Aalto University
School of Science
Degree Programme in Engineering Physics and Mathematics

Vesa Husgafvel

Polyhedral Analysis of Up-peak Traffic Patterns in Elevator Dispatching Problem

Master's Thesis
Espoo, April 6, 2016

Supervisor: Professor Harri Ehtamo
Advisor: Mirko Ruokokoski M.Sc. (Tech.)

Aalto University
School of Science
Degree Programme in Engineering Physics and Mathematics

ABSTRACT OF MASTER'S THESIS

Author:	Vesa Husgafvel		
Title: Polyhedral Analysis of Up-peak Traffic Patterns in Elevator Dispatching Problem			
Date:	April 6, 2016	**Pages:**	viii + 77
Major:	Systems and Operations Research	**Code:**	Mat-2
Supervisor:	Professor Harri Ehtamo		
Advisor:	Mirko Ruokokoski M.Sc. (Tech.)		

Up-peak traffic is a common situation arising in elevator routing, where most of the transportation requests given by passengers are directed from lower floors to upper floors of a building. In this thesis, we examine three up-peak traffic patterns that differ from each other with respect to the number of elevators or the capacity of elevators. Analysis is based on a mixed-integer programming formulation of the elevator dispatching problem (EDP).

Solutions of linear integer optimization problems span a convex hull, which is called a polytope. By examining the structure of a polytope, it is possible to find out special features of the problem to be studied. A central variable in description of a polytope is dimension, which is defined as the number of affinely independent vectors contained in a polytope. In this work, we determine the dimension of each up-peak traffic pattern polytope to be studied, or in the case we are not able to give an exact formula, we determine a lower and an upper bound for the value of dimension. In addition, in each case we determine the number of feasible solutions and the number of arcs in the reduced graph. Results relating to different patterns are compared with each other and with polyhedral results of similar optimization problems that appear in literature.

The obtained results of this work give new, fundamental knowledge of the polyhedral structure of up-peak traffic patterns - a subject that has not been previously studied. We believe that by combining our results with similar research results, e.g., polyhedral results of down-peak traffic patterns, our knowledge of the elevator dispatching problem deepens, which will help in designing of EDP solving algorithms.

Keywords:	Elevator dispatching, elevator routing, integer optimization, up-peak traffic, polytope, polyhedral analysis
Language:	English

Aalto-yliopisto
Perustieteiden korkeakoulu
Teknillisen fysiikan ja matematiikan koulutusohjelma

DIPLOMITYÖN TIIVISTELMÄ

Tekijä:	Vesa Husgafvel		
Työn nimi: Ylösruuhkamuodostelmien polyhedraalianalyysi hissien reititysongelmassa			
Päiväys:	6. huhtikuuta 2016	**Sivumäärä:**	viii + 77
Pääaine:	Systeemi- ja operaatiotutkimus	**Koodi:**	Mat-2
Valvoja:	Professori Harri Ehtamo		
Ohjaaja:	Diplomi-insinööri Mirko Ruokokoski		

Ylösruuhka on yleinen hissien reitityksessä esiintyvä tilanne, jossa suurin osa matkustajien antamista siirtokutsuista kohdistuu rakennuksen alemmista kerroksista ylempiin kerroksiin. Tässä työssä tutkitaan kolmea ylösruuhkamuodostelmaa, jotka eroavat toisistaan käytettävissä olevien hissien lukumäärän tai hissien kapasiteetin suhteen. Analyysi pohjautuu hissien reititysongelman (EDP) lineaariseen sekalukuptimointiformulaatioon.

Lineaaristen kokonaislukuoptimointitehtävien ratkaisut virittävät konveksin kuoren, jota sanotaan tehtävän polytoopiksi. Polytoopin rakennetta tutkimalla on mahdollista selvittää tarkasteltavan tehtävän erityispiirteitä. Keskeinen polytooppia kuvaava suure on dimensio, joka määritellään polytoopin sisältämien affiinisti riippumattomien vektorien lukumääränä. Työssä määrätään kunkin tutkittavan ylösruuhkamuodostelman polytoopin dimensio tai dimension arvolle määrätään ala- ja yläraja. Lisäksi kussakin tapauksessa määrätään käypien ratkaisujen lukumäärä sekä kaarien lukumäärä redusoidussa graafissa. Eri muodostelmiin liittyviä tuloksia vertaillaan sekä keskenään että kirjallisuudessa esiintyvien samankaltaisten optimointiongelmien polyhedraalitulosten kanssa.

Työssä saadut tulokset antavat uutta, perustavanlaatuista tietoa ylösruuhkamuodostelmien polyhedraalirakenteesta - aiheesta jota ei ole aiemmin tutkittu. On uskottavaa että yhdistämällä saatuja tuloksia aiempien tutkimustulosten (esim. alasruuhkamuodostelmia koskevien tulosten) kanssa, hissien reititysongelman tuntemus paranee, mikä puolestaan edistää ratkaisualgoritmien kehitystä.

Asiasanat:	Hissien reititys, kokonaislukuoptimointi, ylösruuhka, polytooppi, polyhedraalianalyysi
Kieli:	Englanti

Acknowledgements

First and foremost, I wish to thank my instructor Mirko Ruokokoski and my supervisor Professor Harri Ehtamo for their guidance and patience during the writing process. Additionally, I am grateful to Kimmo Berg for his comments and suggestions for improvements regarding this thesis. I also want to express my thanks to the Department of Mathematics and Systems Analysis for letting me to work there for most of the time of my studies: if I had not had an opportunity to study alongisde my job, this Master's thesis would not be completed yet.

During my studies, a huge source of strength and happiness for me has been my friends, who were always ready to listen to my worries and support me whenever I had a hard time. I sincerely thank them, especially Maiju, for being my support all these years. Furthermore, special thanks go to my fellow student Ville for his constant willingness to solve my equations, fix my code, but for also being a guy with whom I could enjoy a pint or two.

Finally, I would like to thank my family for supporting me through life and teaching me the values needed to succeed.

Espoo, April 6, 2016

Vesa Husgafvel

Symbols and Abbreviations

Symbols

P	The set of pickup vertices i, $i = 1, \ldots, n$
A	The set of arcs
Q	The capacity of an elevator
D	The set of delivery vertices $n + i$, $i = 1, \ldots, n$
T	The set of origin depot vertices $2n+e$, $e \in \{e_1, \ldots, e_l\}$
0	The common terminal depot vertex
$+0$	The common origin depot vertex
V	The set of vertices, i.e., the union of P, D, T and $\{0\}$
x_{ij}	The binary decision variable indicating whether arc (i, j) is used or not
τ_{ij}	The travel time between vertices i and j
$f(i)$	The floor of vertex i
$d(i)$	The direction of vertex i
t_i	The time when an elevator starts service at vertex i
q_i	The load of an elevator when it leaves vertex i
γ_i	The elapsed time at vertex i, i.e., the time difference between current moment and the moment request was given
ω_i	The number of passengers entering (exiting) the elevator at vertex i
R	The set of reversal arcs
O	The set of arcs which violate service order constraints
X	A forward path
$A(X)$	The arc set of forward path X
$\mathcal{X}$	The family of all forwards paths
$\mathcal{X}'$	The family of extended forwards paths

$EDP^m_{n,l}$	The up-peak traffic problem of n requests and l elevators whose capacity is m. Shorter notation EDP^m_n is used in the case $l = n$
$H^m_{n,l}$	The set of solutions to $EDP^m_{n,l}$ projected onto to the $\mathbf{x}$-space
$G^m_{n,l}$	The reduced graph of $EDP^m_{n,l}$
$A^m_{n,l}$	The set of arcs in $G^m_{n,l}$
$P_{EDP^m_{n,l}}$	The polytope of $EDP^m_{n,l}$
$P_{EDP^m_{n,l}\|x}$	The polytope of $EDP^m_{n,l}$ restricted to the $\mathbf{x}$-space
$D - EDP^\infty_n$	The down-peak traffic problem of n requests and n elevators of infinite capacity
$H^\infty_{D,n}$	The set of solutions to $D-EDP^\infty_n$ projected onto the $\mathbf{x}$-space
$G^\infty_{D,n}$	The reduced graph of $D - EDP^\infty_n$
$A^\infty_{D,n}$	The set of arcs in $G^\infty_{D,n}$
$P_{D-EDP^\infty_n}$	The polytope of $D - EDP^\infty_n$

Abbreviations

(S_1, S_2)	$\{(i,j) \in A : i \in S_1, j \in S_2\}$
(k, S)	$\{(i,j) \in A : i \in \{k\}, j \in S\}$
$x(S_1, S_2)$	$\sum_{(i,j)\in(S_1,S_2)} x_{ij}$
$\Omega(S)$	$\sum_{i\in S} q_i$
$\|S\|$	The cardinality of set S
$\overline{S}$	$V \setminus S$

Contents

Chapter 1

Introduction

1.1 Background

In modern times, when buildings are getting taller and taller, elevator routing is a problem that is increasingly gaining attention. For example, the tallest building in the world, Burj Khalifa, located in Dubai, Arab Emirates, has a height of 828 meters and contains 154 floors that incorporate 57 elevators: one can imagine how an inefficient elevator control system, in such a building, might lead to passengers' extremely long journey or waiting times.

In the most basic form, an elevator group in a building consists of capacitated single-deck elevators, such that each elevator shaft contains one elevator. In high-rise buildings elevators are usually divided into groups and each elevator group is controlled individually. Traditionally, a passenger calls an elevator at her arrival floor by pressing either an up or down button, indicating the desired direction of travel, after which she gives the destination floor inside the elevator. A more sophisticated alternative for calling an elevator is a *destination control system*, in which up and down buttons are replaced with keypads. By using the keypad, a passenger makes a *transportation request* by giving her destination floor to the device, before entering the elevator. After the transportation request is processed, the device guides the passenger to the right elevator - by processing we refer to the time that it takes from the control system to evaluate which elevator is optimal to serve this particular request. There is also a system where the serving elevator is announced later, e.g., KL 118 in Kuala Lumpur, Malaysia. The time spent on processing should be short, typically less than half a second, because if CPU time is long, the situation could have changed so much that the solution is already outdated. In addition, it is uncomfortable for the passenger if she has to wait for a long time before even knowing, which elevator is going to serve her.

Since new requests are received with varying time-intervals[1], elevator routing is also a dynamic problem. A common way to handle this kind of task is to consider it as a *snapshot problem*: each moment defines a static problem, which is then solved whenever a new request arrives and/or a certain amount of time has passed. In addition to the dynamic nature of the problem, elevator routing contains a degree of stochasticity, since each (transportation) request can be viewed as a 3-dimensional random vector, whose components are arrival time, arrival floor, and destination floor of the request.

Elevator routing is a complicated task, which requires simultaneous fulfillment of several conditions in order to be efficient, practical, and comfortable way to transport people. Naturally, elevators have certain capacity restrictions, and journey times of the passengers cannot be arbitrarily long but some other constraints must be satisfied as well. For example, a common principle is that passengers who are travelling upwards are not guided to elevators that are going downwards, and vice versa. Partially due to the complexity of the problem, it took a long time before a formulation in which all constraints are given in exact mathematical form, was presented in [28] by Ruokokoski et al. In Ruokokoski et al. [29] computational experiments are performed to demonstrate the goodness of this formulation compared to traditional methods that use collective control principle together with other heuristic rules.

1.2 Objectives

In this paper, we define the *elevator dispatching problem* (EDP) and, based on [28], formulate it as a snapshot mixed integer linear program. The readers who are not yet familiar with the terminology, a mixed integer linear program refers to an optimization problem in which objective function and constraints are given in a linear form, such that some of the decision variables are required to be integers. In our formulation, we assume that the elevators are administered by a destination control system, and that each elevator shaft contains only one single-deck elevator. Furthermore, the stochastic nature of the problem will not be considered.

Linear constraints of the EDP formulation form a structure called a polytope. The main purpose of this paper is to analyze this polytope and its properties. Due to the complexity of the problem, the structure of the polytope is very challenging to study, so most of the analysis is restricted to two special cases, an up-peak traffic pattern and a down-peak traffic pattern. An

[1] Alexandris [1] showed that arrival process follows a Poisson process.

up-peak traffic pattern is a situation in which most or all of the passengers travel from the lobby to the upper floors of the building; down-peak traffic is the opposite situation. A typical example of up-peak traffic is the morning peak, when people come to work and they must travel upwards in order to get to their offices; similarly, at the end of the day, a down-peak occurs when people are travelling to the lobby to exit the building. Ruokokoski et al. [28] presented some polyhedral results of one down-peak traffic pattern, but the polyhedral structure of up-peak traffic patterns has not earlier been studied at all. Our aim is to fill in this gap by studying separately three up-peak traffic patterns, where each pattern arises from a different assumption on the number of elevators or their capacity. Although up-peak and down-peak traffic patterns are more or less opposite situations, their polyhedral structures are surprisingly different. In order to see these differences, we also present the main polyhedral results of the down-peak traffic pattern studied by Ruokokoski et al. [28].

1.3 Structure

The rest of this thesis is structured as follows: in Chapter 2 we conduct a short literature review in the elevator dispatching problem and introduce some other integer programming problems that are strongly related to elevator routing. In Chapter 3 we present the main concepts of polyhedral and graph theory and derive some combinatorial results that are needed in later chapters. Chapter 4 considers linear and integer linear programming on a general level: we talk about relaxations, modeling techniques, and the role of polyhedral combinatorics in the field of our study. In that chapter, we also describe some of the most common integer programming algorithms. Chapters 5, 6, and 7 focus entirely on the elevator dispatching problem. The notation of the chapters relating to the EDP is tried to kept as similar as possible as that in [28]. Chapter 8 summarizes the main results of our study.

Chapter 2

Literature Review

2.1 Travelling Salesman Problem

One of the most studied problems in the field of optimization is the *travelling salesman problem* (TSP). In the TSP, a salesman must travel through a list of cities such that each city gets visited exactly once, and finally return to the origin city. The question is, which route minimizes the travelling costs. Although the problem is simple to describe, it is very difficult to solve.

The origins of the TSP are not known, but the first mathematical formulation was presented by W.R. Hamilton and Thomas Kirkman in the 1800s. In the 1950s, the problem grew in popularity among scientific circles, when George Dantzig, Delbert Ray Fulkerson, and Selmer M. Johnson expressed the TSP as an integer linear program and solved a large instance of the problem at that time by using the cutting plane method [8]. [1]

Many different forms of the TSP have been studied during the decades. In the *symmetric travelling salesman problem* (sTSP) the travelling costs between each two cities are the same regardless the direction of travel. In contrast, in the *asymmetric travelling salesman problem* (aTSP), the travelling cost depends on the direction, which is often the situation when we consider the prices of flight tickets. If more than one salesman can visit cities, the problem is known as *multi travelling salesman problem* (mTSP). In the mTSP we can, e.g., set a minimum or maximum number of cities, in which one salesman can visit. A comprehensive presentation of different TSP variations can be found in the work "The Traveling Salesman Problem and Its Variations" [14].

[1] The cutting plane method is introduced in Section 4.5.

2.2 Vehicle Routing Problem

The *vehicle routing problem* (VRP) is a generalization of the TSP with the following setup: there are a number of vehicles available, which must deliver goods to a list of customers, each of them having a certain demand. The goods to be delivered are assumed to be of the same product. By requiring that each vehicle must start and end their route at a depot, the objective is to find a set of routes satisfying customers' demands, such that the total route cost is minimized. It is also usually assumed that splitting a customer's delivery is not possible, i.e., each customer must be visited by exactly one vehicle. The first VRP formulation was made by Dantzig and Ramser [9] in order to optimize the costs of petrol deliveries.

The problem described above is the VRP in its very basic form. As in the case of the TSP, there are numerous variations of the VRP and no consensus exists how to classify them. Pisinger and Ropke [26] propose a classification into five different categories:

1. Vehicle routing problem with time windows (VRPTW). In this variant the customers are associated with time windows within which the visits must be made. A typical example of the VRPTW is pizza delivery, in which the maximum delivery time is often fixed to 30-60 minutes.

2. Capacitated vehicle routing problem (CVRP). In the CVRP each vehicle has limited capacity. Some CVRPs can also have restrictions relating to route lenghts or route durations.

3. Multi-depot vehicle routing problem (MDVRP). The MDVRP contains multiple depots from which vehicles can start their routes.

4. Site-dependent vehicle routing problem (SDVRP). In this variant it is stipulated that certain customers can only be served by certain vehicles.

5. Open vehicle routing problem (OVRP). In the OVRP it is not required that vehicles must return to the depot.

The TSP and VRP are similar problems, but the latter incorporates a wider variety of problems; indeed, the VRP of m uncapacitated vehicles can be identified with the mTSP. In certain fields, such as in the logistics industry, transportation optimization can significantly lower the total costs, and as a result, the VRP has been a hot research topic for the past 50 years. Literature reviews on the VRP have been conducted, inter alia, by Laporte and Norbert [20] and Laporte and Osman [21]. The recent developments and publications regarding the subject are covered by Kumar and Panneerselvam [19].

2.3 Pickup and Delivery Problem

In the vehicle routing problem it was assumed that vehicles start their routes from the depot, after which they can deliver the goods by driving straight to the customers. If the goods to be delivered are not in stock, it requires that vehicles must first leave the depot, pick up the goods from somewhere, and then drive back to the depot before the VRP model can be applied. In such cases, it would be more efficient if the goods could be delivered to the customers straight from the pickup places.

In the *pickup and delivery problem* (PDP) each transportation request comprises a pickup place and a destination: a vehicle must first pick up the goods and then drive them to the destination. We require that vehicles must start and end their routes at the depot. The VRP can be considered as a special case of the PDP, in which all pickup places locate at the depot. The objective in the PDP is to divide the transportation requests between vehicles so that the total costs will be minimized.

As in the case of the TSP and the VRP, there are also several extensions for the PDP. E.g., Savelsberg and Sol [30] presented the *general pickup and delivery problem* (GPDP), which consists of a set of depots and allows the vehicles to start or end their routes at any of them. Cortes et al. [6] introduced and formulated another variant known as the *pickup and delivery problem with transfers* (PDPT), in which transferring loads between vehicles is allowed. In the case when the "goods" to be delivered are people, the PDP is called the *dial-a-ride problem* (DARP). The DARPs usually incorporate restrictions on the passengers pickup and delivery times. Additionally, the transportation time of a single passenger is required to stay below a certain maximum limit. A typical example of the DARP is a door-to-door transportation of elderly or disabled people.

2.4 Elevator Dispatching Problem

The elevator dispatching problem (EDP) is a dial-a-ride problem, in which a set of passengers have to be picked up from their arrival floors and transported to their destination floors, so that a given objective function is minimized and a set of constraints are satisfied. Usually, these constraints relate to travel time, capacity of elevators and service order of passengers. Their point is to guarantee that all passengers are delivered to their destination floors as efficiently as possible.

Different forms of the EDP appear in the literature, although elevator routing has historically gained relatively little attention compared to other similar problems, such as the VRP. Elevator routing that is based on a destination control system, is one of the most common variants and also the variant, which we chose for our study. Routing an elevator in a conventional system is difficult as the destinations of passengers are not known in advance. In addition to Ruokokoski et al. [28, 29], destination control systems have been studied by, e.g., Koehler and Ottiger [18] and Tanaka et al. [33].

In some works relating to the EDP it has been assumed that elevator shafts contain more than one car per shaft. If the cars are attached together, passengers at consecutive floors can be served simultaneously, and if the cars are separate, they can move independently as long as collisions are avoided. The former case is known as the multi-deck EDP[16, 32] and the latter as the multi-car EDP [17].

In terms of solving the EDP, most of the research is based on heuristic methods such as artificial intelligence [31], neural networks [23], or local search [22], whereas exact algorithms have been considered only in few articles [15, 28, 33]. The reason for popularity of heuristic methods is due to the complexity of the EDP, the real-time requirement that requests must be responded very quickly but also the lack of a proper mathematical formulation: as mentioned in Chapter 1, the first - and so far only - a complete mathematical formulation for the EDP is provided by Ruokokoski et al. [28].

The relationship between the TSP, VRP, PDP, DARP, and EDP is illustrated in Figure 2.1.

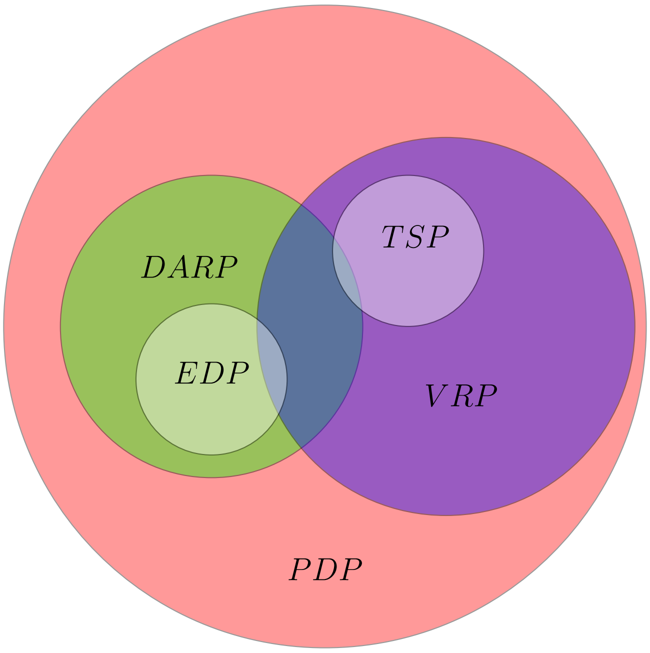


Figure 2.1: A Venn diagram representing the relationship between the TSP, VRP, PDP, DARP, and EDP, where each class refers to the problem in its basic form.

Chapter 3

Polyhedral and Graph Theory

3.1 Polyhedral Theory

A *linear combination* of vectors $\mathbf{x}_i \in \mathbb{R}^n$, $i = 1, \ldots, m$, is a weighted sum

$$\lambda_1 \mathbf{x}_1 + \cdots + \lambda_m \mathbf{x}_m, \tag{3.1}$$

where $\lambda_i \in \mathbb{R}$, $i = 1, \ldots, m$. If $\sum_{i=1}^m \lambda_i = 1$, the linear combination is called *affine*. A set S is affine if for each $\mathbf{x}, \mathbf{y} \in S$, it follows that $\lambda \mathbf{x} + (1-\lambda)\mathbf{y} \in S$. From this definition it follows that affine sets are either lines or (hyper)planes. The set of all affine combinations of elements of $S \subset \mathbb{R}^n$ form the *affine hull* of S, which is denoted by $\text{aff}(S)$. Clearly, for any affine set S, $\text{aff}(S) = S$.

