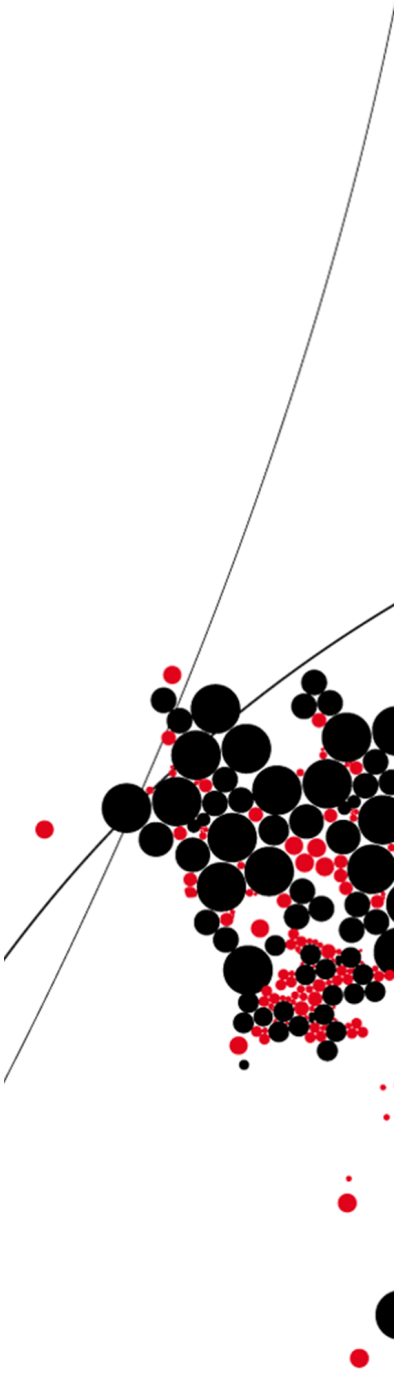
UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering, Mathematics & Computer Science

A polyhedral study of the Travelling Tournament Problem

Marije Renske Siemann

M.Sc. Thesis
March 2020



Supervisor:
dr. M. Walter

Graduation committee:
prof. dr. M.J. Uetz
dr. M. Walter
dr. J.D. Backhoff

Discrete Mathematics and
Mathematical Programming
Department of Applied Mathematics
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Abstract

The Travelling Tournament Problem (TTP) is a sports scheduling problem in which the goal is to find a double round-robin tournament schedule with minimum total travel distance. Additional constraints are the no-repeaters constraint and constraints on the length of home stands and road trips. Without these constraints, the problem is called unconstrained. In this thesis, the TTP is modelled as an integer program and the polytope of solutions is investigated. The dimension of the solution polytope of the unconstrained problem is found and proved. Using this result, it is proved that a certain class of valid inequalities is facet-defining for the unconstrained problem. A separate part of this thesis is spent on analysing the equivalences between three tournament scheduling methods: the circle method, the canonical 1-factorisation and the Kirkman tournament. It is proved that the last two methods are equivalent up to permutation, a result that was missing in electronically available literature.

Preface

This thesis is the result of my final project of the master program Applied Mathematics, and concludes my time as a student. The project was carried out at the University of Twente during the last 7 months. Before I started this project, I did a research internship at a company. Although the company offered me a graduation opportunity, I chose for an internal project at the University of Twente. Not because I did not like the company—quite the contrary—but because I wanted to discover both worlds. Looking back, I am happy with this choice: I have learned what I like and what I do not.

Before I start thanking people, I would like to add a note about the current situation. As a sports enthusiast and athlete, I chose an assignment about scheduling sports tournaments. Ironically, however, at the moment of writing this preface, almost all sports leagues in Europe, and in an increasing number of countries all over the world, are being suspended for an indefinite period of time. This is due to the COVID-19 pandemic that currently rages around the world. The Netherlands are in an "intelligent lockdown", and the presentation of my thesis next week will be held online.

As a result, the last two and a half weeks of writing this thesis did not happen in my familiar office at the university, but in the living room of my student house, after studying on my own in my small student room had proved unsuccessful. Therefore, a first word of thanks goes to my housemates, who accompanied me during these last weeks in our "home office", which greatly improved my productivity.

Of course, I would like to thank my daily supervisor, Matthias Walter. For giving me the opportunity to finish my studies with this research. For his guidance and feedback, for the opportunity to ask questions at any moment, and for his quick responses to e-mails. And finally, for letting me research whatever I like, as long as it results in "nice mathematics". This has resulted in a chapter which is not directly related to the main goal of this thesis, but I did enjoy this part of the research.

It goes without saying that I want to thank my family and friends. Special thanks go to my parents, for their support and unwavering belief that everything will be all right, even though they usually hardly know what I am doing. Hopefully, that will change a bit during the (online) presentation next week. I would like to thank my fellow students for the nice talks during breaks. And of course, I want to thank my friends from Tartaros for the trainings and other activities we did together, which were very welcome distractions from the work.

Finally, I would like to thank the members of the graduation committee for taking the time to read and evaluate my work.

Marije Siemann
March 2020, Enschede

Contents

Chapter 1

Introduction

1.1 Background

All over the world, sports leagues exist. Some leagues, such as the Diamond League (track and field), the UCI World Tour (road cycling), and the Formula One (auto racing), consist of a series of games or races in which multiple teams or individuals compete against each other. However, the larger part of sports leagues, such as the Eredivisie (football) and the MLB (baseball), consist of games between two teams instead of multiple teams. In a typical league, these teams play against each other team twice: once at home and once at the other team's venue. This is the type of league that this thesis focuses on.

In recreational leagues, the distances between the teams are usually not too large. However, in professional leagues, the teams are often spread over an entire country or even multiple countries. Thus, the distances between the teams' stadiums can be rather large. If a team has two or more away games in a row, then excessive travel time and travel cost can be avoided by travelling from the venue of an away game to the venue of the next away game directly, without going home in between. This happens in reality in large competitions such as the NBA (basketball) and is called a road trip.

If a number of venues are close to each other, but far from home, then combining these venues in a road trip saves a large amount of travel distance. This is illustrated in Figure 1.1.

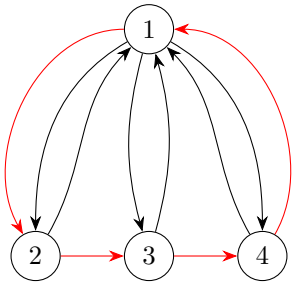


Figure 1.1: In black: team 1 must travel home after the away games against teams 2, 3, and 4, because home games are scheduled in between.
In red: team 1 plays the three away games consecutively.

However, the schedule of one team affects the schedules of the other teams. Thus, simply minimising the travel distance of a single team can lead to unfavourable schedules for other teams. Furthermore,

there are often additional rules to be satisfied, such as a maximum on the number of consecutive home and away games. Consequently, minimising the total travel distance of all teams in the whole schedule is a complicated task. This problem is known in mathematics as the Travelling Tournament Problem (TTP) and is the subject of this thesis.

1.2 Goals

The Travelling Tournament Problem has turned out to be a hard problem. The most common variant of the TTP is proved to be strongly NP-complete in [1]. Intuitively, this is not strange: minimising the travel distance of a single team can be seen as a variant of the travelling salesman problem, which itself is NP-complete. Minimising the travel distance of all teams is even harder: this can be seen as solving a variant of the travelling salesman problem for each team individually, while the individual solutions should also fit together to form a feasible tournament schedule.

When modelling the TTP as an integer program, it turns out that the integrality gap is large, which is one of the reasons that solving the integer program takes very long. This can be improved by adding cutting planes to the program. A special type of cutting planes are facets, which can be seen as the best possible cutting planes. The main goal of this thesis is to reduce the integrality gap by finding facets.

To prove that a valid inequality is indeed facet-defining for a polytope, we must show that its dimension is one less than that of the polytope itself. Thus, it is essential to know the dimension of the polytope. This is an important subgoal of this thesis: computing the dimension of the solution space of the TTP.

During the research, a well-known method to generate tournament schedules was used frequently. While trying to retrace the origin of this method, it was discovered that a certain source, which is often cited as the inventor of this method, cannot be the real inventor. The last subgoal of this thesis, which is not directly related to the main goal but is nevertheless interesting, is to clarify the origin of and the relation between multiple tournament scheduling methods.

1.3 Structure

Chapter 2 describes the Travelling Tournament Problem in more detail. Two variants are introduced, and integer programs are formulated for them. Chapter 3 discusses three different tournament scheduling methods and the equivalences between them. Furthermore, it takes away a misconception about the inventor of a well-known tournament scheduling method. In Chapter 4, the dimension of the solution space of the TTP is figured out. A class of valid inequalities is identified in Chapter 5, and it is proved that these inequalities are facet-defining. Finally, Chapter 6 closes the thesis with the conclusions and recommendations.

It may occur to the reader now that a literature review is missing. This is correct, and has a simple reason: no literature was found in the area of facets and the dimension of the solution space of the TTP. However, for the interested reader, we refer to [2] for an excellent overview of the methods used in tournament scheduling in general and for the TTP in particular.

1.4 Notation

Most of the mathematical notation used in this thesis can be considered standard. Some notation that is not as common or might be ambiguous is clarified in Table 1.1.

Symbol	Meaning
$\lvert\mathbf{x}\rvert$	number of elements in $\mathbf{x}$
$\mathbf{0}_n$	null vector with n elements
$a \equiv b \pmod{n}$	a and b are congruent modulo n
$a \bmod n$	remainder of the division of a by n
$[n]$	the set $\{1, 2, ..., n\}$
$[m, n]$	the set $\{m, m+1, ..., n\}$

Table 1.1: Notation

Chapter 2

The Travelling Tournament Problem

This chapter starts with a precise description of the Travelling Tournament Problem. Next, variants of the TTP and the variants used in this thesis are described. The chapter concludes with integer programming formulations for the used variants.

2.1 Description of the standard TTP

In Section 1.1, we already gave an informal description of the TTP. Here, we make this more precise. Multiple variants of the TTP exist, but we focus on the most predominant one here, which we call "the standard TTP" or "the standard problem".

A round-robin tournament is a tournament in which each team plays each other team a fixed number of times. In the standard TTP, this number is 2, which is called a double round-robin tournament. A double round-robin tournament is often created by concatenating two single round-robin tournament schedules, so that each half of the whole schedule is a single round-robin tournament, but this is not required. Thus, it is allowed that a team plays another team twice in the same half of the schedule.

Each game is allocated to a time slot, or simply slot, such that a team never plays two games in the same slot. In a compact schedule, each team plays a game in each slot. This is possible for an even number of teams, but not when the number of teams is odd. In the standard TTP, an even number of teams compete, and the schedule is compact.

Each game is played at the venue of one of the two competing teams. If a team plays a game at its own venue, this is called a home game, and a game played at the opponent's venue is called an away game. In the standard TTP, where each team plays each other team twice, one of the two games against the same opponent is played at home and the other one away. Thus, if the tournament consists of n teams, each team plays $n-1$ home games and $n-1$ away games, one at each of the other teams' venues. Each team starts and ends the competition at home. Thus, $n-1$ teams will travel from home to an away venue in the first slot, and $n-1$ teams will travel home after the last game.

If a team plays multiple consecutive home games, this is called a home stand, and multiple consecutive away games are called a road trip. The length of a home stand or road trip is defined as the number of games in the home stand or road trip, respectively.

In the standard TTP, there are two additional constraints. First, two teams cannot play the two games between them in two consecutive slots. This is called the no-repeaters constraint. Second, the length of a home stand and road trip has a maximum of three.

2.2 Variants

The main variants differ from the standard TTP in one or more of the following fields (see [3]):

- the maximum length of home stands and road trips is another number than 3, or unconstrained;
- repeaters are allowed;
- the schedule is required to be mirrored: the second half is the same as the first half, but with reversed venues.

In this thesis, we work with two variants of the Travelling Tournament Problem. The first one is the standard problem. In the second one, two types of constraints are ignored: the constraints on the length of a home stand and road trip, and the no-repeaters constraints. This version is referred to as "the unconstrained problem". Although it is less restrictive than the standard problem, the main structure is still intact; the removed constraints make the tournament nicer for the teams, but are not necessary. In [4], it is proved that the unconstrained problem is NP-hard.

The use of the unconstrained problem has the following reason: various proofs have turned out to become much more laborious when the removed constraints are included. With these constraints, it is not guaranteed that simple operations, such as swapping two slots, keep a tournament schedule feasible. The proofs in Chapters 4 and 5 of this thesis apply to the unconstrained problem. However, there is a good chance that these proofs can be extended to proofs for the standard problem using similar proof techniques. Thus, this thesis provides a basis for proving similar theorems for the standard problem and for other variants.

2.3 Integer programming formulations

In this section, we present an integer programming formulation for the standard TTP and the unconstrained TTP.

Parameters:

- number of teams n;
- distance matrix $D,\ d_{ii} = 0\ \forall i,\ d_{ij} = d_{ji}\ \forall i, j$;
- scalar U: maximum length of a home stand or road trip.

Sets:

- slots $k \in \{1, ..., 2(n-1)\}$;
- teams $i, j, s, t \in \{1, ..., n\}$.

Variables:

- $x_{k,i,j} \in \{0, 1\},\ \forall k, i,\ \forall j \neq i$;
- $y_{i,s,t} \in \{0, 1\},\ \forall i, s,\ \forall t \neq s$

with the interpretation

$$\begin{aligned} x_{k,i,j} = 1 \quad &\Longleftrightarrow \quad \text{team } i \text{ plays against team } j \text{ at home in slot } k; \\ y_{i,s,t} = 1 \quad &\Longleftrightarrow \quad \text{team } i \text{ travels from } s \text{ to } t. \end{aligned}$$

The variables $x_{k,i,j}$ will sometimes be referred to as the play variables, and the variables $y_{i,s,t}$ as the travel variables.

Integer program:

$$\min \quad \sum_i \sum_s \sum_{t \neq s} d_{s,t} y_{i,s,t} \tag{2.1}$$

$$\text{s.t.} \quad \sum_{j \neq i} (x_{k,i,j} + x_{k,j,i}) = 1 \qquad \forall k, i \tag{2.2}$$

$$\sum_k x_{k,i,j} = 1 \qquad \forall i,\ \forall j \neq i \tag{2.3}$$

$$x_{k,i,j} + x_{k+1,j,i} \leq 1 \qquad k = 1, ..., 2n-3,\ \forall i,\ \forall j \neq i \tag{2.4}$$

$$\sum_{l=0}^{U} \sum_{j \neq i} x_{k+l,i,j} \leq U \qquad k = 1, ..., 2(n-1) - U,\ \forall i \tag{2.5}$$

$$\sum_{l=0}^{U} \sum_{i \neq j} x_{k+l,i,j} \leq U \qquad k = 1, ..., 2(n-1) - U,\ \forall j \tag{2.6}$$

$$y_{i,s,t} \geq x_{k,s,i} + x_{k+1,t,i} - 1 \qquad k = 1, ..., 2n-3, \forall i,\ \forall s \neq i,\ \forall t \neq s, i \tag{2.7}$$

$$y_{i,i,t} \geq \sum_{j \neq i} x_{k,i,j} + x_{k+1,t,i} - 1 \qquad k = 1, ..., 2n-3,\ \forall i,\ \forall t \neq i \tag{2.8}$$

$$y_{i,t,i} \geq x_{k,t,i} + \sum_{j \neq i} x_{k+1,i,j} - 1 \qquad k = 1, ..., 2n-3,\ \forall i,\ \forall t \neq i \tag{2.9}$$

$$y_{i,i,t} \geq x_{1,t,i} \qquad \forall i,\ \forall t \neq i \tag{2.10}$$

$$y_{i,t,i} \geq x_{2(n-1),t,i} \qquad \forall i,\ \forall t \neq i \tag{2.11}$$

$$x_{k,i,j} \in \{0, 1\} \qquad \forall k, i, j \tag{2.12}$$

$$y_{i,s,t} \in \{0, 1\} \qquad \forall i, s,\ \forall t \neq s \tag{2.13}$$

The objective, in (2.1), is to minimise the total travel distance of all teams. Constraint (2.2) enforces that each team plays exactly once in each slot, either at home or away. Constraint (2.3) implies that each team must play exactly once against each other team at home and once away. It is ensured that two teams do not play each other twice consecutively by constraint (2.4) (no-repeaters constraint). Finally, constraints (2.5) and (2.6) guarantee that a home stand or road trip has length at most U, respectively.

Constraints (2.7) – (2.9) couple the variables $y_{i,s,t}$ to the variables $x_{k,i,j}$. The first one, (2.7), takes care of travelling from an opponent's venue to another opponent's venue. Constraints (2.8) and (2.9) manage the trips from home to an opponent's venue and vice versa, respectively. However, one case is missed by these constraints: as each team must start and finish the tournament at home, a team should also travel from home to an opponent's venue when the first game is played away, and vice versa if the last game is played away. Constraints (2.10) and (2.11) take care of this.

