16 (36000)	–	<b>8</b>
(18, 3 3, 3, 3)	0	36 (36000)	–	<b>30</b>
(18, 4 3, 2, 2, 2)	0	54 (36000)	–	<b>44</b>
(18, 5 2, 2, 2, 2, 1)	32	118 (36000)	–	<b>82</b>
(18, 6 2, 2, 2, 1, 1, 1)	64	176 (36000)	–	<b>114</b>
(18, 7 2, 2, 1, 1, 1, 1, 1)	96	172 (36000)	–	<b>144</b>
(18, 8 2, 1, 1, 1, 1, 1, 1, 1)	128	– (36000)	–	<b>176</b>
(18, 9 1, 1, 1, 1, 1, 1, 1, 1, 1)	160	– (36000)	–	<b>206</b>
(20, 2 5, 5)	0	14 (36000)	–	<b>10</b>
(20, 3 4, 3, 3)	0	70 (36000)	–	<b>32</b>
(20, 4 3, 3, 2, 2)	0	108 (36000)	–	<b>58</b>
(20, 5 2, 2, 2, 2, 2)	0	166 (36000)	–	<b>96</b>
(20, 6 2, 2, 2, 2, 1, 1)	36	168 (36000)	–	<b>134</b>
(20, 7 2, 2, 2, 1, 1, 1, 1)	72	260 (36000)	–	<b>162</b>
(20, 8 2, 2, 1, 1, 1, 1, 1, 1)	108	– (36000)	–	<b>194</b>
(20, 9 2, 1, 1, 1, 1, 1, 1, 1, 1)	144	– (36000)	–	<b>232</b>
(20, 10 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)	180	– (36000)	–	<b>262</b>

**Table 5.8** Proven optimal solutions

$RDP(n, p   \gamma_1, \dots, \gamma_p)$	$RD$	Method
(8, 2 2, 2)	0	MIP-RDP, Exact Method, Matheuristic
(8, 3 2, 1, 1)	12	MIP-RDP, Exact Method, Matheuristic
(8, 4 1, 1, 1, 1)	24	MIP-RDP, Exact Method, Matheuristic
(10, 2 3, 2)	0	MIP-RDP, Matheuristic
(10, 3 2, 2, 1)	16	Matheuristic
(10, 4 2, 1, 1, 1)	32	MIP-RDP, Matheuristic
(10, 5 1, 1, 1, 1, 1)	48	MIP-RDP, Matheuristic
(12, 2 3, 3)	0	MIP-RDP, Matheuristic
(12, 3 2, 2, 2)	0	Constraint Programming (Atan & Cavdaroglu, 2018)
(12, 4 2, 2, 1, 1)	20	Exact Method
(12, 5 2, 1, 1, 1, 1)	40	Exact Method
(12, 6 1, 1, 1, 1, 1, 1)	60	Exact Method
(16, 2 4, 4)	0	MIP-RDP, Exact Method, Matheuristic
(16, 4 2, 2, 2, 2)	0	Exact Method
(16, 5 2, 2, 2, 1, 1)	28	Exact Method
(16, 6 2, 2, 1, 1, 1, 1)	56	Exact Method
(16, 7 2, 1, 1, 1, 1, 1, 1)	84	Exact Method
(16, 8 1, 1, 1, 1, 1, 1, 1, 1)	112	Exact Method

## 6. CONCLUSIONS

In this research, we studied the rest difference problem, where the goal is to construct a timetable that determines both the round and the matchday of each game such that the total rest difference throughout the tournament is minimized. A mixed-integer programming formulation, as well as a matheuristic algorithm, were provided to solve the rest difference problem. We found that the presented MIP-RDP with commercial solvers finds optimal schedules for small  $RDP(n, p | \gamma_1, \dots, \gamma_p)$  instances, i.e., when  $n$  and  $p$  values are small. We also found that the matheuristic algorithm outperforms the MIP-RDP running on commercial solvers as it is more likely to come up with better results than MIP-RDP with increased values of  $n$  and  $p$ .

Furthermore, we developed a polynomial-time exact algorithm, which is guaranteed to find a schedule of zero total rest difference when the number of teams is a positive-integer power of 2, and when the number of games in each day is even. It was also shown that a schedule with zero total rest difference found by this algorithm remains optimal if a matchday is split into two consecutive matchdays, one of which is a one-game matchday. Moreover, we provided some theoretical results for the  $RDP$ . A theorem that sets a lower bound on the value of the  $RD$ , and two theorems that allow to easily obtain optimal schedules for some instances from the optimal schedules of the related instances were provided. Computational experiments for several problem instances were executed, and the best schedules for all instances were provided for reference.

We provide here some future directions for this line of research. As future work, one may want to relax some assumptions considered in this study. For example, we assumed that the number of games allocated to each matchday ( $\gamma_d$  values) is the same in all of the rounds. However, in most round-robin tournaments, the number of games in a matchday (say Saturday) does not have to be the same throughout

the season. In the future, this assumption can be relaxed to define an *RDP* corresponding to round-robin tournaments in practice. On the other hand, as future work, one can consider other fairness criteria such as minimizing the number of breaks or balancing the *coe* value together with rest difference minimization, and investigate how these criteria impact each other. Moreover, one can look at the rest difference in time relaxed tournaments, in which the number of rounds is more than the necessary minimum to finish all the games. In these tournaments, not all teams necessarily play in each round, providing more options to increase the rest duration of a team, which may result in increases or decreases in the rest differences among the opposing teams.

Finally, during the experimental runs of the MIP-RDP model, we have noticed that the Gurobi solver could never find a lower bound greater than zero for the *RDP* instances in which there are no one-game matchdays, i.e.,  $p_1 = 0$ . The question of whether there exists a solution with an  $RD = 0$  in RDP instances with  $p_1 = 0$  is still open, and points out a direction for future research.

## References

- Anagnostopoulos, A., Michel, L., Van Hentenryck, P., & Vergados, Y. (2006). A simulated annealing approach to the traveling tournament problem. *Journal of Scheduling*, 9(2), 177–193.
- Anderson, I. (1999). Balancing carry-over effects in tournaments. In K. Quinn, B. Webb, C. Rowley, & F. Holroyd (Eds.), *Combinatorial designs and their applications* (Vol. 403, p. 1-16). CRC Press.
- Atan, T., & Çavdaroğlu, B. (2018). Minimization of rest mismatches in round robin tournaments. *Computers & Operations Research*, 99, 78-89.
- Bean, J. C., & Birge, J. R. (1980). Reducing travelling costs and player fatigue in the national basketball association. *INFORMS Journal on Applied Analytics*, 10(3), 98-102.
- Bengtsson, H., Ekstrand, J., & Hägglund, M. (2013). Muscle injury rates in professional football increase with fixture congestion: An 11-year follow-up of the UEFA Champions League injury study. *British Journal of Sports Medicine*, 47(12), 743–747.
- Briskorn, D. (2008). Feasibility of home–away–pattern sets for round robin tournaments. *Operations Research Letters*, 36(3), 283-284.
- Briskorn, D. (2009). Combinatorial properties of strength groups in round robin tournaments. *European Journal of Operational Research*, 192(3), 744–754.
- Briskorn, D., & Drexler, A. (2009). IP models for round robin tournaments. *Computers & Operations Research*, 36(3), 837-852.
- Briskorn, D., Drexler, A., & Spieksma, F. C. (2010). Round robin tournaments and three index assignments. *4OR*, 8(4), 365–374.
- Briskorn, D., & Knust, S. (2010). Constructing fair sports league schedules with regard to strength groups. *Discrete Applied Mathematics*, 158(2), 123–135.
- Burke, E., De Werra, D., Silva, J. L., & Raess, C. (2004). Applying heuristic methods to schedule sports competitions on multiple venues extended abstract. *Proceedings PATAT 2004*, 441–444.
- Çavdaroğlu, B., & Atan, T. (2020). Determining matchdays in sports league sched-



- ules to minimize rest differences. *Operations Research Letters*, 48(3), 209-216.
- Chandrasekharan, R. C., Toffolo, T. A., & Wauters, T. (2019). Analysis of a constructive matheuristic for the traveling umpire problem. *Journal of Quantitative Analysis in Sports*, 15(1), 41-57.
- Costa, F. N., Urrutia, S., & Ribeiro, C. C. (2012). An ILS heuristic for the traveling tournament problem with predefined venues. *Annals of Operations Research*, 194(1), 137-150.
- de Werra, D. (1981). Scheduling in sports. In P. Hansen (Ed.), *Annals of discrete mathematics (11)* (Vol. 59, p. 381-395). Amsterdam: North-Holland.
- Di Gaspero, L., & Schaerf, A. (2007). A composite-neighborhood tabu search approach to the traveling tournament problem. *Journal of Heuristics*, 13(2), 189-207.
- Duarte, A. R., Ribeiro, C. C., Urrutia, S., & Haeusler, E. H. (2006). Referee assignment in sports leagues. In *International conference on the practice and theory of automated timetabling* (pp. 158-173).
- Durán, G. (2021). Sports scheduling and other topics in sports analytics: A survey with special reference to Latin America. *TOP*, 29(1), 125-155.
- Easton, K., Nemhauser, G., & Trick, M. (2001). The traveling tournament problem description and benchmarks. In T. Walsh (Ed.), *Principles and practice of constraint programming — cp 2001* (pp. 580-584). Berlin, Heidelberg: Springer Berlin Heidelberg.
- Easton, K., Nemhauser, G., & Trick, M. (2003). Solving the travelling tournament problem: A combined integer programming and constraint programming approach. In E. Burke & P. De Causmaecker (Eds.), *Practice and theory of automated timetabling iv* (pp. 100-109). Berlin, Heidelberg: Springer Berlin Heidelberg.
- Elf, M., Jünger, M., & Rinaldi, G. (2003). Minimizing breaks by maximizing cuts. *Operations Research Letters*, 31(5), 343-349.
- Esteves, P. T., Mikolajec, K., Schelling, X., & Sampaio, J. (2021). Basketball performance is affected by the schedule congestion: NBA back-to-backs under the microscope. *European Journal of Sport Science*, 21(1), 26-35.