A set of vectors $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ is called *linearly independent*, if none of the vectors can be represented as a linear combination of others. Formally,

$$\lambda_1 \mathbf{x}_1 + \cdots + \lambda_m \mathbf{x}_m = \mathbf{0} \Rightarrow \lambda_i = 0 \;\; \forall i = 1, \ldots, m. \tag{3.2}$$

Vectors $\{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_m\}$ are *affinely independent* if $\mathbf{x}_1 - \mathbf{x}_0, \ldots, \mathbf{x}_m - \mathbf{x}_0$ are linearly independent. Linearly independent vectors are always affinely independent, but (in general) not vice versa. If S is a set, which contains $k+1$ but not $k+2$ affinely independent vectors, we say that the *dimension* of S, $\dim(S)$, is k.

When the linear combination (3.1) satisfies conditions $\sum_{i=1}^m \lambda_i = 1$ and $\lambda_i \geq 0$, $\forall i \in \{1, \ldots, m\}$, it is called a *convex combination.* A set S is convex if for each $\mathbf{x}, \mathbf{y} \in S$, and $\lambda \in [0, 1]$, it follows that $\lambda \mathbf{x} + (1-\lambda)\mathbf{y} \in S$. All points in a convex set can be represented as a convex combination of the other points. Geometrically convexity means that the segment joining any two points of a convex set lies entirely within that set. The *convex hull* of an set S, $\text{conv}(S)$, is the smallest convex set containing S. If S is convex, $\text{conv}(S) = S$.

Convexity plays a central role in linear optimization problems, in which the feasible region of solutions is always of the same form: it consists of the intersection of finitely many hyperplanes. Let us give the following definition.

Definition 1. A set $P \subset \mathbb{R}^n$ is a *polyhedron* if it can be represented as the intersection of finitely many half-spaces, i.e.

$$P = \{\mathbf{x} \in \mathbb{R}^n | \mathbf{A}\mathbf{x} \leq \mathbf{b}\}, \mathbf{A} \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^m \tag{3.3}$$

If a polyhedron is bounded, meaning that it fits inside a ball of finite radius, then the polyhedron is called a *polytope*[1] From the point of view of this study, all interesting polyhedra are bounded, and for this reason, forthcoming definitions will be based on polytopes. It is relatively easy to show that each polytope is a convex set [3]. A less trivial fact relates to the "corners" of a polytope. In order to state this result, we first need to define, what a corner means.

Definition 2. Let P be a polytope. A vector $\mathbf{x} \in P$ is an *extreme point* of P if there are no vectors $\mathbf{y}, \mathbf{z} \in P$, $\mathbf{y}, \mathbf{z} \neq \mathbf{x}$, and a scalar $\lambda \in [0, 1]$, such that $\mathbf{x} = \lambda\mathbf{y} + (1 - \lambda)\mathbf{z}$.

Next result shows that a polytope is uniquely determined by its extreme points.

Theorem 1. Let X be the set of extreme points of a non-empty polytope P. Then $\text{conv}(X) = P$.

Proof. See [3]. □

If all the extreme points of P are integers, we say that P is an *integral polytope*. If the extreme points are binary valued, P is a *0-1-polytope*.

A polytope comprises elements of different dimensionality. To see, what this means, consider a polytope P defined by (3.3). A *valid inequality* for P is a linear inequality $\mathbf{d}'\mathbf{x} \leq \mathbf{e}$, where $\mathbf{d}, \mathbf{e} \in \mathbb{R}^n$, if all points of P satisfy that inequality. A valid inequality $\mathbf{d}'\mathbf{x} \leq \mathbf{e}$ is *supporting* if for some point $\mathbf{x}_0 \in P$ it holds $\mathbf{d}'\mathbf{x}_0 = \mathbf{e}$. A set $F \subset P$ whose points satisfy this equality, i.e.,

$$F = \{\mathbf{x} \in P | \mathbf{d}'\mathbf{x}_0 = \mathbf{e}\}, \tag{3.4}$$

is called a *face* of P; we say that F is *induced* by the supporting inequality $\mathbf{d}'\mathbf{x} \leq \mathbf{e}$. One should note that P itself is a face of P, because $\mathbf{0}'\mathbf{x} \leq \mathbf{0}$ is a supporting inequality for any polytope. A 0-dimensional face of P is called a

[1] A reader should note that different definitions of a polytope appear in literature. In this work, we stick with the given definition.

vertex, whereas a 1-dimensional face is known as an *edge*. Suppose now that P is a k-polytope, i.e., the dimension of P is k. Then a $(k-1)$-dimensional face, $(k-1)$-face, is called a *facet*.

An inequality $\mathbf{a}_i'\mathbf{x} \leq \mathbf{b}_i$ in the system $\mathbf{Ax} \leq \mathbf{b}$ is called an *implicit equality* if all feasible solutions of $\mathbf{Ax} \leq \mathbf{b}$ satisfy $\mathbf{a}_i'\mathbf{x} = \mathbf{b}_i$. The subsystem of all implicit equalities is denoted by $(\mathbf{A}^=, \mathbf{b}^=)$. The *rank* of $\mathbf{A}^=$, $\text{rank}(\mathbf{A}^=)$, indicates the number of linearly independent rows or columns in $\mathbf{A}^=$. Next theorem shows how $\dim(P)$ depends on the implicit equalities of P.

Theorem 2. Let $P = \{\mathbf{x} \in \mathbb{R}^n | \mathbf{Ax} \leq \mathbf{b}\}$ be a non-empty polyhedron. Then

$$\dim(P) = n - \text{rank}(\mathbf{A}^=) \tag{3.5}$$

Proof. See [7]. □

When $\mathbf{A}$ is large in size, it may be difficult to determine or even estimate the dimension of P just by trying to find as many affinely independent solutions as possible. All implicit equalities are not usually known, so calculating $\text{rank}(\mathbf{A}^=, \mathbf{b}^=)$ is not possible either. We can, however, create a subsystem of $(\mathbf{A}^=, \mathbf{b}^=)$, say $(\mathbf{A}_S^=, \mathbf{b}_S^=)$, by choosing all known implicit equalities. Since $(\mathbf{A}_S^=, \mathbf{b}_S^=) \subset (\mathbf{A}^=, \mathbf{b}^=)$, it holds that $\text{rank}(\mathbf{A}_S^=) \leq \text{rank}(\mathbf{A}^=)$. Now by using Theorem 2, one gets an upper bound for the dimension of P:

$$\dim(P) \leq n - \text{rank}(\mathbf{A}_S^=) \tag{3.6}$$

If we can now find $n - \text{rank}(\mathbf{A}_S^=) + 1$ affinely independent solutions for P, the obtained upper bound is strict. Unfortunately, showing a set of vectors affinely or linearly independent is very difficult in general. For this reason, the proving is often based on the specific structure of the constraint matrix to be studied. In our study, the following proposition is found useful:

Proposition 1. Let $\{\mathbf{x}^1, \ldots, \mathbf{x}^m\}$ be a set of $n-$vectors, $n \geq m$, and denote the ith entry of $\mathbf{x}^j$ by x_i^j. If there exists a sequence of ordered pairs $(i_1, j_1), \ldots, (i_m, j_m)$, $i_k \in \{1, \ldots, n\}, j_k \in \{1, \ldots, m\}$ such that for each $k = 1, \ldots, m$

1) $x_{i_k}^{j_k} \neq 0$, and

2) $x_{i_k}^{j} = 0 \; \forall j \notin \{j_1, \ldots, j_k\}$,

then $\mathbf{x}^1, \ldots, \mathbf{x}^m$ are linearly independent.

Proof. It is obvious that if $\mathbf{x}^{j_1}, \dots, \mathbf{x}^{j_m}$ are linearly independent, then $\mathbf{x}^1, \dots, \mathbf{x}^m$ are also linearly independent. Consider a linear combination $\lambda_1 \mathbf{x}^{j_1} + \cdots + \lambda_m \mathbf{x}^{j_m}$ for which it holds that $\sum_{k=1}^m \lambda_k \mathbf{x}^{j_k} = \mathbf{0}$. Now,

$$\left(\sum_{k=1}^m \lambda_k \mathbf{x}^{j_k} \right)_{i=i_1} = \sum_{k=1}^m \lambda_k x_{i_1}^{j_k} = \lambda_1 x_{i_1}^{j_1} + \sum_{k=2}^m \lambda_k \underbrace{x_{i_1}^{j_k}}_{=0 \text{ by 2)}} = \lambda_1 x_{i_1}^{j_1} = 0.$$

But since $x_{i_1}^{j_1} \neq 0$ by the first assumption, λ_1 must be 0. Similarly,

$$\left(\sum_{k=1}^m \lambda_k \mathbf{x}^{j_k} \right)_{i=i_2} = \sum_{k=1}^m \lambda_k x_{i_2}^{j_k} = \lambda_1 x_{i_2}^{j_1} + \lambda_2 x_{i_2}^{j_2} + \sum_{k=2}^m \lambda_k x_{i_2}^{j_k} = \lambda_2 x_{i_2}^{j_2} = 0$$

implies that $\lambda_2 = 0$. By continuing the process, one obtains $\lambda_1 = \cdots = \lambda_m = 0$, which shows that $\mathbf{x}^{j_1}, \dots, \mathbf{x}^{j_m}$ are linearly independent. □

For example, if a matrix is defined by

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 0 & 1 & 2 & 0 \\ 4 & 1 & 2 & 3 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$

we can show that its columns are linearly independent by choosing $(i_1, j_1) = (4, 2), (i_2, j_2) = (2, 3), (i_3, j_3) = (1, 4), (i_4, j_4) = (3, 1)$, and then applying Proposition 1. Another useful way to determine the rank of a matrix is given by Proposition 2.

Proposition 2. Let $\mathbf{Ax} = \mathbf{b}$ be a system of linear equations $\mathbf{a}_i' \mathbf{x} = b_i$, where $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, and $n > m$. If $\mathbf{Ax} = \mathbf{b}$ has a solution, and there exists vectors $\mathbf{x}^i \in \mathbb{R}^n$, $i = 1, \dots, m$, such that $\mathbf{a}_k' \mathbf{x}^i = b_k \ \forall k \neq i$ but $\mathbf{a}_i' \mathbf{x}^i \neq b_i$, then $\text{rank}(\mathbf{A}) = m$.

Proof. We show first that vectors $[\mathbf{a}_i', -b_i]$, $i = 1, \dots, m$, are linearly independent. Assume, by contradiction, that this is not true, in which case one of the vectors can be represented as a linear combination of the others. Formally, for some $k \in \{1, \dots, m\}$ $\exists$ $\lambda \in \mathbb{R}^{m-1}$ such that

$$[\mathbf{a}_k', -b_k] = \sum_{i \in \{1, \dots, m\} : i \neq k} [\mathbf{a}_i', -b_i] \lambda_i.$$

By taking the dot product with a vector $[\mathbf{x}^{k'}, 1]$, we obtain

$$
\begin{aligned}
[\mathbf{a}_k', -b_k] \cdot [\mathbf{x}^{k'}, 1] &= \sum_{i \in \{1,\dots,m\}: i \neq k} [\mathbf{a}_i', -b_i] \cdot [\mathbf{x}^{k'}, 1] \lambda_i \\
\Leftrightarrow \quad \mathbf{a}_k' \mathbf{x}^k - b_k &= \sum_{i \in \{1,\dots,m\}: i \neq k} \underbrace{(\mathbf{a}_i' \mathbf{x}^k - b_i)}_{=0} \lambda_i = 0 \\
\Leftrightarrow \quad \mathbf{a}_k' \mathbf{x}^k &= b_k,
\end{aligned}
$$

which is a contradiction. Hence, $[\mathbf{a}_i', -b_i]$, $i = 1, \dots, m$, are linearly independent and $\text{rank}([\mathbf{A}, -\mathbf{b}]) = m$. But since $\mathbf{Ax} = \mathbf{b}$ has a solution, vector $\mathbf{b}$ can be represented as a linear combination of the columns of $\mathbf{A}$, in which case $\text{rank}([\mathbf{A}, -\mathbf{b}]) = \text{rank}(\mathbf{A}) = m$, and the claim follows.

□

3.2 Graph Theory

A *directed graph*, or simply a digraph, $G = (V, A)$ is an ordered pair, where V is a set of *vertices* and A is a set of *arcs* that are ordered pairs of elements of V. An arc between vertices i and j is denoted by (i, j). If arcs are defined as unordered pairs of V, then G is called an *undirected graph*. In such a case we talk about *edges* instead of arcs, and they are denoted by E. An edge between vertices i and j is denoted by $\{i, j\}$. A *loop* is an arc, which connects a vertex to itself, i.e., a pair containing the same element twice. If a graph contains no loops it is called a *simple graph*. A simple undirected graph in which every pair of distinct vertices is connected by a unique arc is called a *complete undirected graph*. A *complete digraph* is defined similarly with the exception that it contains a unique pair of arcs for each pair of vertices, one for both directions.

A *walk* in a graph is a sequence $(v_0, a_1, v_1, a_2, \dots, a_n, v_n)$, where a_i, $i = 1, \dots, n$, is an arc connecting v_{i-1} and v_i, respectively. If $v_0 = v_n$, then the walk is called a *cycle*. A graph which contains no cycles is an *acyclic graph*. A walk is called a *path* if all its vertices are distinct; a walk is called a *trail* if all its arcs are distinct. If the first and the last vertices are the same, but all other vertices are distinct, then a walk is a *closed path*, which is also known as a *circuit*. If each vertex pair (v_{i-1}, v_i), $i = 1, \dots, n$, of a path is connected by a unique arc in the underlying graph, then the path can be denoted by a shorter notion $(v_0, v_1, \dots, v_n)$ without the possibility of a misunderstanding. A graph is *connected* if for any $v, q \in V$ there is an *undirected walk* from v to q, i.e. a walk in which the directions of arcs are ignored. If there is a walk between each pair of vertices, then the graph is *strongly connected*. An acyclic connected, undirected graph is a *tree*. Different graphs are illustrated in Figure 3.1.

Given a directed graph $G = (V, A)$ and a set $S \subset V$, the *cutset* of S, $\delta(S)$, contains arcs whose one endpoint belongs to S and the other one to $\overline{S}$. Formally,

$$\delta(S) = \big\{(i, j) \in A | i \in S, j \notin S \text{ or } i \notin S, j \in S\big\}. \tag{3.7}$$

The set of arcs whose both endpoints lie in S is denoted by $\rho(S)$, i.e.,

$$\rho(S) = \big\{(i, j) \in A | i, j \in S\big\}. \tag{3.8}$$

These sets are also well-defined when G is an undirected graph: one simply needs to replace (i, j) with $\{i, j\}$.

Denote the vertex set of G by $v(G)$ and the arc set by $\alpha(G)$. A graph H is a *subgraph* of G if $v(H) \subset v(G)$ and $\alpha(H) \subset \alpha(G)$. We say that H is a *vertex-induced subgraph* of G if $v(H) \subset v(G)$ and H contains all the arcs of G whose both endpoints are in H, i.e. $\alpha(H) = \{(i, j) \in \alpha(G) | i \in v(H), j \in v(H)\}$. Similarly, H is an *arc-induced subgraph* if $\alpha(H) \subset \alpha(G)$ and H contains all the vertices, which are the endpoints of the arc set $\alpha(H)$, i.e. $v(H) = \{i \in v(G) | \alpha(H) \subset (i, v(G)) \cup (v(G), i)\}$.

3.3 Combinatorics

3.3.1 Selections

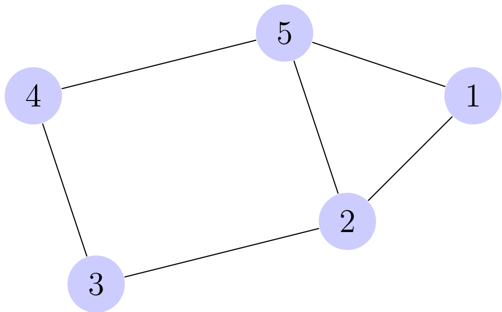
The fundamentals of combinatorics relate to the question in how many ways k objects can be selected from a set of n objects. To answer this question properly, we must first define whether the order in which the objects are selected is significant or not, and can the same object be selected more than once. This leaves four cases to consider:

1. Order not significant and repetitions not allowed. Such a selection is called a *k-combination* of an n-set.

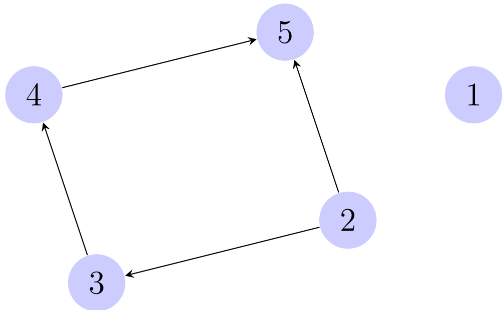
2. Order not significant and repetitions allowed. Such a selection is called a *k-multicombination* of an n-set.

3. Order significant and repetitions not allowed. Such a selection is called a *k-permutation* of an n-set.

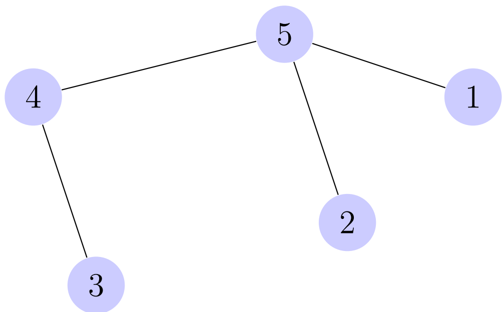
4. Order significant and repetitions allowed. Such a selection is called a *k-tuple* of an n-set.



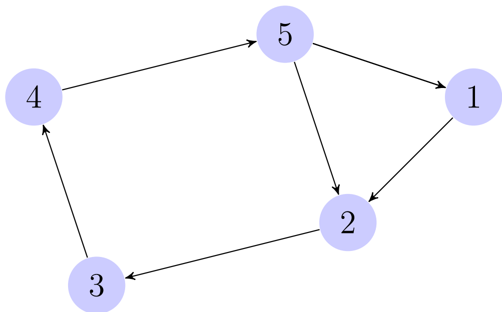
(a) A simple connected, undirected graph



(b) A simple disconnected digraph



(c) A tree



(d) A strongly connected graph

Figure 3.1: Four different graphs defined over same vertex set $V = \{1, \ldots, 5\}$.

The next theorem answers our question in all four cases.

Theorem 3.

1. The number of k-combinations of an n-set is $\binom{n}{k} := \frac{n!}{k!(n-k)!}$, where $n! = n * (n-1) * \cdots * 2 * 1$ is the *factorial* of n.
2. The number of k-multicombinations of an n-set is $\binom{n+k-1}{k}$.
3. The number of k-permutations of an n-set is $\frac{n!}{(n-k)!}$.
4. The number of k-tuples of an n-set is n^k.

Proof. We prove the claims in order 4, 3, 1, and 2. The claim number 4 is obvious because the first object can be selected in n ways, the second object can also be selected in n ways etc., leaving the number of k-tuples as $\underbrace{n * \cdots * n}_{k \text{ times}} = n^k$. When repetitions are not allowed but the order is significant, the first object can be selected in n ways, the second object in $n-1$ ways, and the kth object in $n-(k-1)$ ways. Hence, the number of k-permutations is $n * (n-1) * \cdots * (n-(k-1)) = n!/(n-k)!$. Since selected k objects arise from $k!$ different orders, the number of k-combinations is obtained by dividing the number of k-permutations by $k!$.

The claim number 2 is a bit trickier. Let $x_i \geq 0$, $i = 1, \ldots, n$ be the number of times object i gets chosen. As k objects are selected in total, it must hold that $\sum_{i=1}^{n} x_i = k$, which means that the number of ways of choosing n non-negative integers x_i whose sum is k equals to the number of k-multicombinations. Suppose next that we put $n+k-1$ boxes in a row, and we place a ball in $n-1$ of them, such that each box can contain at most one ball. Let x_1 now denote the number of empty boxes before the first one that contains a ball. Let x_i, $2 \leq i \leq n-1$, be the number of empty boxes between the $(i-1)$st and ith balls, and x_n the number of empty boxes after the $(n-1)$st ball. As $n-1$ boxes out of $n+k-1$ boxes contain a ball, it must hold that $\sum_{i=1}^{n} x_i = (n+k-1)-(n-1) = k$. The number of ways to choose the boxes, where the balls are to be placed in, can be chosen by $\binom{n+k-1}{n-1}$ ways. Since

$$\begin{aligned}\binom{n+k-1}{n-1} &= \frac{(n+k-1)!}{(n-1)!(n+k-1-(n-1))!} = \frac{(n+k-1)!}{(n+k-1-k)!k!} \\ &= \binom{n+k-1}{k},\end{aligned}$$

the claim is proved. □

3.3.2 Principle of Inclusion and Exclusion

Suppose we are given three finite sets A, B, and C, whose elements are known, and we would like to know the number of the elements lying in the union of the sets. Let the *cardinality* of a set, i.e., the number of the elements in that set, be denoted by $|\cdot|$. If the sets are *disjoint*, i.e., $A \cap B = A \cap C = B \cap C = \emptyset$, the cardinality of the union is simply $|A \cup B \cup C| = |A| + |B| + |C|$. In general case this formula is not valid, because the elements in the sets $A \cap B$, $A \cap C$, and $B \cap C$ are counted twice. Hence, the cardinalities of these sets must be subtracted from $|A| + |B| + |C|$. However, if $A \cap B \cap C$ is not empty, this subtraction results in that the elements in $A \cap B \cap C$ will not be counted anymore. Hence, the cardinality of that set must be added after the subtraction, which leads to a formula

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|. \quad (3.9)$$

The situation is illustrated in Figure 3.2.

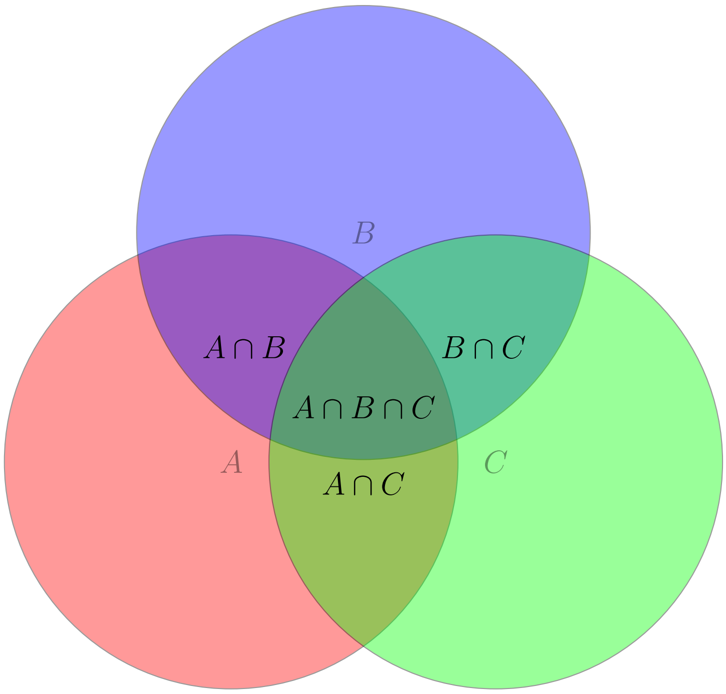


Figure 3.2: A Venn diagram of three intersecting sets.