We model the standard TTP by the integer program specified above, but replace the no-repeaters constraint (2.4) with the following constraint:

$$x_{k,i,j} + x_{k,j,i} + x_{k+1,i,j} + x_{k+1,j,i} \leq 1 \qquad k = 1, ..., 2n-3,\ \forall i,\ \forall j \neq i. \tag{2.14}$$

This strengthens the linear relaxation.

The integer programming formulation of the unconstrained problem is equal to the inter program for the standard problem, except that constraints (2.4) (no repeaters), (2.5) (length of home stand), and (2.6) (length of road trip) are removed.

Chapter 3

Tournament scheduling methods and their equivalence

In subsequent chapters, we will have to generate tournament schedules multiple times. Therefore, this chapter discusses three methods to generate single round-robin tournament schedules. It turns out that the tournament schedules generated by these three methods have a similar structure: they are equivalent up to permutation of the teams. However, the equivalences are not always clear at first sight.

In the next section, we will discuss the three methods. Thereafter, we continue with a proof that two of these tournament scheduling methods are equivalent, a proof of which is missing in electronically available literature.

3.1 Three scheduling methods

We discuss three tournament scheduling methods: the circle method, the canonical 1-factorisation and the Kirkman tournament. In the largest part of this chapter, we are not interested in the venue where games take place: home or away. However, for the canonical 1-factorisation, we add a method to determine a home-away assignment.

3.1.1 Circle method

The circle method is a well-known method to generate a single round-robin tournament schedule. We will explain the method shortly. First, the n teams are arranged in a $2 \times n/2$ array, in any order. The teams in the same column play each other in the first slot. Next, one of the teams is fixed. The other teams move one position, either clockwise or counterclockwise, skipping the fixed team. In the second slot, the teams that are now in the same column play each other. An example of such a rotation is shown in Figure 3.1. These rotations are repeated until the teams are back at their initial position in the array. At this point, each team has played each other team exactly once.

1	2	3	...	$n/2$
n	$n-1$	$n-2$	...	$n/2+1$

$\longrightarrow$

1	n	2	...	$n/2-1$
$n-1$	$n-2$	$n-3$	...	$n/2$

Figure 3.1: Example of a rotation of the circle method. Team 1 is fixed, the other teams move clockwise.

The name circle method originates from a slightly different way of generating the same schedule, in which the teams are arranged in a circle instead of in an array. A clear explanation of this method can be found in [5]. In this article, it is also explained that the circle method is equivalent to another construction, called the canonical 1-factorisation. This construction was first described by De Werra in [6].

3.1.2 Canonical 1-factorisation

The canonical 1-factorisation is a specific 1-factorisation of the complete graph on an even number of vertices n, all of which represent a team. The 1-factors, i.e., perfect matchings, are defined as follows, as described in [7]. The 1-factor F_k is the set of edges specifying the games played in slot k, and is defined by

$$F_k = \{\{n, k\}\} \cup \{\{k+i, k-i\} \mid i = 1, ..., n/2 - 1\} \qquad k = 1, ..., n-1. \tag{3.1}$$

Here, the numbers $k+i$ and $k-i$ are taken modulo $n-1$ as one of the numbers $1, ..., n-1$. The games are denoted by sets of two teams instead of tuples because the order of the teams is irrelevant; we are not concerned with the venues of the games here. The schedule derived from the canonical 1-factorisation is equivalent to the schedule generated by the circle method up to permutation of the teams. When the circle method is applied as in Figure 3.1, with the teams arranged in the same order, team n fixed, and rotating counterclockwise, then the resulting schedule is equal to the schedule from the canonical 1-factorisation.

In the remainder of this chapter, we are not interested in the venues of the games. However, in a later chapter, we will be. Therefore, we add a method to determine which team will play home and away in each game, which is called a home-away assignment. Denote the games by a tuple instead of a set, and let the first team in the tuple play at home. Then the following home-away assignment is suggested in [7]:

- $(2n, i)$ if i is odd, or $(i, 2n)$ if i is even;
- $(i-k, i+k)$ if k is odd, or $(i+k, i-k)$ if k is even.

This assignment has a number of favourable properties, such as the balancedness of the home and away games for each team. We call this home-away assignment the standard home-away assignment.

We now have two methods to generate equivalent tournament schedules: a somewhat geometric way (the circle method), and an algebraic way (the canonical 1-factorisation). There exists a third method, described first by Reverend Kirkman in [8] in the year 1847 already, which can be seen as a greedy method.

3.1.3 Kirkman tournament

A tournament scheduled according to the method described by Kirkman is called a Kirkman tournament. It is constructed as follows. All the games $\{i, j\},\ i < j$ have to be scheduled once. First they are sorted lexicographically: if $i < j$ and $i' < j'$, then $\{i, j\} \leq \{i', j'\}$ if and only if $i < i'$ or $(i = i'$ and $j \leq j')$. Next, the games are scheduled sequentially.

We start with an empty array with $n-1$ columns and a yet unknown number of rows. After completion, column k of the array will contain the games played in slot k. The first game, $\{1, 2\}$, is scheduled at position $(1, 1)$ of the array, i.e., the top-left corner. The next games are scheduled by traversing the array in a left-to-right, top-to-bottom manner, starting from the position of the last scheduled game, and picking the first entry which keeps the schedule feasible. That is, an entry in a column in which none of the teams to be scheduled have played yet. As one only moves forward through the array, a once skipped entry will never be filled, but the column (slot) containing this skipped entry can still be picked in later rows.

Thus, games $\{1,3\}$ up to $\{1,n\}$ are placed at positions $(1,2)$ up to $(1,n-1)$, respectively, as there are no other games in these columns yet. Next, game $\{2,3\}$ has to be scheduled. The first empty entry is at position $(2,1)$. However, this column already contains the game $\{1,2\}$, so that the game $\{2,3\}$ can not be scheduled here—otherwise, team 2 is scheduled twice in slot 1. Similarly, the game cannot be scheduled at position $(2,2)$. The next entry is available however, so that game $\{2,3\}$ is scheduled at position $(2,3)$. This process is continued until all games have been scheduled. An example with 6 teams is shown in Table 3.1. Note that the game $\{2,6\}$ is scheduled at position $(3,2)$ and not at position $(2,2)$ in this example, as one only moves forward through the array.

	c1	c2	c3	c4	c5
r1	$\{1,2\}$	$\{1,3\}$	$\{1,4\}$	$\{1,5\}$	$\{1,6\}$
r2	-	-	$\{2,3\}$	$\{2,4\}$	$\{2,5\}$
r3	-	$\{2,6\}$	-	-	$\{3,4\}$
r4	$\{3,5\}$	-	-	$\{3,6\}$	-
r5	-	$\{4,5\}$	-	-	-
r6	$\{4,6\}$	-	$\{5,6\}$	STOP	

Table 3.1: Kirkman tournament with 6 teams. Dashes indicate skipped entries.

Now we can simply collect the teams in each column to obtain the tournament schedule. It turns out that this method always yields a feasible single round-robin schedule for an even number of teams n, as proved in [9].

Kirkman is often cited as the founder of the circle method (see, e.g., [5], [10], [11], [12]). However, this is incorrect: it can easily be seen that the method above has nothing to do with circles. It has a totally different structure than the circle method. Moreover, Kirkman did not use the method above to produce tournament schedules; it was only a tool used in solving a larger mathematical problem. Surprisingly, the resulting schedule is equivalent up to permutation to the schedule derived from the circle method. However, a non-trivial permutation must be applied to obtain equal schedules and there is no easy way to transform a Kirkman tournament into the circle method. Therefore, Kirkman cannot be considered the founder of the circle method.

To the best of our knowledge, Félix Walecki, a French mathematician, is the real inventor of the circle method. Édouard Lucas, another French mathematician, posed the following recreational problem:

Un pensionnat renferme un nombre pair de jeunes filles qui se promènent tous les jours deux par deux ; on demande comment il faut disposer les promenades de telle sorte qu'une jeune fille se trouve successivement en compagnie de toutes les autres, mais ne puisse s'y trouver plus d'une fois.

Paraphrased translation: an even number of young girls walk in pairs each day. How should the pairs be arranged such that each girl forms a pair with each other girl exactly once?

In his book Récréations mathématiques, published in 1883 [13], Édouard Lucas wrote down the circle method as a solution to this problem, and attributes the solution to Félix Walecki.

3.2 Equivalence of the canonical 1-factorisation and the Kirkman tournament

It is known that the circle method and the canonical 1-factorisation are equivalent (see, e.g., [5]). The equivalence of the Kirkman tournament and these two methods (up to permutation) is non-trivial, however. Presumably, a proof of this equivalence has been published before in [14]. We know this because references to this article have been found, stating that the article contains this proof.

However, the article is very inaccessible. Also, no other articles have been found containing the proof. Therefore, we will prove the equivalence ourselves in this section.

The following plan will be followed to prove the equivalence. First, in Section 3.2.1, we investigate which games take place in each slot in a Kirkman tournament. Thereafter, a permutation is applied to these games in Section 3.2.2. Finally, in Section 3.2.3, it is shown that the permuted Kirkman tournament and the canonical 1-factorisation are equal.

3.2.1 Games in each slot in a Kirkman tournament

Although the rule used to generate a Kirkman tournament is simple, it is not immediately clear which games take place in which slot. In [9], the following theorem is proved by mathematical induction:

Proposition 3.1. [9] In a Kirkman tournament,

- each game $\{i, n\}$, $1 \leq i \leq n-1$, takes place in slot $(2i-2) \bmod (n-1)$;
- each game $\{i, j\}$, $1 \leq i < j \leq n-1$, takes place in slot $(i+j-2) \bmod (n-1)$.

Here, the result of the modulo operation is one of the numbers $1, ..., n-1$.

The theorem can be seen as a function that maps each game to a slot. We would like to know which games take place in each slot; that is, we are looking for the (multivalued) inverse of this function—which strictly speaking does not exist, but we take a practical approach here. To find out which games $\{i, n\}$ and $\{i, j\}$ take place in slot k, we must solve the following two equations, respectively:

$$\begin{aligned} 2i - 2 &\equiv k \pmod{n-1} \\ i + j - 2 &\equiv k \pmod{n-1} \end{aligned}$$

The resulting games, for each slot k, are collected in Table 3.2. It remains to prove that these games are 1) valid, 2) in the correct slot, and 3) exhaustive.

k even	k odd, $k \neq n-1$	$k = n-1$
$\{\frac{k}{2}+1, n\}$	$\{\frac{n+k+1}{2}, n\}$	$\{1, n\}$
$+1 \left(\begin{matrix}\{1, k+1\} \\ \{2, \quad k \quad\} \\ \vdots \quad \vdots \\ \{\frac{k}{2}, \frac{k}{2}+2\}\end{matrix}\right) -1$	$+1 \left(\begin{matrix}\{\ 1\ , k+1\} \\ \{\ 2\ , \quad k \quad\} \\ \vdots \quad \vdots \\ \{\frac{k+1}{2}, \ \frac{k+3}{2}\ \}\end{matrix}\right) -1$	$+1 \left(\begin{matrix}\{2, n-1\} \\ \{3, n-2\} \\ \vdots \quad \vdots \\ \{\frac{n}{2}, \ \frac{n+2}{2}\ \}\end{matrix}\right) -1$
$+1 \left(\begin{matrix}\{k+2,\ n-1\ \} \\ \{k+3,\ n-2\ \} \\ \vdots \quad\quad \vdots \\ \{\ \frac{n+k}{2}\ , \frac{n+k+2}{2}\}\end{matrix}\right) -1$	$+1 \left(\begin{matrix}\{\ k+2\ ,\ n-1\ \} \\ \{\ k+3\ ,\ n-2\ \} \\ \vdots \quad\quad \vdots \\ \{\frac{n+k-1}{2}, \frac{n+k+3}{2}\}\end{matrix}\right) -1$	-

Table 3.2: The games taking place in slot k, $1 \leq k \leq n-1$, in a Kirkman tournament

Validity

A game $\{i, j\}$ is valid if the teams i and j are two distinct integers between 1 and n. In Table 3.2, a distinction is made between even and odd k to ensure that all teams are integer. Slightly abusing

the concept of a set, we will use the phrases 'first team' and 'second team' in the next part to refer to the teams i_1 and i_2, respectively, in the game $\{i_1, i_2\}$.

First, we show that the teams playing each other in each game are distinct. In the top row of the table, one can simply substitute the largest value of k and note that the first team is always smaller than n. In the (non-empty) columns in the middle and bottom rows, a series of games is displayed. The last game of each of these series contains a game $\{i, i+1\}$ for some team i, so these teams are clearly distinct. Furthermore, i is the largest first team and $i+1$ the smallest second team in each of these series. Hence, the other games must also contain distinct teams.

Finally, we should verify that all the teams are between 1 and n. In the previous part, it was already shown that for each game in Table 3.2, the first team is smaller (in team number) than the second team. Therefore, we should verify that the first team is at least equal to 1 and the last team at most equal to n in all games. For the games in the top row, this is trivial. For the middle and bottom rows, it is sufficient to check the first game of each of the series, as this one contains the smallest first team and the largest second team. Checking the values of these teams is also trivial and leads to the conclusion that all teams are between 1 and n.

Having shown that the teams in the games in Table 3.2 are all integer, distinct, and between 1 and n, we can conclude that the games are indeed valid.

Correct slot

Here, we show that each game is placed in the right slot. The top row of the table contains the games in which team n plays; the games $\{i, n\}$. Therefore, we should compute $2i - 2$ and verify that this is equal to $k \pmod{n-1}$ for each game (cf. Theorem 3.1). This turns out to be true; we leave it to the reader to verify this.

The middle and bottom rows contain series of games in which team n does not play. Hence, for each of these games $\{i, j\}$ we should check that $i + j - 2$ equals $k \pmod{n-1}$. As the first team increases by one and the second team decreases by one in each of these series of games, it is clear that $i + j - 2$ is a constant number in each series. Therefore, it is sufficient to check that the first game of each series is in the correct slot. Verifying this is trivial, and it turns out that all games are in the right slot.

Exhaustiveness

Now that it is confirmed that each of the games in Table 3.2 are valid and in the correct slot, it remains to prove that these games are exhaustive, i.e., that no games are missing. Therefore, we count the number of games in each slot and verify that this is equal to $n/2$. In Table 3.3, the number of games in each cell of Table 3.2 are shown.

k even	k odd, $k \neq n-1$	$k = n-1$
1	1	1
$\frac{k}{2}$	$\frac{k+1}{2}$	$\frac{n}{2} - 1$
$\frac{n-k-2}{2}$	$\frac{n-k-3}{2}$	0
$\frac{n}{2}$	$\frac{n}{2}$	$\frac{n}{2}$

Table 3.3: In the top three rows: the number of games in the corresponding cells of Table 3.2. In the bottom row: the column sums.

The number of games in each column sum to $n/2$, proving that no games are missing. Together with the fact that all the games are valid and placed in the correct slot, we can conclude the following:

Theorem 3.2. The games taking place in a Kirkman tournament in each slot k, $1 \leq k \leq n-1$, are given in Table 3.2.

3.2.2 Apply permutation

Now we apply the permutation to the games in Table 3.2. After the permutation, the Kirkman tournament should be equal to the tournament resulting from the canonical 1-factorisation. The following permutation P is used, where t is a team between 1 and n:

$$P(t) = \begin{cases} n & \text{if } t = n \\ 2(t-1) \bmod (n-1) & \text{otherwise} \end{cases}$$

Again, the result of the modulo operation is a number between 1 and $n-1$.

First, we should prove that this is indeed a permutation. That is, the domain and the range of P should be equal. It is easy to see that each team t, $1 \leq t \leq n$, is mapped to a valid team (a team between 1 and n): team n is mapped to itself, and the modulo operation takes care of the other teams.

Besides this, it should be proved that no two teams are mapped to the same team. For team n this is clear. Now suppose to the contrary that there exist two teams t_1 and t_2, $1 \leq t_1 < t_2 \leq n-1$, such that $P(t_1) = P(t_2)$. That is,

$$\begin{aligned} & P(t_1) = P(t_2) \\ \iff \quad & 2(t_2-1) \bmod (n-1) = 2(t_1-1) \bmod (n-1) \\ \iff \quad & 2(t_2-1) = 2(t_1-1) + p(n-1) && \text{(for some integer } p\text{)} \\ \iff \quad & 2(t_2-t_1) = p(n-1) && (3.2) \end{aligned}$$

Note that the left-hand side of the last equation is an even number. On the right-hand side, $n-1$ is an odd number. That implies that p should be even. We distinguish two cases.

Case 1: $p \leq 0$ If p is less than or equal to 0, then $p(n-1)$ is at most 0. However, as t_2 is greater than t_1, $2(t_2-t_1)$ is greater than zero. This gives no solutions.