- Folgado, H., Duarte, R., Marques, P., & Sampaio, J. (2015). The effects of congested fixtures period on tactical and physical performance in elite football. *Journal of Sports Sciences*, *33*(12), 1238–1247.
- Goossens, D. (2018). Optimization in sports league scheduling: Experiences from the Belgian pro league soccer. In G. H. Parlier, F. Liberatore, & M. Demange (Eds.), *Operations research and enterprise systems* (pp. 3–19). Cham: Springer International Publishing.
- Goossens, D., & Spieksma, F. (2009). Scheduling the Belgian soccer league. *INFORMS Journal on Applied Analytics*, *39*(2), 109–118.
- Goossens, D., Yi, X., & Van Bulck, D. (2020). Fairness trade-offs in sports timetabling. In C. Ley & Y. Dominicy (Eds.), *Science meets sports: When statistics are more than numbers* (pp. 213–244). Newcastle upon Tyne: Cambridge Scholars Publishing.
- Goossens, R. D., Kempeneers, J., Koning, H. R., & Spieksma, F. C. (2015). Winning in straight sets helps in Grand Slam tennis. *International Journal of Performance Analysis in Sport*, *15*(3), 1007–1021.
- Guedes, A. C., & Ribeiro, C. C. (2011). A heuristic for minimizing weighted carry-over effects in round robin tournaments. *Journal of Scheduling*, *14*(6), 655–667.
- Günneç, D., & Demir, E. (2019). Fair-fixture: Minimizing carry-over effects in football leagues. *Journal of Industrial & Management Optimization*, *15*(4), 1565–1577.
- Gurobi Optimization Inc. (2021). *Gurobi Optimizer Reference Manual Version 9.1*. Retrieved from <https://tinyurl.com/gurobiref> (Last access: 2021-23-11)
- IBM ILOG CPLEX. (2017). *CPLEX User's Manual Version 12.8*. Retrieved from <https://tinyurl.com/cplexref> (Last access: 2021-23-11)
- Irnich, S. (2010). A new branch-and-price algorithm for the traveling tournament problem. *European Journal of Operational Research*, *204*(2), 218–228.
- Januario, T., & Urrutia, S. (2016). A new neighborhood structure for round robin scheduling problems. *Computers & Operations Research*, *70*, 127–139.
- Januario, T., Urrutia, S., & Gerais-Brazil, M. (2012). An edge coloring heuristic

- based on Vizing's theorem. In *Proc. proceedings of the brazilian symposium on operations research* (pp. 3994–4002).
- Januario, T., Urrutia, S., Ribeiro, C. C., & De Werra, D. (2016). Edge coloring: A natural model for sports scheduling. *European Journal of Operational Research*, *254*(1), 1–8.
- Kendall, G. (2008). Scheduling English football fixtures over holiday periods. *Journal of the Operational Research Society*, *59*(6), 743–755.
- Kendall, G., Knust, S., Ribeiro, C. C., & Urrutia, S. (2010). Scheduling in sports: An annotated bibliography. *Computers & Operations Research*, *37*(1), 1–19.
- Kidd, M. (2010). A tabu-search for minimising the carry-over effects value of a round-robin tournament. *ORiON*, *26*(2).
- Kim, T. (2019). Optimal approach to game scheduling of multiple round-robin tournament: Korea professional baseball league in focus. *Computers & Industrial Engineering*, *136*, 95–105.
- Knust, S. (2008). Scheduling sports tournaments on a single court minimizing waiting times. *Operations Research Letters*, *36*(4), 471–476.
- Knust, S. (2010). Scheduling non-professional table-tennis leagues. *European Journal of Operational Research*, *200*(2), 358–367.
- Lambrechts, E., Ficker, A. M., Goossens, D. R., & Spieksma, F. C. (2018). Round-robin tournaments generated by the circle method have maximum carry-over. *Mathematical Programming*, *172*(1), 277–302.
- Lewis, R., & Thompson, J. (2011). On the application of graph colouring techniques in round-robin sports scheduling. *Computers & Operations Research*, *38*(1), 190–204.
- Miyashiro, R., & Matsui, T. (2005). A polynomial-time algorithm to find an equitable home–away assignment. *Operations Research Letters*, *33*(3), 235–241.
- Miyashiro, R., & Matsui, T. (2006). Semidefinite programming based approaches to the break minimization problem. *Computers and Operations Research*, *33*, 1975–1982.
- Nemhauser, G. L., & Trick, M. A. (1998). Scheduling a major college basketball conference. *Operations Research*, *46*(1), 1–8.

- Post, G., & Woeginger, G. J. (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. (2006). The timetable constrained distance minimization problem. In J. C. Beck & B. M. Smith (Eds.), *Integration of ai and or techniques in constraint programming for combinatorial optimization problems* (pp. 167–181). Berlin, Heidelberg: Springer Berlin Heidelberg.
- Rasmussen, R. V., & Trick, M. A. (2007). A benders approach for the constrained minimum break problem. *European Journal of Operational Research*, 177(1), 198-213.
- Rasmussen, R. V., & Trick, M. A. (2008). Round robin scheduling – a survey. *European Journal of Operational Research*, 188(3), 617-636.
- Rasmussen, R. V., & Trick, M. A. (2009). The timetable constrained distance minimization problem. *Annals of Operations Research*, 171(1), 45.
- 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, C. C. (2012). Sports scheduling: Problems and applications. *International Transactions in Operational Research*, 19(1-2), 201-226.
- Ribeiro, C. C., & Urrutia, S. (2007). Heuristics for the mirrored traveling tournament problem. *European Journal of Operational Research*, 179(3), 775-787.
- Russell, K. G. (1980, 04). Balancing carry-over effects in round robin tournaments. *Biometrika*, 67(1), 127-131.
- Russell, R. A., & Urban, T. L. (2006). A constraint programming approach to the multiple-venue, sport-scheduling problem. *Computers & Operations Research*, 33(7), 1895-1906.
- Sanchez-Sanchez, J., Sanchez, M., Hernandez, D., Ramirez-Campillo, R., Martínez, C., & Nakamura, F. Y. (2019). Fatigue in U12 Soccer-7 players during repeated 1-day tournament games—A pilot study. *The Journal of Strength &*

- Conditioning Research*, 33(11), 3092–3097.
- Sandford, D. (2019). *Frank Lampard not happy Liverpool had two extra days to prepare for super cup game against chelsea*. Retrieved 2019-15-08, from <https://talksport.com/football/587458/frank-lampard-liverpool-having-two-extra-days-to-prepare-as-chelsea-lose-super-cup-clash/> (Last access: 2021-23-11)
- Schönberger, J., Mattfeld, D., & Kopfer, H. (2004). Memetic algorithm timetabling for non-commercial sport leagues. *European Journal of Operational Research*, 153(1), 102-116.
- Scoppa, V. (2015). Fatigue and team performance in soccer: Evidence from the FIFA World Cup and the UEFA European Championship. *Journal of Sports Economics*, 16(5), 482–507.
- Steenland, K., & Deddens, J. A. (1997). Effect of travel and rest on performance of professional basketball players. *Sleep*, 20(5), 366–369.
- Suksompong, W. (2016). Scheduling asynchronous round-robin tournaments. *Operations Research Letters*, 44(1), 96–100.
- Thielen, C., & Westphal, S. (2011). Complexity of the traveling tournament problem. *Theoretical Computer Science*, 412(4), 345-351.
- Trick, M. A. (2001). A schedule-then-break approach to sports timetabling. In E. Burke & W. Erben (Eds.), *Practice and theory of automated timetabling iii* (pp. 242–253). Berlin, Heidelberg: Springer Berlin Heidelberg.
- Urrutia, S., & Ribeiro, C. C. (2006). Maximizing breaks and bounding solutions to the mirrored traveling tournament problem. *Discrete Applied Mathematics*, 154(13), 1932-1938.
- Uthus, D. C., Riddle, P. J., & Guesgen, H. W. (2009). An ant colony optimization approach to the traveling tournament problem. In *Proceedings of the 11th annual conference on genetic and evolutionary computation* (pp. 81–88).
- Van Bulck, D., & Goossens, D. (2020). Handling fairness issues in time-relaxed tournaments with availability constraints. *Computers & Operations Research*, 115, 104856.
- Van Bulck, D., Goossens, D., Schönberger, J., & Guajardo, M. (2020). RobinX:

- A three-field classification and unified data format for round-robin sports timetabling. *European Journal of Operational Research*, 280(2), 568-580.
- van 't Hof, P., Post, G., & Briskorn, D. (2010). Constructing fair round robin tournaments with a minimum number of breaks. *Operations Research Letters*, 38(6), 592-596.
- Van Bulck, D., & Goossens, D. (2020). On the complexity of pattern feasibility problems in time-relaxed sports timetabling. *Operations Research Letters*, 48(4), 452-459.
- Van Bulck, D., Goossens, D. R., & Spieksma, F. C. (2019). Scheduling a non-professional indoor football league: A tabu search based approach. *Annals of Operations Research*, 275(2), 715-730.
- Watanabe, N., Wicker, P., & Yan, G. (2017). Weather conditions, travel distance, rest, and running performance: The 2014 FIFA World Cup and implications for the future. *Journal of Sport Management*, 31(1), 27-43.
- Whetstone, S. (2020). *Moyes moan up about fixture pile up*. Retrieved 2020-23-06, from <https://www.claretandhugh.info/moyes-moan-up-about-fixture-pile-up> (Last access: 2021-23-11)
- Zeng, L., & Mizuno, S. (2013). Constructing fair single round robin tournaments regarding strength groups with a minimum number of breaks. *Operations Research Letters*, 41(5), 506-510.

## APPENDIX A: REFERENCE SOLUTIONS

For reference purposes, we provide the best schedules found for all instances under consideration, i.e., the problem instances given in Table 5.7. While labeling each timetable, we use the expression *an optimal schedule* for the timetables with the optimal RD value, and we use the expression *the best schedule* for the timetables whose RD value is not guaranteed to be optimal.