If sets A, B, and C are denoted by A_1, A_2, and A_3 the formula (3.9) can be written in more compact form

$$\left| \bigcup_{i=1}^{3} A_i \right| = \sum_{\emptyset \neq J \subset \{1,2,3\}} (-1)^{|J|-1} \left| \bigcap_{j \in J} A_j \right|. \quad (3.10)$$

The next theorem generalizes this result:

Theorem 4. Let $(A_1, \ldots, A_n)$ be a family of subsets of X. Then the number of elements lying in the union of A_i's can be counted by the formula

$$\left|\bigcup_{i=1}^{n} A_i\right| = \sum_{\emptyset \neq J \subset \{1,\ldots,n\}} (-1)^{|J|-1} \left|\bigcap_{j \in J} A_j\right| \tag{3.11}$$

Proof. The claim follows easily from the Binomial Theorem but since we do not present it in this work, we skip the proof. See, e.g., [4]. □

3.3.3 Stirling, Bell, and Lah Numbers

Stirling numbers arise in several combinatorial problems. Two different sets of numbers are named after James Stirling, who introduced them in the 18th century. The *Stirling numbers of the first kind*, $s(n, k)$, are the integer coefficients in the *falling factorial expansion* defined by

$$(x)_n := \prod_{k=0}^{n-1} (x - k) = \sum_{k=0}^{n} s(n, k) x^k, \tag{3.12}$$

where $x \in \mathbb{R}$. The reason why these numbers play important role in combinatorics, is that the absolute value of $s(n, k)$ represents the number of permutations of n elements with k cycles. [2]

In terms of this thesis, more interesting numbers are the *Stirling numbers of the second kind* $S(n, k)$, which indicate the number of ways to partition a set of n objects into k non-empty subsets. This property makes them extremely useful for our later purposes. It is quite straightforward to show that $S(n, k)$ can be characterized by the equation

$$\sum_{k=0}^{n} S(n, k)(x)_k = x^n, \tag{3.13}$$

where $(x)_k$ is the kth falling factorial of $x \in \mathbb{R}$. By combining the characterizations (3.12) and (3.13), we see a strong connection between the Stirling

[2] In combinatorics, a cycle means a subset of a permutation whose elements trade places with one another. Due to technical reasons, we do not give a formal definition.

numbers of the first and second kind:

$$\begin{aligned}
&\sum_{k=0}^{n} S(n,k)(x)_k = \sum_{k=0}^{n} S(n,k) \sum_{m=0}^{k} s(k,m)x^m = x^n \ \ \forall x \in \mathbb{R} \\
\Leftrightarrow &\sum_{k=0}^{n} \sum_{m=0}^{\min\{k,n-1\}} S(n,k)s(k,m)x^m + \big(S(n,n)s(n,n) - 1\big)x^n = 0 \ \ \forall x \in \mathbb{R} \\
\Rightarrow &\sum_{k=0}^{n} S(n,k)s(k,m) = \delta_{nm},
\end{aligned} \tag{3.14}$$

where δ_{nm} is the *Kronecker delta*, a function that takes the value of 1 if $n = m$, and 0 otherwise.

Next, we will present an explicit formula for calculating the Stirling numbers of the second kind. We need the following proposition:

Proposition 3. The number of surjections from a set of n elements to a set of k elements is given by

$$\sum_{i=0}^{k} (-1)^i \binom{k}{i} (k-i)^n. \tag{3.15}$$

Proof. Let X be the set of all mappings from $N := \{1, \ldots, n\}$ to $\{1, \ldots, k\}$. Clearly, $|X| = k^n$. Define A_i, for each $i \in \{1, \ldots, k\}$, to be the set of mappings f for which $i \notin f(N)$. The image of each element in N can now be chosen in $k-1$ different ways. As there are n elements in N, $|A_i| = (k-1)^n$. Similarly, if $A_I := \bigcap_{i \in I} A_i$, $I \subset \{1, \ldots, k\}$, denotes the set of mappings for which $I \not\subset f(N)$, we have that $|A_I| = (k - |I|)^n$.

By the definition of a surjection, all elements in $\{1, \ldots, k\}$ must have a preimage, and hence a surjection cannot belong to any of the sets A_i. By using De Morgan's laws and Theorem 4, we find that the number of surjections is equal to

$$\left| \bigcap_{i=1}^{k} \overline{A}_i \right| = \left| \overline{\bigcup_{i=1}^{k} A_i} \right| = \left| X \setminus \bigcup_{i=1}^{k} A_i \right| = |X| - \sum_{\emptyset \neq I \subset \{1,\ldots,k\}} (-1)^{|I|-1} |A_I|.$$

There are $\binom{k}{|I|}$ sets whose cardinality is $|I|$, so we can let $|I|$ run from 1 to k, which gives

$$
\begin{aligned}
|X| - \sum_{\emptyset \neq I \subset \{1,\ldots,k\}} (-1)^{|I|-1} |A_I| &= k^n - \sum_{|I|=1}^{k} (-1)^{|I|-1} \binom{k}{|I|} (k-|I|)^n \\
= \binom{k}{0} (-1)^0 (k-0)^n + \sum_{|I|=1}^{k} (-1)^{|I|} \binom{k}{|I|} (k-|I|)^n \\
= \sum_{i=0}^{k} (-1)^i \binom{k}{i} (k-i)^n,
\end{aligned}
$$

and the claim follows. □

Proposition 4. The Stirling number of the second kind $S(n,k)$ is equal to

$$S(n,k) = \frac{1}{k!} \sum_{j=1}^{k} (-1)^{k-j} \binom{k}{j} j^n. \tag{3.16}$$

Proof. Each surjection from $\{1,\ldots,n\}$ to $\{1,\ldots,k\}$ defines a partition of a set of n elements into k non-empty subsets. Since the order of the parts in a partition is not relevant, the same partition arises from $k!$ surjections, i.e., the number of surjections is $S(n,k)k!$. On the other hand, the same number is given by (3.15), which results in

$$
\begin{aligned}
S(n,k)k! &= \sum_{i=0}^{k} (-1)^i \binom{k}{i} (k-i)^n, \quad i \to k-j \\
\Leftrightarrow \quad S(n,k) &= \frac{1}{k!} \sum_{j=0}^{k} (-1)^{k-j} \binom{k}{k-j} j^n = \frac{1}{k!} \sum_{j=1}^{k} (-1)^{k-j} \binom{k}{j} j^n
\end{aligned}
$$

□

As an immediate consequence of Proposition 4, the number of (all possible) partitions of a set of n elements is

$$B_n = \sum_{k=0}^{n} S(n,k) = \sum_{k=0}^{n} \frac{1}{k!} \sum_{j=1}^{k} (-1)^{k-j} \binom{k}{j} j^n. \tag{3.17}$$

The numbers B_n, $n = 1, 2, \ldots$, are known as the *Bell numbers* named after Eric Temple Bell, who studied them in the 1930s.

In addition to the Stirling numbers of the second kind, which give the answer in how many ways n objects can be partitioned into k non-empty subsets, we are also interested to know in how many ways n objects can be partitioned into k linearly ordered non-empty subsets. The number of such partitions is called the *Lah number* $L(n, k)$. The Lah numbers can be calculated by using the formula of Proposition 5.

Proposition 5. The Lah number $L(n, k)$ is equal to

$$L(n,k) = \binom{n-1}{k-1}\frac{n!}{k!}. \tag{3.18}$$

Proof. We first choose a permutation of n elements, which can be done by $n!$ ways. Then we slice the permutation into k parts by choosing $k-1$ cut points out of $n-1$ possible cut points. This can be done in $\binom{n-1}{k-1}$ ways. When we take into account that the order of the parts is not relevant, i.e., we divide by $k!$, the desired result follows from the multiplicative principle of independent events. □

The Stirling numbers and the Lah numbers are connected by a relation

$$L(n,k) = \sum_{j=1}^{n} s(n,j)S(j,k), \tag{3.19}$$

for which reason the Lah numbers are sometimes referred to as the Stirling numbers of the third kind.

The importance of the Stirling, Bell, and Lah numbers for our study is seen in chapters 6 and 7, when we determine the number of feasible solutions to different EDP traffic patterns.

Chapter 4

Integer Programming

4.1 Classification of Optimization Problems

Discrete optimization problems, such as the VRP, are often formulated via methods of linear programming. By linear programming we mean that a linear cost function is minimized (or maximized) with respect to linear constraints. Linear optimization problems are well-studied and there are numerous algorithms, which are developed to solve different kinds of linear problems. In nonlinear programming the situation is crucially different because algorithms of that field are usually able to find only local exreme values. When optimization problems involve integer constraints, linear modeling is even more important, since integer optimization problems are in general hard to solve, and non-linear modeling would only add to the complexity. In this thesis, we focus only on linear modeling.

Consider a general *linear optimization problem*, LP,

$$
\begin{aligned}
z_{LP} \quad = \quad & \min \quad \mathbf{c}'\mathbf{x} \\
& \text{s.t.} \quad \mathbf{A}\mathbf{x} \leq \mathbf{b},
\end{aligned}
\tag{4.1}
$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^n$, $\mathbf{c} \in \mathbb{R}^m$, and $\mathbf{x} \in \mathbb{R}^n$. If solutions are constrained to be integer-valued, i.e., $\mathbf{x} \in \mathbb{Z}^n$, the problem is called an *integer linear optimization problem*, ILP. In *mixed integer linear problems*, MILP, an integer restriction is set on some of the decision variables, and in *binary linear optimization problems*, BILP, decision variables take on the values 0 or 1. In the case of ILPs and BILPs, the elements of matrix $\mathbf{A}$ and coefficients $\mathbf{b}$ and $\mathbf{c}$ are usually integers or rational numbers.

Many linear programming algorithms require that decision variables are non-negative and for the linear constraints to be in equality form. This is not a real problem, since such a transformation can always be done. The decision variable $\mathbf{x}$ can be written as a sum of two non-negative variables, i.e., $\mathbf{x} = \mathbf{x}^+ - \mathbf{x}^-$, where $\mathbf{x}^+, \mathbf{x}^- \geq \mathbf{0}$. The inequality constraint $\mathbf{Ax} \leq \mathbf{b}$ can be expressed as an equality constraint by introducing a *slack variable* $\mathbf{s}^+ \geq \mathbf{0}$ and by setting $\mathbf{Ax} + \mathbf{s}^+ = \mathbf{b}$. By making the substitutions $\mathbf{y} = [\mathbf{x}^+, \mathbf{x}^-, \mathbf{s}^+]$, $\mathbf{f} = [\mathbf{c}, -\mathbf{c}, \mathbf{0}]$, and $\mathbf{D} = [\mathbf{A}, -\mathbf{A}, \mathbf{I}]$, where $\mathbf{I}$ denotes the identity matrix, the following formulation is obtained:

$$\begin{aligned} z_{LP} \quad = \quad \min \quad & \mathbf{f}'\mathbf{y} \\ \text{s.t.} \quad & \mathbf{Dy} = \mathbf{b}, \\ & \mathbf{y} \geq \mathbf{0}. \end{aligned} \tag{4.2}$$

This formulation is called the *standard form* of a linear programming problem.

4.2 Relaxations

Consider a minimization problem

$$z = \min\{c(\mathbf{x}) |\ \mathbf{x} \in X \subset \mathbb{R}^n\}, \tag{4.3}$$

where c is a function $c : X \rightarrow \mathbb{R}$. A *relaxation* of the problem (4.3) is any minimization problem

$$z_R = \min\{c_R(\mathbf{x}) |\ \mathbf{x} \in X_R \subset \mathbb{R}^n\} \tag{4.4}$$

satisfying conditions
1) $X_R \supseteq X$ and
2) $c_R(\mathbf{x}) \leq c(\mathbf{x})\ \forall \mathbf{x} \in X$.
According to these conditions, it is easy to see that $z_R \leq z$ as $z_R = \min\{c_R(\mathbf{x}) | \mathbf{x} \in X_R \subset \mathbb{R}^n\} \leq \min\{c_R(\mathbf{x}) | \mathbf{x} \in X \subset \mathbb{R}^n\} \leq \min\{c(\mathbf{x}) | \mathbf{x} \in X \subset \mathbb{R}^n\} = z$. Moreover, if $\mathbf{x}_R^*$ is an optimal solution to (4.4) such that $\mathbf{x}_R^* \in X$ and $c_R(\mathbf{x}_R^*) = c(\mathbf{x}_R^*)$, then $\mathbf{x}_R^*$ is also optimal to (4.3).

From the standpoint of this thesis, the most important relaxation is a linear programming relaxation, which means the removal of integer constraints. If an ILP is defined by

$$z_{ILP} = \min\{\mathbf{c}'\mathbf{x} |\ \mathbf{Ax} \leq \mathbf{b}, \mathbf{x} \in \mathbb{Z}^n\}, \tag{4.5}$$

then its *linear programming relaxation* is

$$z_{LP} = \min\{\mathbf{c}'\mathbf{x} |\ \mathbf{Ax} \leq \mathbf{b}, \mathbf{x} \in \mathbb{R}^n\}. \tag{4.6}$$

The cost function is the same in both problems, so whenever an optimal solution of the LP is integer-valued it also solves the ILP. Due to this reason, LP relaxations plays a central role in integer programming algorithms, which will be discussed in Section 4.5.

4.3 Modeling Techniques

4.3.1 Related Variables

A binary decision variable $x \in \{0, 1\}$ is commonly used in situations, where there are two possible choices: either we choose something ($x = 1$) or we do not ($x = 0$). Suppose there are n possible choices and we must choose exactly one. If each possible choice is a binary decision variable x_i, the situation can be expressed by the equation

$$\sum_{i=1}^{n} x_i = 1. \tag{4.7}$$

If at most one choice can be made, the equality sign in the equation is replaced with an inequality sign. Should binary variables be dependent of each other, such that at most $a \in \{0, \dots, n\}$ variables can take the value 1, this can be expressed with a constraint

$$\sum_{i=1}^{n} x_i \leq a. \tag{4.8}$$

Often, there are restrictions on consecutive decision variables: if at most a variables out of m, $m \geq a$, consecutive variables can take the value 1, it can be modeled by a constraint

$$\sum_{i=j}^{j+m-1} x_i \leq a, \quad j \in \{1, \dots, n-m+1\}. \tag{4.9}$$

If a decision x implies another decision y, it can be expressed by

$$x \leq y. \tag{4.10}$$

Additionally, should the same hold vice versa, then

$$x - y = 0. \tag{4.11}$$

4.3.2 Disjunctive Constraints

Suppose we are given m constraints $\mathbf{a}_i'\mathbf{y} \geq b_i$, $i = 1, \dots, m$, where $\mathbf{y} \in \mathbb{R}^{|\mathbf{a}_i|}$, with a requirement that at least k of them must be satisfied. One way to model the requirement, is to introduce m binary variables x_i, $i = 1, \dots, m$, and set

$$(\mathbf{a}_i'\mathbf{y} - b_i)x_i \geq 0, \qquad i = 1, \ldots, m, \tag{4.12}$$

$$\sum_{i=1}^{n} x_i \geq k, \tag{4.13}$$

$$x_i \in \{0, 1\}, \quad i = 1, \ldots, m \tag{4.14}$$

A disadvantage in this system is that the constraints are not anymore in a linear form. This issue can be avoided with the following formulation

$$\mathbf{a}_i'\mathbf{y} \geq b_i - M(1 - x_i), \quad i = 1, \ldots, m, \tag{4.15}$$

$$\sum_{i=1}^{n} x_i \geq k, \tag{4.16}$$

$$x_i \in \{0, 1\}, \qquad i = 1, \ldots, m, \tag{4.17}$$

where M is a large constant satisfying

$$M \geq \max_{y \in \mathbb{R}^{|\mathbf{a}_i|}} (\mathbf{a}_i'\mathbf{y} - b_i). \tag{4.18}$$

In practice, it is wise to choose M to be as small as possible, because optimization problems involving a constraint (4.15) in which M is large tend to be hard to solve.

4.3.3 Degree and Subtour Elimination Constraints

Consider a complete undirected graph $G = (V, E)$, and suppose we want to form a closed path by using vertices from a set $S \subset V$. Also let x_e be a binary decision variable that takes the value 1 if edge e is used in a path and 0 otherwise. Then, in a closed path, the first vertex is visited twice (the first and last vertex are the same) and all other vertices are visited once. It is obvious that a necessary condition for a closed path is a *degree constraint*

$$\sum_{e \in \delta(i)} x_e = 2 \quad \forall i \in S, \tag{4.19}$$

where $\delta(i)$ is the cutset of S and determined by (3.7). The condition (4.19) is still not sufficient to guarantee that chosen edges form a closed path, because it allows so called subtours. E.g., if $S = \{1, 2, 3, 4, 5, 6\} \subset V$, we can set that $x_{\{1,2\}} = x_{\{2,3\}} = x_{\{3,1\}} = x_{\{4,5\}} = x_{\{5,6\}} = x_{\{6,4\}} = 1$ and notice that the condition (4.19) is satisfied, although the solution now contains two distinct closed paths. This problem can be tackled with a *subtour elimination constraint*

$$\sum_{e \in \rho(S')} x_e \leq |S'| - 1, \quad \forall S' \subset S, \;\; S' \neq \emptyset, S, \tag{4.20}$$

where $\rho(S')$ is given by the equation (3.8). A disadvantage of the constraint is that it involves $2^{|S|} - 1$ inequalities that a solution must satisfy. Unfortunately, due to the nature of integer programming, this is a problem, which is often difficult to circumvent.

4.3.4 Precedence Constraints

Consider a complete directed graph $G = (V, A)$ where $\{+0, p, q, 0\} \subset V$. Suppose we want to form an open path containing all vertices such that it starts from vertex $+0$, ends at vertex 0, and vertex p must be visited before vertex q. Since $+0$ is the start vertex, 0 the end vertex, and all vertices must be visited, we get the conditions

$$\sum_{(+0,j) \in A} x_{+0,j} = 1, \tag{4.21}$$

$$\sum_{(i,+0) \in A} x_{i,+0} = 0, \tag{4.22}$$

$$\sum_{(0,j) \in A} x_{0,j} = 0, \tag{4.23}$$

$$\sum_{(i,0) \in A} x_{i,0} = 1, \tag{4.24}$$

$$\sum_{(i,j) \in \delta(k)} x_{ij} = 2 \quad \forall k \in V \setminus \{+0, 0\}, \tag{4.25}$$

where x_{ij} is a binary decision variable indicating whether an arc (i, j) is used or not. In order to guarantee that p is visited before q, we define a family of sets, $\mathcal{S}_{pq}$, by

$$\mathcal{S}_{pq} = \left\{ S \subset V | \{p, 0\} \subset \overline{S}, \{q, +0\} \subset S \right\}$$

and require that

$$\sum_{(i,j) \in A:\; i \in \overline{S},\; j \in S} x_{ij} \geq 1 \quad \forall S \in \mathcal{S}_{pq}. \tag{4.26}$$

The idea behind the constraint is the following: since $S_1 := V \setminus \{p, 0\}$ belongs to $\mathcal{S}_{pq}$, it follows that

$$\sum_{(0,j)\in A: j\in S_1} x_{0j} + \sum_{(p,j)\in A: j\in S_1} x_{pj} \geq 1 \Rightarrow \sum_{(p,j)\in A: j\in S_1} x_{pj} \geq 1 \Rightarrow x_{pi_1} = 1$$

for some $i_1 \in S_1$. By choosing next $S_2 := V \setminus \{p, 0, i_1\}$, we get $x_{i_1,i_2} = 1$ for some $i_2 \in S_2$. Now, it is obvious that after no more than $|V| - 3$ steps, we obtain a path from p to q.

If there are several pair of vertices (p_i, q_i), $i = 1, \ldots, n$, such that p_i must always precede q_i, the constraints (4.26) can be written in the form

$$\sum_{(i,j)\in A:\ i\in\overline{S},\ j\in S} x_{ij} \geq 1 \quad \forall S \in \mathcal{S}, \tag{4.27}$$

where

$$\mathcal{S} = \big\{S \subset V | \exists i \in \{1, \ldots, n\} \text{ s.t. } \{p_i, 0\} \subset \overline{S}, \{q_i, +0\} \subset S\big\}.$$

The constraints (4.27) are often referred as *precedence constraints*, and they were introduced by Balas et al. [2].

4.3.5 Strong Formulations

In linear programming, the time which is needed to solve a problem depends primarily on the number of constraints and variables that are used in the formulation. Hence, the fewer constraints and variables the formulation contains, the better it is. In integer programming the situation is crucially different: the goodness of a formulation is determined by the goodness of its linear relaxation. To see what is meant by "goodness", consider an integer programming problem

$$\begin{aligned} z \quad = \quad \min \quad & \mathbf{c}'\mathbf{x} \\ \text{s.t.} \quad & \mathbf{x} \in X, \end{aligned} \tag{4.28}$$

where X is the set of feasible solutions, and the problem where X is replaced by its convex hull:

$$\begin{aligned} z^* \quad = \quad \min \quad & \mathbf{c}'\mathbf{x} \\ \text{s.t.} \quad & \mathbf{x} \in \text{conv}(X) \end{aligned} \tag{4.29}$$

By Theorem 1, we know that conv(X) is a polyhedron, and hence (4.29) is a linear programming problem. The optimal solution of an LP is always an extreme point of the underlying polyhedron, and since all extreme points are now integers, we have that $z^* = z$. Linear programming problems can be solved efficiently [1] so, whenever the convex hull of integer solutions is known, problem (4.28) is also efficient to solve. An unfortunate fact is that convex hulls are rarely known and they can comprise exponentially many linear constraints. Even if a convex hull may be impractical to form, good approximations might still be available. This is useful, since integer programming algorithms, such as the branch-and-bound, find the solution to an interger programming problem by solving a sequence of LP problems, whose number depends on the quality of the approximation. In practice, the better the convex hull is known, the smaller is the number of LPs to be solved. The following definition provides a means of quantifying the *quality of a formulation*:

Definition 3. Let A and B be two formulations of the same integer programming problem. If LP_A and LP_B denote the feasible sets of the corresponding LP relaxations, formulation A is said to be at least as *strong* as formulation B if $LP_A \subset LP_B$.

4.4 Polyhedral Combinatorics in Integer Programming

4.4.1 Polyhedral Combinatorics in General

Polyhedral combinatorics is a branch of mathematics that studies the problems of counting and describing the faces of polytopes (or polyhedra). Research in this area can be divided into two groups: mathematical and optimization orientation. Mathematicians are interested in the number of different dimensional faces of polytopes and how they are connected. A key tool in this approach is the *f-vector* of a polytope: if the dimension of a polytope is d, its f-vector is $(f_0, f_1, \ldots, f_{d-1})$, where f_k, $k = 0, \ldots, d-1$, represents the number of k-dimensional faces. If we concatenate the number one at each end of the vector, we get the *extended f-vector* $(1, f_0, f_1, \ldots, f_{d-1}, 1) = (f_{-1}, f_0, \ldots, f_d)$. The coefficients of the extended f-vector satisfy Euler's formula

$$\sum_{k=-1}^{d} (-1)^k f_k = 0, \tag{4.30}$$

[1] In this context, "efficiently" means that the problem can be solved in polynomial time, which, in turn, means that the computational complexity of the problem grows polynomially with respect to the size of the problem.

which can be considered as the most important (known) relation between the coefficients. For instance, it is easy to see that the extended f-vector of a (3-dimensional) cube, which is $(1, 8, 12, 6, 1)$, satisfies the formula.