Case 2: $p \geq 2$ If p is greater than or equal to 2, then $p(n-1)$ is at least $2(n-1)$. However, $2(t_2-t_1) \leq 2((n-1)-1) = 2(n-2) < 2(n-1)$. This gives no solutions.

Hence, there are no solutions to equation (3.2). This shows that no two distinct teams are mapped to the same team by P, and hence, P is a valid permutation.

First we apply the permutation to the games in the top row of Table 3.2, the games in which team n plays. The second team in these games is always team n, which is mapped to itself. The images of the first team under the permutation P are given below:

$$\begin{array}{llllll} k \text{ even} & P\left(\frac{k}{2}+1\right) & \equiv 2\left(\frac{k}{2}+1-1\right) & = k & & \pmod{n-1} \\ k \text{ odd}, k \neq n-1 & P\left(\frac{n+k+1}{2}\right) & \equiv 2\left(\frac{n+k+1}{2}-1\right) & = n+k-1 \equiv k & & \pmod{n-1} \\ k = n-1 & P(1) & \equiv 2(1-1) & = 0 & \equiv n-1 & \pmod{n-1} \end{array}$$

Similar computations are executed for the games in the middle and bottom rows of Table 3.2. The resulting games are given in Table 3.4. Note that the modulo operation has not been applied yet to the games in the middle and bottom rows, as this depends on the value of k.

We call this schedule the P-permuted Kirkman tournament.

k even	k odd, $k \neq n-1$	$k = n-1$
$\{k, n\}$	$\{k, n\}$	$\{n-1, n\}$
$+2\left(\begin{matrix}\{ & 0 & , & 2k & \} \\ \{ & 2 & , & 2k-2 & \} \\ & \vdots & & \vdots & \end{matrix}\right)-2$ $\{k-2,\ k+2\}$	$+2\left(\begin{matrix}\{ & 0 & , & 2k & \} \\ \{ & 2 & , & 2k-2 & \} \\ & \vdots & & \vdots & \end{matrix}\right)-2$ $\{k-1,\ k+1\}$	$+2\left(\begin{matrix}\{ & 2 & , & 2n-4 & \} \\ \{ & 4 & , & 2n-6 & \} \\ & \vdots & & \vdots & \end{matrix}\right)-2$ $\{n-2,\ n\}$
$+2\left(\begin{matrix}\{ & 2k+2 & , & 2n-4 & \} \\ \{ & 2k+4 & , & 2n-6 & \} \\ & \vdots & & \vdots & \end{matrix}\right)-2$ $\{n+k-2,\ n+k\}$	$+2\left(\begin{matrix}\{ & 2k+2 & , & 2n-4 & \} \\ \{ & 2k+4 & , & 2n-6 & \} \\ & \vdots & & \vdots & \end{matrix}\right)-2$ $\{n+k-3, n+k+1\}$	-

Table 3.4: The games taking place in slot k, $1 \leq k \leq n-1$, in a Kirkman tournament after applying permutation P. The teams in the middle and bottom rows should be taken modulo $n-1$ as one of the numbers $1, ..., n-1$ after substituting a value for k.

3.2.3 Equality of permuted Kirkman tournament and canonical 1-factorisation

It remains to prove that the P-permuted Kirkman tournament is equal to the schedule following from the canonical 1-factorisation. We differentiate between the games in which team n does and does not play.

Looking in the top row of Table 3.4, it is clear that team n plays against team k in slot k in the P-permuted Kirkman tournament. Recall the definition of the canonical 1-factorisation:

$$F_k = \{\{n, k\}\} \cup \{\{k+i, k-i\} \mid i = 1, ..., n/2-1\} \qquad k = 1, ..., n-1 \tag{3.1}$$

The first game of each factor F_k, $\{n, k\}$, shows that team n also plays against team k in slot k. Thus, P-permuted Kirkman tournament and the canonical 1-factorisation are equal with respect to the games in which team n plays.

Next, we consider the games in which team n does not play. To show that the tournaments are also equal with regard to these games, we take two steps. First, we show that the sum of the teams playing against each other in slot k is always equal to $2k \bmod (n-1)$ in both tournaments. Thereafter, we prove that this implies that the tournaments are equal.

Sum of the teams in each game

In the canonical 1-factorisation, the games in which team n does not play is given by the set $\{\{k+i, k-i\} \mid i = 1, ..., n/2-1\}$ in slot k. Here, $k+i$ and $k-i$ should be taken modulo $n-1$. Adding these two numbers, we see that the sum of the teams playing each other in slot k is equal to $2k \bmod (n-1)$.

The same holds for the P-permuted Kirkman tournament. The first games in the columns of the middle row of Table 3.4 are $\{0, 2k\}$ and $\{1, k+1\}$. The sum of the teams is $2k$ in both games. In each series of games, the first team always decreases by 2 and the second team always increases by 2. Therefore, the sum does not change. Hence, the sum of the teams playing against each other in the middle row of Table 3.4 is always $2k$.

In the bottom row of Table 3.4, the first game of the series is $\{2k+2, 2n-4\}$ in both non-empty columns. Summing these teams yields $(2k+2)+(2n-4) = 2k+2(n-1) \equiv 2k \pmod{n-1}$. The same argument as in the preceding paragraph leads to the conclusion that this holds for all the games in the series: the sum of the teams playing each other is equal to $2k \bmod (n-1)$.

Equality of the tournaments

Now we use this property to prove that the tournaments must be equal. Suppose that there exist three teams $t_1, t_2, t_3 \in [n-1]$, all distinct, such that

$$\begin{aligned} t_1 + t_2 &\equiv 2k \pmod{n-1} \\ t_1 + t_3 &\equiv 2k \pmod{n-1}. \end{aligned}$$

That is, for some integer p,

$$\begin{aligned} & t_1 + t_2 = t_1 + t_3 + p(n-1) \\ \iff \quad & t_2 - t_3 = p(n-1) \end{aligned} \tag{3.3}$$

To find solutions to this equation, we distinguish two cases.

Case 1: $p = 0$ This implies that $t_2 = t_3$. However, t_2 and t_3 were assumed to be distinct. This gives no solutions.

Case 2: $|p| \geq 1$ Then $|t_2 - t_3| = |p(n-1)| = |p|(n-1) \geq n-1$. However, as $1 \leq t_2, t_3 \leq n-1$, we have that $|t_2 - t_3| \leq (n-1) - 1 = n-2$. This gives no solutions.

Hence, there are no solutions to equation (3.3). Therefore, there can not exist two distinct teams t_2 and t_3 which can play against team t_1 in slot k. Thus, the opponent of team t_1 in slot k is unique, and hence, the P-permuted Kirkman tournament and the canonical 1-factorisation are equal with respect to the games in which team n does not play. We can conclude the following:

Theorem 3.3. The tournament schedule following from the canonical 1-factorisation, as given in Equation (3.1), and the P-permuted Kirkman tournament, as given in Table 3.4, are equal. Therefore, the Kirkman tournament is equivalent up to permutation to the tournament schedules following from the canonical 1-factorisation and the circle method.

Chapter 4

Dimension of the solution space of the TTP

In this chapter, we figure out the dimension of the polytope of solutions to the Travelling Tournament Problem. In Sections 4.1 and 4.2, we focus on the polytope of play vectors. In Section 4.3, this is extended to full solutions, including the travel vectors. The whole chapter applies to the unconstrained problem.

4.1 Redundant equations

Let $\mathbf{x}_n$ denote the vector of the play variables $x_{k,i,j}$ for n teams, n even. Let the set of indices of $\mathbf{x}_n$ be given by

$$A_n := \{(k,i,j) \mid k \in \{1,...,2(n-1)\},\ i,j \in \{1,...,n\},\ j \neq i\}.$$

Formally, if B and C are arbitrary sets, then B^C is the set of all functions from C to B. We slightly abuse this notation by identifying such a function with a vector which has its indices in C and its values in B. Then, let

$$X_n := \left\{\mathbf{x}_n \in \{0,1\}^{A_n} \,\middle|\, \mathbf{x}_n \text{ satisfies (2.2) and (2.3)}\right\}. \tag{4.1}$$

That is, X_n is the set of all feasible double round-robin schedules for n teams, ignoring the no-repeaters constraints (equations (2.4)) and the constraints on the length of home stands and road trips (equations (2.5) and (2.6)).

In this section, we investigate which equations are redundant. First recall equations (2.2) and (2.3):

$$\sum_{j \neq i}(x_{k,i,j} + x_{k,j,i}) = 1 \qquad \forall k, i \tag{2.2}$$

$$\sum_{k} x_{k,i,j} = 1 \qquad \forall i,\ \forall j \neq i \tag{2.3}$$

It can be shown that the equations (2.2) for one slot k are a linear combination of the other equations. We show this for slot 1, w.l.o.g., and for an arbitrary team i. In the first line, we work with the left-hand side of equations (2.2). In the second line, a part of these are rewritten to the left-hand side of

equations (2.3). Then these are replaced with the right-hand sides of equations (2.2) and (2.3), after which the linear dependence follows:

$$\begin{aligned}
\sum_{j\neq i}(x_{1,i,j}+x_{1,j,i}) &= \sum_{k=1}^{2(n-1)}\sum_{j\neq i}(x_{k,i,j}+x_{k,j,i}) - \sum_{k=2}^{2(n-1)}\sum_{j\neq i}(x_{k,i,j}+x_{k,j,i}) \\
&= \sum_{j\neq i}\left[\sum_{k=1}^{2(n-1)} x_{k,i,j}\right] + \sum_{j\neq i}\left[\sum_{k=1}^{2(n-1)} x_{k,j,i}\right] - \sum_{k=2}^{2(n-1)}\left[\sum_{j\neq i}(x_{k,i,j}+x_{k,j,i})\right] \\
&= \sum_{j\neq i} 1 + \sum_{j\neq i} 1 - \sum_{k=2}^{2(n-1)} 1 \\
&= (n-1)+(n-1)-(2(n-1)-1) \\
&= 1.
\end{aligned}$$

Hence, these equations are redundant. Now we show that the remaining equations are linearly independent.

First we show that the equations (2.2) are linearly independent for all slots $k \geq 2$. That is, the coefficient matrix of these equations has full row rank. Let the coefficient vectors corresponding to each of these equations be denoted by $\mathbf{v}_{k,i}$ and let $a_{k,i}$ be the associated scalars. So, we must show that the only solution to

$$\sum_{k\geq 2}\sum_{i} a_{k,i}\mathbf{v}_{k,i} = 0 \tag{4.2}$$

is the trivial solution $a_{k,i} = 0$ for all $k \geq 2$ and for all i.

Note that the variable $x_{k,s,t}$ only appears in the following two equations:

$$\begin{aligned}
\sum_{j\neq s}(x_{k,s,j}+x_{k,j,s}) &= 1 \\
\sum_{j\neq t}(x_{k,t,j}+x_{k,j,t}) &= 1.
\end{aligned}$$

The coefficients of $x_{k,s,t}$ are 1 in both equations and the scalars corresponding to these equations are $a_{k,s}$ and $a_{k,t}$. It follows that $a_{k,s}+a_{k,t}$ must be equal to 0 in any solution to (4.2). This holds for all slots $k \geq 2$ and for all pairs of distinct teams s and t. Then, it is easy to see that all coefficients $a_{k,i}$ must be equal to zero. Hence, these equations are linearly independent.

Next, it is easy to see that equations (2.3) cannot be written as a linear combination of the other equations, since each of these equations contains a unique variable: $x_{1,i,j}$. This one does not appear in any other equation anymore, since the equations (2.2) for slot 1 were shown to be redundant. Therefore, all remaining equations are linearly independent, and the equations (2.2) for slot 1 are the only redundant equations.

4.2 Dimension of $\mathrm{conv}(X_n)$

The set X_n of all feasible double round-robin schedules is defined by equations (2.2) and (2.3). Equations (2.2) are defined for each slot and for each team. For one slot, they are redundant. This yields a total of $(2(n-1)-1)n = n(2n-3)$ irredundant equations. Equations (2.3) are defined for every pair of two distinct teams. That gives $n(n-1)$ equations. Hence, the total number of irredundant equations equals $n(2n-3)+n(n-1) = n((2n-3)+(n-1)) = n(3n-4)$.

However, we cannot conclude yet that the dimension of the polytope $\text{conv}(X_n)$ is equal to $|\mathbf{x}_n| - n(3n-4)$, where $|\mathbf{x}_n|$ denotes the number of variables in $\mathbf{x}_n$. It could be possible that there exist irredundant valid equations for X_n which are not explicitly present in the formulation of X_n. In this section, we will show that such an equation does not exist.

This section is structured as follows: first we transform the set X_n into an isomorphic set $\overline{X}_n$, which will turn out handy later. This will be explained in the next section. Next, we sketch an outline of the proof. To complete the proof, we will have to construct a number of feasible double round-robin schedules. Therefore, we introduce an algorithm to produce such schedules, named the circle method. Using this algorithm, we construct the schedules needed for the proof. The section is concluded with a summary of the proof.

4.2.1 Transform X_n into $\overline{X}_n$

In Section 4.1, we showed that the equations (2.2) for slot 1 were redundant. These equations make the proof unnecessarily complicated. Therefore, we transform the set X_n into a new set $\overline{X}_n$ in which we remove these equations. Note that the variables $x_{1,i,j}$ now only appear in equations (2.3), a unique one in each equation. We can project out these variables, thereby transforming equations (2.3) into inequalities. This gives the following constraints:

$$\sum_{j \neq i} (x_{k,i,j} + x_{k,j,i}) = 1 \qquad \forall k \geq 2, \forall i \tag{4.3}$$

$$\sum_{k \geq 2} x_{k,i,j} \leq 1 \qquad \forall i,\ \forall j \neq i \tag{4.4}$$

Let $\overline{\mathbf{x}}_n$ denote the vector of play variables for n teams, but only for the slots $k \geq 2$. Let the corresponding set of indices of $\overline{\mathbf{x}}_n$ be given by

$$\overline{A}_n := \{(k,i,j) \mid k \in \{2, ..., 2(n-1)\},\ i,j \in \{1, ..., n\},\ j \neq i\}$$

Now, let

$$\overline{X}_n := \left\{ \overline{\mathbf{x}}_n \in \{0,1\}^{\overline{A}_n} \,\middle|\, \overline{\mathbf{x}}_n \text{ satisfies (4.3) and (4.4)} \right\}.$$

Clearly, any element $\mathbf{x}_n$ of X_n, i.e., a feasible double round-robin schedule, can be transformed (affinely) into an element of $\overline{X}_n$ by setting $\overline{x}_{k,i,j} = x_{k,i,j}$ for $k \geq 2$. Similarly, any element $\overline{\mathbf{x}}_n$ of the set $\overline{X}_n$ can be transformed back (affinely) into a feasible double round-robin schedule by setting $x_{k,i,j} = \overline{x}_{k,i,j}$ for $k \geq 2$ and $x_{1,i,j} = 1 - \sum_{k \geq 2} \overline{x}_{k,i,j}$ for all distinct i and j. Hence, X_n and $\overline{X}_n$ are isomorphic and their convex hulls have the same dimension.

4.2.2 Proof outline

In this section, we sketch an outline of the proof. The general idea is as follows: assume that some valid equation exists for $\overline{X}_n$ that is irredundant. Then derive constraints on the coefficients of the equation, to show that the equation is a linear combination of the equations we already know. This is a contradiction, showing that no additional irredundant valid equations can exist. Therefore, the dimension of $\text{conv}(\overline{X}_n)$ is equal to $|\overline{\mathbf{x}}_n|$ minus the number of (linearly independent) equations in the definition of $\overline{X}_n$. As X_n and $\overline{X}_n$ are isomorphic, the dimensions of their convex hulls agree.

Now we develop this idea in more detail. Let $\mathbf{a}^\mathsf{T} \overline{\mathbf{x}}_n = \beta$ be an arbitrary valid equation for $\overline{X}_n$. We will derive constraints on the coefficient vector $\mathbf{a}$ to show that this valid equation is a linear combination of the equations (4.3) that we already know. These constraints have the form $\mathbf{b}_\ell \mathbf{a} = 0$, where $\mathbf{b}_\ell$ is a row vector. The vectors $\mathbf{b}_\ell$ will be called constraint vectors and are collected in a matrix $\mathbf{B}$, one in each row. Thus, any coefficient vector $\mathbf{a}$ must satisfy $\mathbf{B}\mathbf{a} = \mathbf{0}$.

We write the equations (4.3) in matrix form: $\mathbf{K}\overline{\mathbf{x}}_n = \mathbf{1}$. Here, the rows of $\mathbf{K}$ are the coefficients of the **K**nown equations (4.3). Now, we want to show that any solution $\mathbf{a}$ to $\mathbf{Ba} = \mathbf{0}$ is a linear combination of the rows of $\mathbf{K}$. In other words, we want to show that the null space of $\mathbf{B}$, null($\mathbf{B}$), is a subspace of the row space of $\mathbf{K}$, row($\mathbf{K}$): null($\mathbf{B}$) $\subseteq$ row($\mathbf{K}$).