**Table A.1** An optimal schedule ( $RD = 0$ ) of the  $RDP(8, 2|2, 2)$

Round	Day 1		Day 2	
1	1-2	4-3	5-6	8-7
2	1-3	2-4	5-7	6-8
3	1-4	8-5	2-3	7-6
4	1-5	2-6	4-8	3-7
5	1-6	4-7	5-2	8-3
6	1-7	2-8	5-3	6-4
7	1-8	7-2	5-4	3-6

**Table A.2** An optimal schedule ( $RD = 12$ ) of the  $RDP(8, 3|2, 1, 1)$

Round	Day 1		Day 2	Day 3
1	1-2	4-3	5-6	8-7
2	1-3	2-4	5-7	6-8
3	1-4	8-5	2-3	7-6
4	1-5	2-6	4-8	3-7
5	1-6	4-7	5-2	8-3
6	1-7	2-8	5-3	6-4
7	1-8	7-2	5-4	3-6

**Table A.3** An optimal schedule ( $RD = 24$ ) of the  $RDP(8, 4|1, 1, 1, 1)$ 

Round	Day 1	Day 2	Day 3	Day 4
1	1-2	4-3	5-6	8-7
2	1-3	2-4	5-7	6-8
3	1-4	8-5	2-3	7-6
4	1-5	2-6	4-8	3-7
5	1-6	4-7	5-2	8-3
6	1-7	2-8	5-3	6-4
7	1-8	7-2	5-4	3-6

**Table A.4** An optimal schedule ( $RD = 0$ ) of the  $RDP(10, 2|3, 2)$ 

Round	Day 1			Day 2	
1	1-8	2-5	6-9	3-7	4-10
2	1-5	3-10	6-8	2-9	4-7
3	1-3	2-7	6-10	4-9	5-8
4	1-10	2-3	4-8	5-9	6-7
5	1-2	7-9	8-10	3-4	5-6
6	1-7	2-10	8-9	3-5	4-6
7	1-9	4-5	7-10	2-8	3-6
8	3-8	5-7	9-10	1-4	2-6
9	1-6	3-9	5-10	2-4	7-8

**Table A.5** An optimal schedule ( $RD = 16$ ) of the  $RDP(10, 3|2, 2, 1)$ 

Round	Day 1		Day 2		Day 3
1	2-7	6-8	3-10	1-9	4-5
2	1-5	7-8	2-6	4-10	3-9
3	1-8	3-4	2-10	6-9	5-7
4	2-9	4-8	5-6	7-10	1-3
5	2-4	8-9	3-5	6-7	1-10
6	1-7	4-9	2-8	3-6	5-10
7	2-5	3-8	1-4	7-9	6-10
8	9-10	5-8	1-6	2-3	4-7
9	1-2	8-10	3-7	4-6	5-9



**Table A.6** An optimal schedule ( $RD = 32$ ) of the  $RDP(10, 4|2, 1, 1, 1)$ 

Round	Day 1		Day 2	Day 3	Day 4
1	2-7	6-8	3-10	1-9	4-5
2	1-5	7-8	2-6	4-10	3-9
3	1-8	3-4	2-10	6-9	5-7
4	2-9	4-8	5-6	7-10	1-3
5	2-4	8-9	3-5	6-7	1-10
6	1-7	4-9	2-8	3-6	5-10
7	2-5	3-8	1-4	7-9	6-10
8	9-10	5-8	1-6	2-3	4-7
9	1-2	8-10	3-7	4-6	5-9

**Table A.7** An optimal schedule ( $RD = 48$ ) of the  $RDP(10, 5|1, 1, 1, 1, 1)$ 

Round	Day 1	Day 2	Day 3	Day 4	Day 5
1	2-7	6-8	3-10	1-9	4-5
2	1-5	7-8	2-6	4-10	3-9
3	1-8	3-4	2-10	6-9	5-7
4	2-9	4-8	5-6	7-10	1-3
5	2-4	8-9	3-5	6-7	1-10
6	1-7	4-9	2-8	3-6	5-10
7	2-5	3-8	1-4	7-9	6-10
8	9-10	5-8	1-6	2-3	4-7
9	1-2	8-10	3-7	4-6	5-9

**Table A.8** An optimal schedule ( $RD = 0$ ) of the  $RDP(12, 2|3, 3)$ 

Round	Day 1			Day 2		
1	1-8	4-11	5-7	2-10	3-12	6-9
2	1-11	4-5	9-12	2-3	6-10	7-8
3	1-4	2-6	5-9	3-8	7-10	11-12
4	3-10	4-6	8-12	1-9	2-5	7-11
5	2-7	3-4	9-11	1-5	6-12	8-10
6	2-11	3-9	5-8	1-6	4-7	10-12
7	2-9	4-10	8-11	1-12	3-5	6-7
8	1-7	2-8	3-6	4-9	5-12	10-11
9	1-2	3-7	4-12	5-11	6-8	9-10
10	1-3	5-10	6-11	2-4	7-12	8-9
11	2-12	3-11	7-9	1-10	4-8	5-6

**Table A.9** An optimal schedule ( $RD = 0$ ) of the  $RDP(12, 3|2, 2, 2)$ 

Round	Day 1		Day 2		Day 3	
1	1-2	3-4	5-6	7-8	9-10	11-12
2	1-3	10-12	5-7	2-4	9-11	6-8
3	2-7	6-11	4-5	1-10	8-9	3-12
4	6-7	4-10	1-5	8-12	3-9	2-11
5	3-11	6-10	4-7	1-12	2-9	5-8
6	10-11	1-7	4-12	5-9	3-6	2-8
7	2-6	3-8	7-10	5-12	4-9	1-11
8	6-8	1-4	2-3	7-12	9-11	5-10
9	5-11	3-7	4-6	1-8	9-10	2-12
10	3-5	2-10	7-11	4-8	9-12	1-6
11	1-9	3-10	7-8	6-12	2-5	4-11

**Table A.10** An optimal schedule ( $RD = 20$ ) of the  $RDP(12, 4|2, 2, 1, 1)$ 

Round	Day 1		Day 2		Day 3	Day 4
1	1-2	3-4	5-6	7-8	9-10	11-12
2	1-3	10-12	5-7	2-4	9-11	6-8
3	2-7	6-11	4-5	1-10	8-9	3-12
4	6-7	4-10	1-5	8-12	3-9	2-11
5	3-11	6-10	4-7	1-12	2-9	5-8
6	10-11	1-7	4-12	5-9	3-6	2-8
7	2-6	3-8	7-10	5-12	4-9	1-11
8	6-8	1-4	2-3	7-12	9-11	5-10
9	5-11	3-7	4-6	1-8	9-10	2-12
10	3-5	2-10	7-11	4-8	9-12	1-6
11	1-9	3-10	7-8	6-12	2-5	4-11

**Table A.11** An optimal schedule ( $RD = 40$ ) of the  $RDP(12, 5|2, 1, 1, 1, 1)$ 

Round	Day 1		Day 2	Day 3	Day 4	Day 5
1	1-2	3-4	5-6	7-8	9-10	11-12
2	1-3	10-12	5-7	2-4	9-11	6-8
3	2-7	6-11	4-5	1-10	8-9	3-12
4	6-7	4-10	1-5	8-12	3-9	2-11
5	3-11	6-10	4-7	1-12	2-9	5-8
6	10-11	1-7	4-12	5-9	3-6	2-8
7	2-6	3-8	7-10	5-12	4-9	1-11
8	6-8	1-4	2-3	7-12	9-11	5-10
9	5-11	3-7	4-6	1-8	9-10	2-12
10	3-5	2-10	7-11	4-8	9-12	1-6
11	1-9	3-10	7-8	6-12	2-5	4-11

**Table A.12** An optimal schedule ( $RD = 60$ ) of the  $RDP(12, 6|1, 1, 1, 1, 1, 1)$ 

Round	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6
1	1-2	3-4	5-6	7-8	9-10	11-12
2	1-3	10-12	5-7	2-4	9-11	6-8
3	2-7	6-11	4-5	1-10	8-9	3-12
4	6-7	4-10	1-5	8-12	3-9	2-11
5	3-11	6-10	4-7	1-12	2-9	5-8
6	10-11	1-7	4-12	5-9	3-6	2-8
7	2-6	3-8	7-10	5-12	4-9	1-11
8	6-8	1-4	2-3	7-12	9-11	5-10
9	5-11	3-7	4-6	1-8	9-10	2-12
10	3-5	2-10	7-11	4-8	9-12	1-6
11	1-9	3-10	7-8	6-12	2-5	4-11

**Table A.13** The best schedule ( $RD = 2$ ) of the  $RDP(14, 2|4, 3)$ 

Round	Day 1				Day 2		
1	1-12	4-8	5-10	11-2	3-6	7-9	13-14
2	1-4	5-8	14-6	12-7	2-10	11-3	9-13
3	1-6	3-13	2-9	12-8	7-5	11-10	4-14
4	3-12	4-7	5-11	10-14	1-9	2-8	6-13
5	1-13	9-8	7-11	12-5	2-6	14-3	10-4
6	3-10	7-13	5-9	4-6	1-8	2-14	12-11
7	1-14	3-5	6-9	12-2	4-13	8-11	10-7
8	5-6	7-8	10-13	12-14	1-2	3-9	4-11
9	1-11	3-2	7-14	13-5	4-9	8-10	6-12
10	2-13	4-12	6-8	7-3	1-5	9-10	11-14
11	1-10	11-9	6-7	12-13	2-4	3-8	5-14
12	6-10	14-8	11-13	12-9	1-7	3-4	5-2
13	4-5	6-11	2-7	8-13	1-3	9-14	10-12