From the point of view of this study, the more essential topic in polyhedral combinatorics is the optimization aspect. In the field of optimization, computer and systems scientists study the faces of specific polytopes that arise from integer programming problems. Analysis is often restricted to just 0-1 polytopes, because many integer problems, including the TSP and the PDP, can be formulated as BILPs. In particular interests are the facets 0-1 polytopes. In next two subsections, Section 4.4.2 and Section 4.4.3, we will present some polyhedral results and facet-defining inequalities for the TSP and the TSP with pickup and delivery.

4.4.2 Symmetric Travelling Salesman Problem

Consider the symmetric travelling salesman problem (TSP) introduced in Section 2.1. Suppose we are given an undirected graph $G = (V, E)$, where each edge $e \in E$ is associated with a travelling costs c_e. The problem can be formulated as the following BILP

$$\begin{aligned}
\min \quad & \sum_{e \in E} c_e x_e \\
\text{s.t.} \quad & \sum_{e \in \delta(i)} x_e = 2, && \forall i \in V, && (4.31) \\
& \sum_{e \in \rho(S)} x_e \leq |S| - 1, && \forall S \subset V, \ \ S \neq \emptyset, V, && (4.32) \\
& x_e \in \{0, 1\}, && \forall e \in E, && (4.33)
\end{aligned}$$

where $x_e = 1$ if edge e is used in a solution, and 0 otherwise. When $|V| = n \geq 3$, it is easy to see that the number of edges is

$$|E| = \binom{n}{2} = \frac{(n-1)n}{2} \tag{4.34}$$

and the number of feasible solutions is

$$|H| = \frac{n!}{2}. \tag{4.35}$$

We define the *symmetric travelling salesman polytope* P^E_{TSP} as the convex hull of all feasible solutions, i.e.,

$$P^E_{TSP} = \text{conv}\{\mathbf{x} \in \{0, 1\}^{|E|} | \ \mathbf{x} \text{ satisfies (4.31) and (4.32)}\}. \tag{4.36}$$

By Theorem 2, it is immediate that $\dim(P^E_{TSP}) = |E| - \text{rank}(P^E_{TSP}) \leq |E| - \text{rank}(\{\mathbf{x} \in \mathbb{R}^{|E|} | \; \mathbf{x} \text{ satisfies } (4.31)\}) = (n-1)n/2 - n = n(n-3)/2$. Grötschel and Padberg [13] showed that the inequality holds as an equality:

$$\dim(P^E_{TSP}) = \frac{1}{2}n(n-3). \tag{4.37}$$

By using this result, they also showed that inequalities

$$x_e \leq 1, \;\; e \in E, \tag{4.38}$$

and

$$x_e \geq 0, \;\; e \in E, \tag{4.39}$$

define facets of P^E_{TSP} whenever $n \geq 4$ and $n \geq 5$, respectively. These inequalities are often considered as "trivial" facets, although showing that they are actually facet-defining is not trivial at all. Another family of facets is given by the next proposition:

Proposition 6. Let $n \geq 6$ and $\{u, v, w, u_1, v_1, w_1\} \subset V$. Then the inequality

$$x_{\{u,v\}} + x_{\{u,w\}} + x_{\{v,w\}} + x_{\{u,u_1\}} + x_{\{v,v_1\}} + x_{\{w,w_1\}} \leq 4 \tag{4.40}$$

defines a facet of P^E_{TSP}.

Proof. See [13]. □

By counting the number of inequalities in (4.38), (4.39), and (4.40), we find that P^E_{TSP} comprises at least $n(n-1)/2 + n(n-1)/2 + n(n-1)(n-2)(n-3)(n-4)(n-5) \approx n^5$ many facets. This gives rise to the question of how many facets are needed to form the whole convex hull. In the case of the symmetric travelling salesman problem, Yannakakis [34] showed that at least exponentially many facets are needed; indeed, a lower bound for this number is of the order $2^{\sqrt{n}}$. One should note that the exponential number of facets in the symmetric case does not automatically mean that the number of facets would be as large in the asymetric case; in fact, this problem remained unsolved for a few decades until Fiori et al. [11] proved that exponentially many facets are also needed in the case of the asymmetric travelling salesman problem. This means that generating all facets of a travelling salesman polytope is not an efficient way to solve the original problem.

4.4.3 Symmetric Travelling Salesman Problem with Pickup and Delivery

Consider the symmetric travelling salesman problem with pickup and delivery, TSPPD, i.e., the TSP in which each request comprises a pickup vertex and a delivery vertex that must be visited. Let $G = (V, E)$ be a complete undirected graph, where V is composed of pickup vertices $P = \{1, \ldots, n\}$, delivery vertices $D = \{n+1, \ldots, 2n\}$, a start depot vertex $+0$, and a terminal depot vertex 0. In reality $+0$ and 0 represent the same depot but the presence of two different depot vertices simplify the formulation. The problem can be formulated as follows:

$$
\begin{aligned}
\min \quad & \sum_{e \in E} c_e x_e && && \\
\text{s.t.} \quad & \sum_{e \in \delta(i)} x_e = 2, && \forall i \in V, && (4.41) \\
& \sum_{e \in \rho(S)} x_e \leq |S| - 1, && \forall S \subset V, \;\; S \neq \emptyset, V, && (4.42) \\
& \sum_{e \in \delta(S)} x_e \geq 4, && \forall S \in \mathcal{S} && (4.43) \\
& x_{\{+0,0\}} = 1, && && (4.44) \\
& x_e \in \{0, 1\}, && \forall e \in E, && (4.45)
\end{aligned}
$$

where $\mathcal{S} = \big\{S \subset V | \exists i \in \{1, \ldots, n\} \text{ s.t. } \{i, 0\} \subset \overline{S}, \{n+i, +0\} \subset S\big\}$. The constraints (4.43) represent the precedence constraints of an undirected graph, and they can be easily derived from the precedence constraints (4.27) that are defined for a directed graph. The formulation we presented is originally from Ruland [24] with the exception that instead of (4.42) he used the constraints

$$
\sum_{e \in \delta(S)} x_e \geq 2, \quad \forall S \subset V, \;\; S \neq \emptyset, V, \qquad (4.46)
$$

which is an alternative representation of the subtour elimination constraints. The formulation we gave and the one from Ruland are not just equivalent, but they are also equally strong (see Appendix A).

The number of edges in the TSPPD graph, $|E|$, and the number of feasible solutions to the problem, $|H|$, are given by the next propositions:

Proposition 7. The number of edges in the TSPPD graph is equal to

$$
|E| = 2n^2 + n + 1 \qquad (4.47)
$$

Proof. Let S_1 and S_2 be subsets of V and define $|\{S_1, S_2\}| := |\{\{i, j\} \in E | i \in S_1, j \in S_2\}|$. When the symmetry is taken into account, we find that the number of edges is

$$\begin{aligned} |E| &= \frac{1}{2}\Big(\sum_{i=1}^{n} \underbrace{|\{i, V\}|}_{=2n^2} + \sum_{i=1}^{n} \underbrace{|\{n+i, V\}|}_{=2n^2} + \underbrace{|\{+0, V\}|}_{=n+1} + \underbrace{|\{0, V\}|}_{=n+1} \Big) \\ &= 2n^2 + n + 1 \end{aligned}$$

□

Proposition 8. The number of feasible solutions to the TSPPD is

$$|H| = \frac{(2n)!}{2^n} \tag{4.48}$$

Proof. Let H_{i-1} be the set of feasible solutions when $|P| = i - 1$. Clearly, any solution in H_{i-1} can be extended to a solution of H_i by requiring that the path goes also through vertices i and $n+i$. Since i must be visited before $n + i$, the number of such extensions is $\binom{2i}{2}$. Thus,

$$\begin{aligned} |H| = |H_n| = \binom{2n}{2}|H_{n-1}| = \prod_{i=1}^{n}\binom{2i}{2}|H_1| = \prod_{i=1}^{n}\frac{(2i)!}{2!(2i-2)!} = \prod_{i=1}^{n}\frac{2i(2i-1)}{2} \\ = \frac{(2n)!}{2^n} \end{aligned}$$

□

Ruland [24] was the first who studied the polyhedral structure of the TSPPD polytope, which is defined as

$$P^E_{TSPPD} = \text{conv}\{\mathbf{x} \in \{0,1\}^{|E|} | \ \mathbf{x} \text{ satisfies (4.41) - (4.44)}\}. \tag{4.49}$$

Ruland showed that the dimension of P^E_{TSPPD} is at most $2n^2 - n - 2$, but was not able to give an exact formula. Ten years later, Dumitrescu [10] proved that $2n^2 - n - 2$ is also a lower bound for the dimension, i.e.,

$$\dim(P^E_{TSPPD}) = 2n^2 - n - 2. \tag{4.50}$$

Ruland had proposed several valid inequalities that are satisfied by the TSPPD, and now when the dimension was known, it was also possible to study whether these inequalities are facet-defining or not. E.g., Dumitrescu stated and proved the following proposition:

Proposition 9. For any $H = \{i_1, \dots, i_m\} \subset P$, the inequality

$$\sum_{e \in \rho(H)} x_e + \sum_{j=1}^{m} x_{\{i_j, n+i_j\}} \leq |H| \tag{4.51}$$

defines a facet of the TSPPD polytope.

Proof. See [10]. □

Now, we can see that the TSPPD also has trivial facets: by choosing $H = \{i\}$, $i \in P$ in Proposition 9, it follows that $x_{\{i,n+i\}} \leq 1$ is a facet for any $i \in P$. To the best of our knowledge, it is unclear if the TSPPD has any other trivial facets of the form $x_e \geq 0$ or $x_e \leq 1$. Dumitrescu showed that some of the precedence constraints are facet-defining as well:

Proposition 10. The inequality

$$\sum_{e \in \delta(S)} x_e \geq 4, \tag{4.52}$$

is a facet of the TSPPD for any $S \in \mathcal{S}' := \big\{S \subset V | \exists!\ i \in \{1, \dots, n\}$ s.t. $\{i, 0\} \subset \overline{S}, \{n+i, +0\} \subset S\big\}^2$.

Proof. See [10]. □

4.5 Integer Programming Algorithms

In Section 4.5, we consider the integer programming problem

$$\begin{aligned} z_{ILP} \quad = \quad \min \quad & \mathbf{c}'\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b}, \\ & \mathbf{x} \in \mathbb{Z}^n_+, \end{aligned} \tag{4.53}$$

and its linear programming relaxation

$$\begin{aligned} z_{LP} \quad = \quad \min \quad & \mathbf{c}'\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b}, \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned} \tag{4.54}$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^n$, $\mathbf{c} \in \mathbb{R}^m$, $\mathbf{x} \in \mathbb{R}^n$, and $n > m$.

[2] ∃! "denotes that there is a unique"

4.5.1 Simplex Algorithm

4.5.1.1 Basic Solutions

Consider the linear programming problem in the standard form (4.54). It can be assumed that the rows of $\mathbf{A}$ are linearly independent; were they not, either some of the constraints can be eliminated or the problem has no solutions. Neither of the cases are thus interesting to study.

Since the rows of $\mathbf{A}$ are linearly independent, $\mathbf{A}$ must contain m linearly independent columns. By choosing such columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(m)}$, where $\mathbf{A}_{B(i)}$ indicates the $B(i)$th column of $\mathbf{A}$, we obtain a *basis matrix* $\mathbf{B} = [\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(m)}]$. Because $\mathbf{B}$ is invertible, it determines the values of *basic variables* by $\mathbf{x}_B := \mathbf{B}^{-1}\mathbf{b} = [x_{B(1)}, \ldots, x_{B(m)}]'$. The rest of the decision variables are *nonbasic variables* $\mathbf{x}_N$, where N denotes the index set of nonbasic variables. When all nonbasic variables are set to zero, basic variables and nonbasic variables define a *basic solution* $\mathbf{x} = [\mathbf{x}'_N, \mathbf{x}'_B]'$; in addition, if $\mathbf{x} \geq \mathbf{0}$, the solution is called a *basic feasible solution*.

Let P be the polyhedron defined by the constraints of the problem, and suppose we are given a basis matrix $\mathbf{B}$ that defines a basic feasible solution $\mathbf{x} \in P$. Since P is a convex set, for each point $\mathbf{x} \in P$ there exists a vector $\mathbf{d} \in \mathbb{R}^n$ such that $\mathbf{x} + \theta\mathbf{d} \in P$ for some $\theta > 0$. Such a vector $\mathbf{d}$ is called a *feasible direction* at $\mathbf{x}$.

Consider next a situation in which we move from point $\mathbf{x}$ to direction $\mathbf{d} = [\mathbf{d}'_B, \mathbf{d}'_N]'$, such that $d_j = 1$ for some $j \in N$ and $d_i = 0$ $\forall i \neq j$, $i \in N$. In order to stay in the feasible region, we must determine θ so that $\mathbf{A}(\mathbf{x} + \theta\mathbf{d}) = \mathbf{b} \Leftrightarrow \theta\mathbf{A}\mathbf{d} = \mathbf{0}$. If one assumes that $x_{B(i)} > 0$ $\forall i \in \{1, \ldots, m\}$, then $\theta > 0$, in which case

$$\begin{aligned} \mathbf{0} = \mathbf{A}\mathbf{d} = \sum_{i=1}^{n} \mathbf{A}_i d_i = \sum_{i=1}^{m} \mathbf{A}_{B(i)} d_{B(i)} + \mathbf{A}_j = \mathbf{B}\mathbf{d}_B + \mathbf{A}_j \\ \Leftrightarrow \ \mathbf{d}_B = -\mathbf{B}^{-1}\mathbf{A}_j. \end{aligned} \tag{4.55}$$

The obtained vector $\mathbf{d}$ is called the jth *basic direction*. The case when $x_{B(i)} = 0$ for some $i \in \{1, \ldots, m\}$ will be considered in Section 4.5.1.4.

4.5.1.2 Reduced Costs

In terms of solving the problem (4.54), it is essential to know how the value of the cost function $\mathbf{c}'\mathbf{x}$ changes when moved along the $j$the basic direction $\mathbf{d}$: a unit displacement ($\theta = 1$) along $\mathbf{d}$ causes $\bar{c}_j$ units change in the cost function, where

$$\bar{c}_j = \mathbf{c}'(\mathbf{x} + \mathbf{d}) - \mathbf{c}'\mathbf{x} = \mathbf{c}'\mathbf{d} = \mathbf{c}'_B\mathbf{d}_B + c_j = c_j - \mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A}_j. \tag{4.56}$$

Quantity $\overline{c}_j$ is known as the *reduced cost* of the variable x_j, and it can be interpreted as follows: c_j indicates the cost of one unit increase in the variable x_j, whereas the term $-\mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A}_j$ represents the cost of the requirement that $\mathbf{Ax} = \mathbf{b}$ must hold. Since our goal is the minimization of the cost function, we are interested in variables x_j for which $\overline{c}_j < 0$. Indeed, it can be shown that $\mathbf{x}$ is an optimal solution if $\overline{\mathbf{c}} = [\overline{\mathbf{c}}'_N, \overline{\mathbf{c}}'_B]' \geq \mathbf{0}$ [3]. It is sufficient to consider only nonbasic variables, since the reduced costs of basic variables $x_{B(i)}$, $i = 1, \ldots, m$, are always 0:

$$\overline{c}_{B(i)} = c_{B(i)} - \mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A}_{B(i)} = c_{B(i)} - \mathbf{c}'_B\mathbf{e}_i = c_{B(i)} - c_{B(i)} = 0,$$

where $\mathbf{e}_i$ is the ith basis vector of $\mathbb{R}^n$.

4.5.1.3 New Basic Solution

Suppose $\overline{c}_j < 0$ for some $j \in N$. By moving $\theta \geq 0$ units along the jth basic direction, the change in the cost function is $\theta\overline{c}_j < 0$, which makes it desirable to maximize θ. Since we need to stay in the feasible region, one must have that $\mathbf{A}(\mathbf{x} + \theta\mathbf{d}) = \mathbf{b}$ and $\mathbf{x} + \theta\mathbf{d} \geq \mathbf{0}$. The former condition is always satisfied by the construction of $\mathbf{d}$. The latter condition leaves two cases to consider:

1. If $\mathbf{d} \geq \mathbf{0}$, then $\mathbf{x} + \theta\mathbf{d} \geq \mathbf{0}\ \forall\theta \geq 0$, in which case $\theta = \infty$ and $z_{LP} = -\infty$.
2. If $d_i < 0$ for some $i \in \{1, \ldots, n\}$, then $x_i + \theta d_i$ becomes negative for large enough θ, which sets a requirement $\theta \leq -x_i/d_i$. Since $x_i + \theta d_i$ must be non-negative for all $i \in \{1, \ldots, n\}$, the maximum value of θ is given by

$$\theta^* = \min_{\{i=1,\ldots,n|d_i<0\}} \left(-\frac{x_i}{d_i} \right) = \min_{\{i=1,\ldots,m|d_{B(i)}<0\}} \left(-\frac{x_{B(i)}}{d_{B(i)}} \right), \tag{4.57}$$

where the last equality follows from the fact that $d_i = 0\ \forall i \in N \setminus \{j\}$, and $d_j = 1$. Assume now that $\theta < \infty$ and let $l = \arg\min_i(-x_{B(i)}/d_{B(i)})$. If the new feasible solution is denoted by $\mathbf{y} = \mathbf{x} + \theta^*\mathbf{d}$, then $y_j = \theta^*$ and $y_{B(l)} = x_{B(l)} + \theta^* d_{B(l)} = x_{B(l)} + d_{B(l)} * (-x_{B(l)}/d_{B(l)}) = 0$. Since $y_j > 0$ and $y_{B(l)} = 0$, it gives rise to change our basis matrix $\mathbf{B} = [\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(l)}, \ldots, \mathbf{A}_{B(m)}]$ such that the column $\mathbf{A}_{B(l)}$ is replaced with the column $\mathbf{A}_j$. The new matrix obtained is

$$\overline{\mathbf{B}} = [\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(l-1)}, \mathbf{A}_j, \mathbf{A}_{B(l+1)}, \ldots, \mathbf{A}_{B(m)}], \tag{4.58}$$

which can be shown to be a basis matrix that determines a basic feasible solution $\mathbf{y} = \mathbf{x} + \theta^*\mathbf{d}$ [3].

4.5.1.4 Degeneracy

If some of the basic variables are zero, i.e. $x_{B(l)} = 0$ for some $l \in \{1, \dots, m\}$, the basic solution is said to be *degenerate*. In addition, if $d_{B(l)} < 0$ for the chosen basis direction $\mathbf{d}$, degeneracy means that the non-negativity condition $x_{B(l)} + \theta^* d_{B(l)} \geq 0$ is satisfied only if $\theta^* = 0$. Although the new solution obtained is equivalent to the old one, we can still change the basis matrix ($\mathbf{B} \hookrightarrow \overline{\mathbf{B}}$) in the hope that the next basis change would lead us to a better solution. If the new basic variable is selected by random, it is possible that this procedure will lead back to the initial basis, and thereby cause an infinite loop. This undesirable phenomenon is called *cycling*.

Cycling can be avoided by using certain rules for basis changes. One of these rules is known as *Bland's rule*, which can be described as follows:

1. Find the smallest j for which the reduced cost is negative and let the corresponding variable enter the basis.
   Formally, $j = \min\{k \in N | \overline{c}_k < 0\}$.

2. Out of all variables x_i which satisfy condition (4.57), choose the one with the smallest value of i. In other words,
   $l = \min\left\{B(i) \in B | -x_{B(i)}/d_{B(i)} = \theta^*, d_{B(i)} < 0\right\}$.

Under Bland's rule cycling cannot occur, and the algorithm terminates after a finite number of steps (See [5]).

4.5.1.5 Simplex Iteration

The simplex algorithm, which solves a linear programming problem (4.53), can be described with the following five-step iteration:

1. Determine a basis matrix $\mathbf{B} = [\mathbf{A}_{B(1)}, \dots, \mathbf{A}_{B(m)}]$ and the corresponding basic feasible solution $\mathbf{x}$.

2. Calculate the reduced costs $\overline{c}_j = c_j - \mathbf{c}'_B \mathbf{B}^{-1} \mathbf{A}_j$ of all nonbasic variables x_j, $j \in N$. If $\overline{c}_j \geq 0 \; \forall j \in N$, then $\mathbf{x}$ is an optimal solution, and the algorithm terminates; otherwise, choose an index $j = \min\{k \in N | \overline{c}_k < 0\}$.

3. Determine the jth basic direction $\mathbf{d} = \mathbf{B}^{-1} \mathbf{A}_j$. If $\mathbf{d} \geq \mathbf{0}$, the optimal cost is $-\infty$, and the algorithm terminates.

4. If $d_i < 0$ for some $i \in B$, set $\theta^* = \min_{\left\{i=1,\dots,m | d_{B(i)} < 0\right\}} \left(-\frac{x_{B(i)}}{d_{B(i)}} \right)$.

5. Choose an index $l = \min\left\{B(i) \in B | -x_{B(i)}/d_{B(i)} = \theta^*, d_{B(i)} < 0\right\}$.
   Form a new basis $\overline{\mathbf{B}}$ by replacing the column $\mathbf{A}_{B(l)}$ with the column

$\mathbf{A_j}$. When $\mathbf{y}$ denotes the new basic feasible solution, the values of the new basic variables are $y_j = \theta^*$ and $y_{B(i)} = x_{B(i)} + \theta^* d_{B(i)}$ $\forall i \in \{1, \ldots, m\} \setminus \{l\}$. Return to step 2.