As each row $\mathbf{k}_\ell$ of $\mathbf{K}$ contains the coefficients of a valid equation for $\overline{X}_n$, we know that $\mathbf{B}\mathbf{k}_\ell = \mathbf{0}$ holds for all ℓ. Hence, row($\mathbf{K}$) $\subseteq$ null($\mathbf{B}$). Therefore, to show that null($\mathbf{B}$) $\subseteq$ row($\mathbf{K}$) also holds, it is sufficient to prove that row($\mathbf{K}$) and null($\mathbf{B}$) have the same dimension. That is, the rows of $\mathbf{B}$ must contain $|\overline{\mathbf{x}}_n| - \text{rank}(\mathbf{K})$ linearly independent constraint vectors $\mathbf{b}_\ell$.

We already proved that the rows of $\mathbf{K}$ are linearly independent. Thus, rank($\mathbf{K}$) $= n(2n-3)$. This yields

$$\begin{aligned} |\overline{\mathbf{x}}_n| - \text{rank}(\mathbf{K}) &= n(n-1)(2n-3) - n(2n-3) \\ &= n((n-1)-1)(2n-3) \\ &= n(n-2)(2n-3) \end{aligned}$$

So, we have to find $n(n-2)(2n-3)$ linearly independent constraint vectors $\mathbf{b}_l$. We do this as follows: let $\overline{\mathbf{x}}_n, \overline{\mathbf{z}}_n \in \overline{X}_n$ be two feasible solution vectors to the transformed problem. Then, $\mathbf{a}^\mathsf{T}\overline{\mathbf{x}}_n = \beta$ and $\mathbf{a}^\mathsf{T}\overline{\mathbf{z}}_n = \beta$ hold. Hence, $\mathbf{a}^\mathsf{T}\overline{\mathbf{x}}_n = \mathbf{a}^\mathsf{T}\overline{\mathbf{z}}_n$ or $\mathbf{a}^\mathsf{T}(\overline{\mathbf{z}}_n - \overline{\mathbf{x}}_n) = 0$, and we have that $(\overline{\mathbf{z}}_n - \overline{\mathbf{x}}_n)^\mathsf{T}$ is a valid constraint vector that can be added to the matrix $\mathbf{B}$.

In order to find $n(n-2)(2n-3)$ linearly independent constraint vectors $(\overline{\mathbf{z}}_n - \overline{\mathbf{x}}_n)^\mathsf{T}$, we try to construct sparse constraint vectors. This makes it easier to show that the vectors are linearly independent. These sparse vectors are generated as follows: a vector $\overline{\mathbf{x}}_n$ is chosen, and the vector $\overline{\mathbf{z}}_n$ is constructed from $\overline{\mathbf{x}}_n$ by applying a local change to the schedule which keeps the vector feasible. Examples of such local changes are swapping the venues of the games between two teams, or swapping all games of two slots.

This completes the plan of the proof. Now we must generate the constraint vectors. Therefore, we will have to construct feasible double round-robin schedules. We use the circle method, as described in Chapter 3, to generate single round-robin schedules. To generate a double round-robin schedule, two single round-robin schedules are concatenated. Now, we will look into two types of local changes to generate the constraint vectors.

4.2.3 Local change: home-away swap

The first local change we consider is swapping the venues of the games between two teams, called a 'home-away swap'. That is, two teams i and j play each other twice, once home and once away. The home game becomes the away game and vice versa for each team. Clearly, if the vector $\overline{\mathbf{x}}_n$ is a feasible vector, then applying a home-away swap to this vector again yields a feasible vector $\overline{\mathbf{z}}_n$. Because the vectors $\overline{\mathbf{x}}_n$ and $\overline{\mathbf{z}}_n$ are very similar, the difference vector is sparse.

At this point, we can finally exploit the properties of the transformed set $\overline{X}_n$. Suppose we apply a home-away swap to two teams of which one of the two games between them is played in the first slot. Then the difference of the vectors $\overline{\mathbf{x}}_n$ and $\overline{\mathbf{z}}_n$ becomes even sparser than when we would have used the vectors $\mathbf{x}_n$ and $\mathbf{z}_n$ of the original set X_n.

Now we will describe the home-away swap in detail, first for the original set X_n. Suppose team i and team j play each other in slot 1 at team i's venue and in slot $k \geq 2$ at team j's venue. That is, $x_{1,i,j} = 1$ and $x_{k,j,i} = 1$. After the swap, we obtain the vector $\mathbf{z}_n$ which differs from $\mathbf{x}_n$ in the following entries: $z_{1,i,j} = 0$, $z_{1,j,i} = 1$, $z_{k,i,j} = 1$ and $z_{k,j,i} = 0$. Therefore, the only nonzero entries in the difference vector $\mathbf{c} = (\mathbf{z}_n - \mathbf{x}_n)^\mathsf{T}$ are $c_{1,i,j} = -1$, $c_{1,j,i} = 1$, $c_{k,i,j} = 1$, and $c_{k,j,i} = -1$. The vectors $\overline{\mathbf{x}}_n$ and $\overline{\mathbf{z}}_n$ of the transformed problem do not have an entry for the first slot, and hence, the difference vector $\mathbf{b} = (\overline{\mathbf{z}}_n - \overline{\mathbf{x}}_n)^\mathsf{T}$ only has two nonzero entries: $b_{k,i,j} = 1$ and $b_{k,j,i} = -1$.

Of course, we still have to prove that there exists a feasible schedule in which teams i and j play each other in slot 1 and in slot k. This is, however, trivial: we can use the circle method, as described in Section 3.1.1, to generate a (single) round-robin schedule. We then duplicate this schedule, and randomly choose which of the two games between two teams is played at home and away. Next, we can swap all games in two slots to obtain another feasible schedule. So, we can simply swap the slots such that one of the games between team i and team j is played in slot 1 and the other one in slot k.

It is easy to see that the set of constraint vectors generated by performing home-away swaps for every slot $k \geq 2$ and for every unique pair of distinct teams i and j is linearly independent. After all, the value of the variable $b_{k,i,j}$ is nonzero only in the constraint concerning the home-away swap of teams i and j in slots 1 and k. We select those pairs i and j for which $i < j$. The number of these unique pairs of distinct teams equals $n(n-1)/2$ and the number of slots except the first slot is $2n-3$. Thus, we already have $n(n-1)(2n-3)/2$ linearly independent constraint vectors. These constraint vectors are collected in the matrix $\mathbf{B}$.

The constraint vectors are similar for the different slots k and slot 1 between which you perform a home-away swap. For $n = 4$ and an arbitrary slot $k \geq 2$, the interesting entries of the constraint vectors are shown in the matrix in Figure 4.1. The other entries of these vectors are zero.

$x_{k,1,2}$	$x_{k,1,3}$	$x_{k,1,4}$	$x_{k,2,1}$	$x_{k,2,3}$	$x_{k,2,4}$	$x_{k,3,1}$	$x_{k,3,2}$	$x_{k,3,4}$	$x_{k,4,1}$	$x_{k,4,2}$	$x_{k,4,3}$
1	0	0	−1	0	0	0	0	0	0	0	0
0	1	0	0	0	0	−1	0	0	0	0	0
0	0	1	0	0	0	0	0	0	−1	0	0
0	0	0	0	1	0	0	−1	0	0	0	0
0	0	0	0	0	1	0	0	0	0	−1	0
0	0	0	0	0	0	0	0	1	0	0	−1

Figure 4.1: The nonzero entries of the constraint vectors generated by performing a home-away swap between slot 1 and slot k for $n = 4$. The entries coloured in red highlight the row echelon form. The blue colour highlights the other nonzero entries.

Note that the variables in Figure 4.1 are lexicographically ordered. That is, if two variables are indexed by (k, i, j) and (k', i', j'), then $(k, i, j) \leq (k', i', j')$ if and only if $k < k'$ or ($k = k'$ and $i < i'$) or ($k = k'$ and $i = i'$ and $j \leq j'$). Furthermore, we ordered the rows such that the matrix is in row echelon form. We do the same for the whole matrix $\mathbf{B}$: the variables are also lexicographically ordered, and the rows are ordered such that $\mathbf{B}$ is in row echelon form. This is possible because each pivot is in a unique column.

Now we switch to another type of local change, the partial slot swap, to find the remaining constraint vectors.

4.2.4 Local change: partial slot swap

With the home-away swaps, we found $n(n-1)(2n-3)/2$ linearly independent constraint vectors. The number of independent constraint vectors still needed then equals:

$$\begin{aligned} n(n-2)(2n-3) - \frac{n(n-1)(2n-3)}{2} &= n(2n-3)\left((n-2) - \frac{n-1}{2}\right) \\ &= n(2n-3)\frac{2(n-2)-(n-1)}{2} \\ &= n(2n-3)\frac{n-3}{2} \end{aligned}$$

To find these vectors, we consider another local change, called the 'partial slot swap'. Suppose two games are played in the first slot by four teams, and the same four teams play another two games in another slot $k \geq 2$. Then we can swap the games of these four teams between slot 1 and slot k. An example of such a swap is displayed graphically in Figure 4.2.

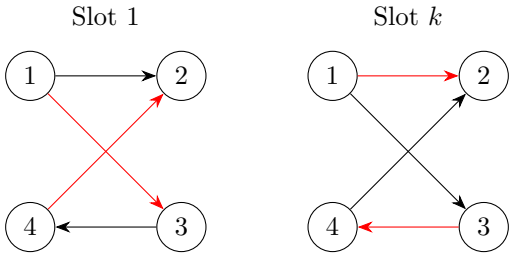


Figure 4.2: Example of a partial slot swap. In black, the original games. In red, the games after the swap. The arrow heads indicate the venue of the games.

The above description of the partial slot swap is a little inaccurate. According to this description, Figure 4.3 also depicts a valid partial slot swap. However, this is actually a double home-away swap, and therefore, this type is not considered a partial slot swap.

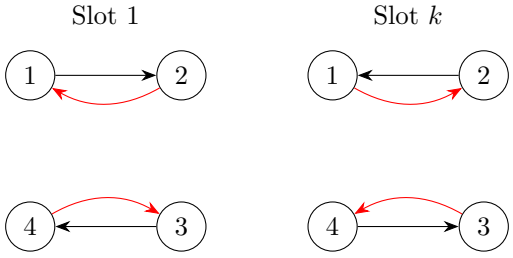


Figure 4.3: Example of a double home-away swap. This is not regarded a partial slot swap.

We now describe this swap mathematically, first for the vectors in the original set X_n. Suppose teams i, j, s, and t, all distinct, are involved in a partial slot swap. Team i and j play at team i's venue in slot 1 and team s and t play at team s's venue in slot 1: $x_{1,i,j} = 1$ and $x_{1,s,t} = 1$. In slot k, team i' plays home against team j' and team s' plays home against team t': $x_{k,i',j'} = 1$ and $x_{k,s',t'} = 1$. Here, the set $\{i', j', s', t'\}$ is equal to the set $\{i, j, s, t\}$. Now, we swap these games in slot 1 and k to obtain a new vector $\mathbf{z}_n$. The changed entries in $\mathbf{z}_n$ compared to $\mathbf{x}_n$ are $z_{1,i,j} = 0$, $z_{1,s,t} = 0$, $z_{1,i',j'} = 1$, $z_{1,s',t'} = 1$, $z_{k,i,j} = 1$, $z_{k,s,t} = 1$, $z_{k,i',j'} = 0$, and $z_{k,s',t'} = 0$.

The vectors in the transformed set $\overline{X}_n$ do not have an entry for the first slot, so that the difference vector $\mathbf{b} = (\overline{\mathbf{z}}_n - \overline{\mathbf{x}}_n)^\mathsf{T}$ has only four nonzero entries: $b_{k,i,j} = 1$, $b_{k,s,t} = 1$, $b_{k,i',j'} = -1$, and $b_{k,s',t'} = -1$. With this type of constraint vectors, it is possible to construct the remaining $n(2n-3)(n-3)/2$ linearly independent constraint vectors. The proof consists of two parts. Of course, we cannot perform a partial slot swap if there does not exist a schedule in which $x_{1,i,j} = 1$, $x_{1,s,t} = 1$, $x_{k,i',j'} = 1$, and $x_{k,s',t'} = 1$. Therefore, we will first show that for every possible partial slot swap, there exists a feasible schedule on which this swap can be performed. Next, we will show how we can select the right swaps to obtain $n(2n-3)(n-3)/2$ additional linearly independent constraint vectors.

Existence of schedules

Again, we use the circle method twice to obtain two feasible single round-robin schedules. Because we can specify the games in the first slot as we like, we can simply choose the first single round-robin

schedule to contain the games (i, j) and (s, t) in the first slot, and the second schedule to contain the games (i', j') and (s', t') in the first slot. Next, we concatenate these two schedules to obtain a double round-robin schedule and swap the slot containing the games (i', j') and (s', t') with slot k. This clearly yields a feasible double round-robin schedule with the right properties for a partial slot swap.

Selecting partial slot swaps

Now, we must select partial slot swaps to obtain the remaining linearly independent constraint vectors. We do this by only choosing those constraint vectors which can be added to $\mathbf{B}$ while maintaining the row echelon form. This is done by only adding a constraint vector $\mathbf{b}$ to $\mathbf{B}$ if its pivot entry corresponds to a free variable in $\mathbf{B}$. Then, this vector can be inserted into $\mathbf{B}$ such that the matrix remains in row echelon form. This is repeated until we have added $n(2n-3)(n-3)/2$ vectors. Because the rows of $\mathbf{B}$ remain linearly independent by construction during this process, this ensures that we have found enough linearly independent constraint vectors.

So, we have to find $n(2n-3)(n-3)/2$ partial slot swaps of which the first nonzero entry corresponds to a unique free variable in $\mathbf{B}$. Note that the constraint vectors obtained by the home-away swaps and the partial slot swaps are similar for the swaps between the different slots k and slot 1. Therefore, we can focus on the swaps concerning one particular slot k, so that we have to find $n(n-3)/2$ additional constraint vectors for this slot.

The constraint vectors originating from the home-away swaps have their pivots in the columns corresponding to those variables $x_{k,i,j}$ for which $i < j$. Hence, the free variables in $\mathbf{B}$ are those variables $x_{k,i,j}$ for which $i > j$. Thus, for a given team $i \in [n]$, there are $i-1$ free variables for each slot k: $x_{k,i,1}$ up to $x_{k,i,i-1}$. Then the number of free variables from $i = 2$ to $i = n-1$ for one slot k equals

$$\sum_{i=2}^{n-1}(i-1) = \sum_{i=1}^{n-2} i = \frac{(n-2)(n-1)}{2} = \frac{n^2-3n+2}{2}$$
$$= \frac{n^2-3n}{2} + 1 \quad = \frac{n(n-3)}{2} + 1.$$

This is exactly one more than the number of vectors we need: $n(n-3)/2$. Thus, if we can find constraint vectors of which the set of indices of the pivot entries is equal to the set

$$S_k = \{(k, i, j) \mid i \in \{2, ..., n-1\},\, j < i,\, (i, j) \neq (n-1, n-2)\}, \tag{4.5}$$

then we are done. This is possible, as shown in Algorithm 1.

It can easily be verified that the games (s, t), (i', j'), and (s', t') are all valid and lexicographically larger than (i, j). Furthermore, i, j, s, and t are all distinct and $\{i, j, s, t\} = \{i', j', s', t'\}$. Therefore, the constraint vector $\mathbf{b}$ that is returned by the algorithm is indeed a feasible constraint vector derived from the partial slot swap of the games (i, j) and (s, t) in slot 1 with the games (i', j') and (s', t') in slot k.

Applying this algorithm to each of the tuples of indices in the set S_k (4.5) yields $n(n-3)/2$ constraint vectors, all linearly independent from the other row vectors in $\mathbf{B}$ because the pivot entry in each of the constraint vectors corresponds to a unique free variable. Then the new constraint vectors can be inserted into the rows of $\mathbf{B}$ such that $\mathbf{B}$ remains in row echelon form. This is done for each slot k greater than 1.

To illustrate this process, we extend Figure 4.1 to include the (nonzero entries of the) constraint vectors derived from partial slot swaps between slot 1 and slot k. This is shown in Figure 4.4.

The matrix contains two new rows. The first new row corresponds to the 'else' clause of Algorithm 1, the last one to the 'if' clause. Note that the matrix is indeed in row echelon form.