**Table A.14** The best schedule ( $RD = 12$ ) of the  $RDP(14, 3|3, 2, 2)$ 

Round	Day 1			Day 2		Day 3	
1	5-11	7-13	6-8	9-14	12-10	1-3	4-2
2	1-2	3-4	5-13	8-11	10-9	6-7	12-14
3	1-4	8-9	10-11	13-3	5-2	6-12	7-14
4	3-5	6-14	13-10	4-11	9-7	1-8	2-12
5	9-13	8-2	11-7	3-6	5-10	1-12	4-14
6	2-6	3-10	8-13	4-12	11-9	1-14	7-5
7	1-7	2-13	3-8	4-9	5-14	6-10	11-12
8	1-9	2-7	4-13	5-6	10-14	3-11	8-12
9	2-10	7-3	11-14	4-8	12-13	1-5	6-9
10	11-13	9-2	3-14	5-8	7-12	6-4	1-10
11	1-6	2-11	13-14	4-10	8-7	3-9	5-12
12	4-7	10-8	9-5	2-14	12-3	1-13	11-6
13	1-11	2-3	4-5	6-13	7-10	8-14	9-12

**Table A.15** The best schedule ( $RD = 40$ ) of the  $RDP(14, 4|2, 2, 2, 1)$ 

Round	Day 1		Day 2		Day 3		Day 4
1	3-6	8-13	4-5	10-14	1-7	9-11	2-12
2	14-5	8-3	2-7	11-12	13-6	9-1	4-10
3	9-6	2-11	3-14	8-5	7-12	10-13	1-4
4	3-5	12-13	11-6	10-1	2-9	8-14	4-7
5	2-6	8-10	4-14	12-5	1-11	9-7	3-13
6	1-5	2-8	4-12	10-6	7-11	13-14	3-9
7	3-11	10-12	1-8	6-4	7-13	9-14	2-5
8	2-14	11-4	1-13	10-3	5-7	9-12	6-8
9	1-3	2-4	8-12	11-14	5-10	9-13	6-7
10	9-10	11-13	1-2	5-6	7-8	12-14	3-4
11	2-13	11-10	4-8	5-9	3-12	7-14	1-6
12	2-10	7-3	1-12	6-14	4-13	11-5	8-9
13	1-14	2-3	4-9	5-13	6-12	7-10	8-11

**Table A.16** The best schedule ( $RD = 62$ ) of the  $RDP(14, 5|2, 2, 1, 1, 1)$ 

Round	Day 1		Day 2		Day 3	Day 4	Day 5
1	11-13	12-14	3-7	5-8	2-4	6-9	1-10
2	2-12	10-9	4-7	8-13	1-6	3-5	11-14
3	1-14	6-3	2-13	10-7	5-11	9-12	4-8
4	1-7	4-9	8-12	13-10	5-6	2-11	3-14
5	1-4	8-10	13-12	5-14	7-9	2-6	3-11
6	1-8	9-11	4-10	5-12	13-14	6-7	2-3
7	4-12	6-14	1-9	13-3	5-10	8-11	2-7
8	2-8	5-9	4-14	11-7	1-13	6-10	3-12
9	4-11	6-13	2-5	7-14	8-9	3-10	1-12
10	1-2	3-9	14-8	13-4	5-7	10-12	6-11
11	6-12	11-10	2-14	13-9	7-8	1-3	4-5
12	3-4	6-8	2-9	10-14	11-12	7-13	1-5
13	1-11	2-10	3-8	4-6	5-13	7-12	9-14

**Table A.17** The best schedule ( $RD = 86$ ) of the  $RDP(14, 6|2, 1, 1, 1, 1, 1)$ 

Round	Day 1		Day 2	Day 3	Day 4	Day 5	Day 6
1	5-6	7-14	2-12	11-13	1-4	9-10	3-8
2	1-13	9-3	4-14	6-7	2-11	8-10	5-12
3	5-10	8-12	2-7	4-13	3-14	1-9	6-11
4	1-6	13-14	7-8	3-4	10-12	2-5	9-11
5	2-9	7-10	1-8	6-14	4-12	3-13	5-11
6	3-5	4-11	6-8	1-14	7-9	2-10	12-13
7	3-6	7-12	1-10	4-5	8-11	2-13	9-14
8	6-10	9-13	3-12	2-14	4-8	1-5	7-11
9	2-8	6-13	4-7	1-11	3-10	9-12	5-14
10	5-9	8-13	12-14	10-11	4-6	1-2	3-7
11	2-3	4-10	11-14	8-9	6-12	1-7	5-13
12	2-4	6-9	10-14	5-8	7-13	1-12	3-11
13	1-3	2-6	4-9	5-7	8-14	10-13	11-12

**Table A.18** The best schedule ( $RD = 112$ ) of the  $RDP(14, 7|1, 1, 1, 1, 1, 1, 1)$ 

Round	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
1	4-11	10-14	6-8	9-12	7-13	1-3	2-5
2	4-10	3-5	11-14	6-12	2-13	8-9	1-7
3	7-9	2-4	6-11	3-10	12-13	1-8	5-14
4	4-8	3-11	6-7	10-13	1-5	12-14	2-9
5	1-10	9-14	3-7	8-11	5-13	2-12	4-6
6	5-8	7-10	11-13	4-12	1-9	3-14	2-6
7	11-12	5-10	1-2	7-8	3-9	4-13	6-14
8	13-14	2-8	1-12	10-11	6-9	5-7	3-4
9	5-9	2-11	4-7	10-12	3-6	8-13	1-14
10	3-12	2-7	9-11	4-5	8-14	1-13	6-10
11	2-3	8-10	4-14	9-13	1-6	7-12	5-11
12	1-4	9-10	7-11	2-14	3-8	5-12	6-13
13	1-11	2-10	3-13	4-9	5-6	7-14	8-12

**Table A.19** An optimal schedule ( $RD = 0$ ) of the  $RDP(16, 2|4, 4)$ 

Round	Day 1				Day 2			
1	1-3	2-4	5-7	6-8	9-11	10-12	13-15	14-16
2	1-4	3-2	5-8	7-6	9-12	11-10	13-16	14-15
3	1-2	5-6	3-4	8-7	9-10	13-14	12-11	15-16
4	1-5	2-6	3-7	4-8	9-13	10-14	11-15	12-16
5	1-6	4-7	3-8	2-5	9-14	12-15	11-16	10-13
6	1-7	2-8	3-5	4-6	9-15	10-16	11-13	12-14
7	1-8	9-16	3-6	11-14	2-7	10-15	4-5	12-13
8	1-9	2-10	3-11	4-12	5-13	6-14	7-15	8-16
9	1-10	4-11	3-12	2-9	5-14	8-15	7-16	6-13
10	1-11	2-12	3-9	4-10	5-15	6-16	7-13	8-14
11	1-12	8-13	3-10	6-15	5-16	4-9	7-14	2-11
12	1-13	2-14	3-15	4-16	5-9	6-10	7-11	8-12
13	1-14	4-15	3-16	2-13	5-10	8-11	7-12	6-9
14	1-15	2-16	3-13	4-14	5-11	6-12	7-9	8-10
15	1-16	2-15	3-14	4-13	5-12	6-11	7-10	8-9

**Table A.20** The best schedule ( $RD = 18$ ) of the  $RDP(16, 3|3, 3, 2)$ 

Round	Day 1			Day 2			Day 3	
1	3-4	6-7	10-15	1-5	2-8	11-14	13-12	9-16
2	2-5	4-6	7-15	13-9	12-16	14-8	1-11	10-3
3	1-3	12-9	6-15	2-4	5-16	14-7	8-13	11-10
4	13-5	8-10	3-15	2-16	7-11	12-6	1-9	4-14
5	2-6	7-16	12-11	3-5	8-15	13-10	1-4	9-14
6	3-7	5-8	4-9	2-11	10-12	13-15	1-14	6-16
7	7-5	2-10	12-15	3-9	8-4	13-14	1-6	11-16
8	3-14	6-11	13-4	12-5	8-9	1-16	2-15	10-7
9	3-6	13-11	8-16	4-5	14-10	9-15	1-12	7-2
10	1-2	5-14	11-8	6-13	9-10	12-7	3-16	15-4
11	2-14	7-9	12-4	5-11	10-6	15-16	1-8	3-13
12	1-13	16-10	4-7	2-9	11-15	14-12	3-8	6-5
13	1-7	4-10	13-16	3-11	9-5	14-15	2-12	8-6
14	2-3	7-13	12-8	4-16	9-11	14-6	1-10	5-15
15	1-15	2-13	3-12	4-11	5-10	6-9	7-8	14-16

**Table A.21** An optimal schedule ( $RD = 0$ ) of the  $RDP(16, 4|2, 2, 2, 2)$ 

Round	Day 1		Day 2		Day 3		Day 4	
1	1-3	2-4	5-7	6-8	9-11	10-12	13-15	14-16
2	1-4	3-2	5-8	7-6	9-12	11-10	13-16	14-15
3	1-2	5-6	3-4	8-7	9-10	13-14	12-11	15-16
4	1-5	2-6	3-7	4-8	9-13	10-14	11-15	12-16
5	1-6	4-7	3-8	2-5	9-14	12-15	11-16	10-13
6	1-7	2-8	3-5	4-6	9-15	10-16	11-13	12-14
7	1-8	9-16	3-6	11-14	2-7	10-15	4-5	12-13
8	1-9	2-10	3-11	4-12	5-13	6-14	7-15	8-16
9	1-10	4-11	3-12	2-9	5-14	8-15	7-16	6-13
10	1-11	2-12	3-9	4-10	5-15	6-16	7-13	8-14
11	1-12	8-13	3-10	6-15	5-16	4-9	7-14	2-11
12	1-13	2-14	3-15	4-16	5-9	6-10	7-11	8-12
13	1-14	4-15	3-16	2-13	5-10	8-11	7-12	6-9
14	1-15	2-16	3-13	4-14	5-11	6-12	7-9	8-10
15	1-16	2-15	3-14	4-13	5-12	6-11	7-10	8-9