4.5.2 Branch and Bound Algorithm

Let $ILP(F)$ be the ILP defined as

$$\begin{aligned} z_{ILP} \quad = \quad \min \quad & \mathbf{c}'\mathbf{x} \\ \text{s.t.} \quad & \mathbf{x} \in F, \\ & \mathbf{x} \in \mathbb{Z}^n_+, \end{aligned} \tag{4.59}$$

where F is a set of linear equalities and inequalities. The problem is usually too hard to solve directly since the decision variables are required to be integers. However, a lower bound for z_{ILP} can be obtained by solving the linear programming relaxation $LP(F)$ of $ILP(F)$, i.e.

$$\begin{aligned} b(F) \quad = \quad \min \quad & \mathbf{c}'\mathbf{x} \\ \text{s.t.} \quad & \mathbf{x} \in F, \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \tag{4.60}$$

This problem is easy to solve with any linear programming algorithm such as simplex (see Section 4.5.1). If the optimal solution of $LP(F)$, $\mathbf{x}^*$, is an integer vector, it also solves the original problem; otherwise, $LP(F)$ is divided into subproblems of which feasible regions contain all integer solutions but not the vector $\mathbf{x}^*$. This splitting of the search space is called *branching*. Suppose the ith variable of $\mathbf{x}^*$ is not an integer and define sets F_1 and F_2 such that

$$F_1 = \{\mathbf{x} \in F |\ x_i \leq \lfloor x_i^* \rfloor\} \text{ and } F_2 = \{\mathbf{x} \in F |\ x_i \geq \lceil x_i^* \rceil\}. \tag{4.61}$$

Then $\mathbf{x}^*$ belongs to neither of the sets, but $(F_1 \cup F_2) \cap \mathbb{Z}^n_+ = F \cap \mathbb{Z}^n_+$. If $LP(F_1)$ or $LP(F_2)$ gives no integer solution, the search spaces of F_1 and F_2 must also be branched, which generates additional subproblems. Suppose this procedure is continued until some of the subproblems $LP(F_i)$, $i \in I$, where I is an index set, has an integer solution. Let such problem be $LP(F_k)$ and denote the value of the objective function by $U := b(F_k)$. Branching of the search space of F_k can now be stopped since the value of the objective function cannot improve anymore on this branch. Moreover, as U provides an upper bound for z_{ILP}, every subproblem $LP(F_i)$ for which $b(F_i) \geq U$ can be eliminated. Whenever a better integer solution is found, the value of U is updated, resulting in the elimination of further subproblems. Due to this elimination process, the name of the algorithm carries the word "bound".

4.5.2.1 Branch and Bound Iteration

The branch and bound algorithm can be described with the following six steps:

1. Set $U = \infty$.
2. Choose an $LP(F_i)$ from the list of subproblems. If the list is empty, the optimal solution is the one that corresponds to U; if this happens when $U = \infty$, $ILP(F)$ has no solution.
3. If $LP(F_i)$ is infeasible, eliminate it from the list of subproblems. Otherwise, calculate the lower bound $b(F_i)$.
4. If $b(F_i) \geq U$, the subproblem can be eliminated. Return to step 2.
5. If $b(F_i) < U$, and the solution of the problem $\mathbf{x}^* \in \mathbb{Z}_+^n$, set $U = b(F_i)$ and eliminate $LP(F_i)$. Return to step 2.
6. If $b(F_i) < U$, and $\mathbf{x}^* \notin \mathbb{Z}_+^n$, create new subproblems according to rule (4.61). Return to step 2.

4.5.3 Cutting Plane Method

The basic idea of the cutting plane method is to solve the integer programming problem (4.53) by solving a sequence of linear programming problems. First, solve the LP relaxation (4.54) and find its optimal solution $\mathbf{x}^*$. If $\mathbf{x}^* \in \mathbb{Z}_+^n$, then it also solves the problem (4.53). Otherwise, we find an inequality that is satisfied by all integer solutions, but not by $\mathbf{x}^*$; in other words, a plane defined by the inequality "cuts out" $\mathbf{x}^*$. By adding this inequality to the linear programming problem and solving it again, we get a better approximation of the integer optimum. This procedure is repeated until the solution found is an integer.

Usually, the way how cutting planes are formed uses the specific structure of the ILP to be solved. We present Gomory's cutting plane method [12] that generates cuts for any standard form integer programming problem (4.53) by using the steps of the simplex algorithm. The method was the first cutting plane method that was guaranteed to terminate in finite time. Let $\mathbf{x} = [\mathbf{x}_B', \mathbf{x}_N']'$ be a basic feasible solution to (4.54), where $\mathbf{x}_B$ and $\mathbf{x}_N$ denote basic and nonbasic variables, respectively. Since $\mathbf{x}$ is a basic feasible solution, it satisfies the constraint $\mathbf{A}\mathbf{x} = \mathbf{b}$, which is the same as

$$[\underbrace{\mathbf{A}_B}_{=\mathbf{B}}, \mathbf{A}_N] * [\mathbf{x}_B', \mathbf{x}_N']' = \mathbf{B}\mathbf{x}_B + \mathbf{A}_N\mathbf{x}_N = \mathbf{b}.$$

By multiplying both sides with $\mathbf{B}^{-1}$, we get

$$\mathbf{x}_B + \mathbf{B}^{-1}\mathbf{A}_N\mathbf{x}_N = \mathbf{B}^{-1}\mathbf{b}.$$

Let $\overline{a}_{ij} = (\mathbf{B}^{-1}\mathbf{A}_j)_i$, and $\overline{a}_{i0} = (\mathbf{B}^{-1}\mathbf{b})_i$, in which case

$$x_i + \sum_{j\in N} \overline{a}_{ij}x_j = \overline{a}_{i0}.$$

Since $x_j \geq 0$ for all $j \in \{B, N\}$, it holds that

$$x_i + \sum_{j\in N} \lfloor\overline{a}_{ij}\rfloor x_j \leq x_i + \sum_{j\in N} \overline{a}_{ij}x_j = \overline{a}_{i0}.$$

Furthermore, since x_j should be an integer, and the sums and products of integers are always integers, we obtain the inequality

$$x_i + \sum_{j\in N} \lfloor\overline{a}_{ij}\rfloor x_j \leq \lfloor\overline{a}_{i0}\rfloor, \tag{4.62}$$

which is valid for all integer solutions. Suppose that $\mathbf{B}$ represents the basis matrix of an optimal solution $\mathbf{x}^*$, where $x_i^* \notin \mathbb{Z}$. Now, $x_i^* = \overline{a}_{i0}$, and $x_j^* = 0$ $\forall j \in N$ in which case

$$x_i + \sum_{j\in N} \lfloor\overline{a}_{ij}\rfloor x_j \leq \lfloor\overline{a}_{i0}\rfloor \Leftrightarrow \overline{a}_{i0} \leq \lfloor\overline{a}_{i0}\rfloor.$$

But this is a contradiction since $\overline{a}_{i0}$ was assumed to be fractional. Hence, (4.62) is not satisfied by $\mathbf{x}^*$, and the inequality defines a cutting plane.

4.5.3.1 Cutting Plane Iteration

Cutting plane method can be described with the following steps:

1. Solve the linear programming problem (4.54). Let $\mathbf{x}^*$ be an optimal solution.

2. If $\mathbf{x}^*$ is integer, then $\mathbf{x}^*$ also solves the integer programming problem (4.53), and the algorithm terminates.

3. If $\mathbf{x}^*$ is not integer, form a linear inequality constraint that is satisfied by all integer solutions but not by $\mathbf{x}^*$ and add it to the problem (4.54). Return to step 1.

Chapter 5

Elevator Dispatching Problem (EDP)

5.1 Formulation

Let $G = (V, A)$ be a directed graph where V denotes the set of vertices and $A \subset V \times V$ the set of arcs. In addition, let the number of transportation requests be n and the number of elevators l. Each request $i \in \{1, \ldots, n\}$ is associated with two vertices: a *pickup vertex* i, which corresponds to the arrival floor of the request, and a *delivery vertex* $n+i$, which corresponds to the destination floor. Pickup and delivery vertex sets are denoted by P and D, respectively. Each elevator $e \in \{e_1, \ldots, e_l\}$ is associated with an *origin depot vertex* $2n + e$ indicating the initial location of the elevator. Origin depot vertex set is denoted by T. After an elevator has served the requests (possibly none) that were assigned to it, the elevator ends its route to a *common terminal depot vertex* 0. In the following, 0 is used for notational and modeling convenience. Now, we can define $V = P \cup D \cup T \cup \{0\}$ and $A = \{(i,j)|\ i \neq j, i, j \in P \cup D\} \cup \{(i,j)|\ i \in T, j \in P \cup D\} \cup \{(i,0)|\ i \in P \cup D \cup T\}$.

The solutions to *EDP* are sets of paths starting from T and ending at 0, so that each vertex in P and D is visited by some elevator. A feasible solution also satisfies a set of constraints, which we divide into 8 different categories [28]:

1) ***In- and out-degree constraints*** guarantee that each pickup and delivery vertex is visited exactly once. In order to take into account these conditions, a binary decision variable x_{ij} is associated with each arc $(i,j) \in A$. The variable takes the value of 1 if some elevator goes straight from vertex i to vertex j in the solution, and 0 otherwise. In-degree and out-degree constraints can now be employed, respectively, by equations

$$x(V,i) = 1, \quad \forall i \in P \cup D, \tag{5.1}$$

$$x(i,V) = 1, \quad \forall i \in V \setminus \{0\}, \tag{5.2}$$

where $x(V,i)$ and $x(i,V)$ are abbreviations for expressions $\sum_{(j,k)\in A:k\in\{i\},j\in V} x_{jk}$ and $\sum_{(k,j)\in A:k\in\{i\},j\in V} x_{kj}$.

2) *Precedence constraints* require that the pickup vertex of each request is visited before the delivery vertex, and both vertices must be visited by the same elevator. In other words, if elevator e serves request i, there must be a path through vertices $2n+e$, i and $n+i$, respectively. In order to express these constraints formally, we define a family $\mathcal{S}$ of all vertex subsets S such that $\mathcal{S}= \{S \subset V| \ \exists! \ i \in P$ s.t. $i \notin S$ and $n+i \in S, \ 0 \notin S, T \subset S, 2+|T| \leq |S| \leq |V| - |T| - 2\}$. Now, the precedence constraints follow from the inequalities

$$x(\overline{S},S) \geq 1, \quad \forall S \in \mathcal{S}, \tag{5.3}$$

where $\overline{S} = V \setminus S$ and $x(\overline{S},S) = \sum_{(i,j)\in A:i\in\overline{S},j\in S} x_{ij}$.

3) *Fixing constraints*. There are three different types of transportation requests: *on-board requests*, *assigned requests*, and *non-assigned requests*. These requests represent passengers, respectively, who have been picked up but have not been delivered, passengers that have been assigned to some elevator but not yet picked up, and passengers which are neither assigned to an elevator nor picked up. On-board requests and assigned requests are called *fixed requests*, as they cannot be reassigned. If request i is fixed to some elevator $\hat{e}$, then there must be a path from origin depot vertex $2n+\hat{e}$ to vertex $n+i$. Define a family $\mathcal{F}$ of all vertex subsets F by $\mathcal{F}= \{F \subset V| \ \exists! \ (n+i) \in F$ s.t. $2n+\hat{e} \notin F,$ whenever i is fixed to elevator $\hat{e}\}$, in which case the fixing constraints can be expressed in the form

$$x(\overline{F},F) \geq 1, \quad \forall F \in \mathcal{F} \tag{5.4}$$

4) *Load constraints (consistency, capacity, initial conditions)*. Assume that each elevator has a maximum capacity of Q. Each vertex $i \in V$ is associated with a load ω_i, expressing the number of passengers entering the elevator ($i \in P$), exiting the elevator ($i \in V$), or the initial load of the elevator ($i \in T$). In addition, conditions $\omega_0 = 0$, $\omega_{2n+e} \geq 0 \ \forall (2n+e) \in T$, $\omega_i = -\omega_{n+i} \ \forall i \in P$, and $\omega_i \leq Q \ \forall i \in V$ are assumed to hold.

Let $q_i \in \mathbb{R}_+$ be a decision variable, which represents the load of an elevator upon leaving vertex i. Consistency, capacity, and initial conditions for the load can now be formulated as follows:

$$q_j \geq q_i + \omega_j - \min\{Q, Q + \omega_i\}(1 - x_{ij}), \quad \forall (i,j) \in A, \tag{5.5}$$

$$\max\{0, \omega_i\} \leq q_i \leq \min\{Q, Q + \omega_i\}, \quad \forall i \in P \cup D, \tag{5.6}$$

$$q_i = \omega_i, \quad \forall i \in T, \tag{5.7}$$

$$q_i = 0, \quad i = 0. \tag{5.8}$$

5) *Time constraints (consistency, time window)*. We define the *waiting time* of a passenger as the time from the moment the request was given to the moment the passenger enters the elevator. Accordingly, the *journey time* is defined as the time from the moment the request was given to the moment the passenger exits the elevator at the destination floor. In addition, the time from the instant request $i \in P$ was given to the current moment is called an *elapsed time* γ_i. Let t_i be a continuous decision variable indicating the time at which an elevator begins service at vertex i. By requiring that

$$a_i \leq t_i \leq b_i, \quad \forall i \in V, \tag{5.9}$$

where

$$b_i = \begin{cases} W - \gamma_i & \text{if } i \in P, \\ J - \gamma_i & \text{if } i \in D, \\ \infty & \text{if } i \in V \setminus (P \cup D), \end{cases}$$

it can be guaranteed that neither the waiting nor the journey time of a passenger exceeds given parameters W and J. The earliest time at which service may begin at vertex i, a_i, is greater than 0 only if a request device is not located next to the elevators, in which case the walking time from the device to the elevators is taken into account; otherwise, a_i can be set to 0. Time windows of vertices $i \in V \setminus (P \cup D)$ are assumed to be relaxed, i.e., $[a_i, b_i] = [0, \infty)$. Consistency of time variables follows from the set of inequalities

$$t_j \geq t_i + \tau_{ij} - \max\{0, b_i + \tau_{ij} - a_j\}(1 - x_{ij}), \quad \forall (i,j) \in A, \tag{5.10}$$

where τ_{ij} represents a *travel time* between vertices i and j. The length of the travel time τ_{ij} depends on the floors and vertex types (P, D, T or 0) of i and j. Especially, if $j = 0$, then $\tau_{ij} = 0$. More detailed description of travel times is presented in references [27, 28].

6) ***Boarding constraints*** stipulate that an elevator cannot leave a floor before it is full or all of the passengers who are assigned to that elevator, waiting at the current floor, and travelling the same direction as the elevator, have entered the elevator. Let the floor of vertex i be $f(i)$. The direction of vertex i, $i \in P \cup D$, is denoted by $d(i)$ and it is defined as follows:

$$d(i) = \begin{cases} \chi_{\{f(n+i)>f(i)\}} * 1 + \chi_{\{f(n+i)<f(i)\}} * (-1) & \text{if } i \in P, \\ d(i-n) & \text{if } i \in D. \end{cases}$$

Denote the load of vertex set $S \subset V$ by $\Omega(S)$ and define a family $\mathcal{X}$ of *forwards paths* X, which are of the form $X = (2n+e, k_1, \ldots, k_i, \ldots, k_j, \ldots, k_r)$, where $k_i, k_r \in P$, $d(k_i) = d(k_r)$, $f(k_i) = f(k_r)$, $f(k_i) \neq f(k_j)$ such that either $\Omega(\{2n+e, k_1, \ldots, k_i\}) < Q$ or $\Omega(\{2n+e, k_1, \ldots, k_i\}) = Q$, but $k_r < k_i$. The boarding constraints can now be expressed by inequalities

$$\sum_{(i,j)\in A(X)} x_{ij} \leq |A(X)| - 1, \quad \forall X \in \mathcal{X}, \tag{5.11}$$

where $A(X)$ is the arc set of X.

7) ***Reversal constraints*** relate to situations in which an elevator changes its direction: it is usually assumed that passengers cannot travel the opposite direction in respect of their destination floors, so we require that the elevator must be empty each time the direction is changed. The set of *reversal arcs* is defined by

$$\begin{aligned} R =& \{(n+i, j) \in A |\ n+i \in D, j \in P, d(i) \neq d(j)\} \cup \\ & \{(n+i, j) \in A |\ n+i \in D, j \in P, d(i) = d(j), d(i)f(n+i) > d(j)f(j)\}, \end{aligned}$$

in which case the reversal constraints can be expressed in the form

$$q_i \leq (1 - x_{ij}) \min\{Q, Q + \omega_i\}. \quad \forall (i,j) \in R. \tag{5.12}$$

8) ***Service order constraints*** reflect the following three assumptions, which are usually given in elevator routing problems.
"*i) Waiting passengers at a floor cannot enter the servicing elevator car before all on-board passengers who are going to leave the elevator at that floor finish leaving the elevator.*
ii) If there is more than one passenger boarding the elevator at a floor, then they board the elevator in the ascending order of their arrival times.
iii) If there is more than one passenger leaving the car at a floor, then they leave the elevator in the reverse order of their boarding." [28] Define O as the

set of arcs which contradict assumptions i)-iii). Formally, $O = O_i \cup O_{ii} \cup O_{iii}$, where

$$\begin{aligned}
O_i &= \{(i, n+j) \in A |\ i \in P, n+j \in D, f(i) = f(n+j), d(i) = d(n+j)\} \\
O_{ii} &= \{(i, j) \in A |\ i, j \in P, j < i, f(i) = f(j), d(i) = d(j)\} \\
O_{iii} &= \{(n+i, n+j) \in A |\ i, j \in P, f(n+i) = f(n+j), d(n+i) = d(n+j), \\
&\quad d(i) = 1, \{f(i) < f(j) \text{ or } \{f(i) = f(j) \text{ and } i < j\}\}\} \cup \\
&\quad \{(n+i, n+j) \in A |\ i, j \in P, f(n+i) = f(n+j), d(n+i) = d(n+j), \\
&\quad d(i) = -1, \{f(i) > f(j) \text{ or } \{f(i) = f(j) \text{ and } i < j\}\}\},
\end{aligned}$$

in which case the service order constraints are

$$x_{ij} = 0, \quad \forall (i, j) \in O. \tag{5.13}$$

By minimizing the average waiting time of the passengers, which is one of the most common objective functions among elevator routing problems, with respect to constraints (5.1)-(5.13), the following mixed integer linear problem is obtained:

$$\min \sum_{i \in P} \frac{\omega_i}{\Omega(P)} (t_i + \gamma_i), \tag{5.14}$$

subject to

$$\begin{aligned}
x(V, i) &= 1, && \forall i \in P \cup D, \\
x(i, V) &= 1, && \forall i \in V \setminus \{0\}, \\
x(\overline{S}, S) &\geq 1, && \forall S \in \mathcal{S}, \\
x(\overline{F}, F) &\geq 1, && \forall F \in \mathcal{F}, \\
t_j &\geq t_i + \tau_{ij} - \max\{0, b_i + \tau_{ij} - a_j\}(1 - x_{ij}), && \forall (i, j) \in A, \\
a_i \leq t_i &\leq b_i, && \forall i \in V, \\
\sum_{(i,j) \in A(X)} x_{ij} &\leq |A(X)| - 1, && \forall X \in \mathcal{X}, \\
q_j &\geq q_i + \omega_j - \min\{Q, Q + \omega_i\}(1 - x_{ij}), && \forall (i, j) \in A, \\
\max\{0, \omega_i\} \leq q_i &\leq \min\{Q, Q + \omega_i\}, && \forall i \in P \cup D, \\
q_i &= \omega_i, && \forall i \in T, \\
q_i &= 0, && i = 0, \\
q_i &\leq (1 - x_{ij}) \min\{Q, Q + \omega_i\}, && \forall (i, j) \in R, \\
x_{ij} &= 0, && \forall (i, j) \in O, \\
x_{ij} &\in \{0, 1\}, && \forall (i, j) \in A, \\
t_i, q_i &\in \mathbb{R}_+, && \forall i \in V
\end{aligned}$$

5.2 Polyhedral Analysis

Polyhedral analysis of the EDP is extremely challenging in the general case, since the numbers of constraints and decision variables in the formulation depend not only on the number of requests, but also on how the requests are located in the building. Due to this reason, we can properly study the structure of the EDP polytope only under very specific cases, including up-peak traffic and down-peak traffic, which are the topics of Chapters 6 and 7. Therefore, the main point of this section is to give just essential definitions and state some useful propositions that can be applied in the analysis of up-peak and down-peak traffic patterns.

Polyhedral analysis is often desirable to carry out when the number of decision variables in the formulation is the smallest possible. For this reason, we show a way, how to eliminate the load variables q_i from the EDP formulation: replace constraints (5.5)-(5.8), (5.12), and (5.11) with *rounded capacity constraints*

$$x(S,\overline{S}) \geq \max\left\{1, \left\lceil \frac{|\Omega(S)|}{Q} \right\rceil \right\} \quad \forall S \subset V \setminus \{0\}, |S| \geq 2, \tag{5.15}$$

and *extended boarding constraints*

$$\sum_{(i,j)\in A(X)} x_{ij} \leq |A(X)| - 1, \forall X \in \mathcal{X}' \tag{5.16}$$

where $\mathcal{X}'$ is the union of $\mathcal{X}$ and all forward paths of the form $X = (2n + e, k_1, \ldots, k_{r-1}, k_r)$ where $(k_{r-1}, k_r) \in R$, and $\Omega(X \setminus \{k_r\}) > 0$. Rounded capacity constraints are originally presented in the context of VRPs [25], but they apply directly to the EDP formulation as well. The extension of $\mathcal{X}$ reflects the assumption that reversals are forbidden: since elevator e reverses its direction either at vertex k_{r-1} or k_r so that its load is positive, such a path must be infeasible. Due to the elimination of load variables, polyhedral analysis is now possible to be carried out in the $(\mathbf{x}, \mathbf{t})$-space instead of the $(\mathbf{x}, \mathbf{t}, \mathbf{q})$-space. Let us define the EDP polytope.

Definition 4. The polytope of the EDP, P_{EDP}, is the convex hull of the feasible solutions projected onto the $(\mathbf{x}, \mathbf{t})$-space, i.e.,

$$\begin{aligned} P_{EDP} = \operatorname{conv} \big\{ (\mathbf{x}, \mathbf{t}) \in \{0,1\}^{|A|} \times \mathbb{R}_+^{|V|} \mid (\mathbf{x}, \mathbf{t}) \text{ satisfies } (5.1) - (5.4), \\ (5.9) - (5.10), (5.13), \text{ and } (5.15) - (5.16) \big\} \end{aligned} \tag{5.17}$$

From now on, when we say that $(\mathbf{x}, \mathbf{t})$ is a feasible solution we refer to the fact that $(\mathbf{x}, \mathbf{t})$ satisfies the constraints of Definition 4. Suppose that we are given a set of feasible solutions $(\mathbf{x}^i, \mathbf{t}^i)$, $i = 1, \dots, m$, whose $\mathbf{x}$ variables are affinely independent. Next, we show that if one of the solutions is "flexible" with respect to all time windows $[a_j, b_j]$, i.e., for some $i \in \{1, \dots, m\}$ $t^i_j < b_j$ $\forall j \in V$, then the dimension of P_{EDP} is at least $m + |V| - 1$. In order to prove the claim, we first need two definitions.

Definition 5. A feasible solution $(\mathbf{x}, \mathbf{t})$ of the EDP is called *ϵ-flexible* if for some $\epsilon > 0$ a solution $(\mathbf{x}, \mathbf{t} + \epsilon * \mathbf{1}_{|\mathbf{t}|})$ is also feasible, where $\mathbf{1}_{|\mathbf{t}|}$ is the all-ones vector of size $|\mathbf{t}|$.