Algorithm 1: Construct partial slot swap vector

Input : Number of teams n, slot k, index (i, j) with $j < i < n-1$ or $(i = n-1$ and $j < n-2)$

Output: Constraint vector $\mathbf{b}$ following from the valid partial slot swap between games (i, j) and (s, t) in slot 1 and games (i', j') and (s', t') in slot k, such that (s, t), (i', j') and (s', t') are all lexicographically larger than (i, j)

if $j < i - 1$ **then**
  $(s, t) \leftarrow (i + 1, j + 1)$
  $(i', j') \leftarrow (i, j + 1)$
  $(s', t') \leftarrow (i + 1, j)$
else
  $(s, t) \leftarrow (i + 1, i + 2)$
  $(i', j') \leftarrow (i, i + 2)$
  $(s', t') \leftarrow (i + 1, j)$
end
$\mathbf{b} \leftarrow \mathbf{0}_{|\overline{\mathbf{x}}_n|}$
$b_{k,i,j}$, $b_{k,s,t} \leftarrow 1$
$b_{k,i',j'}$, $b_{k,s',t'} \leftarrow -1$
return b

$x_{k,1,2}$	$x_{k,1,3}$	$x_{k,1,4}$	$x_{k,2,1}$	$x_{k,2,3}$	$x_{k,2,4}$	$x_{k,3,1}$	$x_{k,3,2}$	$x_{k,3,4}$	$x_{k,4,1}$	$x_{k,4,2}$	$x_{k,4,3}$
1	0	0	−1	0	0	0	0	0	0	0	0
0	1	0	0	0	0	−1	0	0	0	0	0
0	0	1	0	0	0	0	0	0	−1	0	0
0	0	0	1	0	−1	−1	0	1	0	0	0
0	0	0	0	1	0	0	−1	0	0	0	0
0	0	0	0	0	1	0	0	0	0	−1	0
0	0	0	0	0	0	1	−1	0	−1	1	0
0	0	0	0	0	0	0	0	1	0	0	−1

Figure 4.4: The matrix in Figure 4.1 augmented with the constraint vectors from the partial slot swaps. The rows with the green pivots are new.

The proof is now completed. In the next section, the proof will be summarised.

4.2.5 Conclusion

In Section 4.2.2, we sketched an outline of the proof that the dimension of $\text{conv}(\overline{X}_n)$ equals $|\overline{\mathbf{x}}_n|$ minus the rank of $\mathbf{K}$. The outlined steps were followed in the subsequent sections. We summarise them shortly. We found $n(n-2)(2n-3)$ linearly independent constraint vectors in total, that were collected in the matrix $\mathbf{B}$. This number is equal to the rank of $\mathbf{K}$, and hence, the null space of $\mathbf{B}$ is equal to the row space of $\mathbf{K}$. Therefore, there do not exist additional valid equations for $\overline{X}_n$ that are not explicitly present in its formulation. Thus, the dimension of the convex hull of $\overline{X}_n$ is equal to the number of variables minus the rank of $\mathbf{K}$:

$$\begin{aligned}\dim(\text{conv}(\overline{X}_n)) &= |\overline{\mathbf{x}}_n| - \text{rank}(\mathbf{K}) \\ &= |\overline{\mathbf{x}}_n| - \text{nullity}(\mathbf{B}) \\ &= \text{rank}(\mathbf{B}) \\ &= n(n-2)(2n-3).\end{aligned}$$

Now, we conclude the proof with a theorem.

Theorem 4.1. The dimension of $\text{conv}(\overline{X}_n)$ is equal to $n(n-2)(2n-3)$. As $\overline{X}_n$ and X_n are isomorphic, the same holds for $\text{conv}(X_n)$.

4.3 Dimension of $\text{conv}(Y_n)$

In the previous section, the dimension of the convex hull of X_n was demonstrated. However, in this set, only the play variables $x_{k,i,j}$ were considered. In this section, the result is extended to the travel variables $y_{i,s,t}$. First we define the set Y_n. Next, an outline of the proof is sketched, after which the proof follows.

4.3.1 Definition of Y_n

Let $\mathbf{y}_n$ denote the vector of the travel variables $y_{i,s,t}$ for n teams, n even. Let the set of indices of $\mathbf{y}_n$ be given by

$$B_n := \{(i,s,t) \mid i,s,t \in \{1,...,n\},\, t \neq s\}.$$

Then, let

$$Y_n := \{(\mathbf{x}_n, \mathbf{y}_n) \in \{0,1\}^{A_n} \times \{0,1\}^{B_n} \mid \mathbf{x}_n \in X_n,\, \mathbf{y}_n \text{ satisfies } (2.7), (2.8), (2.9), (2.10), (2.11)\}. \quad (4.6)$$

That is, an element of the set Y_n is a tournament schedule (the x-values), together with values for the travel variables (the y-values). However, note that the travel variables are only constrained by inequalities of the type 'greater or equal than'. When solving the integer program, the costs are minimised and hence, the values of the travel variables become as small as possible. Then they reflect the real travel movements of the teams. However, this is not necessarily the case for the elements of Y_n; a travel variable $y_{i,s,t}$ can be equal to 1 even though team i never travels from s to t.

4.3.2 Proof outline

We now investigate the dimension of the convex hull of this set Y_n. The number of equations in the definitions of X_n and Y_n is equal; only inequalities were added. We want to show that the dimension of $\text{conv}(Y_n)$ is equal to the dimension of $\text{conv}(X_n)$ plus the number of elements of $\mathbf{y}_n$. This is done in the same way as for the set X_n: we assume that some valid equation exists, and then show that this valid equation must be a linear combination of the known equations.

So, let $\mathbf{a}^\mathsf{T}\mathbf{x}_n + \mathbf{b}^\mathsf{T}\mathbf{y}_n = \gamma$ be an arbitrary valid equation for Y_n. We derive constraints on the coefficients $\mathbf{b}$ by applying local changes to elements of the set Y_n such that the modified elements are in Y_n as well. However, constructing explicit elements of Y_n is rather laborious. We can avoid doing so by using a well-known property of double round-robin schedules, that uses the concept of a break. If a team plays either home or away twice in a row, this is called a break. It is known that in any double round-robin schedule on n teams, at least $2n-2$ breaks occur [15]. Thus, any double round-robin schedule on n teams, $n \geq 4$, contains at least two breaks.

We call a break that occurs when a team plays two games either home or away in a row a 'home break' or 'away break', respectively. It is easy to see that the number of home and away breaks in any double round-robin schedule must be equal—otherwise, the number of teams playing home and away is not equal in some slot, which is clearly impossible. This implies that in each double round-robin schedule on n teams, $n \geq 4$, at least one away break occurs. That is, team i plays consecutively away at team s and at team t for some $i,s,t \in [n]$, all distinct. From this, we can derive a couple of things.

First we address the variables of the type $y_{i,i,t}$, then $y_{i,t,i}$ and finally $y_{i,s,t}$. Here, i, s and t are all distinct.

4.3.3 Variables of the type $y_{i,i,t}$

First, as team i plays away at team t's venue only once and travels there from team s, team i never travels directly from i (home) to t. There are two constraints on the variable $y_{i,i,t}$: (2.8) and (2.10). As team i never travels directly from i to t, constraint (2.8) does not restrict the value of $y_{i,i,t}$. The same holds for constraint (2.10), as team i does not play its first game against team t; the game against team s precedes it. Thus, $y_{i,i,t}$ can take both the values 0 and 1.

We take advantage of this by picking a random element of Y_n, choosing i, s and t such that i plays consecutively at team s and at team t (which is possible as argued in the previous section), and creating two elements of Y_n by setting the value of $y_{i,i,t}$ to 0 and 1. We name these two elements $(\mathbf{x}_n, \mathbf{y}_n)$ and $(\mathbf{x}'_n, \mathbf{y}'_n)$, respectively. As they are elements of Y_n, we know that $\mathbf{a}^\mathsf{T}\mathbf{x}_n + \mathbf{b}^\mathsf{T}\mathbf{y}_n = \gamma$ and $\mathbf{a}^\mathsf{T}\mathbf{x}'_n + \mathbf{b}^\mathsf{T}\mathbf{y}'_n = \gamma$ hold. Also, $\mathbf{x}_n$ and $\mathbf{x}'_n$ are equal. Hence,

$$\begin{aligned} 0 &= \gamma - \gamma \\ &= (\mathbf{a}^\mathsf{T}\mathbf{x}'_n + \mathbf{b}^\mathsf{T}\mathbf{y}'_n) - (\mathbf{a}^\mathsf{T}\mathbf{x}_n + \mathbf{b}^\mathsf{T}\mathbf{y}_n) \\ &= \mathbf{b}^\mathsf{T}(\mathbf{y}'_n - \mathbf{y}_n). \end{aligned}$$

Because the difference vector $\mathbf{c} = \mathbf{y}'_n - \mathbf{y}_n$ has only one nonzero entry, namely $c_{i,i,t} = 1$, this implies that $b_{i,i,t} = 0$. By permuting the teams, this argument can be repeated for all $i, t \in [n]$, $i \neq t$. Thus, $b_{i,i,t} = 0$ for all distinct teams i and t.

4.3.4 Variables of the type $y_{i,t,i}$

In this part, we swap s and t: we assume that team i plays consecutively away first at team t and then at team s. Given that team i travels from team t to team s, we can conclude that team i never travels directly from team t back to team i (home). Thus, constraint (2.9) does not restrict the value of $y_{i,t,i}$. The same holds for constraint (2.11), as team i does not play its last game against team t; the game against team s succeeds it. Thus, $y_{i,t,i}$ can take both the values 0 and 1.

We again pick a random element of Y_n and construct two elements from it, one with value 0 and one with value 1 for the variable $y_{i,t,i}$. This gives two elements of Y_n which differ only in this variable. Thus, using the same technique as in the previous section, we can conclude that the coefficient $b_{i,t,i}$ of the valid equation must be equal to zero. By permuting the teams, the same follows for all distinct teams i and t.

4.3.5 Variables of the type $y_{i,s,t}$

In this part, we rename t to j. Thus, team i plays consecutively away first at team s and then at team j. We let t be a team distinct from i, j and s. In total, we have 4 distinct teams now, which is possible as $n \geq 4$. As team i travels from team s to team j, team i never travels directly from team s to team t. Thus, constraint (2.7) does not restrict the value of $y_{i,s,t}$. There are no other constraints which apply to $y_{i,s,t}$. Thus, $y_{i,s,t}$ can take both the values 0 and 1. Now by following the same approach as in the previous two sections, we can conclude that the coefficient $b_{i,s,t}$ of the valid equation must be equal to zero for all distinct teams i, s and t.

4.3.6 Conclusion

In the previous three sections, we showed that in any valid equation $\mathbf{a}^\mathsf{T}\mathbf{x}_n + \mathbf{b}^\mathsf{T}\mathbf{y}_n = \gamma$ for Y_n, the values $b_{i,i,t}$, $b_{i,t,i}$ and $b_{i,s,t}$ must all be equal to zero for i, s and t distinct. Together, this implies that

$\mathbf{b} = \mathbf{0}$. Thus, any valid equation for Y_n must be of the type $\mathbf{a}^\mathsf{T}\mathbf{x}_n = \gamma$.

Now, any valid equation of the type $\mathbf{a}^\mathsf{T}\mathbf{x}_n = \gamma$ for Y_n is automatically also a valid equation for X_n, as this equation only concerns the x-variables and X_n and Y_n have the same constraints applying to the x-variables. The reverse is also true: as Y_n is an extension of X_n in which no extra constraints are added to the original variables, any valid equation for X_n is also a valid equation for Y_n. Thus, the valid equations of X_n and Y_n of the type $\mathbf{a}^\mathsf{T}\mathbf{x}_n = \gamma$ are equal for both sets.

In Section 4.2, we already investigated valid equations of the type $\mathbf{a}^\mathsf{T}\mathbf{x}_n = \gamma$ and concluded that no additional valid equations exist apart from those already known. Thus, the same applies to Y_n. Also, no valid equations with nonzero coefficient vector $\mathbf{b}$ exist. Hence, we can conclude that the dimension of the convex hull of Y_n is equal to the dimension of the convex hull of X_n plus the number of elements of $\mathbf{y}_n$:

$$\begin{aligned}\dim(\text{conv}(Y_n)) &= \dim(\text{conv}(X_n)) + |\mathbf{y}_n| \\ &= n(n-2)(2n-3) + n^2(n-1) \\ &= n(3n^2 - 8n + 6).\end{aligned}$$

We conclude the section with a corollary.

Corollary 4.1.1. The dimension of $\text{conv}(Y_n)$ is equal to $n(3n^2 - 8n + 6)$. This is equal to the dimension of $\text{conv}(X_n)$ plus the number of elements in $\mathbf{y}_n$.

Chapter 5

Facets

The hardness of the Travelling Tournament Problem is at least partially caused by the weak linear relaxation, leading to a large integrality gap. By adding cutting planes, we can reduce the gap. In this chapter, we focus on a specific class of cutting planes: facets. First, the integrality gap is discussed. Next, using specialised software, a class of facets is identified that reduces the integrality gap. However, as the software can only be applied to a finite number of instances, we need to prove mathematically that the facets found by the software are indeed facets for all instances. The chapter is concluded with this proof, which is, as in Chapter 4, for the unconstrained problem.

5.1 Integrality gap

To test the effect of the addition of facets, the set of National League (NL) instances is used. The National League is a baseball league in the United States and Canada which is part of the Major League Baseball (MLB), where the TTP has its origin. The National League consists of 16 teams, but smaller instances are created by taking a subset of the teams. This way, the instances NL4 up to NL16 are created, one for each even number between 4 and 16 [16]. To give an idea, the data of the NL4 instance can be found in Table 5.1. The data of the other instances are accessible on [3]. Here, also the current best upper and lower bounds can be found. At the moment of writing, the instances NL12, NL14, and NL16 have not yet been solved.

Team	ATL	NYM	PHI	MON
ATL	0	745	665	929
NYM	745	0	80	337
PHI	665	80	0	380
MON	929	337	380	0

Table 5.1: The distances between the teams of the NL4 instance.
Note that the distances are symmetric.

First, we investigate the integrality gap. We use the standard problem for this, because the solution values are known for this problem, but not for the unconstrained problem. The software Gurobi [17] was used to compute the optimal solution to the linear relaxation. The integrality gap of the instances NL4 up to NL12 can be found in Table 5.2, and is large for all instances.

Now we should add facets to reduce the integrality gap. In the next section, we use specialised software to identify facets.

	NL4	NL6	NL8	NL10	NL12-l	NL12-u
IP value	8276	23916	39721	59436	108629	110729
LP value	2004	2186	2686	2980	4736	4736
Int. gap	4.13	10.9	14.8	19.9	22.9	23.4

Table 5.2: In the first two body rows, the objective values of the optimal solutions to the integer program and its linear relaxation. In the last row, the integrality gap. For the NL12 instance, two values are shown: for the best known lower bound (NL12-l) and upper bound (NL12-u).

5.2 Finding facets

The software library IPO (Investigating Polyhedra by Oracles, [18]), the theory and details of which are described in [19], can be used to compute facets of a polytope. IPO was applied to the standard version of the NL4 instance, including the no-repeaters constraints and the constraints on the length of home stands and road trips; see Table 5.1 for the data. Let P be the convex hull of the set of solutions of the NL4 instance. To be precise, we try to find facets of the polytope P.

A screenshot of the output of IPO is displayed in Figure 5.1. In short, IPO works as follows. First, the optimal solution to the LP relaxation of the integer program (IP) is computed. Next, a facet or equation of the polytope P is computed which separates the current solution to the LP relaxation from P. This facet or equation is added to the IP, after which this procedure is repeated.

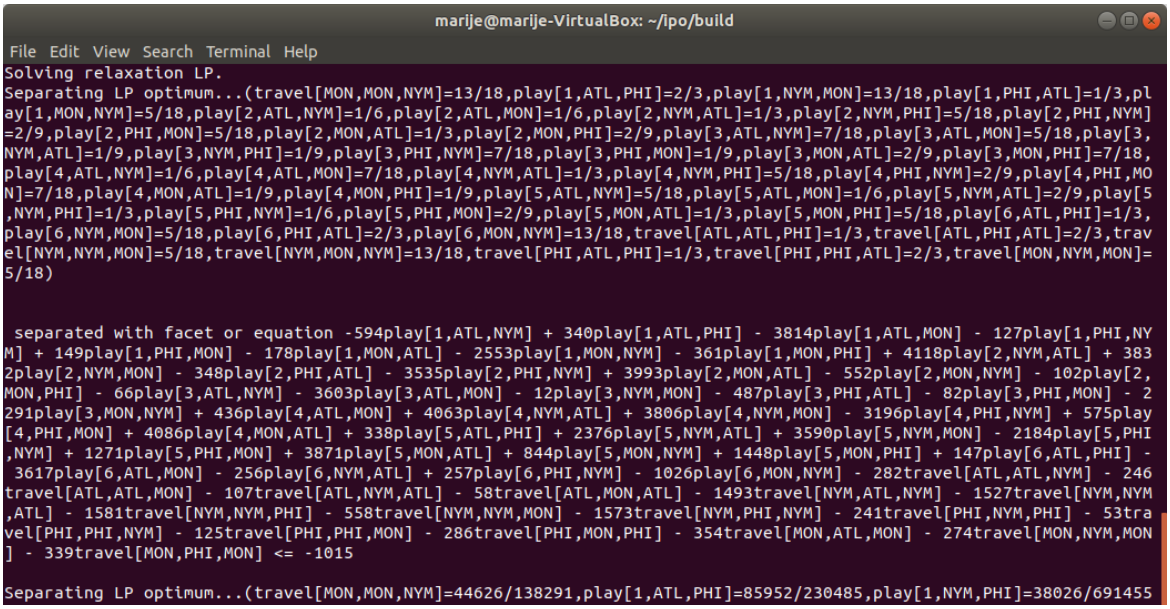


Figure 5.1: Output of IPO when applied to the NL4 instance

Most facets that are found by IPO appear to be specific to the instance to which it is applied, like the facet displayed in Figure 5.1, and are therefore unusable in other instances. However, a small number of the facets that were found did have a physical interpretation and are applicable to other instances. An example of such a facet is

```
play[2,PHI,NYM] + play[3,NYM,PHI] + play[3,PHI,NYM] + play[4,PHI,NYM] <= 1.
```

We can interpret this as follows: if team NYM plays home against team PHI in slot 3, then team NYM cannot play away against team PHI in slot 2, 3 or 4. This follows logically from the fact that team NYM can neither play two games in the same slot, nor can it play against team PHI in slot 2 or 4 because of the no-repeaters constraints. (Note that IPO was applied to the standard problem, including these constraints.)