**Table A.22** An optimal schedule ( $RD = 28$ ) of the  $RDP(16, 5|2, 2, 2, 1, 1)$ 

Round	Day 1		Day 2		Day 3		Day 4	Day 5
1	1-3	2-4	5-7	6-8	9-11	10-12	13-15	14-16
2	1-4	3-2	5-8	7-6	9-12	11-10	13-16	14-15
3	1-2	5-6	3-4	8-7	9-10	13-14	12-11	15-16
4	1-5	2-6	3-7	4-8	9-13	10-14	11-15	12-16
5	1-6	4-7	3-8	2-5	9-14	12-15	11-16	10-13
6	1-7	2-8	3-5	4-6	9-15	10-16	11-13	12-14
7	1-8	9-16	3-6	11-14	2-7	10-15	4-5	12-13
8	1-9	2-10	3-11	4-12	5-13	6-14	7-15	8-16
9	1-10	4-11	3-12	2-9	5-14	8-15	7-16	6-13
10	1-11	2-12	3-9	4-10	5-15	6-16	7-13	8-14
11	1-12	8-13	3-10	6-15	5-16	4-9	7-14	2-11
12	1-13	2-14	3-15	4-16	5-9	6-10	7-11	8-12
13	1-14	4-15	3-16	2-13	5-10	8-11	7-12	6-9
14	1-15	2-16	3-13	4-14	5-11	6-12	7-9	8-10
15	1-16	2-15	3-14	4-13	5-12	6-11	7-10	8-9

**Table A.23** An optimal schedule ( $RD = 56$ ) of the  $RDP(16, 6|2, 2, 1, 1, 1, 1)$ 

Round	Day 1		Day 2		Day 3	Day 4	Day 5	Day 6
1	1-3	2-4	5-7	6-8	9-11	10-12	13-15	14-16
2	1-4	3-2	5-8	7-6	9-12	11-10	13-16	14-15
3	1-2	5-6	3-4	8-7	9-10	13-14	12-11	15-16
4	1-5	2-6	3-7	4-8	9-13	10-14	11-15	12-16
5	1-6	4-7	3-8	2-5	9-14	12-15	11-16	10-13
6	1-7	2-8	3-5	4-6	9-15	10-16	11-13	12-14
7	1-8	9-16	3-6	11-14	2-7	10-15	4-5	12-13
8	1-9	2-10	3-11	4-12	5-13	6-14	7-15	8-16
9	1-10	4-11	3-12	2-9	5-14	8-15	7-16	6-13
10	1-11	2-12	3-9	4-10	5-15	6-16	7-13	8-14
11	1-12	8-13	3-10	6-15	5-16	4-9	7-14	2-11
12	1-13	2-14	3-15	4-16	5-9	6-10	7-11	8-12
13	1-14	4-15	3-16	2-13	5-10	8-11	7-12	6-9
14	1-15	2-16	3-13	4-14	5-11	6-12	7-9	8-10
15	1-16	2-15	3-14	4-13	5-12	6-11	7-10	8-9



**Table A.24** An optimal schedule ( $RD = 84$ ) of the  $RDP(16, 7|2, 1, 1, 1, 1, 1, 1)$ 

Round	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	
1	1-3	2-4	5-7	6-8	9-11	10-12	13-15	14-16
2	1-4	3-2	5-8	7-6	9-12	11-10	13-16	14-15
3	1-2	5-6	3-4	8-7	9-10	13-14	12-11	15-16
4	1-5	2-6	3-7	4-8	9-13	10-14	11-15	12-16
5	1-6	4-7	3-8	2-5	9-14	12-15	11-16	10-13
6	1-7	2-8	3-5	4-6	9-15	10-16	11-13	12-14
7	1-8	9-16	3-6	11-14	2-7	10-15	4-5	12-13
8	1-9	2-10	3-11	4-12	5-13	6-14	7-15	8-16
9	1-10	4-11	3-12	2-9	5-14	8-15	7-16	6-13
10	1-11	2-12	3-9	4-10	5-15	6-16	7-13	8-14
11	1-12	8-13	3-10	6-15	5-16	4-9	7-14	2-11
12	1-13	2-14	3-15	4-16	5-9	6-10	7-11	8-12
13	1-14	4-15	3-16	2-13	5-10	8-11	7-12	6-9
14	1-15	2-16	3-13	4-14	5-11	6-12	7-9	8-10
15	1-16	2-15	3-14	4-13	5-12	6-11	7-10	8-9

**Table A.25** An optimal schedule ( $RD = 112$ ) of the  $RDP(16, 8|1, 1, 1, 1, 1, 1, 1, 1)$ 

Round	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8
1	1-3	2-4	5-7	6-8	9-11	10-12	13-15	14-16
2	1-4	3-2	5-8	7-6	9-12	11-10	13-16	14-15
3	1-2	5-6	3-4	8-7	9-10	13-14	12-11	15-16
4	1-5	2-6	3-7	4-8	9-13	10-14	11-15	12-16
5	1-6	4-7	3-8	2-5	9-14	12-15	11-16	10-13
6	1-7	2-8	3-5	4-6	9-15	10-16	11-13	12-14
7	1-8	9-16	3-6	11-14	2-7	10-15	4-5	12-13
8	1-9	2-10	3-11	4-12	5-13	6-14	7-15	8-16
9	1-10	4-11	3-12	2-9	5-14	8-15	7-16	6-13
10	1-11	2-12	3-9	4-10	5-15	6-16	7-13	8-14
11	1-12	8-13	3-10	6-15	5-16	4-9	7-14	2-11
12	1-13	2-14	3-15	4-16	5-9	6-10	7-11	8-12
13	1-14	4-15	3-16	2-13	5-10	8-11	7-12	6-9
14	1-15	2-16	3-13	4-14	5-11	6-12	7-9	8-10
15	1-16	2-15	3-14	4-13	5-12	6-11	7-10	8-9

**Table A.26** The best schedule ( $RD = 8$ ) of the  $RDP(18, 2|5, 4)$ 

Round	Day 1					Day 2			
1	6-3	12-16	2-9	10-4	7-17	1-8	5-13	11-15	14-18
2	1-14	17-5	6-4	9-7	3-10	2-12	16-8	13-15	11-18
3	16-11	7-6	12-15	13-18	17-4	1-9	2-5	8-3	10-14
4	13-6	14-3	8-11	9-5	15-17	1-10	4-16	2-7	12-18
5	1-4	12-7	6-8	11-5	14-17	16-10	9-15	13-3	2-18
6	13-9	5-8	6-12	4-14	15-18	1-17	11-7	16-3	10-2
7	1-7	5-15	10-11	8-13	14-12	2-16	4-18	9-6	17-3
8	1-5	12-11	2-3	9-4	7-14	6-18	10-13	17-16	15-8
9	1-12	4-5	18-8	2-14	16-6	3-7	9-11	17-13	15-10
10	4-15	6-14	12-17	7-13	16-5	1-18	3-11	9-10	8-2
11	1-3	4-12	8-10	17-6	9-18	2-11	7-5	13-16	15-14
12	12-9	16-7	8-4	5-14	3-18	1-6	17-10	11-13	15-2
13	3-4	5-12	17-11	10-6	7-18	1-15	16-9	8-14	2-13
14	1-2	4-7	6-11	13-14	15-16	3-12	8-9	10-5	17-18
15	1-11	14-16	9-3	13-4	5-18	2-6	7-15	10-12	8-17
16	2-17	4-11	6-15	9-14	5-3	1-13	8-12	10-7	16-18
17	1-16	2-4	3-15	5-6	7-8	9-17	10-18	11-14	12-13

**Table A.27** The best schedule ( $RD = 30$ ) of the  $RDP(18, 3|3, 3, 3)$ 

Round	Day 1			Day 2			Day 3		
1	2-8	16-3	6-18	1-7	10-12	14-15	4-17	13-11	9-5
2	1-15	8-16	4-9	3-6	5-11	10-14	2-18	17-13	12-7
3	12-18	11-14	17-7	4-15	9-16	6-10	1-8	3-5	2-13
4	1-5	11-12	14-17	18-9	10-13	16-6	2-15	4-7	8-3
5	12-5	2-7	9-6	1-14	11-17	16-15	3-13	18-10	4-8
6	2-17	6-7	13-4	9-12	1-11	8-15	16-10	5-14	3-18
7	2-4	5-16	13-6	3-10	9-11	14-18	8-7	1-12	15-17
8	3-14	16-13	4-6	1-17	11-18	8-12	2-5	7-15	10-9
9	2-10	16-7	3-4	5-17	8-11	12-6	1-18	13-14	15-9
10	11-15	16-12	13-18	4-14	6-17	5-8	1-9	10-7	2-3
11	4-5	7-13	8-18	6-14	12-15	11-16	1-2	3-9	10-17
12	3-17	8-9	15-6	2-16	12-14	4-18	13-5	1-10	11-7
13	1-13	17-9	6-8	3-15	12-4	16-18	2-11	5-10	7-14
14	3-12	10-11	8-17	5-7	9-13	15-18	1-6	14-2	4-16
15	7-9	6-2	3-11	17-12	13-15	5-18	1-4	8-10	14-16
16	6-11	4-10	5-15	14-8	9-2	13-12	7-3	1-16	17-18
17	1-3	2-12	4-11	5-6	7-18	9-14	10-15	17-16	8-13

**Table A.28** The best schedule ( $RD = 44$ ) of the  $RDP(18, 4|3, 2, 2, 2)$ 

Round	Day 1			Day 2		Day 3		Day 4	
1	5-18	12-16	14-11	2-15	8-17	1-4	3-7	6-13	10-9
2	1-7	4-3	14-18	2-17	9-6	5-12	10-13	8-15	11-16
3	5-13	9-12	10-8	1-3	4-14	2-6	15-16	7-18	17-11
4	2-9	7-11	12-13	3-14	10-5	1-15	8-4	6-16	17-18
5	4-6	15-17	16-18	10-14	13-11	1-8	5-9	2-7	3-12
6	4-17	13-14	15-9	1-5	10-16	6-18	12-7	2-3	8-11
7	4-15	6-7	14-5	1-16	13-9	2-8	12-18	3-11	10-17
8	2-16	10-11	9-18	4-5	7-15	6-14	12-8	1-13	3-17
9	5-8	18-11	16-7	1-17	13-3	2-10	9-4	6-12	14-15
10	5-7	12-15	13-17	1-11	10-6	3-9	8-18	2-4	14-16
11	5-15	11-6	12-17	7-13	9-14	3-8	4-16	1-10	2-18
12	3-16	6-15	5-17	7-9	8-14	1-2	11-12	4-13	10-18
13	3-15	10-4	13-18	5-16	6-17	2-12	11-9	1-14	8-7
14	8-16	13-15	9-17	1-12	4-18	2-11	7-14	3-10	6-5
15	2-14	11-5	15-17	8-13	9-16	1-18	6-3	4-12	10-7
16	2-5	3-18	16-13	1-6	8-9	10-12	14-17	4-7	11-15
17	1-9	2-13	3-5	4-11	6-8	7-17	10-15	12-14	16-18