Definition 6. Let $(\mathbf{x}, \mathbf{t})$ be a feasible solution of the EDP in which passengers are served by $l \leq |T|$ elevators such that elevator e_k, $k = 1, \dots, l$, visits s_{e_k} vertices from the set $P \cup D$. We define relations "$\preceq_{\mathbf{x}}$" and "$\prec_{\mathbf{x}}$" as follows: $i \preceq_{\mathbf{x}} j$ if the position of vertex $i \in V$ precedes or is equal to the position of vertex $j \in V$ in the sequence

$$(2n + e_1, \dots, 2n + e_{|T|}, i_{(e_1,1)}, i_{(e_1,2)}, \dots, i_{(e_1,s_{e_1})}, i_{(e_2,1)}, \dots, i_{(e_l,s_{e_l})}, 0), \quad (5.18)$$

where $i_{(e_k,r)} \in P \cup D$, $k = 1, \dots, l$, $r = 1, \dots, s_{e_k}$, denotes the rth vertex visited by elevator e_k on route $\mathbf{x}$. If the position of i precedes the position of j, then $i \prec_{\mathbf{x}} j$.

Now, we prove the claim by showing that $|V| + 1$ affinely independent solutions arise from any ϵ-flexible solution.

Proposition 11. Let $\mathbf{h}^i = (\mathbf{x}^i, \mathbf{t}^i)$, $i = 1, \dots, m$, be a set of feasible solutions of the EDP, where $\mathbf{x}^1, \dots, \mathbf{x}^m$ are affinely independent. If $\mathbf{h}^i$ is ϵ-flexible for some $i \in \{1, \dots, m\}$, then $\dim(P_{EDP}) \geq m + |V| - 1$.

Proof. Without loss of generality, it can be assumed that $\mathbf{h}^m$ is ϵ-flexible. Let $\mathbf{z} := (\mathbf{x}, \mathbf{t})$ denote $\mathbf{h}^m = (\mathbf{x}^m, \mathbf{t}^m)$, and define a set of solutions $\{\mathbf{z}^i_\epsilon\}_{i \in V}$ as follows:

$$\mathbf{z}^i_\epsilon = \{(\mathbf{x}^i_\epsilon, \mathbf{t}^i_\epsilon) |\ \mathbf{x}^i_\epsilon = \mathbf{x},\ (t^i_\epsilon)_k = (t_k + \epsilon)\chi_{\{i \preceq_{\mathbf{x}} k\}} + t_k \chi_{\{k \prec_{\mathbf{x}} i\}}\ \forall k \in V\}$$

Clearly, all $\mathbf{z}^i_\epsilon$'s are feasible solutions. Consider a set $H := \{\mathbf{h}^i\}_{i=1,\dots,m-1} \cup \{\mathbf{z}\} \cup \{\mathbf{z}^i_\epsilon\}_{i \in V}$. The vectors of H are affinely independent if

$$\sum_{i=1}^{m-1} \lambda_i (\mathbf{h}^i - \mathbf{z}) + \sum_{i \in V} \gamma_i (\mathbf{z}^i_\epsilon - \mathbf{z}) = \mathbf{0} \quad \Rightarrow \quad \lambda_i, \gamma_i = 0 \ \forall i.$$

which is the same as

$$\begin{bmatrix} \sum_{i=1}^{m-1} \lambda_i(\mathbf{x}^i - \mathbf{x}) + \sum_{i\in V} \gamma_i(\mathbf{x}_\epsilon^i - \mathbf{x}) \\ \sum_{i=1}^{m-1} \lambda_i(\mathbf{t}^i - \mathbf{t}) + \sum_{i\in V} \gamma_i(\mathbf{t}_\epsilon^i - \mathbf{t}) \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \Rightarrow \lambda_i, \gamma_i = 0 \;\; \forall i$$

$$\Leftrightarrow \begin{bmatrix} \sum_{i=1}^{m-1} \lambda_i(\mathbf{x}^i - \mathbf{x}) \\ \sum_{i=1}^{m-1} \lambda_i(\mathbf{t}^i - \mathbf{t}) + \sum_{i\in V} \gamma_i(\mathbf{t}_\epsilon^i - \mathbf{t}) \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \Rightarrow \lambda_i, \gamma_i = 0 \;\; \forall i$$

As $\mathbf{x}^1, \ldots, \mathbf{x}^m$ are affinely independent, it follows that $\lambda_i = 0 \, \forall i \in \{1, \ldots, m-1\}$. It remains to show that

$$\sum_{i\in V} \gamma_i(\mathbf{t}_\epsilon^i - \mathbf{t}) = \mathbf{0} \Rightarrow \gamma_i = 0 \;\; \forall i \in V.$$

Re-index variables $\mathbf{t}_\epsilon^i$ and γ_i, $i \in V$, so that i corresponds the ith vertex in the sequence (5.18). By construction of $\mathbf{z}_\epsilon^i$'s,

$$\sum_{i=1}^{|V|} \gamma_i(\mathbf{t}_\epsilon^i - \mathbf{t}) = \epsilon * \Big[\gamma_1, \gamma_1 + \gamma_2, \cdots, \sum_{i=1}^{|V|-1} \gamma_i, \sum_{i=1}^{|V|} \gamma_i\Big]^T = \mathbf{0} \Rightarrow \gamma_i = 0 \;\; \forall i \in V,$$

and since the cardinality of H is $m + |V|$, the claim follows. □

The reader should note, that the concept of ϵ-flexibility is primarily a tool in our proofs due to the following reason: a feasible solution can always be made ϵ-flexible just by slightly increasing the values of all upper bounds b_i, $i \in V$. Since these increases can be arbitrarily small, the value of the objective function to be minimized essentially stays the same.

Most of the constraints in the EDP polytope relate to $\mathbf{x}$-variables, so that $\mathbf{x}$ and $\mathbf{t}$ are dependent only through time consistency constraints (5.10). Next, we demonstrate how every feasible $\mathbf{x}$-route can be extended to the $(\mathbf{x}, \mathbf{t})$-space, so that these constraints are satisfied.

Definition 7. A vector $\mathbf{x} \in \{0,1\}^{|A|}$ is a *feasible* $\mathbf{x}$*-route* of the EDP if it satisfies constraints (5.1)-(5.4), (5.13), and (5.15)-(5.16).

Definition 8. Let $\mathbf{x}$ be a feasible $\mathbf{x}$-route of the EDP. If the time vector $\mathbf{t} = \mathbf{t}(\mathbf{x})$ where

$$\begin{aligned} \mathbf{t}(\mathbf{x}) = \big\{ \mathbf{t} \in \mathbb{R}_+^{2n+2} | \; & x_{kl} t_l = x_{kl} \max\{t_k + \tau_{kl}, a_l\} \; \forall (k,l) \in A \setminus (V, \{0\}), \\ & t_i = 0 \; \forall i \in T, \; t_0 = \max_{k\in D}\{t_k\} \big\}, \end{aligned} \tag{5.19}$$

we say that $\mathbf{t}$ *is induced by* $\mathbf{x}$.

For each feasible $\mathbf{x}$-route the solution $(\mathbf{x}, \mathbf{t}(\mathbf{x}))$ clearly satisfies the time consistency constraints, but nothing guarantees that $\mathbf{t}(\mathbf{x})$ would also satisfy the time window constraints (5.9). If we make a further assumption that these constraints are relaxed, then every feasible $\mathbf{x}$-route implies a feasible solution, which enables us to consider $\mathbf{x}$ and $\mathbf{t}$ dimensions separately. Thus, it is useful to define also an EDP polytope that is restricted to the $\mathbf{x}$-space.

Definition 9. The EDP polytope restricted to the $\mathbf{x}$-space is

$$P_{EDP|_x} = \text{conv}\big\{\mathbf{x} \in \{0,1\}^{|A|} \mid \mathbf{x} \text{ satisfies } (5.1) - (5.4), (5.13), \text{ and } (5.15) - (5.16)\big\}, \tag{5.20}$$

Whenever the time window constraints are relaxed, the dimensions of P_{EDP} and $P_{EDP|_x}$ are connected by a simple relation:

Proposition 12. If the time window constraints (5.9) are relaxed, the dimension of the EDP polytope is given by

$$\dim(P_{EDP}) = \dim(P_{EDP|_x}) + |V| \tag{5.21}$$

Proof. Let the dimension of $P_{EDP|_x}$ be m, i.e. $P_{EDP|_x}$ contains $m+1$ affinely independent $\mathbf{x}$-routes $\mathbf{x}^i$, $i = 1, \ldots, m+1$. Since the time windows are relaxed, each solution induced by $\mathbf{x}$, $(\mathbf{x}^i, \mathbf{t}(\mathbf{x}^i))$, is feasible. In addition, all feasible solutions are ϵ-flexible so, by Proposition 11, $\dim(P_{EDP}) \geq m + |V| = \dim(P_{EDP|_x}) + |V|$. On the other hand, according to equations (3.5) and (3.6), $\dim(P_{EDP}) = |A| + |V| - \text{rank}(P_{EDP}) \leq |A| + |V| - \text{rank}(P_{EDP|_x}) = \dim(P_{EDP|_x}) + |V|$, which gives the result. □

Chapter 6

EDP: Up-peak Traffic Pattern

6.1 General Assumptions

Up-peak traffic is a situation in a building in which all or most of the requests are heading upwards. A common example of up-peak traffic is the morning peak in office buildings when people come to work and they travel from the lobby to the upper floors of the building to their offices. The purpose of this section is to analyze the polyhedral structure of three different up-peak traffic patterns. All three cases are based on four general assumptions:

A1. There are n non-assigned requests, which are ordered according to their arrival times:
$\mathcal{F}=\emptyset$ and $\gamma_1 > \gamma_2 > \cdots > \gamma_n \geq 0$.

A2. All requests come from the first floor and the destination floors of the requests are in ascending order:
$f(i) = 1 \ \forall i \in P$ and $f(n+i) > f(n+j) > 1 \ \forall i > j \quad i, j \in P$

A3. Each vertex has a relaxed time window:
$[a_i, b_i] = [0, \infty) \ \forall i \in V$.

A4. All elevators are in the same state and they are initially located at the first floor:
$f(2n+e) = 1 \ \forall (2n+e) \in T$.

By saying that the elevators are in the same state, it means that they are empty, they are located at the same floor, their doors are in the same position, and none of them have fixed requests. Such elevators are called symmetrical. In order to simplify the forthcoming polyhedral analysis, the symmetry is eliminated by replacing the origin depot vertex set T by one vertex, which we call a *common origin depot vertex* $+0$. Now, instead of demanding that each elevator must end its route to the common terminal depot vertex, we set the out-degree of $+0$ to be at most $|T|$:

$$x(+0, V) \leq |T|. \tag{6.1}$$

In other words, if some elevator has no assignments, it stays at vertex $+0$. This modification changes the out-degree constraints (5.2) into the form

$$x(i, V) = 1, \quad \forall i \in P \cup D. \tag{6.2}$$

The general formulation of the EDP can be simplified by using assumptions A1-A4. First, all requests are non-assigned, which makes fixing constraints (5.4) redundant. Also, constraints (5.9) are unnecessary, because each vertex has a relaxed time window. Time consistency conditions can be written in the form

$$t_j \geq t_i + \tau_{ij} - M(1 - x_{ij}), \quad \forall (i, j) \in A, \tag{6.3}$$

where M is a large positive constant, so that $M \geq t_i + \tau_{ij}\ \forall (i, j) \in A$ in all feasible solutions. It can be assumed that the longest total journey time is reached when there is only one serving elevator, which can carry one passenger at a time. In this case

$$M = \tau_{+0,1} + \sum_{i=1}^{n-1} (\tau_{i,n+i} + \tau_{n+i,i+1}) + \tau_{n,2n}.$$

Due to the reversal constraints, direction of an elevator cannot be changed unless it is empty; therefore,

$$x_{n+i,n+j} = 0 \quad \forall n + j < n + i, \quad n + i, n + j \in D. \tag{6.4}$$

Service order constraints can also be simplified: according to assumption A2, $f(i) \neq f(n+j) \neq f(n+i)\ \forall i \in P, \forall n+i, n+j \in D$, and thus, $O_i = O_{iii} = \emptyset$ for any feasible solution $\mathbf{x}^*$. As $O_{ii} = \{(i, j) \in A|\ i, j \in P, j < i, f(i) = f(j), d(i) = d(j)\} = \{(i, j) \in A|\ i, j \in P, j < i\}$, service order constraints are employed by equations

$$x_{ij} = 0 \quad \forall i > j, \quad i, j \in P. \tag{6.5}$$

In addition to formulas (6.1)-(6.5), the following proposition is also found useful.

Proposition 13. If assumptions A1-A4 hold, then the equation

$$x_{ij} - x_{n+i,n+j} = 0, \quad \forall i, j \in P, n+i, n+j \in D, \tag{6.6}$$

is valid for the EDP.

Proof. If $i > j$, then $x_{ij} = 0$ by (6.5) and $x_{n+i,n+j} = 0$ by (6.4). Hence, it can be assumed that $i < j$. Suppose $x_{ij} = 1$. There cannot be a request k, $i < k < j$, which is assigned to the same elevator as requests i and j, because passengers are assumed to board the elevator in the ascending order of their arrival times. Since $n+i$ and $n+j$ must be visited by the same elevator and any vertex $n+k$, $i < k < j$ between them cannot be visited, it follows that either $x_{n+i,n+j} = 1$ or $x_{n+j,n+i} = 1$. The latter option contradicts constraint (6.4), and thus, $x_{n+i,n+j} = 1$.

Suppose $x_{n+i,n+j} = 1$. If it were $x_{ij} = 0$, there would be request k such that $i < k < j$ and $x_{ik} = 1$. But since $x_{n+i,n+j} = 1$, vertices $n+i, n+k, n+j$ cannot be visited without violating reversal constraints, so it must hold that $x_{ij} = 1$, which completes the proof. □

Next, we study separately three different cases: in the first case the number of elevators equals the number of requests and elevators have unlimited capacity. In the second case, the number of elevators equals the number of requests, but elevators have restricted capacity. In the third case, elevators have unlimited capacity, but the number of elevators is less than the number of requests.

6.2 Case 1: No Restrictions

6.2.1 Assumptions

In addition to assumptions A1-A4, we assume that

A5. There are as many elevators as requests: $|T| = l = n$

A6. Elevators have unlimited capacity: $Q = \infty$

The EDP which satisfies conditions A1-A6 is denoted by EDP_n^∞. Since the capacity of elevators is unlimited (A6) and all passengers travel upwards from the lobby (A2), the boarding constraints (5.11) force that all passengers who are assigned to the same elevator must have boarded before the elevator can leave the floor. In practice, it means that after the elevator goes up, returning to the lobby is not possible, i.e.,

$$x_{n+i,j} = 0 \quad \forall n+i \in D, \forall j \in P. \tag{6.7}$$

As a consequence of (6.7) and (6.4), load variables can be omitted from the formulation without introducing rounded capacity constraints (5.15) or extended boarding constraints (5.16). By removing redundant arcs, one obtains a reduced graph G_n^∞ presented in Figure 6.1.

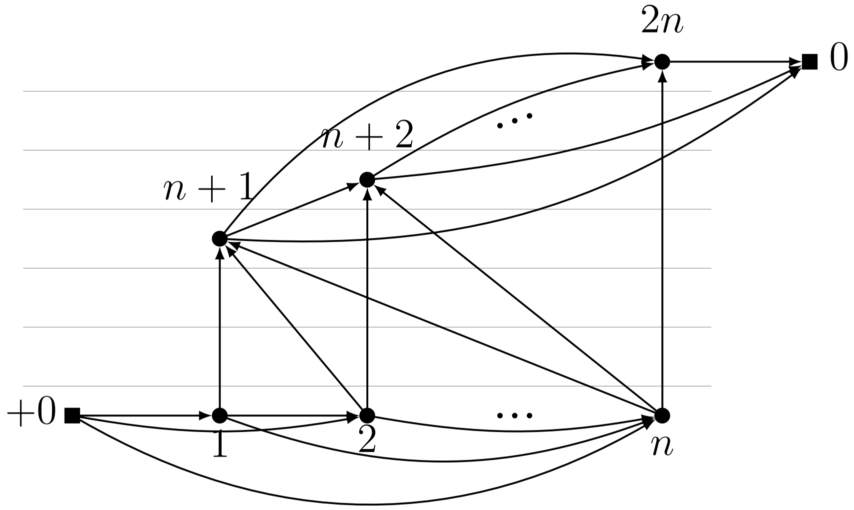


Figure 6.1: The reduced graph, G_n^∞, of the problem EDP_n^∞.

6.2.2 Polyhedral Analysis

We define the polytope of EDP_n^∞ as the convex hull of feasible solutions of the problem under assumptions A1-A6, i.e.,

$$P_{EDP_n^\infty} := \text{conv}\{(\mathbf{x}, \mathbf{t}) \in \{0,1\}^{|A_n^\infty|} \times \mathbb{R}_+^{2n+2} | \, (\mathbf{x}, \mathbf{t}) \text{ satisfies } (5.1), (5.3), \text{ and } (6.2) - (6.7)\}, \quad (6.8)$$

where A_n^∞ is the set of arcs in graph G_n^∞. Polytope $P_{EDP_n^\infty|x}$ is defined similarly but without time consistency constraints (6.3). In the next theorem we determine the number of feasible EDP_n^∞ solutions projected onto $\mathbf{x}$-space: the number of solutions is counted only in the $\mathbf{x}$-space since there are clearly infinitely many solutions in the $(\mathbf{x}, \mathbf{t})$-space. One should note that projected solutions are now the same thing as feasible $\mathbf{x}$-routes, since time window constraints are relaxed. The set of projected solutions is denoted by H_n^∞.

Theorem 5. The number of feasible solutions to EDP_n^∞ in the $\mathbf{x}$-space is the nth Bell number,

$$|H_n^\infty| = B_n = \sum_{k=0}^{n} S(n,k) = \sum_{k=0}^{n} \frac{1}{k!} \sum_{j=1}^{k} (-1)^{k-j} \binom{n}{k} j^n \tag{6.9}$$

Proof. We know that n requests can be assigned to n elevators by B_n different ways, where B_n is the nth Bell number and indicates the number of partitions of a set. According to constraint (6.5), the pickup order is unique. The delivery order is also unique, which is an immediate consequence of constraints (5.1), (6.2), and (6.6). Thus, the number of feasible solutions is B_n. □

Lemma 1. The number of arcs in G_n^∞ is $|A_n^\infty| = \frac{3}{2}n(n+1)$

Proof. $|(+0, V)| = n$
$|(i, V)| = |(i, P)| + |(i, D)| = n - i + i = n, \quad i = 1, \ldots, n$
$|(n+i, V)| = |(n+i, D)| + |(n+i, 0)| = n - i + 1, \quad i = 1, \ldots, n$
$\Rightarrow |A_n^\infty| = n + \sum_{i=1}^{n} n + \sum_{i=1}^{n}(n-i+1) = n + n^2 + n^2 - \frac{1}{2}n(n+1) + n = \frac{3}{2}n(n+1)$. □

In order to determine the dimension of EDP_n^∞, consider constraints (5.1), (6.2), and (6.6). Constraint $x_{n-1,n} - x_{2n-1,2n} = 0$ is a linear combination of equations $x(V, n) = 1$, $x(V, 2n) = 1$, and $x_{i,n} - x_{n+i,2n} = 0$, $i = 1, \ldots, n-2$ since $x(V, n) - x(V, 2n) - \sum_{i=1}^{n-2}(x_{in} - x_{n+i,2n}) = 1 - 1 - 0 \Leftrightarrow x_{n-1,n} - x_{2n-1,2n} = 0$. The next lemma shows that the rest of the equations are linearly independent.

Lemma 2. Consider graph G_n^∞. If $[\mathbf{A}_S, -\mathbf{b}_S]$ is the matrix defined by constraints (5.1), (6.2), and

$$x_{ij} - x_{n+i,n+j} = 0, \quad i, j \in P, \quad i < j, \quad i \le n-2, \tag{6.10}$$

then the rows of the matrix are linearly independent and $\text{rank}(\mathbf{A}_S) = \frac{1}{2}n^2 + \frac{7}{2}n - 1$.

Proof. We show that the equations in system $\mathbf{A}_S\mathbf{x} = \mathbf{b}_S$ are linearly independent by finding a vector $\mathbf{x} \in \{0,1\}^{|A_n^\infty|}$ for each equation $\mathbf{a}_i'\mathbf{x} = b_i$, $i = 1, \ldots, |\mathbf{b}_S|$, such that $\mathbf{a}_k'\mathbf{x} = b_k \ \forall k \neq i$ but $\mathbf{a}_i'\mathbf{x} \neq b_i$. We have six different cases to consider:
1a) $x(V, i) = 1$, $i \in P$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$, $R = (+0, 1, \ldots, i-1, i+1, n, \ldots, n+i-1, n+i+1, \ldots, 2n, 0) \cup (i, n+i, 0)$, and $\chi_{A(R)}$ denotes the characteristic function of the arc set of R.

1b) $x(V, n+i) = 1$, $n+i \in D$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$ and $R = (n+i, 2n-1, 0) \cup (+0, 1, \ldots, i-1, i+1, \ldots, n-2, n+1, \ldots, n+i-1, n+i+1, \ldots, 2n-2, 0) \cup (+0, i, n-1, n, 2n, 0)$.
2a) $x(i, V) = 1$, $i \in P$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$ and $R = (+0, i) \cup (+0, n, n+i, 0) \cup (+0, 1, \ldots, i-1, i+1, \ldots, n-1, n+1, \ldots, n+i-1, n+i+1, \ldots, 2n, 0)$
2b) $x(n+i, V) = 1$, $n+i \in D$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$ and $R = (+0, i, n+i) \cup (+0, n, n+i, 0) \cup (+0, 1, \ldots, i-1, i+1, \ldots, n+i-1, n+i+1, \ldots, 2n, 0)$
3a) $x_{ij} - x_{n+i,n+j} = 0$, $i \in P \setminus \{n-1, n\}$, $j \in P, i+2 \leq j$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$ and $R = (+0, 1 \ldots, i, n+1, \ldots, n+i, n+j, \ldots, 2n-1, 0) \cup (+0, i+1, \ldots, j-1, n+i+1, \ldots, n+j-1, 0) \cup (+0, j, \ldots, n, 2n, 0)$
3b) $x_{ij} - x_{n+i,n+j} = 0$, $i \in P \setminus \{n-1, n\}$, $j \in P, i+1 = j$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$ and $R = (+0, 1, \ldots, i, n+1, \ldots, n+i, n+j, \ldots, 2n-1, 0) \cup (+0, j, \ldots, n, 2n, 0)$.