However, addition of all the facets of this type does not lead to a reduction of the integrality gap. The same holds for most of the other classes of facets that were found with IPO. It seems that the large integrality gap is caused mainly by the weak coupling between the play and the travel variables, which leads to low values of the travel variables. Hence, it is not surprising that facets in which only the play variables appear are not effective to reduce the integrality gap.

One class of facets found with IPO involves only travel variables. An example of such a facet is

```
travel[MON,NYM,ATL] + travel[MON,NYM,PHI] + travel[MON,NYM,MON] >= 1.
```

The interpretation is that team MON must travel to one of the other teams from team NYM. This follows logically from the fact that team MON starts and ends the competition at home, and plays away against team NYM once during the competition. Thus, after this away game has been played, team MON cannot stay at team NYM's venue.

Of course, this does not only hold for leaving a venue, but also for travelling to a venue. Each team should travel to and leave each of the other teams' venues exactly once (inequality: at least once). Thus, we can describe this class of facets as the "flow conservation facets". A special case is leaving and returning to the home venue. In the unconstrained problem, this must happen at least once. In the standard problem, the number of times the home venue must be left depends on the problem size. Because of the constraints on the length of a home stand and road trip, at most 3, one cannot visit all 5 venues of the other teams in a 6-team instance without leaving and returning to the home venue at least twice. Similarly, for the NL8 instance, this number is at least 3, for the NL10 instance at least 3 as well, and for the NL4 instance at least 4.

Now, we add all these facets to the instances NL4 up to NL12 and compute the objective values of the optimal solutions to the linear relaxations. The standard problem is used for this. The results are given in Table 5.3.

	NL4	NL6	NL8	NL10	NL12-l	NL12-u
IP value	8276	23916	39721	59436	108629	110729
LP value	8016	16638	29889	38572	78047	78047
Int. gap	1.03	1.44	1.33	1.54	1.42	1.39

Table 5.3: The same type of data as in Table 5.2, but now, the flow conservation facets have been added. This leads to a lower integrality gap.

The integrality gap has decreased significantly for all instances. Moreover, the gap does not increase monotonically anymore with the size of the instance.

Because this is the only class of facets found with IPO that has significant effects on the integrality gap, the remainder of this chapter focuses on the flow conservation facets. The software was only applied to the NL4 instance. Thus, a mathematical proof that the valid inequalities found with IPO are indeed facet-defining for all instances is lacking at this moment. In the next section, this proof is given. As the proof is for the unconstrained problem, all valid inequalities are of the type "≥ 1".

5.3 Facet proof

In this section, the proof that the flow conservation facets found with IPO are indeed facets of the unconstrained problem will be given. One special case remains unproven; only computational results are given for this case. First we define precisely the statement to be proved.

Let $P_n = \text{conv}(X_n)$ and $Q_n = \text{conv}(Y_n)$. We want to prove that the following inequalities are facet-defining for Q_n:

$$\sum_{s\in[n]\setminus\{t\}} y_{i,s,t} \geq 1 \quad \forall i,t$$
$$\sum_{s\in[n]\setminus\{t\}} y_{i,t,s} \geq 1 \quad \forall i,t,$$

or equivalently, that the following faces are facets:

$$\{(\mathbf{x}_n,\mathbf{y}_n) \in Q_n \mid \sum_{s\in[n]\setminus\{t\}} y_{i,s,t} = 1\} \quad \forall i,t$$
$$\{(\mathbf{x}_n,\mathbf{y}_n) \in Q_n \mid \sum_{s\in[n]\setminus\{t\}} y_{i,t,s} = 1\} \quad \forall i,t.$$

By symmetry, it is sufficient to prove this for only one of these two types of facets: either those regarding travelling to a venue, or those regarding leaving a venue. We choose for the first one: travelling to a venue. For convenience, we give names to the associated inequalities and faces:

$$\ell_{n,i,t} : \sum_{s\in[n]\setminus\{t\}} y_{i,s,t} \geq 1 \tag{5.1}$$
$$F_{n,i,t} := \{(\mathbf{x}_n,\mathbf{y}_n) \in Q_n \mid \sum_{s\in[n]\setminus\{t\}} y_{i,s,t} = 1\}. \tag{5.2}$$

The indices take the following values: n is even and at least 4, and i and t range from 1 to n.

Now, we are ready to start with the proof. First an outline of the proof is sketched in the next section.

5.3.1 Proof outline

Two proof techniques were considered for this proof. The first one is very straightforward: trying to find $\dim(Q_n)$ affinely independent vectors which are contained in the face $F_{n,i,t}$. The second one is a bit more sophisticated, but resembles a proof technique used before. The face $F_{n,i,t}$ is contained in some facet. This facet is defined by some facet-defining inequality ℓ_n. By applying local changes to tournament schedules, like in Sections 4.2.3 and 4.2.4, we can deduce properties of the coefficients of ℓ_n. The goal is then to find enough properties of the coefficients so that we can conclude that ℓ_n is equivalent to our inequality $\ell_{n,i,t}$, proving that $F_{n,i,t}$ is a facet.

Both techniques have advantages and disadvantages. To make best use of the advantages of both methods, we use a combination of both techniques. The next sections are structured as follows. First, we use the local change technique to derive properties of some of the coefficients of the facet-defining inequality ℓ_n. Next, we look for affinely independent vectors contained in $F_{n,i,t}$, but in a lower dimensional polytope. Finally, these two results are combined to prove that $F_{n,i,t}$ is a facet.

5.3.2 Local changes

Let $n \geq 4$, n even, be an arbitrary tournament size and choose $i, t \in [n]$ arbitrarily. The inequality $\ell_{n,i,t}$ (5.1) is clearly valid: this follows from the physical interpretation as described in Section 5.2. The face $F_{n,i,t}$ (5.2) is contained in some facet of Q_n. Call this facet F_n and let the facet-defining inequality be

$$\ell_n : \mathbf{a}^\mathsf{T}\mathbf{x}_n + \mathbf{b}^\mathsf{T}\mathbf{y}_n \leq \gamma.$$

Let $\mathbf{x}_n \in X_n$ and compute the associated travel vector $\mathbf{y}_n$ such that values of all elements are minimised. Because the travel variables only depend on the play variables and not on each other, this is easy to do. The vector $\mathbf{y}_n$ now reflects the real travel movements. This implies that $(\mathbf{x}_n, \mathbf{y}_n) \in F_{n,i,t}$.

Many elements of $\mathbf{y}_n$ are equal to zero. Suppose the variable $y_{i',s',t'}$ is equal to zero and does not appear in the inequality $\ell_{n,i,t}$. Now increase its value to 1. This yields the new vector $\mathbf{y}'_n$. The schedule $(\mathbf{x}_n, \mathbf{y}'_n)$ is also contained in $F_{n,i,t}$. As $F_{n,i,t}$ is contained in F_n and $(\mathbf{x}_n, \mathbf{y}_n)$ and $(\mathbf{x}_n, \mathbf{y}'_n)$ are both in $F_{n,i,t}$, this implies that they are also in F_n. Thus, the following equations hold:

$$\begin{aligned} \mathbf{a}^\mathsf{T}\mathbf{x}_n + \mathbf{b}^\mathsf{T}\mathbf{y}_n &= \gamma \\ \mathbf{a}^\mathsf{T}\mathbf{x}_n + \mathbf{b}^\mathsf{T}\mathbf{y}'_n &= \gamma, \end{aligned}$$

and hence, by subtracting them,

$$\begin{aligned} & \mathbf{b}^\mathsf{T}\mathbf{y}'_n - \mathbf{b}^\mathsf{T}\mathbf{y}_n = 0 \\ \iff \quad & \mathbf{b}^\mathsf{T}(\mathbf{y}'_n - \mathbf{y}_n) = 0. \end{aligned}$$

Now, as $\mathbf{y}'_n$ only differs from $\mathbf{y}_n$ in one position, namely at index (i', s', t'), this implies that $b_{i',s',t'} = 0$. In Sections 4.3.3, 4.3.4, and 4.3.5, we already showed that it is possible to construct tournament schedules such that each travel variable is equal to zero in at least one schedule. Using these schedules, and minimising the entries of the travel variables, we can repeat the procedure above for all variables $y_{i',s',t'}$ which do not appear in the inequality $\ell_{n,i,t}$. Thus, for all indices (i', s', t') which do not appear in $\ell_{n,i,t}$, $b_{i',s',t'}$ must be equal to zero.

Let $\overline{\mathbf{y}}_n$ and $\overline{\mathbf{b}}$ indicate the travel vector and coefficient vector, respectively, restricted to the indices appearing in the inequality $\ell_{n,i,t}$. Thus, these vectors have $n-1$ elements: one for each of the indices (i, s, t) where $s \in [n] \setminus \{t\}$. Now, we can write the facet-defining inequality ℓ_n as follows:

$$\ell_n : \mathbf{a}^\mathsf{T}\mathbf{x}_n + \overline{\mathbf{b}}^\mathsf{T}\overline{\mathbf{y}}_n \leq \gamma.$$

5.3.3 Affinely independent vectors

In the previous section, we showed that the travel variables which do not appear in the inequality $\ell_{n,i,t}$ are irrelevant in a sense. Therefore, we restrict the travel variables to the appearing variables in this section. Like in $\overline{\mathbf{y}}_n$ and $\overline{\mathbf{b}}$, we use a bar to indicate the restricted versions of several entities. Among them are $\overline{Y_n}$, $\overline{Q}_n$ and $\overline{F}_{n,i,t}$. The definitions speak for themselves.

The goal of this section is to find $\dim(\overline{Q}_n)$ affinely independent vectors which are contained in the restricted face $\overline{F}_{n,i,t}$, proving that $\overline{F}_{n,i,t}$ is a facet of the restricted polytope $\overline{Q}_n$. This does not yet prove that $F_{n,i,t}$ is also a facet of Q_n, but it does bring us closer to a proof.

First set of vectors

To begin with, we can find $\dim(P_n) + 1$ affinely independent play vectors $\mathbf{x}_n$. Let $p = \dim(P_n) + 1$. Call these vectors $\mathbf{x}^1, ..., \mathbf{x}^p$ and collect them in the set $S_n^{\mathbf{x}}$. To avoid notational clutter, we

omit the subscript n on the play variables. Compute the associated travel vectors with minimised values, and restrict these vectors to the appearing variables. This yields the vectors $\overline{\mathbf{y}}^1, ..., \overline{\mathbf{y}}^p$. The restricted tournament schedules $(\mathbf{x}^1, \overline{\mathbf{y}}^1), ..., (\mathbf{x}^1, \overline{\mathbf{y}}^1)$ are contained in $\overline{F}_{n,i,t}$. Thus, we have already found $\dim(P_n) + 1$ affinely independent vectors.

Now we compute the remaining number of vectors to be found. In accordance with the dimension of Q_n, the dimension of $\overline{Q}_n$ is equal to the dimension of P_n plus the number of elements of $\overline{\mathbf{y}}_n$. $\overline{\mathbf{y}}_n$ contains $n-1$ elements. Thus, the remaining number of vectors is

$$\dim(\overline{Q}_n) - (\dim(P_n) + 1) = \dim(P_n) + (n-1) - (\dim(P_n) + 1) = n - 2.$$

To find these vectors, we first deduce the properties they should have.

Properties of the remaining vectors

Choose a set of $n-2$ play vectors: $\mathbf{u}^1, ..., \mathbf{u}^{n-2} \in X_n$. As the set of vectors $\mathbf{x}^1, ..., \mathbf{x}^p$ is an affine basis of the affine hull of X_n, we can write each vector $\mathbf{u}^j$ as an affine combination of these vectors. Hence, there exist scalars γ_j^m such that

$$\begin{cases} \mathbf{u}^j = \gamma_j^1 \mathbf{x}^1 + ... + \gamma_j^p \mathbf{x}^p \\ \sum\limits_{m=1}^{p} \gamma_j^m = 1 \end{cases} \qquad \forall j \in [n-2].$$

Let the associated travel vectors of the play vectors $\mathbf{u}^j$, with minimised values and restricted to the appearing variables, be $\overline{\mathbf{v}}^1, ..., \overline{\mathbf{v}}^{n-2}$. Thus, $(\mathbf{u}^j, \overline{\mathbf{v}}^j) \in \overline{F}_{n,i,t}$ for all $j \in [n-2]$. The whole set of vectors

$$S_n := \left\{(\mathbf{x}^j, \overline{\mathbf{y}}^j)\right\}_{j\in[p]} \cup \left\{(\mathbf{u}^j, \overline{\mathbf{v}}^j)\right\}_{j\in[n-2]} \tag{5.3}$$

is affinely independent if and only if the only solution to

$$\begin{cases} \alpha_1 \mathbf{x}^1 + ... + \alpha_p \mathbf{x}^p + \beta_1 \mathbf{u}^1 + ... + \beta_{n-2} \mathbf{u}^{n-2} = 0 & (5.4) \\ \alpha_1 \overline{\mathbf{y}}^1 + ... + \alpha_p \overline{\mathbf{y}}^p + \beta_1 \overline{\mathbf{v}}^1 + ... + \beta_{n-2} \overline{\mathbf{v}}^{n-2} = 0 & (5.5) \\ \sum\limits_{j=1}^{p} \alpha_j + \sum\limits_{j=1}^{n-2} \beta_j = 0 & (5.6) \end{cases}$$

is the trivial solution $\alpha_j = 0$ for all $j \in [p]$ and $\beta_j = 0$ for all $j \in [n-2]$.

The first equation, (5.4), can be rewritten as follows:

$$\begin{aligned} & \alpha_1 \mathbf{x}^1 + ... + \alpha_p \mathbf{x}^p + \beta_1 \mathbf{u}^1 + ... + \beta_{n-2} \mathbf{u}^{n-2} = 0 \\ \iff \quad & \alpha_1 \mathbf{x}^1 + ... + \alpha_p \mathbf{x}^p + \beta_1 \sum_{m=1}^{p} \gamma_1^m \mathbf{x}^m + ... + \beta_{n-2} \sum_{m=1}^{p} \gamma_{n-2}^m \mathbf{x}^m = 0 \\ \iff \quad & \left(\alpha_1 + \sum_{j=1}^{n-2} \beta_j \gamma_j^1\right) \mathbf{x}^1 + ... + \left(\alpha_p + \sum_{j=1}^{n-2} \beta_j \gamma_j^p\right) \mathbf{x}^p = 0 \end{aligned} \tag{5.7}$$

Combining equations (5.7) and (5.6) and the fact that the vectors $\mathbf{x}^1, ..., \mathbf{x}^p$ are affinely independent, we can conclude that for all $m \in [p]$,

$$\alpha_m + \sum_{j=1}^{n-2} \beta_j \gamma_j^m = 0.$$

This gives a new expression for the α_j. We can use these to rewrite equation (5.5):

$$\begin{aligned} & \alpha_1\overline{\mathbf{y}}^1 + ... + \alpha_p\overline{\mathbf{y}}^p + \beta_1\overline{\mathbf{v}}^1 + ... + \beta_{n-2}\overline{\mathbf{v}}^{n-2} = 0 \\ \iff \quad & -\overline{\mathbf{y}}^1 \sum_{j=1}^{n-2} \beta_j\gamma_j^1 - ... - \overline{\mathbf{y}}^p \sum_{j=1}^{n-2} \beta_j\gamma_j^p + \beta_1\overline{\mathbf{v}}^1 + ... + \beta_{n-2}\overline{\mathbf{v}}^{n-2} = 0 \\ \iff \quad & \beta_1 \left(\overline{\mathbf{v}}^1 - \sum_{m=1}^{p} \gamma_1^m \overline{\mathbf{y}}^m \right) + ... + \beta_{n-2} \left(\overline{\mathbf{v}}^{n-2} - \sum_{m=1}^{p} \gamma_{n-2}^m \overline{\mathbf{y}}^m \right) = 0. \end{aligned}$$

Hence, if the vectors

$$\overline{\mathbf{w}}^j := \overline{\mathbf{v}}^j - \sum_{m=1}^{p} \gamma_j^m \overline{\mathbf{y}}^m \quad \forall j \in [n-2] \tag{5.8}$$

are linearly independent, then all β_j must be equal to zero. Next, if all β_j are equal to zero, then equations (5.4) and (5.6) reduce to

$$\begin{cases} \alpha_1\mathbf{x}^1 + ... + \alpha_p\mathbf{x}^p = 0 \\ \sum\limits_{j=1}^{p} \alpha_j = 0, \end{cases}$$

and as the $\mathbf{x}^j$ are all affinely independent, this implies that all α_j should be equal to zero too.