**Table A.29** The best schedule ( $RD = 82$ ) of the  $RDP(18, 5|2, 2, 2, 2, 1)$ 

Round	Day 1		Day 2		Day 3		Day 4		Day 5
1	11-14	13-16	2-12	9-15	1-7	6-18	4-10	8-17	3-5
2	11-13	14-16	3-7	9-12	1-18	15-2	4-8	6-17	5-10
3	6-15	12-16	1-5	13-14	4-17	11-7	3-9	8-10	2-18
4	2-9	7-17	3-8	12-13	1-15	10-18	4-11	6-16	5-14
5	2-7	5-16	1-11	4-14	3-12	10-15	6-13	8-18	9-17
6	1-14	12-10	7-16	13-8	6-9	11-15	3-4	17-18	2-5
7	8-9	15-16	7-13	12-14	2-4	3-17	5-18	11-6	1-10
8	5-17	10-6	1-4	11-18	2-3	8-16	7-12	13-15	9-14
9	2-16	10-17	1-8	7-14	3-11	12-15	4-18	13-9	5-6
10	10-16	13-18	2-14	11-17	3-15	9-5	1-12	8-7	4-6
11	4-9	10-13	6-7	14-15	1-3	16-18	2-17	11-5	8-12
12	1-16	12-17	2-11	6-14	3-18	10-9	4-13	8-5	7-15
13	8-15	14-18	1-17	9-7	3-10	4-16	2-6	11-12	5-13
14	8-14	11-10	1-9	2-13	4-5	6-12	3-16	15-17	7-18
15	1-13	6-3	2-8	9-11	10-14	16-17	4-7	5-12	15-18
16	3-13	14-17	4-15	7-5	1-6	2-10	9-16	12-18	8-11
17	1-2	3-14	4-12	5-15	6-8	7-10	9-18	11-16	13-17

**Table A.30** The best schedule ( $RD = 114$ ) of the  $RDP(18, 6|2, 2, 2, 1, 1, 1)$ 

Round	Day 1		Day 2		Day 3		Day 4	Day 5	Day 6
1	4-13	12-8	1-2	7-16	9-15	17-18	6-11	3-5	10-14
2	12-13	15-16	5-10	11-17	3-14	6-4	1-7	2-8	9-18
3	5-14	8-18	7-9	13-15	1-3	12-16	4-11	2-6	10-17
4	1-12	4-3	5-9	7-13	2-17	10-6	8-15	11-16	14-18
5	6-17	13-9	2-15	14-11	5-12	10-8	1-4	3-7	16-18
6	6-9	7-18	3-10	12-15	1-16	13-17	8-14	2-11	4-5
7	6-18	11-8	2-5	4-17	1-13	10-7	3-15	9-12	14-16
8	6-8	7-15	2-4	5-13	3-17	16-9	12-14	11-18	1-10
9	2-13	8-4	3-16	15-6	9-14	10-18	1-11	12-17	5-7
10	1-18	13-8	3-6	9-10	4-14	11-15	2-16	7-12	5-17
11	4-15	7-17	1-8	3-9	2-12	6-14	13-18	5-16	10-11
12	1-6	5-11	3-8	15-17	2-9	12-18	4-7	13-14	10-16
13	2-10	7-14	1-15	8-17	9-11	13-16	5-6	4-12	3-18
14	3-12	11-13	1-17	16-6	2-14	10-15	7-8	4-9	5-18
15	1-5	8-9	2-7	3-11	6-13	12-10	16-17	4-18	14-15
16	6-12	11-7	1-9	14-17	2-3	15-18	10-13	5-8	4-16
17	1-14	2-18	3-13	4-10	5-15	6-7	8-16	9-17	11-12

**Table A.31** The best schedule ( $RD = 144$ ) of the  $RDP(18, 7|2, 2, 1, 1, 1, 1, 1)$ 

Round	Day 1		Day 2		Day 3	Day 4	Day 5	Day 6	Day 7
1	1-5	11-16	9-10	13-18	3-15	4-14	7-12	8-17	2-6
2	4-13	7-6	2-8	10-15	1-16	9-18	3-14	5-11	12-17
3	1-9	2-15	5-17	13-7	11-18	12-14	10-16	4-6	3-8
4	2-9	16-17	3-6	10-12	4-8	7-18	5-15	11-14	1-13
5	2-16	10-3	8-18	17-9	7-14	1-11	6-15	4-12	5-13
6	10-14	16-18	2-3	12-13	11-15	8-9	1-6	4-5	7-17
7	2-17	6-5	10-11	12-16	1-8	14-18	4-7	9-15	3-13
8	4-15	12-11	6-17	16-8	1-14	3-7	9-13	5-18	2-10
9	8-14	12-15	2-18	13-10	3-5	6-16	1-4	7-9	11-17
10	8-12	10-18	6-9	14-15	5-16	2-13	3-4	1-17	7-11
11	3-11	9-12	2-5	6-14	13-16	1-7	15-18	4-17	8-10
12	1-2	3-12	4-18	17-10	7-16	8-15	5-14	6-13	9-11
13	6-11	8-13	2-12	10-4	17-18	1-3	9-14	15-16	5-7
14	3-17	16-14	6-8	11-13	7-15	12-18	2-4	5-9	1-10
15	3-16	9-4	13-15	14-17	5-10	2-7	8-11	6-18	1-12
16	13-14	15-17	2-11	4-16	1-18	3-9	7-10	6-12	5-8
17	1-15	2-14	3-18	4-11	5-12	6-10	7-8	9-16	13-17

**Table A.32** The best schedule ( $RD = 176$ ) of the  $RDP(18, 8|2, 1, 1, 1, 1, 1, 1, 1)$ 

Round	Day 1		Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8
1	3-16	13-6	5-18	2-17	4-7	1-9	10-15	8-12	11-14
2	9-10	13-16	3-7	2-4	5-17	15-18	1-6	8-14	11-12
3	1-15	7-2	4-5	6-8	3-17	11-18	12-14	10-13	9-16
4	1-2	10-16	3-8	5-7	4-11	9-17	13-14	12-18	6-15
5	9-13	12-15	6-14	3-7	11-17	4-18	2-16	1-5	8-10
6	3-11	6-12	9-14	2-18	13-15	1-10	7-17	4-16	5-8
7	2-14	5-16	4-8	3-12	13-18	9-15	1-7	10-17	6-11
8	2-8	15-16	9-18	6-7	5-14	3-13	4-12	1-17	10-11
9	7-14	16-18	2-15	3-5	1-11	8-9	12-17	4-13	6-10
10	10-12	13-17	14-18	4-6	3-9	5-15	1-8	7-16	2-11
11	1-4	3-15	10-18	14-17	7-11	8-16	12-13	2-5	6-9
12	7-8	12-16	11-13	4-15	17-18	10-14	1-3	2-6	5-9
13	5-6	11-16	1-14	15-18	3-10	2-9	7-12	4-17	8-13
14	5-11	9-12	7-13	14-15	1-16	6-18	8-17	3-4	2-10
15	8-18	15-11	7-9	4-10	14-16	1-13	5-12	2-3	6-17
16	4-14	11-9	3-6	16-17	8-15	2-12	1-18	5-13	7-10
17	1-12	2-13	3-14	4-9	6-16	7-18	8-11	15-17	5-10

**Table A.33** The best schedule ( $RD = 206$ ) of the  $RDP(18, 9|1, 1, 1, 1, 1, 1, 1, 1, 1)$ 

Round	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9
1	7-11	4-18	5-10	6-17	8-13	14-15	2-9	12-16	1-3
2	10-17	8-14	2-12	3-16	4-7	9-15	5-11	1-13	6-18
3	4-16	1-6	9-11	2-8	5-15	3-7	14-17	13-18	10-12
4	6-11	12-14	3-15	9-16	10-13	5-8	17-18	1-4	2-7
5	5-17	16-15	2-4	11-12	1-7	13-14	8-18	3-6	9-10
6	5-16	1-18	8-10	6-14	2-11	4-12	3-9	7-13	15-17
7	3-11	9-12	13-15	4-8	6-10	2-14	5-18	1-16	7-17
8	3-12	8-11	2-5	1-17	7-14	10-15	4-6	16-18	9-13
9	4-15	7-10	5-12	9-18	13-16	11-17	1-14	2-6	3-8
10	2-3	15-18	6-8	16-17	11-14	4-10	1-9	5-7	12-13
11	4-14	10-11	3-18	7-9	5-13	8-17	1-12	6-16	2-15
12	2-16	1-5	10-14	3-13	4-11	12-17	7-18	8-9	6-15
13	4-17	6-9	11-13	1-2	10-16	8-15	3-5	14-18	7-12
14	11-16	1-15	7-8	12-18	2-17	3-10	6-13	4-9	5-14
15	4-13	3-17	5-6	7-16	9-14	1-10	8-12	2-18	11-15
16	6-7	12-15	14-16	1-8	11-18	3-4	5-9	13-17	2-10
17	1-11	2-13	3-14	4-5	6-12	7-15	8-16	9-17	10-18