Because the system also has a feasible solution, e.g., $\mathbf{x} = \chi_{A(R)}$ where $R = (+0, 1, \ldots, 2n, 0)$, so, by Proposition 2, (5.1), (6.2), and (6.10) are linearly independent equations, and the rank of the system is $\text{rank}(\mathbf{A}_S) = \text{rank}([\mathbf{A}_S, -\mathbf{b}_S]) = 2n + 2n + \frac{1}{2}n(n-1) - 1 = \frac{1}{2}n^2 + \frac{7}{2}n - 1$. □

Theorem 6. The dimension of $P_{EDP_n^\infty}$ is $n^2 + 3$.

Proof. We first determine the dimension of $P_{EDP_n^\infty|x}$ and then extend the result to the $(\mathbf{x}, \mathbf{t})$-space. By using Theorem 2, Lemma 1, and Lemma 2, respectively, an upper bound for the dimension of $P_{EDP_n^\infty|x}$ is obtained: $\dim(P_{EDP_n^\infty|x}) = |A_n^\infty| - \text{rank}(P_{EDP_n^\infty}|x) = \frac{3}{2}n(n+1) - \text{rank}(P_{EDP_n^\infty|x}) \leq \frac{3}{2}n(n+1) - (\frac{1}{2}n^2 + \frac{7}{2}n - 1) = n^2 - 2n + 1$. Here the usage of Lemma 2 is based on the fact that $\mathbf{A}_S$ is a submatrix of the constraint matrix of $P_{EDP_n^\infty|x}$, and therefore, $\text{rank}(\mathbf{A}_S) \leq \text{rank}(P_{EDP_n^\infty|x})$. We show that the upper bound holds as equality by finding $n^2 - 2n + 2$ affinely independent $\mathbf{x}$-routes, which belong to polytope $P_{EDP_n^\infty|x}$.

Let H_1, H_2, H_3, and Z be sets of feasible $\mathbf{x}$-routes defined as follows: $H_1 = \{\mathbf{h}_1^{i,n+j}\}_{i,j \in P: 1 < j < i}$, $H_2 = \{\mathbf{h}_2^{ij}\}_{i,j \in P: 1 < j < i}$, $H_3 = \{\mathbf{h}_3^i\}_{i \in P \setminus \{1\}}$, $Z = \{z\}$, where

$$\begin{aligned}
\mathbf{h}_1^{i,n+j} &= \left\{\mathbf{x} \in H_n^\infty |\ x_{ji} = x_{i,n+j} = 1,\ x_{k,n+k} = 1\ \ \forall k \in P \setminus \{i,j\}\right\} \\
\mathbf{h}_2^{ij} &= \left\{\mathbf{x} \in H_n^\infty |\ x_{1i} = x_{ij} = x_{j,n+1} = 1,\ x_{k,n+k} = 1\ \ \forall k \in P \setminus \{1,i,j\}\right\} \\
\mathbf{h}_3^{i} &= \left\{\mathbf{x} \in H_n^\infty |\ x_{1i} = x_{i,n+1} = 1,\ x_{k,n+k} = 1\ \ \forall k \in P \setminus \{1,i\}\right\}, \text{ and} \\
\mathbf{z} &= \left\{\mathbf{x} \in H_n^\infty |\ x_{k,n+k} = 1\ \ \forall k \in P\right\}
\end{aligned}$$

When $H_1 \cup H_2 \cup H_3 \cup Z$ is denoted by H, we can make the following observations:

1. For each $i, j \in P$, $1 < j < i$, the only solution in H in which $x_{i,n+j} = 1$ is $\mathbf{h}_1^{i,n+j}$.

2. For each $i, j \in P$, $1 < j < i$, the only solution in $H \setminus H_1$ in which $x_{ij} = 1$ is $\mathbf{h}_2^{ij}$.

3. The only solution in $H \setminus (H_1 \cup H_2)$ in which $x_{2,n+1} = 1$ is $\mathbf{h}_3^2$.

4. For each i, $3 \leq i \leq n$, the only solution in $H \setminus (H_1 \cup H_2 \cup \mathbf{h}_3^2)$ in which $x_{1i} = 1$ is $\mathbf{h}_3^i$.

5. The only solution in $H \setminus (H_1 \cup H_2 \cup H_3) = Z$ is $\mathbf{z}$.

These observations guarantee that there exists a sequence of ordered pairs $\big((i_k, j_k)\big)_{k=1}^{|H|}$ that satisfies the conditions of Proposition 1, and hence, the vectors of H are linearly independent. Since linearly indepedent vectors are affinely independent as well, and $|H| = |H_1| + |H_2| + |H_3| + |Z| = \frac{(n-2)(n-1)}{2} + \frac{(n-2)(n-1)}{2} + n - 1 + 1 = n^2 - 2n + 2$, the dimension of $P_{EDP_n^\infty|x}$ is $n^2 - 2n + 1$. Because time window constraints are relaxed, we know by Proposition 12 that $\dim(P_{EDP_n^\infty}) = \dim(P_{EDP_n^\infty|x}) + |V| = n^2 - 2n + 1 + 2n + 2 = n^2 + 3$.

□

6.3 Case 2: Restricted Capacity

6.3.1 Assumptions

In addition to assumptions A1-A4, it is assumed that

A5. There are as many elevators as requests: $|T| = l = n$

A6'. Elevators can simultaneously carry at most m, $1 \leq m \leq n$, passengers, and each request represents a single passenger: $Q = m$, $\omega_i = 1$, $\omega_{n+i} = -1$, $i = 1, \ldots, n$.

The EDP which satisfies conditions A1-A5 and A6' is denoted by EDP_n^m. A reduced graph of the problem is denoted by G_n^m.

Contrary to Case 1, all arcs of the form $x_{n+i,j}$, $n + i \in D$, $j \in P$ are not excluded. Due to assumption A6', elevators which have more than m requests to serve cannot carry all of their passengers at the same time, but need to return to the lobby at some point in order to complete the task (See Figure 6.2). Hence, load variables q_i cannot be eliminated by the equation (6.7) but we need to use rounded capacity constraints (5.15).

6.3.2 Polyhedral Analysis

The polytope of EDP_n^m is

$$P_{EDP_n^m} := \text{conv}\{(\mathbf{x}, \mathbf{t}) \in \{0,1\}^{|A_n^m|} \times \mathbb{R}_+^{2n+2} | \, (\mathbf{x}, \mathbf{t}) \text{ satisfies } (5.1), (5.3), (5.11), (5.15), \text{ and } (6.2) - (6.6)\}, \tag{6.11}$$

Polytope $P_{EDP_n^m|x}$ is defined similarly.

Theorem 7. The number of feasible solutions to EDP_n^m in the $\mathbf{x}$-space is the nth Bell number,

$$|H_n^m| = |H_n^\infty| = B_n \tag{6.12}$$

Proof. Since there are no restrictions on how many requests can be assigned to one elevator, the number of solutions remains the same as in Case 1. The only difference compared to Case 1 is, that if more than m requests are assigned to some elevator, it cannot pick up all passengers at one time (See Figure 6.2). Hence, $|H_n^m| = |H_n^\infty|$. □

Lemma 3. The number of arcs in G_n^m is

$$|A_n^m| = \Big(\frac{3}{2}n(n+1) + \frac{(n-m)(n-m+1)}{2}\Big)\chi_{\{m\geq 2\}} + \Big(\frac{1}{2}n^2 + \frac{5}{2}n\Big)\chi_{\{m=1\}}$$

Proof. Suppose first that $m \geq 2$. Clearly, $A_n^\infty \subset A_n^m$. In addition, returning from delivery vertices back to pick-up vertices is now possible if the number of requests assigned to some elevator is more than its capacity. Service order constraints, however, impose a restriction $x_{n+i,j} = 0 \; \forall i > j, \; i,j \in P$, which gives $|(n+i, P)| = (n-i)\chi_{\{i\geq m\}}$. By summing over $i \in P$, we obtain that $|A_n^m| = |A_n^\infty| + \sum_{i=1}^n (n-i)\chi_{\{i\geq m\}} = |A_n^\infty| + \sum_{i=m}^n (n-i) = \frac{3}{2}n(n+1) + \frac{(n-m)(n-m+1)}{2}$.

Suppose now that $m = 1$. Since $m = 1$, each elevator can carry only one passenger at a time and this passenger must be delivered to her destination floor before the next passenger can be picked up. Therefore, $x_{ij} = x_{n+i,n+j} = x_{i,n+j} = 0 \; \forall i,j \in P, i \neq j$. For this reason,

$$\begin{aligned} |A_n^1| &= |(+0, V)| + \sum_{i=1}^n \Big(|(i, V)| + |(n+i, V)|\Big) \\ &= |(+0, P)| + \sum_{i=1}^n \Big(|(i, n+i)| + |(n+i, P)| + |(n+i, 0)|\Big) \\ &= n + \sum_{i=1}^n \Big(1 + n - i + 1\Big) = \frac{1}{2}n^2 + \frac{5}{2}n. \end{aligned}$$

□

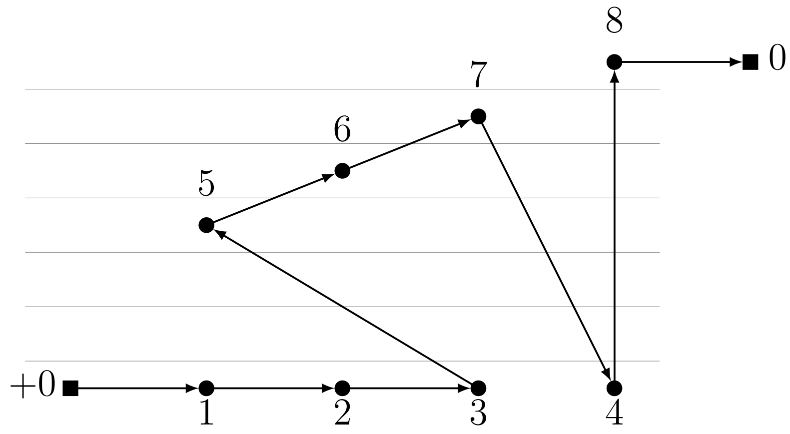


Figure 6.2: A feasible solution to EDP_n^m when $n = 4$ and $m = 3$.

Next, we determine the dimension of $P_{EDP_n^m}$, which is divided into three parts: we study separately cases $m = 1$, $m = 2$, and $m \geq 3$. We start with the case $m \geq 3$ because the result in that case follows almost immediately from Theorem 6.

Theorem 8. When $m \geq 3$, the dimension of $P_{EDP_n^m}$ is $n^2+3+\frac{(n-m)(n-m+1)}{2}$.

Proof. By Proposition 12, $\dim(P_{EDP_n^m|x}) = \dim(P_{EDP_n^m}) - (2n+2)$, so it is enough to show that the dimension of $P_{EDP_n^m|x}$ is $n^2 + 3 + \frac{(n-m)(n-m+1)}{2} - (2n+2)$. The dimension cannot be larger than this since $\dim(P_{EDP_n^m|x}) \leq |A_n^m| - \text{rank}(\mathbf{A}_S) = n^2 + 3 + \frac{(n-m)(n-m+1)}{2} - (2n+2)$, where $\mathbf{A}_S$ refers to the constraint matrix defined in Lemma 2. We show that $P_{EDP_n^m|x}$ contains $n^2 + 3 + \frac{(n-m)(n-m+1)}{2} - (2n+2) + 1$ affinely independent $\mathbf{x}$-routes. Let sets H_1, H_2, H_3, and Z be defined as in Theorem 6 with the exception that H_n^∞ and A_n^∞ are replaced by H_n^m and A_n^m, respectively. These $\mathbf{x}$-routes clearly belong to polytope $P_{EDP_n^m|x}$. Moreover, let $H_4 := \{\mathbf{h}_4^{n+i,j}\}_{i,j\in P:j>i\geq m}$, where

$$\mathbf{h}_4^{n+i,j} = \left\{\mathbf{x} \in H_n^m |\ x_{n+i,j} = 1,\ x(+0,V) = n-m-1\right\}.$$

By observing that the only $\mathbf{x}$-route in $H := H_1 \cup H_2 \cup H_3 \cup H_4 \cup Z$ where $x_{n+i,j} = 1$, $i,j \in P, j > i \geq m$, is $\mathbf{h}_4^{n+i,j}$, we can apply Proposition 1 since $\mathbf{x}$-routes in $H_1 \cup H_2 \cup H_3 \cup Z$ are already known to be linearly independent. By Lemma 3, the cardinality of H_4 is $\frac{(n-m)(n-m+1)}{2}$, and hence, $|H| = |H_1 \cup H_2 \cup H_3 \cup H_4 \cup Z| = |H_1 \cup H_2 \cup H_3 \cup Z| + |H_4| = n^2 - 2n + 2 + \frac{(n-m)(n-m+1)}{2}$ as desired. □

Next, the dimension of $P_{EDP_n^m}$ is determined when $m = 1$ and $m = 2$. We need the following lemmas:

Lemma 4. Consider graph G_n^1. If $[\mathbf{A}_S, -\mathbf{b}_S]$ is the matrix defined by constraints (6.2) and

$$x(V,i) = 1, \quad \forall i \in P, \tag{6.13}$$

then the rows of the matrix are linearly independent and $\text{rank}(\mathbf{A}_S) = 3n$.

Proof. We use the sama technique as in Lemma 2. There are 3 different cases to consider:
1a) $x(V,i) = 1$, $i \in P$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$,
$R = (+0, 1, n+1, 2, n+2, \dots, n+i-1, 0) \cup (i, n+i, i+1, \dots, 0)$
2a) $x(i,V) = 1$, $i \in P$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$,
$R = (+0, 1, n+1, 2, n+2, \dots, i) \cup (n+i, i+1, n+i+1, \dots, 0)$
2b) $x(n+i,V) = 1$, $n+i \in D$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$,
$R = (+0, 1, n+1, 2, n+2, \dots, i, n+i) \cup (+0, i+1, n+i+1, \dots, 0)$.

Because the system also has a feasible solution $\mathbf{x} = \chi_{A(R)}$ where $R = (+0, 1, n+1, 2, n+2, \dots, n, 2n, 0)$, so, by Proposition 2, (6.2) and (6.13) are linearly independent equations and $\text{rank}(\mathbf{A}_S) = \text{rank}([\mathbf{A}_S, -\mathbf{b}_S]) = 2n+n = 3n$. □

Lemma 5. Consider graph G_n^2. If $[\mathbf{A}_S, -\mathbf{b}_S]$ is the matrix defined by constraints (6.2), (6.13),

$$x_{ij} = x_{n+i,n+j}, \quad i < j, \quad i,j \in P, \text{ and} \tag{6.14}$$

$$x_{ij} = x_{j,n+i}, \quad i < j, \quad i,j \in P \tag{6.15}$$

then the rows of the matrix are linearly independent and $\text{rank}(\mathbf{A}_S) = n^2 + 2n$.

Proof. There are 5 different cases to consider:
1) $x(V,i) = 1$, $i \in P$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$,
$R = \bigcup_{k \in P \setminus \{i\}} (+0, k, n+k, 0) \cup (i, n+i, 0)$
2a) $x(i,V) = 1$, $i \in P$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$,
$R = \bigcup_{k \in P \setminus \{i\}} (+0, k, n+k, 0) \cup (+0, i) \cup (n+i, 0)$
2b) $x(n+i,V) = 1$, $n+i \in D$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$,
$R = \bigcup_{k \in P \setminus \{i\}} (+0, k, n+k, 0) \cup (+0, i, n+i)$
3) $x_{ij} = x_{n+i,n+j}$, $i < j$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$,
$R = \bigcup_{k \in P \setminus \{i,j\}} (+0, k, n+k, 0) \cup (+0, i, j, n+i, 0) \cup (n+j, 0)$
4) $x_{ij} = x_{j,n+i}$, $i < j$, is not satisfied when $\mathbf{x} = \chi_{A(R)}$,
$R = \bigcup_{k \in P \setminus \{i,j\}} (+0, k, n+k, 0) \cup (+0, i, j, n+j, 0) \cup (n+i, n+j, 0)$

Because the system also has a feasible solution $\mathbf{x} = \chi_{A(R)}$ where $R = (+0, 1, 2, n+1, n+2, 3, 4, n+3, n+4, \dots, 2n, 0)$, so, by Proposition 2, the rows of $\mathbf{A}_S$ are linearly independent, and $\text{rank}(\mathbf{A}_S) = \text{rank}([\mathbf{A}_S, -\mathbf{b}_S]) = n + n + n + n(n-1)/2 + n(n-1)/2 = n^2 + 2n$. □

Theorem 9. The dimension of $P_{EDP_n^1}$ is $\frac{1}{2}n^2 + \frac{3}{2}n + 2$.

Proof. By Proposition 12, $\dim(P_{EDP_n^1|x}) = \dim(P_{EDP_n^1}) - (2n+2)$, so it is enough to show that the dimension of $P_{EDP_n^1|x}$ is $\frac{1}{2}n^2 + \frac{3}{2}n + 2 - (2n+2) = \frac{1}{2}n^2 - \frac{1}{2}n$. The dimension cannot be larger than this since $\dim(P_{EDP_n^1|x}) \leq |A_n^1| - \text{rank}(\mathbf{A}_S) = \frac{1}{2}n^2 + \frac{5}{2}n - 3n = \frac{1}{2}n^2 - \frac{1}{2}n$, where $\mathbf{A}_S$ refers to the constraint matrix defined in Lemma 4. We show that $P_{EDP_n^1|x}$ contains $\frac{1}{2}n^2 - \frac{1}{2}n + 1$ affinely independent **x**-routes. Let sets H_4 and Z be defined as in Theorem 8. All **x**-routes in H_4 and Z clearly belong to $P_{EDP_n^1}$. By Theorem 8, we know that $H_4 \cup Z$ is a linearly independent set and since $|H_4 \cup Z| = |H_4| + |Z| = \frac{(n-1)(n-1+1)}{2} + 1 = \frac{1}{2}n^2 - \frac{1}{2}n + 1$, the claim is proved. □

Theorem 10. The dimension of $P_{EDP_n^2}$ is $n^2 + 3$.

Proof. It is enough to show that $\dim(P_{EDP_n^2|x}) = n^2 - 2n + 1$. The dimension cannot be larger than this since $\dim(P_{EDP_n^2|x}) = |A_n^2| - \text{rank}(P_{EDP_n^2|x}) \leq \frac{3}{2}n(n+1) + \frac{(n-2)(n-1)}{2} - \text{rank}(\mathbf{A}_S) = \frac{3}{2}n(n+1) + \frac{(n-2)(n-1)}{2} - (n^2 + 2n) = n^2 - 2n + 1$, where $\mathbf{A}_S$ is defined as in Lemma 5. We show that there are $n^2 - 2n + 2$ affinely independent **x**-routes in $P_{EDP_n^2|x}$.

Let Z be defined as in Theorem 6 (with the exception that H_n^∞ is replaced by H_n^2) and let H_1 and H_2 be the following sets of vectors:
$H_1 = \{\mathbf{h}_1^{ij}\}_{i,j \in P: 1 \leq i < j}$, $H_2 = \{\mathbf{h}_2^{n+i,j}\}_{i,j \in P: 2 \leq i < j}$, where

$$\mathbf{h}_1^{ij} = \left\{\mathbf{x} \in H_n^2 |\ x_{ij} = 1,\ x_{k,n+k} = 1\ \ \forall k \in P \setminus \{i,j\}\right\} \text{ and}$$
$$\mathbf{h}_2^{n+i,j} = \left\{\mathbf{x} \in H_n^2 |\ x_{1i} = x_{n+i,j} = 1,\ x_{k,n+k} = 1\ \ \forall k \in P \setminus \{1,i,j\}\right\}$$

Let $H := H_1 \cup H_2 \cup Z$. Clearly, $H \subset P_{EDP_n^2|x}$. The only **x**-route in H, where $x_{n+i,j} = 1,\ i,j \in P, 1 \leq i < j$, is $\mathbf{h}_2^{n+i,j}$ and the only solution in $H \setminus H_2$ where $x_{ij} = 1,\ i,j \in P, 2 \leq i < j$, is $\mathbf{h}_1^{ij}$. Hence, by Proposition 1, all vectors of H are linearly independent, and $|H| = |H_1| + |H_2| + |Z| = \sum_{i=1}^{n-1}(n-i) + \sum_{i=2}^{n-1}(n-i) + 1 = n^2 - 2n + 2$ as desired. □

6.4 Case 3: Restricted Number of Elevators

6.4.1 Assumptions

In addition to assumptions A1-A4, we assume that

A5'. The number of elevators is $|T| = l$, $2 \leq l \leq n$.

A6. Elevators have unlimited capacity.

The EDP which satisfies conditions A1-A4, A5', and A6 is denoted by $EDP^{\infty}_{n,l}$. The reduced graph of the problem is denoted by $G^{\infty}_{n,l}$.

Since the elevators have unlimited capacity, load variables can be eliminated by using simplified boarding constraints (6.7). Assumption A5' is taken into account by equation (6.1).

6.4.2 Polyhedral Analysis

The polytope of $EDP^{\infty}_{n,l}$ is

$$P_{EDP^{\infty}_{n,l}} := \operatorname{conv}\{(\mathbf{x},\mathbf{t}) \in \{0,1\}^{|A^{\infty}_{n,l}|} \times \mathbb{R}^{2n+2}_{+} | \ (\mathbf{x},\mathbf{t}) \text{ satisfies } (5.1),(5.3),(6.1)-(6.6), \text{ and } (6.7)\} \tag{6.16}$$

Polytope $P_{EDP^{\infty}_{n,l}|x}$ is defined similarly. Unlike in Case 1 or 2, there are now fewer solutions in the $\mathbf{x}$-space:

Theorem 11. The number of feasible solutions to $EDP^{\infty}_{n,l}$ in the $\mathbf{x}$-space is

$$|H^{\infty}_{n,l}| = \sum_{k=1}^{l} S(n,k) = \sum_{k=1}^{l} \frac{1}{k!} \sum_{j=1}^{k} (-1)^{k-j} \binom{n}{k} j^n, \tag{6.17}$$

where $S(n,k)$ refers to the Stirling number of the second kind.

Proof. We are looking for the number of ways to partition n passengers into k elevators, where $k = 1,\dots,l$. By the definition of the Stirling numbers of the second kind, the number of such partitions is $\sum_{k=1}^{l} S(n,k)$. □

Lemma 6. The number of arcs in $G^{\infty}_{n,l}$ is $|A^{\infty}_{n,l}| = \Big(\frac{3}{2}n(n+1)\Big)\chi_{\{l\geq 2\}} + \Big(2n+1\Big)\chi_{\{l=1\}}$

Proof. If $l \geq 2$, then $A^{\infty}_{n,l} = A^{\infty}_{n}$ and the claim holds. If $l = 1$, all requests are served by one elevator and since the capacity of the elevator is unrestricted, all passengers must be boarded at one time. Hence, the one and only solution to the problem in the $\mathbf{x}-$space is $\mathbf{x} = \chi_{A(R)}$, where $R = (+0,1,\dots,n,n+1,\dots,2n,0)$. All arcs which are not in $A(R)$ are redundant and can be removed. As $|A(R)| = 2n+1$, the claim follows. □

In order to get a hint, how the dimension of $P_{EDP^{\infty}_{n,l}}$ depends on l or n, we determine the dimensions of $P_{EDP^{\infty}_{n,l}|x}$ numerically in cases $n,l \in \{1,2,3,4,5,6\}$, $l \leq n$. The results are presented in Table 6.1. A survey of the values shows that the number of elevators seems to have diminishing effect on dimension when n increases. Moreover, in some cases the number of solutions clearly restricts the magnitude of the dimension. We set the following proposition:

Table 6.1: The dimension of $P_{EDP^{\infty}_{n,l}|x}$ as a function of l and n. The number in parentheses represents the number of feasible $\mathbf{x}$-routes, $|H^{\infty}_{n,l}|$.