Thus, if the vectors $\overline{\mathbf{w}}^j$ are linearly independent, then the only solution to the set of equations (5.4), (5.5), and (5.6) is the trivial solution. Therefore, the set of vectors S_n (5.3) is affinely independent. This gives us the following task: finding vectors $\mathbf{u}^1, ..., \mathbf{u}^{n-2} \in X_n$ such that the vectors $\overline{\mathbf{w}}^j$ are linearly independent. Keep in mind that we did not pose any restrictions on the set of play vectors $\{\mathbf{x}^j\}_{j\in[p]}$, apart from the fact that the set should be affinely independent. This gives us some freedom.

Technique to construct vectors $\mathbf{u}^j$

The following technique was found to construct vectors $\mathbf{u}^j$. We focus on a specific facet: $\overline{F}_{n,n,1}$. Thus, $i = n$ and $t = 1$. First recall the canonical 1-factorisation (note that the factors F_k have nothing to do with the facets F_n):

$$F_k = \{\{n, k\}\} \cup \{\{k + i, k - i\} \mid i = 1, ..., n/2 - 1\} \qquad k = 1, ..., n - 1. \tag{3.1}$$

We start with a schedule on n teams generated with the canonical 1-factorisation, and apply the standard home-away assignment as described in Section 3.1.2. Thus, team n plays alternately home and away. A double round-robin schedule is created by copying the schedule, reverting the home-away assignment and then concatenating the copy to the original schedule. Now swap teams 2 and $n - 1$. For $n = 6$, the resulting schedule is shown in Table 5.4. Note that without the swap, team n plays the other teams in increasing order.

Now, in slot $n - 1$, team n plays home against team 2. In slot n, team n plays away against team 1. Create a second schedule by performing two home-away swaps on the first schedule: between team n and team 1, and between team n and team 2. Create a third schedule by performing one home-away swap on the first schedule: only between team n and team 2. Call the play vector of the first schedule $\mathbf{x}_1^2$, the second one $\mathbf{x}_2^2$, and the third one $\mathbf{x}_3^2$. The superscript 2 comes from the swap between team $n - 1$ and team 2. The subscript does not refer to the number of teams in this case.

We focus on slots 1, $n - 1$, and n. The opponents of team n in these slots of the three schedules are given in Table 5.5. A minus indicates an away game and vice versa.

	Slot									
Team	1	2	3	4	5	6	7	8	9	10
1	−6	+3	−2	+5	−4	+6	−3	+2	−5	+4
2	+5	−4	+1	−3	−6	−5	+4	−1	+3	+6
3	+4	−1	−6	+2	−5	−4	+1	+6	−2	+5
4	−3	+2	−5	+6	+1	+3	−2	+5	−6	−1
5	−2	+6	+4	−1	+3	+2	−6	−4	+1	−3
6	+1	−5	+3	−4	+2	−1	+5	−3	+4	−2

Table 5.4: Tournament on 6 teams, generated with the canonical 1-factorisation. Teams 2 and 5 are swapped. The table shows the opponent of each team in each slot. A minus indicates an away game and vice versa.

Slot	$\mathbf{x}_1^2$	$\mathbf{x}_2^2$	$\mathbf{x}_3^2$	$\mathbf{u}^2$
1	+1	−1	+1	−1
$n-1$	+2	−2	−2	+2
n	−1	+1	−1	+1

Table 5.5: The opponents of team n in slots 1, $n-1$, and n in the schedules $\mathbf{x}_j^2$, $j \in [3]$, and $\mathbf{u}^2$.

We can create a new play vector, $\mathbf{u}^2$, by computing the following affine combination of the other play vectors:

$$\mathbf{u}^2 = \mathbf{x}_1^2 + \mathbf{x}_2^2 - \mathbf{x}_3^2.$$

This yields again a feasible play vector. The opponents of team n in this schedule are also shown in Table 5.5.

Now we compute the corresponding travel vectors of these four schedules, with minimised values and restricted to variables appearing in the facet $\overline{F}_{n,n,1}$. Call these vectors $\overline{\mathbf{y}}_j^2$, $j \in [3]$, and $\overline{\mathbf{v}}^2$. They contain the variables with indices $(n, s, 1)$, $s \geq 2$. These variables indicate whether team n travels from team s to team 1. Take a look at their values, in Table 5.6.

Index	$\overline{\mathbf{y}}_1^2$	$\overline{\mathbf{y}}_2^2$	$\overline{\mathbf{y}}_3^2$	$\overline{\mathbf{v}}^2$
$(n,2,1)$	0	1	0	0
$(n,3,1)$	0	0	0	0
⋮	⋮	⋮	⋮	⋮
$(n,n-1,1)$	0	0	0	0
$(n,n,1)$	1	0	1	1

Table 5.6: The values of the vectors $\overline{\mathbf{y}}_j^2$, $j \in [3]$, and $\overline{\mathbf{v}}^2$. All nonzero values are shown.

Recall equation (5.8):

$$\overline{\mathbf{w}}^j := \overline{\mathbf{v}}^j - \sum_{m=1}^{p} \gamma_j^m \overline{\mathbf{y}}^m \quad \forall j \in [n-2]. \tag{5.8}$$

In our case, only three travel vectors are involved: $\overline{\mathbf{y}}_1^2$, $\overline{\mathbf{y}}_2^2$, and $\overline{\mathbf{y}}_3^2$, with coefficients +1, +1, and −1,

respectively. Using Table 5.6, it is easy to see that the only two nonzero values of the vector

$$\overline{\mathbf{w}}^2 = \overline{\mathbf{v}}^2 - (\overline{\mathbf{y}}_1^2 + \overline{\mathbf{y}}_2^2 - \overline{\mathbf{y}}_3^2)$$

are $+1$ at index $(n, 2, 1)$ and -1 at index $(n, n, 1)$. Thus, the vector $\overline{\mathbf{w}}^2$ is nonzero, and the set containing only this vector is linearly independent. Hence, with $\overline{\mathbf{u}}^2$, we have found the first vector that we were looking for. Do not forget that we should also choose the vectors $\mathbf{x}_1^2$, $\mathbf{x}_2^2$, and $\mathbf{x}_3^2$ to be part of the set $S_n^{\mathbf{x}}$ of affinely independent play vectors—otherwise, $\overline{\mathbf{u}}^2$ will be formed by a different affine combination, which should not happen.

Finding the remaining vectors

To find the other vectors, a simple trick can be used: swapping team 2 with team $3, ..., n-1$, one by one. This gives us the vectors $(\mathbf{u}^3, \overline{\mathbf{v}}^3), ..., (\mathbf{u}^{n-1}, \overline{\mathbf{v}}^{n-1})$. Also, we obtain the vectors $\mathbf{x}_1^j$, $\mathbf{x}_2^j$, $\mathbf{x}_3^j$, $\overline{\mathbf{y}}_1^j$, $\overline{\mathbf{y}}_2^j$, and $\overline{\mathbf{y}}_3^j$ for $j \in [3, n-1]$ by performing the same swap. Compute the vectors

$$\overline{\mathbf{w}}^j = \overline{\mathbf{v}}^j - (\overline{\mathbf{y}}_1^j + \overline{\mathbf{y}}_2^j - \overline{\mathbf{y}}_3^j).$$

Similar to $\overline{\mathbf{w}}^2$, the vector $\overline{\mathbf{w}}^j$ for $j \in [3, n-1]$ has two nonzero values: $+1$ at index $(n, j, 1)$ and -1 at index $(n, n, 1)$. Thus, each vector $\overline{\mathbf{w}}^j$, $j \in [2, n-1]$, has a unique nonzero entry: at index $(n, j, 1)$. Hence, the set $\{\overline{\mathbf{w}}^2, ..., \overline{\mathbf{w}}^{n-1}\}$ is linearly independent.

It seems that we are done now. Unfortunately, there is one problem: the vectors $\mathbf{x}_1^j$, $\mathbf{x}_2^j$, and $\mathbf{x}_3^j$ for $j \in [2, n-1]$ should all be part of the set $S_n^{\mathbf{x}}$ of affinely independent play vectors. However, they are not affinely independent, and thus, we cannot add them all. Fortunately, this can be fixed.

First we investigate which vectors can be added. We start with the set $S_n^{\mathbf{x}} = \{\mathbf{x}_1^2, \mathbf{x}_2^2, \mathbf{x}_3^2\}$. Note that the vectors $\mathbf{x}_1^j$ and $\mathbf{x}_3^j$ for $j \in [3, n-1]$ all have a unique nonzero entry. In the play vector $\mathbf{x}_1^j$, team n plays home against team j in slot $n-1$. In the vector $\mathbf{x}_3^j$, team n plays away against team j in slot $n-1$. Thus, we can add these vectors to our set $S_n^{\mathbf{x}}$ while keeping it affinely independent, resulting in the set

$$S_n^{\mathbf{x}} = \{\mathbf{x}_1^2, \mathbf{x}_2^2, \mathbf{x}_3^2\} \cup \{\mathbf{x}_1^j, \mathbf{x}_3^j\}_{j \in [3, n-1]}.$$

Now we have to find alternatives for the vectors $\mathbf{x}_2^j$, $j \in [3, n-1]$. Therefore, look at the opponents of team n in the second half of the schedule. For $n = 8$, these are given in Table 5.7.

Slot	$\mathbf{x}_2^2$	$\mathbf{x}_2^3$	$\mathbf{x}_2^4$	$\mathbf{x}_2^5$	$\mathbf{x}_2^6$	$\mathbf{x}_2^7$
8	$+1$	$+1$	$+1$	$+1$	$+1$	$+1$
9	$+7$	$+7$	$+7$	$+7$	$+7$	$+2$
10	-3	-2	-3	-3	-3	-3
11	$+4$	$+4$	$+2$	$+4$	$+4$	$+4$
12	-5	-5	-5	-2	-5	-5
13	$+6$	$+6$	$+6$	$+6$	$+2$	$+6$
14	$+2$	$+3$	$+4$	$+5$	$+6$	$+7$

Table 5.7: The opponents of team 8 in the play vectors $\mathbf{x}_2^j$ in slots 8 up to 14 for $n = 8$.

Now we swap some slots in the range n up to (and including) $2n-3$, or the range 8 up to 13 for $n = 8$. This range is suited for this, for two reasons. First, swapping slots in this range does not change the travel vectors $\overline{\mathbf{y}}_1^j$, $\overline{\mathbf{y}}_2^j$, and $\overline{\mathbf{y}}_3^j$ for $j \in [3, n-1]$, because team n still travels to team 1 from the same venue; the away game against team 1 takes place in slot 1.

Second, we want to make sure that the vectors $\mathbf{u}^j = \mathbf{x}_1^j + \mathbf{x}_2^j - \mathbf{x}_3^j$ are feasible play vectors. Because the vectors $\mathbf{x}_1^j$ and $\mathbf{x}_3^j$ have the same values for the slots in the range n up to $2n-3$, we have that $\mathbf{u}^j = \mathbf{x}_2^j$ for the entries in this range; $\mathbf{x}_1^j$ and $\mathbf{x}_3^j$ cancel out. Thus, if we swap some slots in the range n up to $2n-3$ in the vector $\mathbf{x}_2^j$, the same happens in $\mathbf{u}^j$, ensuring that this play vector stays feasible.

As we can see in Table 5.7, in the vectors $\mathbf{x}_2^j$, $j \in [2, n-1]$, the opponents of team n in each of the slots from $n+1$ up to $2n-3$ consist of two teams: the 'regular' opponent in all but one vectors, and once team 2 as opponent. In slot n, team 1 is the only opponent. Thus, it is possible to swap two slots in such a way that a new opponent is introduced in at least one of these slots. Hence, the resulting vector can be added to the set $S_n^{\mathbf{x}}$. We choose the following system of swaps, so that we do not introduce the same opponent in the same slots: swap slots $n+j-3$ and $n+j-2$ in vector $\mathbf{x}_2^j$. Let the resulting play vectors be $\hat{\mathbf{x}}_2^j$, $j \in [3, n-1]$. The result on the schedule of team 8 for $n = 8$ is shown in Table 5.8.

Slot	$\hat{\mathbf{x}}_2^2$	$\hat{\mathbf{x}}_2^3$	$\hat{\mathbf{x}}_2^4$	$\hat{\mathbf{x}}_2^5$	$\hat{\mathbf{x}}_2^6$	$\hat{\mathbf{x}}_2^7$
8	+1	+7	+1	+1	+1	+1
9	+7	+1	−3	+7	+7	+2
10	−3	−2	+7	+4	−3	−3
11	+4	+4	+2	−3	−5	+4
12	−5	−5	−5	−2	+4	+6
13	+6	+6	+6	+6	+2	−5
14	+2	+3	+4	+5	+6	+7

Table 5.8: The opponents of team 8 in the play vectors $\hat{\mathbf{x}}_2^j$ in slots 8 up to 14 for $n = 8$. Colours indicate which slots are swapped; no special meaning otherwise.

This system of swaps works for all $n \geq 4$. Now, we obtain new vectors $\hat{\mathbf{u}}^j$ and we can add the vectors $\hat{\mathbf{x}}_2^j$, $j \in [3, n-1]$ to the set $S_n^{\mathbf{x}}$:

$$\hat{\mathbf{u}}^j = \mathbf{x}_1^j + \hat{\mathbf{x}}_2^j - \mathbf{x}_3^j \qquad \forall j \in [3, n-1]$$
$$S_n^{\mathbf{x}} = \{\mathbf{x}_1^2, \mathbf{x}_2^2, \mathbf{x}_3^2\} \cup \{\mathbf{x}_1^j, \mathbf{x}_3^j\}_{j\in[3,n-1]} \cup \{\hat{\mathbf{x}}_2^j\}_{j\in[3,n-1]}.$$

Conclusion

We found a set of affinely independent play vectors $S_n^{\mathbf{x}}$ from which we can construct the vectors $\mathbf{u}^2, \hat{\mathbf{u}}^3, ..., \hat{\mathbf{u}}^{n-1}$. The corresponding travel vectors are $\overline{\mathbf{y}}_1^j, \overline{\mathbf{y}}_2^j, \overline{\mathbf{y}}_3^j$ and $\overline{\mathbf{v}}^j$ for $j \in [2, n-1]$. With these travel vectors, we compute the set of vectors $\{\overline{\mathbf{w}}^2, ..., \overline{\mathbf{w}}^{n-1}\}$. This set is linearly independent.

The set $S_n^{\mathbf{x}}$ contains $3(n-2)$ affinely independent vectors. This can be extended to an affine basis of X_n. Call these vectors $\mathbf{x}^{3(n-2)+1}, ..., \mathbf{x}^p$ and let the corresponding travel vectors, with minimised values and restricted to the appearing variables, be $\overline{\mathbf{y}}^{3(n-2)+1}, ..., \overline{\mathbf{y}}^p$.

Now, the following set of $\dim(\overline{Q}_n)$ vectors, all of which are contained in the fact $\overline{F}_{n,n,1}$, is affinely independent:

$$\left\{(\mathbf{x}_j^2, \overline{\mathbf{y}}_j^2)\right\}_{j\in[3]} \cup \left\{(\mathbf{x}_1^j, \overline{\mathbf{y}}_1^j), (\hat{\mathbf{x}}_2^j, \overline{\mathbf{y}}_2^j), (\mathbf{x}_3^j, \overline{\mathbf{y}}_3^j)\right\}_{j\in[3,n-1]} \cup \left\{(\mathbf{x}^j, \mathbf{y}^j)\right\}_{j\in[3(n-2)+1,p]}$$
$$\cup \left\{(\mathbf{u}^2, \overline{\mathbf{v}}^2)\right\} \cup \left\{(\hat{\mathbf{u}}^j, \overline{\mathbf{v}}^j)\right\}_{j\in[3,n-1]}.$$

This proves that $\overline{F}_{n,n,1}$ is a facet of the polytope $\overline{Q}_n$. Now, by symmetry, $\overline{F}_{n,i,t}$ is a facet of $\overline{Q}_n$ for any two distinct teams i and t.