**Table A.34** The best schedule ( $RD = 10$ ) of the  $RDP(20, 2|5, 5)$ 

Round	Day 1					Day 2				
1	1-19	6-16	4-8	12-11	13-9	2-20	5-18	7-14	15-17	10-3
2	2-15	18-17	12-4	16-11	9-19	1-6	20-3	13-8	10-7	14-5
3	2-11	6-20	4-15	16-19	18-9	1-3	5-13	14-10	8-7	12-17
4	4-16	7-12	15-9	11-6	19-20	1-5	3-14	8-10	13-17	2-18
5	11-9	10-18	3-13	15-16	12-20	1-2	4-7	8-5	17-14	6-19
6	1-8	3-15	18-12	9-20	10-16	2-5	4-17	13-11	7-19	6-14
7	4-5	10-12	11-19	20-16	13-14	1-9	3-8	7-6	18-15	17-2
8	1-15	6-18	17-3	16-13	10-19	2-7	11-4	8-9	5-20	14-12
9	1-10	11-5	4-2	6-13	16-17	3-18	8-12	9-14	20-7	19-15
10	1-11	8-20	4-13	19-18	12-9	2-10	5-16	7-3	14-15	6-17
11	16-7	9-4	13-18	15-10	14-20	1-12	3-6	5-17	19-2	8-11
12	3-5	9-10	11-17	19-12	18-14	1-4	2-6	7-15	16-8	13-20
13	5-19	17-10	11-18	15-8	13-7	1-20	6-12	3-9	4-14	16-2
14	2-3	9-6	7-18	13-19	4-20	1-14	8-17	16-12	5-10	11-15
15	1-16	7-9	10-11	3-19	17-20	2-13	5-6	15-12	4-18	8-14
16	1-7	14-2	9-16	3-11	17-19	4-6	5-12	10-20	13-15	18-8
17	1-17	7-11	13-12	10-4	18-20	2-9	5-15	16-3	8-6	14-19
18	1-18	20-11	13-10	14-16	6-15	2-12	9-5	7-17	4-3	8-19
19	1-13	2-8	3-12	4-19	5-7	6-10	9-17	11-14	15-20	16-18

**Table A.35** The best schedule ( $RD = 32$ ) of the  $RDP(20, 3|4, 3, 3)$ 

Round	Day 1				Day 2			Day 3		
1	3-13	6-17	9-5	14-7	2-18	19-16	4-15	1-20	10-11	12-8
2	3-6	4-18	16-15	9-7	2-19	17-14	12-10	1-11	13-5	8-20
3	2-3	7-16	11-20	10-14	4-12	9-15	13-17	1-5	6-18	8-19
4	3-14	5-19	11-4	17-12	1-6	8-18	15-13	2-7	9-10	16-20
5	3-12	11-17	8-5	16-10	1-13	18-15	14-19	2-9	20-7	4-6
6	1-15	17-8	7-6	13-19	3-10	5-12	4-9	2-14	11-16	18-20
7	4-13	7-8	10-5	14-16	1-3	2-20	17-15	6-19	11-18	12-9
8	3-15	11-19	8-16	18-12	6-9	10-13	17-20	1-2	4-7	5-14
9	6-10	19-18	14-13	9-20	2-12	5-4	16-17	1-7	3-8	11-15
10	1-17	10-20	5-2	19-9	3-4	7-11	14-18	6-13	15-8	12-16
11	2-10	16-6	7-18	17-19	1-9	15-12	5-20	3-11	14-4	13-8
12	14-8	6-5	12-1	16-18	3-9	13-11	4-20	2-17	19-15	7-10
13	7-17	4-8	9-11	5-18	2-15	12-13	10-19	20-3	6-14	16-1
14	1-14	8-11	20-6	15-10	2-13	16-3	9-18	4-17	19-12	5-7
15	3-18	7-12	11-14	17-5	19-4	6-8	10-1	2-16	20-15	9-13
16	3-17	9-16	19-7	13-20	5-11	6-15	14-12	1-18	4-2	10-8
17	7-13	17-9	14-15	10-18	3-19	12-20	5-16	1-4	2-8	6-11
18	2-6	5-3	7-15	12-11	1-8	9-14	19-20	17-10	13-18	4-16
19	1-19	2-11	3-7	4-10	5-15	6-12	8-9	13-16	14-20	17-18

**Table A.36** The best schedule ( $RD = 58$ ) of the  $RDP(20, 4|3, 3, 2, 2)$ 

Round	Day 1			Day 2			Day 3		Day 4	
1	7-11	20-16	18-12	4-19	6-17	9-5	1-14	13-2	3-15	10-8
2	3-8	13-5	6-19	4-9	7-18	12-20	1-2	11-16	10-15	14-17
3	1-16	12-17	8-13	2-9	7-4	18-20	5-6	14-15	3-19	11-10
4	1-8	6-15	11-19	3-10	17-13	5-18	7-14	12-16	2-4	9-20
5	6-11	8-15	13-18	10-5	9-12	2-14	4-20	16-7	1-19	17-3
6	4-16	18-15	5-14	6-13	10-9	19-20	2-12	7-3	1-17	11-8
7	3-12	13-14	15-19	2-7	5-4	8-17	6-9	16-18	1-11	10-20
8	5-3	20-11	19-2	9-16	13-15	14-12	1-10	6-18	4-8	7-17
9	1-6	3-16	8-7	4-10	9-13	14-18	5-11	12-19	2-20	17-15
10	3-13	11-12	4-14	8-5	6-16	15-2	9-18	10-19	1-7	17-20
11	1-20	9-7	15-12	3-14	18-10	4-6	2-5	17-19	13-11	8-16
12	2-11	6-14	3-18	7-10	8-19	15-20	4-12	13-16	1-9	5-17
13	2-18	12-13	16-10	1-5	19-7	15-4	9-17	14-11	3-6	8-20
14	20-3	11-17	16-2	14-9	7-12	5-19	1-15	4-18	6-8	10-13
15	3-11	12-5	9-15	1-4	8-18	17-2	16-14	13-19	6-10	7-20
16	1-12	11-4	17-18	5-15	14-10	16-19	6-7	13-20	2-8	3-9
17	7-13	15-11	8-9	1-18	19-14	2-3	4-17	16-5	6-20	12-10
18	2-10	7-15	16-17	14-8	6-12	5-20	9-11	18-19	1-3	4-13
19	1-13	2-6	3-4	5-7	8-12	9-19	10-17	11-18	14-20	15-16

**Table A.37** The best schedule ( $RD = 96$ ) of the  $RDP(20, 5|2, 2, 2, 2, 2)$ 

Round	Day 1		Day 2		Day 3		Day 4		Day 5	
1	6-19	17-10	1-5	3-11	13-16	14-18	2-12	7-20	4-9	8-15
2	6-16	14-7	13-18	15-20	2-5	8-9	1-11	4-12	3-10	17-19
3	9-20	19-10	11-12	13-15	1-4	2-18	5-8	7-6	3-17	16-14
4	2-10	19-20	5-6	9-11	4-7	12-15	14-17	16-18	1-13	8-3
5	2-19	11-5	4-18	9-6	3-13	7-15	8-17	14-12	1-16	10-20
6	6-18	17-15	4-11	5-19	7-12	16-20	2-13	8-14	1-10	9-3
7	6-17	12-20	2-11	10-9	13-14	16-19	1-8	15-18	3-7	5-4
8	5-18	8-19	3-4	9-13	11-14	17-20	7-16	12-10	1-15	6-2
9	7-11	13-17	3-19	16-12	4-10	14-20	2-15	9-5	1-6	8-18
10	2-9	3-16	6-8	11-17	1-18	4-20	5-15	14-10	7-13	12-19
11	1-20	3-2	4-17	7-19	9-18	13-12	6-11	10-15	5-14	8-16
12	2-20	16-5	1-3	7-17	8-10	12-18	9-14	11-15	4-19	13-6
13	13-19	18-17	2-16	12-8	4-6	10-11	5-7	9-15	1-14	3-20
14	2-8	7-10	11-13	18-19	3-14	5-20	4-15	16-17	1-9	6-12
15	1-17	12-9	2-7	4-16	5-10	8-13	6-20	19-14	3-15	11-18
16	3-18	11-20	6-14	8-7	2-4	9-17	10-16	15-19	1-12	5-13
17	1-19	9-7	10-13	18-20	8-11	15-16	3-6	5-12	2-17	14-4
18	4-8	13-20	1-7	10-18	2-14	6-15	3-12	5-17	9-19	16-11
19	1-2	3-5	4-13	6-10	7-18	8-20	9-16	11-19	12-17	14-15

**Table A.38** The best schedule ( $RD = 134$ ) of the  $RDP(20, 6|2, 2, 2, 2, 1, 1)$ 

Round	Day 1		Day 2		Day 3		Day 4		Day 5	Day 6
1	1-5	6-17	3-4	18-19	8-12	15-20	2-9	7-11	10-14	13-16
2	1-6	5-12	7-9	15-18	2-16	14-13	4-19	11-10	8-20	3-17
3	2-18	6-5	1-12	7-13	10-16	14-19	3-20	15-9	4-11	8-17
4	6-12	16-19	3-9	8-11	10-13	17-20	2-14	5-18	1-7	4-15
5	8-18	17-16	6-9	12-19	1-13	4-14	7-15	10-20	3-11	2-5
6	3-13	12-17	5-11	10-15	1-4	14-20	8-16	9-18	6-19	2-7
7	5-14	16-18	1-11	7-19	8-9	10-17	2-6	4-20	13-15	3-12
8	1-19	10-8	6-20	15-12	4-13	14-16	2-3	5-9	7-18	11-17
9	2-11	17-18	1-10	6-13	3-15	14-9	4-16	12-20	5-7	8-19
10	7-8	9-10	1-15	16-20	3-14	13-18	2-17	12-4	5-19	6-11
11	5-13	8-15	7-10	14-17	3-18	6-4	1-20	19-11	2-12	9-16
12	5-15	14-8	6-10	12-16	4-18	11-9	1-3	7-17	13-20	2-19
13	4-9	10-12	3-7	13-19	2-20	18-11	1-17	15-14	5-8	6-16
14	2-15	10-18	9-12	13-17	3-19	11-20	1-14	5-16	4-7	6-8
15	4-8	5-20	1-16	15-11	2-10	6-3	9-13	17-19	7-14	12-18
16	7-12	11-16	9-17	19-20	3-10	6-15	2-13	5-4	1-8	14-18
17	3-5	15-17	6-14	7-16	8-13	12-11	1-18	4-2	9-20	10-19
18	11-13	12-14	4-10	5-17	6-7	18-20	1-2	3-8	9-19	15-16
19	1-9	2-8	3-16	4-17	5-10	6-18	7-20	11-14	12-13	15-19