Unfortunately, the author realised too late that $i = t$ is a special case, and time is lacking now to write down a formal proof that $\overline{F}_{n,i,i}$ is indeed a facet of $\overline{Q}_n$ for all $i \in [n]$. The point where the proof for this case could fail, is that the set of vectors $\{\overline{\mathbf{w}}^2, ..., \overline{\mathbf{w}}^{n-1}\}$ is not linearly independent.

As a practical substitute for a proof, computations were carried out to compute these vectors for $i = t = n$. It turned out that, for $n = 4, 6, 8, 10, 12, 14$, and 16—the sizes that are used in practice—the nonzero entries of $\overline{\mathbf{w}}^j$ are:

- value -1 at index (n, j, n) for all $j \in [2, n-1]$;
- value -1 at index $(n, n-3, n)$ for $j = n-1$ if $n \geq 6$.

Thus, it is clear that the set of vectors $\{\overline{\mathbf{w}}^2, ..., \overline{\mathbf{w}}^{n-1}\}$ is linearly independent for n between 4 and 16. Therefore, $\overline{F}_{n,i,i}$ is a facet of $\overline{Q}_n$ for n between 4 and 16, n even. A formal proof for all n for the case $i = t$ should not be too complex; all components are present, but it is simply laborious to prove what values the vectors $\overline{\mathbf{w}}^j$ take, and time is lacking.

We now turn to the last part of the original problem we were proving: that the face $F_{n,i,t}$ is a facet of the polytope Q_n.

5.3.4 Combining the previous results

In Section 5.3.2, we assumed that the face $F_{n,i,t}$ is contained in a facet F_n of Q_n and showed that the facet-defining inequality ℓ_n of F_n does not contain travel variables which do not appear in the inequality $\ell_{n,i,t}$ of $F_{n,i,t}$. Thus, ℓ_n can be written as

$$\ell_n : \mathbf{a}^\mathsf{T}\mathbf{x}_n + \overline{\mathbf{b}}^\mathsf{T}\overline{\mathbf{y}}_n \leq \gamma.$$

In Section 5.3.3, we showed that the restricted face $\overline{F}_{n,i,t}$ is a facet of the restricted polytope $\overline{Q}_n$. For the case $i = t$, the proof is incomplete; a small part is missing. However, for the values of n between 4 and 16, which are used in practice, computations were carried out to show that $\overline{F}_{n,i,i}$ is a facet of $\overline{Q}_n$.

Now, these results should be combined to show that $F_{n,i,t}$ is a facet of Q_n. The proof is split into two parts. First, we prove a lemma. Next, we use this to complete the proof.

Facet-defining inequalities of Q_n and $\overline{Q}_n$

First we prove the following lemma, relating facets of Q_n to facets of $\overline{Q}_n$.

Lemma 5.1. Suppose we have a facet-defining inequality of the polytope Q_n. Furthermore, suppose that in this inequality, only variables appear which are also in $\overline{Q}_n$. Then the facet-defining inequality for Q_n is also facet-defining for $\overline{Q}_n$.

Proof. Let $\mathbf{a}^\mathsf{T}\mathbf{x}_n + \overline{\mathbf{b}}^\mathsf{T}\overline{\mathbf{y}}_n \leq \gamma$ be some facet-defining inequality of Q_n and let the corresponding facet be

$$F_{Q_n} = \{(\mathbf{x}_n, \mathbf{y}_n) \in Q_n \mid \mathbf{a}^\mathsf{T}\mathbf{x}_n + \overline{\mathbf{b}}^\mathsf{T}\overline{\mathbf{y}}_n = \gamma\}.$$

Similarly, let

$$F_{\overline{Q}_n} = \{(\mathbf{x}_n, \overline{\mathbf{y}}_n) \in \overline{Q}_n \mid \mathbf{a}^\mathsf{T}\mathbf{x}_n + \overline{\mathbf{b}}^\mathsf{T}\overline{\mathbf{y}}_n = \gamma\}.$$

As F_{Q_n} is a facet of Q_n, it contains $\dim(Q_n)$ affinely independent vectors. Let these vectors be

$$(\mathbf{x}^1, \mathbf{y}^1), ..., (\mathbf{x}^{q_n}, \mathbf{y}^{q_n}),$$

where $q_n = \dim(Q_n)$.

To prove that $F_{\overline{Q}_n}$ is a facet of $\overline{Q}_n$, we must show that it contains $\dim(\overline{Q}_n)$ affinely independent vectors. Suppose to the contrary that this is not the case. So, $F_{\overline{Q}_n}$ contains at most $r < \dim(\overline{Q}_n)$ affinely independent vectors. Let $\overline{\mathbf{y}}^j$, $j \in [q_n]$, be the restricted versions of the vectors $\mathbf{y}^j$. Thus, of the vectors

$$(\mathbf{x}^1, \overline{\mathbf{y}}^1), ..., (\mathbf{x}^{q_n}, \overline{\mathbf{y}}^{q_n}),$$

at most r are affinely independent, or equivalently, of the vectors

$$(1, \mathbf{x}^1, \overline{\mathbf{y}}^1), ..., (1, \mathbf{x}^{q_n}, \overline{\mathbf{y}}^{q_n}),$$

at most r are linearly independent. Now put these vectors in a matrix:

$$\overline{M} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \mathbf{x}^1 & \mathbf{x}^2 & \cdots & \mathbf{x}^{q_n} \\ \overline{\mathbf{y}}^1 & \overline{\mathbf{y}}^2 & \cdots & \overline{\mathbf{y}}^{q_n} \end{bmatrix}.$$

As the matrix contains at most r linearly independent columns, we have that $\operatorname{rank}(\overline{M}) \leq r$, and hence, the matrix contains at most r linearly independent rows. Now, we extend the matrix $\overline{M}$ to the full matrix M by replacing the vectors $\overline{\mathbf{y}}^j$ by $\mathbf{y}^j$:

$$M = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \mathbf{x}^1 & \mathbf{x}^2 & \cdots & \mathbf{x}^{q_n} \\ \mathbf{y}^1 & \mathbf{y}^2 & \cdots & \mathbf{y}^{q_n} \end{bmatrix}.$$

Thus, we have added $\left|\mathbf{y}^j\right| - \left|\overline{\mathbf{y}}^j\right|$ new rows. Therefore, M contains at most $r + \left|\mathbf{y}^j\right| - \left|\overline{\mathbf{y}}^j\right|$ linearly independent rows. This gives

$$\begin{aligned} \operatorname{rank}(M) &\leq r + \left|\mathbf{y}^j\right| - \left|\overline{\mathbf{y}}^j\right| \\ &< \dim(\overline{Q}_n) + \left|\mathbf{y}^j\right| - \left|\overline{\mathbf{y}}^j\right| \\ &= \dim(Q_n). \end{aligned}$$

Thus, M contains less than $\dim(Q_n)$ linearly independent columns. However, this implies that the vectors

$$(1, \mathbf{x}^1, \mathbf{y}^1), ..., (1, \mathbf{x}^{q_n}, \mathbf{y}^{q_n})$$

are not linearly independent, and hence, that the vectors

$$(\mathbf{x}^1, \mathbf{y}^1), ..., (\mathbf{x}^{q_n}, \mathbf{y}^{q_n})$$

are not affinely independent. This is a contradiction. Thus, $F_{\overline{Q}_n}$ must contain $\dim(\overline{Q}_n)$ affinely independent vectors, and hence, it is a facet of $\overline{Q}_n$. □

Completing the proof

Recall the first part of Section 5.3.2. We assumed that our face $F_{n,i,t}$ (5.2), defined by the inequality $\ell_{n,i,t}$ (5.1), is contained in the facet F_n. Next, we showed that that the inequality defining F_n can be written as

$$\ell_n : \mathbf{a}^{\mathsf{T}}\mathbf{x}_n + \overline{\mathbf{b}}^{\mathsf{T}}\overline{\mathbf{y}}_n \leq \gamma.$$

$\overline{F}_{n,i,t}$ is the restricted version of $F_{n,i,t}$, defined by the same inequality $\ell_{n,i,t}$. Similarly, $\overline{F}_n$ is the restricted version of F_n, defined by the same inequality ℓ_n.

As ℓ_n only contains variables which appear in $\overline{Q}_n$, Lemma 5.1 shows that ℓ_n is facet-defining for $\overline{Q}_n$, and hence, that $\overline{F}_n$ is a facet of $\overline{Q}_n$. Furthermore, we showed that $\overline{F}_{n,i,t}$ is a facet of $\overline{Q}_n$ in Section 5.3.3 (except for the case $i = t$ for $n \geq 18$, which we momentarily ignore).

First we show that $\overline{F}_{n,i,t} \subseteq \overline{F}_n$. So, suppose that $(\mathbf{x}_n, \overline{\mathbf{y}}_n) \in \overline{F}_{n,i,t}$. Thus, there exists a vector $\mathbf{y}_n$ which is an extension of $\overline{\mathbf{y}}_n$ such that $(\mathbf{x}_n, \mathbf{y}_n) \in F_{n,i,t}$. As $F_{n,i,t}$ is contained in F_n, this implies that $(\mathbf{x}_n, \mathbf{y}_n) \in F_n$. And hence, $(\mathbf{x}_n, \overline{\mathbf{y}}_n) \in \overline{F}_n$. Thus, $\overline{F}_{n,i,t} \subseteq \overline{F}_n$. Combining this with the fact that $\overline{F}_{n,i,t}$ and $\overline{F}_n$ are both facets, this shows that $\overline{F}_{n,i,t} = \overline{F}_n$.

This does not imply that their facet-defining inequalities $\ell_{n,i,t}$ and ℓ_n are equal, as the polytope $\overline{Q}_n$ is not full-dimensional. However, it does imply that the one inequality can be obtained from the other by scalar multiplication and addition of linear combinations of the equality set of $\overline{Q}_n$. Thus, we can say that these inequalities are "equivalent with respect to the facets of $\overline{Q}_n$". As the equality set of Q_n is equal to the equality set of $\overline{Q}_n$, this implies that $\ell_{n,i,t}$ and ℓ_n are "equivalent with respect to the facets of Q_n" as well. And hence, as ℓ_n is facet-defining for Q_n, so is $\ell_{n,i,t}$. Thus, $F_{n,i,t}$ is a facet of Q_n.

We conclude the proof with a theorem.

Theorem 5.2. The inequalities

$$\sum_{s \in [n] \setminus \{t\}} y_{i,s,t} \geq 1$$
$$\sum_{s \in [n] \setminus \{t\}} y_{i,t,s} \geq 1$$

are proved to be facet-defining for the polytope Q_n in the following cases:

- for all distinct $i, t \in [n]$, for all $n \geq 4$, n even;
- for $i \in [n]$ and $i = t$, for $n \in \{4, 6, 8, 10, 12, 14, 16\}$.

Note that for the case $i = t$ and $n \geq 18$, it is very unlikely that the inequalities are *not* facet-defining, but a formal proof is missing.

Chapter 6

Conclusions and recommendations

6.1 Conclusions

The Travelling Tournament problem was modelled as an integer program. The main goal of this research was to reduce the integrality gap of this integer program by adding a specific type of cutting planes: facets.

Before we started searching facets, first we demonstrated the dimension of the solution polytope of the integer program. This was done in two steps: first only for the part of the polytope with the play variables, and next this was extended to the full polytope. The results are given in Theorem 4.1 and Corollary 4.1.1, respectively.

Next, we started looking for facets. Using the software library IPO, facets of the solution polytope were found. Most facets appeared to be specific to the instance to which the software was applied, but a few had a physical interpretation so that they could be generalised to other instances. Only one class of facets found in this way, which we called the "flow conservation facets", had a significant effect on the integrality gap.

Thereafter, it was proved that this class of facets is indeed facet-defining for the unconstrained TTP, using a combination of two proof techniques. For one special case, the proof is missing. However, computations showed that the proof technique used in the other cases seems to be working for the special case as well. Therefore, it is expected that this gap can be filled quite easily. The precise result can be found in Theorem 5.2.

During the research, we needed methods to generate tournament schedules. We spent a chapter on discussing three methods: the circle method, the canonical 1-factorisation and the Kirkman tournament. Kirkman is often cited as the inventor of the circle method, but we showed that this is a misconception. Furthermore, we proved that the Kirkman tournament is equivalent up to permutation to the canonical 1-factorisation. Presumably, this proof has been done before, but it was not electronically available. Thus, our proof fills this gap.

6.2 Recommendations for further research

We have a number of suggestions for further research. First of all, it would be good to complete the missing part of the facet proof. It is expected that this is not very complex, but it can be a laborious task.

The proofs in Chapters 4 and 5 of this thesis applied to the unconstrained TTP. The next step would be to extend this to the standard TTP. It seems likely that similar proof techniques can be used for the standard TTP as those used for the unconstrained TTP.

The last suggestion is the most important one in our opinion. Although we showed that the integrality gap is reduced significantly by adding the flow conservation facets, the integer program is still much slower after the addition of the facets than current state-of-the-art methods for solving TTP instances. The question is whether the facets can be incorporated in these methods to make them faster. It would be good to investigate this.

Bibliography

[1] C. Thielen and S. Westphal, "Complexity of the traveling tournament problem," *Theor. Comput. Sci.*, vol. 412, no. 4&5, pp. 345–351, 2011, doi:10.1016/j.tcs.2010.10.001.

[2] R. V. Rasmussen and M. A. Trick, "Round robin scheduling – a survey," *Eur. J. Oper. Res.*, vol. 188, no. 3, pp. 617–636, 2008, doi:10.1016/j.ejor.2007.05.046.

[3] M. A. Trick, "Challenge Traveling Tournament Instances," https://mat.tepper.cmu.edu/TOURN/, (accessed Mar. 24, 2020).

[4] R. Bhattacharyya, "Complexity of the Unconstrained Traveling Tournament Problem," *Oper. Res. Lett.*, vol. 44, no. 5, pp. 649–654, 2016, doi:10.1016/j.orl.2016.07.011.

[5] D. Froncek, "Scheduling a Tournament," in *Mathematics and Sports*, J. A. Gallian, Ed. Math. Assoc. Amer., 2010, pp. 203–216, doi:10.5948/UPO9781614442004.018.

[6] D. de Werra, "Scheduling in Sports," in *Studies on Graphs and Discrete Programming*, ser. North-Holland Mathematics Studies, P. Hansen, Ed. North-Holland, 1981, vol. 59, pp. 381–395, doi:10.1016/S0304-0208(08)73478-9.

[7] A. Drexl and S. Knust, "Sports league scheduling: Graph- and resource-based models," *Omega*, vol. 35, no. 5, pp. 465–471, 2007, doi:10.1016/j.omega.2005.08.002.

[8] T. P. Kirkman, "On a problem in combinations," *Cam. Dub. Math. J.*, vol. 2, pp. 191–204, 1847.

[9] I. Anderson, *Combinatorial Designs and Tournaments*. Oxford, U.K.: Oxford Univ. Press, 1997, ch. 8.1.

[10] M. Goerigk and S. Westphal, "A combined local search and integer programming approach to the traveling tournament problem," *Ann. Oper. Res.*, vol. 239, pp. 343–354, 2016, doi:10.1007/s10479-014-1586-6.

[11] C. C. Ribeiro, "Sports scheduling: Problems and applications," *Int. Trans. Oper. Res.*, vol. 19, no. 1&2, pp. 201–226, 2012, doi:10.1111/j.1475-3995.2011.00819.x.

[12] E. Lambrechts *et al.*, "Round-robin tournaments generated by the Circle Method having maximum carry-over," *Math. Program.*, vol. 172, no. 1&2, pp. 277–302, 2018, doi:10.1007/s10107-017-1115-x.

[13] Édouard Lucas, *Récréations mathématiques*. Paris, France: Gauthier-Villars, 1883, ch. Les jeux de demoiselles, pp. 176–180.

[14] I. Anderson, "Kirkman and GK_{2n}," *Bull. Inst. Combin. Appl.*, vol. 3, pp. 111–112, 1991.

[15] D. de Werra, "Some models of graphs for scheduling sports competitions," *Discrete Appl. Math.*, vol. 21, no. 1, pp. 47–65, 1988, doi:10.1016/0166-218X(88)90033-9.

[16] K. Easton, G. L. Nemhauser, and M. A. Trick, "Solving the Traveling Tournament Problem: A Combined Integer Programming and Constraint Programming Approach," in *Practice and Theory of Automated Timetabling IV*, E. Burke and P. D. Causmaecker, Eds., Gent, Belgium, Aug. 2002, pp. 100–109, doi:10.1007/978-3-540-45157-0_6.

[17] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2020. [Online]. Available: http://www.gurobi.com

[18] M. Walter, "IPO - Investigating Polyhedra by Oracles," 2019. [Online]. Available: https://bitbucket.org/matthias-walter/ipo/src/master/

[19] M. Walter, "Investigating Polyhedra by Oracles and Analyzing Simple Extensions of Polytopes," Ph.D. dissertation, Fak. für Math., Otto-von-Guericke-Univ. Magdeburg, Magdeburg, ST, 2016.