**Table A.39** The best schedule ( $RD = 162$ ) of the  $RDP(20, 7|2, 2, 2, 1, 1, 1, 1)$ 

Round	Day 1		Day 2		Day 3		Day 4	Day 5	Day 6	Day 7
1	5-19	13-10	1-3	6-14	7-11	8-20	2-12	17-18	4-9	15-16
2	1-10	9-17	2-18	6-3	11-20	19-14	4-15	5-13	7-8	12-16
3	10-17	15-19	1-9	12-13	2-11	8-5	6-18	4-20	7-16	3-14
4	1-11	7-14	8-12	17-19	5-20	13-9	2-6	3-18	10-15	4-16
5	3-16	8-17	4-10	12-20	5-9	11-14	15-18	1-7	2-19	6-13
6	1-18	16-8	9-11	13-15	6-19	10-20	4-5	3-12	14-17	2-7
7	12-17	19-20	3-7	5-10	2-14	8-18	1-16	9-15	11-13	4-6
8	11-16	17-20	4-13	10-18	2-3	12-19	6-9	8-14	5-7	1-15
9	2-8	11-17	4-12	18-19	13-20	15-14	1-5	10-16	3-9	6-7
10	7-9	13-14	3-10	15-11	2-17	4-18	12-15	8-19	1-20	6-16
11	9-14	12-18	1-8	10-11	2-5	7-13	15-17	6-20	16-19	3-4
12	6-11	10-8	9-18	17-16	1-14	3-20	7-12	5-15	2-13	4-19
13	3-17	15-7	5-12	10-9	8-11	13-19	1-6	16-18	2-4	14-20
14	3-15	11-19	6-8	9-12	1-13	4-14	5-17	2-16	18-20	7-10
15	6-15	12-11	3-19	9-8	1-4	2-10	5-14	7-18	16-20	13-17
16	1-2	6-12	4-17	7-20	8-15	13-16	9-19	10-14	5-18	3-11
17	3-5	9-16	2-20	15-11	1-12	14-18	8-13	4-7	10-19	6-17
18	1-17	7-19	2-9	15-20	3-13	8-4	11-18	12-14	5-16	6-10
19	1-19	2-15	3-8	4-11	5-6	7-17	9-20	10-12	13-18	14-16



**Table A.40** The best schedule ( $RD = 194$ ) of the  $RDP(20, 8|2, 2, 1, 1, 1, 1, 1, 1)$ 

Round	Day 1		Day 2		Day 3	Day 4	Day 5	Day 6	Day 7	Day 8
1	3-17	7-6	1-2	5-19	4-14	16-18	9-15	13-20	10-12	8-11
2	1-6	12-20	4-18	15-13	3-5	9-11	8-10	16-19	2-14	7-17
3	2-17	13-18	3-9	11-19	1-12	8-14	6-20	7-16	5-10	4-15
4	11-12	13-17	6-16	15-10	1-3	4-8	2-18	9-19	14-20	5-7
5	3-16	10-6	1-13	8-7	12-15	14-19	11-17	5-20	9-18	2-4
6	1-8	10-16	11-15	18-20	3-7	6-14	17-19	4-9	2-5	12-13
7	2-12	4-19	5-13	7-14	9-17	1-10	8-16	11-20	15-18	3-6
8	4-13	11-16	2-10	9-7	5-14	8-20	1-17	12-19	6-15	3-18
9	5-17	8-19	3-15	14-10	2-7	13-16	9-20	1-18	6-12	4-11
10	3-14	13-9	4-6	5-8	12-18	7-20	10-19	1-11	15-17	2-16
11	3-12	10-18	2-11	9-14	4-20	16-17	5-6	1-15	8-13	7-19
12	7-13	11-18	2-9	4-12	3-10	16-20	1-5	15-19	14-17	6-8
13	16-5	12-9	7-18	17-8	4-10	6-19	11-13	1-14	2-3	15-20
14	6-11	8-18	3-19	14-13	4-7	5-9	1-20	10-17	12-16	2-15
15	4-5	6-18	1-9	13-19	7-10	15-16	2-20	11-14	3-8	12-17
16	2-19	12-8	1-4	14-16	7-15	17-20	3-11	5-18	6-13	9-10
17	2-8	6-9	3-20	13-10	1-19	17-18	4-16	7-12	5-11	14-15
18	1-16	9-8	5-15	14-12	7-11	18-19	10-20	2-6	3-13	4-17
19	1-7	2-13	3-4	5-12	6-17	8-15	9-16	10-11	14-18	19-20

**Table A.41** The best schedule ( $RD = 232$ ) of the  $RDP(20, 9|2, 1, 1, 1, 1, 1, 1, 1, 1)$ 

Round	Day 1		Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9
1	6-9	14-15	1-20	7-18	5-19	12-13	10-16	3-4	11-17	2-8
2	4-10	18-20	5-13	6-15	12-19	2-11	1-7	3-17	8-16	9-14
3	1-3	10-18	8-14	15-19	9-16	6-13	2-4	7-11	12-17	5-20
4	1-10	7-4	8-19	9-11	3-15	12-20	2-13	6-16	5-17	14-18
5	6-20	10-8	7-19	5-14	1-4	11-13	3-9	17-18	2-16	12-15
6	6-8	7-14	1-5	12-16	15-17	11-19	4-18	9-13	10-20	2-3
7	13-18	17-19	7-12	6-14	3-10	11-15	1-8	4-9	5-16	2-20
8	2-5	13-14	10-15	1-12	18-19	4-16	7-17	9-20	8-11	3-6
9	1-16	11-20	5-15	3-8	7-9	12-18	6-17	4-19	2-14	10-13
10	1-6	10-14	3-7	13-19	8-15	4-17	2-18	9-12	16-20	5-11
11	4-8	7-6	5-9	2-12	10-19	11-16	1-14	17-20	3-13	15-18
12	4-5	13-20	3-18	6-12	7-8	10-11	1-15	14-17	16-19	2-9
13	5-18	16-14	3-20	7-15	9-19	2-10	6-11	4-13	1-17	8-12
14	7-20	17-8	5-10	11-12	1-13	16-18	9-15	2-19	3-14	4-6
15	8-20	16-15	7-13	4-14	3-19	5-12	10-17	9-18	1-11	2-6
16	6-10	14-19	4-12	15-20	7-16	8-13	3-5	9-17	11-18	1-2
17	6-19	14-20	10-12	5-7	1-18	8-9	2-17	4-15	3-11	13-16
18	5-6	8-18	13-17	3-16	12-14	4-11	2-15	19-20	1-9	7-10
19	1-19	2-7	3-12	4-20	5-8	6-18	9-10	11-14	13-15	16-17

**Table A.42** The best schedule ( $RD = 262$ ) of the  $RDP(20, 10|1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$

Round	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10
1	14-19	12-20	11-16	13-15	1-2	10-18	5-9	3-4	6-17	7-8
2	2-16	19-20	5-10	12-14	3-6	11-15	8-9	1-13	4-18	7-17
3	13-17	1-15	6-14	9-18	8-11	2-20	3-12	5-19	10-16	4-7
4	9-14	3-7	5-16	15-18	6-13	2-12	4-10	1-17	8-19	11-20
5	3-14	13-18	6-12	4-16	11-19	1-10	8-20	2-17	5-15	7-9
6	12-16	8-17	4-6	10-19	7-15	1-11	3-13	9-20	14-18	2-5
7	6-11	10-15	13-20	17-19	1-7	2-14	3-9	5-18	4-12	8-16
8	14-16	6-20	7-13	10-11	1-19	4-8	2-9	12-18	3-5	15-17
9	2-18	3-15	16-19	6-7	9-12	4-11	10-13	1-8	14-20	5-17
10	10-12	14-17	5-11	7-19	6-16	4-9	3-18	1-20	8-13	2-15
11	16-17	8-15	7-11	5-12	1-4	3-20	18-19	6-9	10-14	2-13
12	13-14	4-20	15-16	9-19	2-10	5-7	1-3	8-12	6-18	11-17
13	4-14	15-19	9-13	12-17	16-20	1-5	6-8	2-7	3-10	11-18
14	1-16	3-11	9-17	12-13	4-19	7-10	2-6	5-8	14-15	18-20
15	5-6	7-14	15-20	13-19	11-12	1-9	3-16	8-18	2-4	10-17
16	11-13	9-16	5-14	1-18	6-15	12-19	7-20	4-17	8-10	2-3
17	1-14	7-12	2-8	9-11	16-18	4-15	5-13	6-19	10-20	3-17
18	6-10	9-15	1-12	3-19	2-11	8-14	13-16	7-18	4-5	17-20
19	1-6	2-19	3-8	4-13	5-20	7-16	9-10	11-14	12-15	17-18

# Curriculum Vitae

## Personal Information

Name: Tasbih Tuffaha

Nationality: Palestinian

## Academic Background

Bachelor of Civil Engineering, AN-Najah National University, Palestine, (2009).

Master of Engineering Management, AN-Najah National University, Palestine, (2015).

## Languages

Arabic – Native

English – Fluent

Turkish – Beginner / Basics

## Work Experience

(2011–2015): Civil Engineer, Engineering Studies and Consulting Center “AMAR”, Palestine.

(2010 – 2011): Civil Engineer, Nablus Municipality, Palestine.

## Publications and Presentations Derived from the Thesis

Tuffaha T., Cavdaroglu B., Atan T. (2021). Timetabling Round Robin Tournaments with the Consideration of Rest Durations, Proceedings of the 13th International Conference on the Practice and Theory of Automated Timetabling - PATAT 2021 Volume II, Pages 389-397, Bruges, Belgium. [http://www.patatconference.org/patat2020/proceedings/papers/33.PATAT\\_2020\\_v2\\_paper\\_65.pdf](http://www.patatconference.org/patat2020/proceedings/papers/33.PATAT_2020_v2_paper_65.pdf)