$n \setminus l$	1	2	3	4	5	6
1	0(1)	#	#	#	#	#
2	0(1)	1(2)	#	#	#	#
3	0(1)	3(4)	4(5)	#	#	#
4	0(1)	7(8)	9(14)	9(15)	#	#
5	0(1)	15(16)	16(41)	16(51)	16(52)	#
6	0(1)	25(32)	25(122)	25(187)	25(202)	25(203)

Proposition 14. The dimension of $EDP^{\infty}_{n,l}$, $1 \leq l \leq n$, is bounded from below by

$$\dim(P_{EDP^{\infty}_{n,l}}) \geq \min\left\{\frac{n(n+1)}{2}, |H^{\infty}_{n,l}| - 1\right\} + 2n + 2 \tag{6.18}$$

and from above by

$$\dim(P_{EDP^{\infty}_{n,l}}) \leq \min\left\{(n-1)^2, |H^{\infty}_{n,l}| - 1\right\} + 2n + 2 \tag{6.19}$$

In addition, if $n \geq 6$, then

$$\left(\frac{n(n+1)}{2}\right)\chi_{\{l\geq 2\}} + 2n + 2 \leq \dim(P_{EDP^{\infty}_{n,l}}) \leq (n-1)^2\chi_{\{l\geq 2\}} + 2n + 2 \tag{6.20}$$

Proof. We first determine lower and upper bounds for the dimension of $P_{EDP^{\infty}_{n}|x}$ and then extend the results to the $(\mathbf{x}, \mathbf{t})$-space. Clearly, all feasible $\mathbf{x}$-routes in polytope $P_{EDP^{\infty}_{n,l}|x}$ satisfy the constraint matrix $\mathbf{A}_S$ given in Lemma 2. Hence, $\dim(P_{EDP^{\infty}_{n,l}|x}) \leq |A^{\infty}_{n}| - \text{rank}(\mathbf{A}_S) \leq \frac{3}{2}n(n+1) - (\frac{1}{2}n^2 + \frac{7}{2}n - 1) = n^2 - 2n + 1 = (n-1)^2$. On the other hand, every vector in $P_{EDP^{\infty}_{n,l}|x}$ is a linear combination, or more precisely a convex combination, of the feasible $\mathbf{x}$-routes, which means that the dimension of $P_{EDP^{\infty}_{n,l}|x}$ is at most $|H^{\infty}_{n,l}| - 1$. By combining these two facts, one obtains $\dim(P_{EDP^{\infty}_{n,l}|x}) \leq \min\left\{(n-1)^2, |H^{\infty}_{n,l}| - 1\right\}$.

Suppose $l \geq 2$. Moreover, we can assume that $n \geq 7$ since the values in Table 6.1 show that the claim holds for cases $n \leq 6$. The idea of the proof is to form a set of feasible $\mathbf{x}$-routes containing arcs that are used only in one of these $\mathbf{x}$-routes. Define $H^{k\setminus n-k}$ as the set of feasible $\mathbf{x}$-routes in which requests are taken care of by two elevators, such that one elevator serves k requests and the other $n-k$ requests. Let $H_1 = \{\mathbf{h}_1^i\}_{i\in P}, H_2 = \{\mathbf{h}_2^{ji}\}_{j,i\in P: i-j\geq 4}, H_3 = \{\mathbf{h}_3^{i,n+j}\}_{i,j\in P:1\leq i-j\leq n-4}, Z = \{\mathbf{z}\}$, where

$$\begin{aligned}\mathbf{h}_1^i &= \{\mathbf{x}\in H_{n,l}^{\infty}\cap H^{1\backslash n-1} |\ x_{i,n+i}=1\},\\ \mathbf{h}_2^{ji} &= \{\mathbf{x}\in H_{n,l}^{\infty}\cap H^{2\backslash n-2} |\ x_{ji}=1\},\\ \mathbf{h}_3^{i,n+j} &= \{\mathbf{x}\in H_{n,l}^{\infty}\cap H^{2\backslash n-2} |\ x_{i,n+j}=1\},\\ \mathbf{z} &= \{\mathbf{x}\in H_{n,l}^{\infty}\cap H^{0\backslash n}\}\end{aligned}$$

When $H_1\cup H_2\cup H_3\cup Z$ is denoted by H, we can make the following observations:

1. For each $i\in P$, the only x-route in H in which $x_{i,n+i}=1$ is $\mathbf{h}_1^i$.
2. For each $i-j\geq 4$, $i,j\in P$, the only x-route in $H\setminus H_1$ in which $x_{jj}=1$ is $\mathbf{h}_2^{ji}$.
3. For each $1\leq i-j\leq n-4$, $i,j\in P$, the only x-route in $H\setminus(H_1\cup H_2)$ in which $x_{i,n+j}=1$ is $\mathbf{h}_3^{i,n+j}$.
4. The only $\mathbf{x}$-route in $H\setminus(H_1\cup H_2\cup H_3)=Z$ is $\mathbf{z}$.

Now, by Proposition 1, the vectors of H are linearly independent. The cardinality of H can be obtained by using the principle of inclusion and exclusion, i.e., Theorem 4:

$$\begin{aligned}&|H| = |Z| + |H_1\cup H_2\cup H_3| = 1 + |H_1\cup H_2\cup H_3|\\ &= 1 + |H_1| + H_2| + |H_3| - |H_1\cap H_2| - |H_1\cap H_3| - |H_2\cap H_3| + |H_1\cap H_2\cap H_3|,\end{aligned}$$

where

$$\begin{aligned}|H_1| &= \sum_{j=1}^{n} 1 = n,\\ |H_2| &= \sum_{i,j\in P: i-j\geq 4} = \sum_{i=5}^{n}\sum_{j=1}^{i-4} 1\\ &= \sum_{i=5}^{n}(i-4) = \frac{(n-5+1)(5+n)}{2} - 4(n-5+1) = \frac{1}{2}(n^2-7n+12),\\ |H_3| &= \sum_{i,j\in P:1\leq i-j\leq n-4} = \sum_{i=2}^{n-3}\sum_{j=1}^{i-1} 1 + \sum_{i=n-2}^{n}\sum_{j=i-(n-4)}^{i-1} 1\\ &= \sum_{i=2}^{n-3}(i-1) + \sum_{i=n-2}^{n}(n-4) = \frac{1}{2}(n^2-n-12),\end{aligned}$$

$$
\begin{aligned}
&|H_1 \cap H_2| \le |\{\mathbf{x} \in H^{1\backslash n-1} \cap H^{2\backslash n-2}\}| = 0 \Rightarrow |H_1 \cap H_2| = 0,\\
&|H_1 \cap H_3| \le |\{\mathbf{x} \in H^{1\backslash n-1} \cap H^{2\backslash n-2}\}| = 0 \Rightarrow |H_1 \cap H_3| = 0,\\
&|H_2 \cap H_3| = |\{\mathbf{x} \in H^{2\backslash n-2} : 4 \le i-j \le n-4, \quad i,j \in P\}|\\
&= \Big(\sum_{i=5}^{n-3}\sum_{j=1}^{i-4} 1 + \sum_{i=n-2}^{n}\ \sum_{j=i-(n-4)}^{i-4} 1\Big)\chi_{\{n\ge 8\}} = \Big(\sum_{i=5}^{n-3}(i-4) + \sum_{i=n-2}^{n}(n-7)\Big)\chi_{\{n\ge 8\}}\\
&= \Big(\frac{(n-7)(n-6)}{2} + 3(n-7)\Big)\chi_{\{n\ge 8\}} = \frac{1}{2}(n^2-7n),\\
&|H_1 \cap H_2 \cap H_3| \le |H_1 \cap H_2| = 0 \Rightarrow |H_1 \cap H_2 \cap H_3| = 0.
\end{aligned}
$$

Hence,

$$
\begin{aligned}
|H| &= 1 + n + \frac{1}{2}(n^2 - 7n + 12) + \frac{1}{2}(n^2 - n - 12) - \frac{1}{2}(n^2 - 7n),\\
&= \frac{1}{2}n(n+1) + 1.
\end{aligned}
$$

in which case the dimension of $P_{EDP^{\infty}_{n,l}|x}$ is at least $|H| - 1 = \frac{1}{2}n(n+1)$. Recall that the lower bound is valid only if $l \ge 2$ and $n \ge 7$: when n and l are small enough, the number of feasible **x**-routes is the restrictive factor. By combining these facts together with the upper bound of the dimension, we obtain a chain of inequalities

$$
\min\Big\{\frac{n(n+1)}{2}, |H^{\infty}_{n,l}| - 1\Big\} \le \dim(P_{EDP^{\infty}_{n,l}|x}) \le \min\Big\{(n-1)^2, |H^{\infty}_{n,l}| - 1\Big\} \tag{6.21}
$$

Whenever $l \ge 2$, it holds that $|H^{\infty}_{n,l}| - 1 \ge |H^{\infty}_{n,2}| - 1 = S(n,1) + S(n,2) - 1 = 1 + 2^{n-1} - 1 - 1 = 2^{n-1} - 1$. In addition, if $n \ge 6$, it is easy to prove that $2^{n-1} - 1 \ge (n-1)^2 \ge \frac{n(n+1)}{2}$. Now, it follows that $\min\Big\{\frac{n(n+1)}{2}, |H^{\infty}_{n,l}| - 1\Big\} = \frac{n(n+1)}{2}$ and $\min\Big\{(n-1)^2, |H^{\infty}_{n,l}| - 1\Big\} = (n-1)^2$. On the other hand, if $l = 1$, then $|H^{\infty}_{n,1}| = 1$ regardless the value of n, in which case $\min\Big\{\frac{n(n+1)}{2}, |H^{\infty}_{n,1}| - 1\Big\} = 0$. Hence, when $n \ge 6$, it holds that

$$
\Big(\frac{n(n+1)}{2}\Big)\chi_{\{l\ge 2\}} \le \dim(P_{EDP^{\infty}_{n,l}|x}) \le (n-1)^2\chi_{\{l\ge 2\}}. \tag{6.22}
$$

The inequalities to be proven, (6.18), (6.19), and (6.20), follow from inequalities (6.21) and (6.22) by applying Proposition 12. □

Proposition 14 gives only bounds within the dimension must lie, so one might be interested in the goodness of these bounds. In practice, the lower bound seems to be rather poor because a more careful inspection of Table 6.1 reveals that the obtained upper bound is, in fact, achieved in all cases when $n, l \in \{1, \ldots, 6\}$, $l \leq n$. This result also makes sense intuitively: the number of feasible **x**-routes, $|H^{\infty}_{n,l}|$, increases very rapidly with respect to n, so we can consider it probable that for large enough n, $H^{\infty}_{n,l}$ would contain $(n-1)^2 + 1$ linearly independent **x**-routes, regardless the value of l, $l \neq 1$. Since no real knowledge about cases $n, l \geq 7$ is available, we end this chapter with a following conjecture:

Conjecture 1. The dimension of $P_{EDP^{\infty}_{n,l}}$, $1 \leq l \leq n$, is

$$\dim(P_{EDP^{\infty}_{n,l}}) = \min\left\{(n-1)^2, |H^{\infty}_{n,l}| - 1\right\} + 2n + 2 \qquad (6.23)$$

Chapter 7

EDP: Down-peak Traffic Pattern

7.1 Assumptions

Down-peak traffic is a situation in which all or most of the requests are travelling downwards. A typical example of down-peak traffic is the peak at the end of the day, when people are travelling to the lobby in order to exit the building. Unlike in the case of up-peak traffic, we consider only one down-peak traffic pattern that was studied by Ruokokoski et al.[28]. The analysis is based on the following five assumptions:

B1. There are n non-assigned requests such that
$f(1) > \ldots > f(n),\ f(n+1) = \ldots = f(2n) = 1$

B2. Each vertex has a relaxed time window:
$[a_i, b_i] = [0, \infty)\ \forall i \in V.$

B3. Elevators are symmetrical and they initially locate at the first floor:
$f(2n+e) = 1\ \forall (2n+e) \in T.$

B4. The number of elevators is equal to the number of requests: $|T| = n$

B5. Elevators have unlimited capacity: $Q = \infty$

A reader should note that assumptions B2, B3, B4, and B5 are the same as assumptions A3, A4, A5, and A6, respectively - we just prefer to use different notation for up-peak and down-peak traffic patterns. We denote the down-peak traffic pattern that satisfies assumptions B1-B5 by $D - EDP_n^\infty$ and the corresponding reduced graph by $G_{D,n}^\infty$. The reduced graph is presented in Figure 7.1.

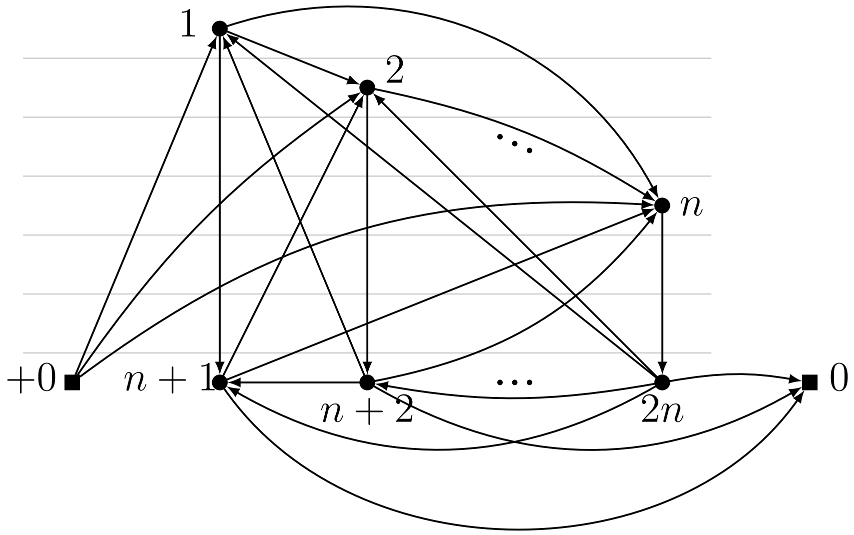


Figure 7.1: The reduced graph $G^{\infty}_{D,n}$ of the studied down-peak traffic pattern.

7.2 Polyhedral Analysis

The polytope of $D - EDP^{\infty}_{n}$ is the convex hull of feasible solutions of the problem under assumptions B1-B5, i.e.,

$$P_{D-EDP^{\infty}_{n}} := \text{conv}\Big\{(\mathbf{x},\mathbf{t}) \in \{0,1\}^{|A^{\infty}_{D,n}|} \times \mathbb{R}^{2n+2}_{+} | \ (\mathbf{x},\mathbf{t}) \text{ satisfies } (5.1), (5.3),(5.13),(5.16), \text{ and } (6.2)-(6.3)\Big\} \tag{7.1}$$

where $A^{\infty}_{D,n}$ is the set of arcs in the reduced graph $G^{\infty}_{D,n}$. Next we determine the number of feasible solutions in the $\mathbf{x}$-space, i.e., the number of feasible $\mathbf{x}$-routes.

Theorem 12. The number of feasible solutions to $D - EDP^{\infty}_{n}$ in the $\mathbf{x}$-space is

$$|H^{\infty}_{D,n}| = \sum_{m=1}^{n} \sum_{k=l}^{n} S(n,k)L(k,l), \tag{7.2}$$

where $S(n,k)$ is the Stirling number of the second kind (3.16), and $L(k,l)$ the Lah number (3.18).

Proof. Suppose that requests are served by $l \leq n$ elevators, which leave from the lobby $k \geq l$ times in total. During each service trip, an elevator serves a cluster of requests by picking up a set of passengers and then by returning to the lobby. The service order in each cluster is unique since an elevator can change the direction of travel only if it is empty. Clusters that are assigned to an elevator can be served in any order. Hence, the number of ways to form k clusters equals $S(n,k)$, and the number of ways to assign k clusters to l elevators equals $L(k,l)$. Therefore, according to the multiplicative principle, n requests can be divided into k clusters that are served by m elevators in $S(n,k)L(k,l)$ different ways. The claim follows by summing $S(n,k)L(k,l)$ over $l = 1,\dots,n$ and $k = l,\dots,n$. □

If we compare the number given by Theorem 12 with the number of feasible **x**-routes in the corresponding up-peak traffic problem EDP_n^∞, i.e., B_n, it is obvious that the former number is significantly larger. The reason why these symmetrical traffic patterns are so different, with respect to the number of **x**-routes, are the boarding constraints (5.11): in the up-peak traffic pattern passengers are picked up from the same floor, and since elevators have unlimited capacity, all passengers assigned to the same elevator, must be picked up before it can leave the current floor. In the down-peak traffic pattern, each request comes from a different floor, so examination of boarding constraints is not necessary. In fact, in the case of $D-EDP_n^\infty$, we could use a simplified form for the extended boarding constraints (5.16), so that they incorporate only reversal requirements. Formally,

$$\sum_{(i,j)\in A(X)} x_{ij} \le |A(X)| - 1 \;\; \forall X \in \mathcal{X}^*, \tag{7.3}$$

where $\mathcal{X}^*$ is the union of all forward paths of the form
$X = (2n+e, k_1, \dots, k_{r-1}, k_r)$ where $(k_{r-1}, k_r) \in R$, and $\Omega(X \setminus \{k_r\}) > 0$. By relaxing constraints (6.7) in the formulation of EDP_n^∞, we obtain $|H_n^\infty| = |H_{D,n}^\infty|$.

The number of arcs in the reduced graph $G_{D,n}^\infty$ and the dimension of $P_{D-EDP_n^\infty}$ are given in the next lemma and theorem, respectively.

Lemma 7. The number of arcs in $G_{D,n}^\infty$ equals

$$|A_{D,n}^\infty| = 2n^2 + n. \tag{7.4}$$

Proof. Since $|(\{j\}, \overline{\{j\}})| = n - j + 1$ for $j = 1,\dots,n$, $|(\{n+j\}, \overline{\{n+j\}})| = n + j - 1$ for $j = 1,\dots,n$, and $|(\{+0\}, \overline{\{+0\}})| = n$, hence $|A_{D,n}^\infty| = n + 1/2n(n+1) + n^2 + 1/2n(n-1) = 2n^2 + n$. □

Theorem 13. The dimension of $P_{D-EDP_n^\infty}$ is

$$\dim(P_{D-EDP_n^\infty}) = \frac{1}{2}(3n^2 + n + 4). \tag{7.5}$$

Proof. The proof is similar to the proof of Theorem 6. See [28] for details. □

Chapter 8

Conclusions

Based on the study of Ruokokoski et al. [28], we gave a snapshot mixed-integer linear formulation for the elevator dispatching problem (EDP) in a general case. Although, our formulation incorporates three different kinds of variables, which are routing variables $\mathbf{x}$, time variables $\mathbf{t}$, and load variables $\mathbf{q}$, we showed that polyhedral analysis can be carried out in the $(\mathbf{x}, \mathbf{t})$-space instead of the $(\mathbf{x}, \mathbf{t}, \mathbf{q})$-space. This observation relies on the fact that load variables are "artificial" by nature, thereby enabling their elimination. Furthermore, if neither waiting times nor journey times of passengers are restricted, i.e., time window constraints are relaxed, polyhedral analysis can be carried out just in the $\mathbf{x}$-space, which after the results can be extended to $(\mathbf{x},\mathbf{t})$-space by using a simple formula.

Most of our study focused on the analysis of up-peak traffic patterns. We analyzed three different up-peak traffic patterns: in Case 1 it was assumed that there are as many elevators as transport requests, and that the capacity of elevators is unlimited. In Case 2 there were as many elevators as transport requests but elevators had restricted capacity, and in Case 3 elevators had unlimited capacity, but the number of elevators was less than the number of requests. Depending on the case the general formulation of the EDP was simplified as much as possible. In each case, we determined the number of feasible solutions in the $\mathbf{x}$-space and counted the number of arcs in the reduced graph. In addition, the dimension of each EDP polytope, defined as the convex hull of the feasible solutions, was studied. An exact formula for the value of dimension was given in Case 1 and Case 2, whereas in Case 3 we found a plausible formula, but could not show its validity. Instead, in Case 3 we determined non-trivial lower and upper bounds for the dimension. Summaries of the obtained results are given in Table 8.1 and Table 8.2, where, for comparison reasons, we also present similar results for TSP_{2n} and $TSPPD_n$. If we compare the formulas of Table 8.1 with each other, we can note that whenever $n \geq 3$,

$$\begin{aligned} |H(TSP_{2n})| &> |H(TSPPD_n)| > |H(D-EDP_n^{\infty})| > |H(EDP_n^{\infty})| \\ &= |H(EDP_n^m)| \geq |H(EDP_{n,l}^{\infty}|l \geq 2)| > |H(EDP_{n,l}^{\infty}|l = 1)| \end{aligned}$$

where $|H(\cdots)|$ refers to the number of solutions in the $\mathbf{x}$-space. Validity of the inequality $|H(TSPPD_n)| > |H(D-EDP_n^{\infty})|$ is a non-trivial thing, and its proof is skipped here. A similar sequence of inequalities holds for the number of arcs too, but in a slightly different order. Assuming $n \geq 3$, we have

$$\begin{aligned} |A(TSPPD_n)| &> |A(TSP_{2n})| > |A(D-EDP_n^{\infty})| > |A(EDP_n^m|m \geq 2)| \\ &\geq |A(EDP_n^{\infty})| = |A(EDP_{n,l}^{\infty}|l \geq 2)| > |A(EDP_n^m|m = 1)| \\ &> |A(EDP_{n,l}^{\infty}|l = 1)|. \end{aligned}$$

Recall that the graph of $TSPPD_n$ contains $2n+2$ vertices instead of $2n$ vertices, which is the reason for the direction of the first inequality. Another interesting detail in this sequence is that the number of arcs in the up-peak traffic patterns attains its maximum, when the capacity of elevators is two. If we assume that $n \geq 5$ and the validity of Conjecture 1, then the order of inequalities for dimensions is exactly the same as in the case of the number of the arcs. The requirement that n must be at least five, follows from the fact the dimension of EDP polytopes is determined in the $(\mathbf{x}, \mathbf{t})$-space and not in the $\mathbf{x}$-space, unlike the dimensions of the TSP and the TSPPD. Should the same analysis be carried out just in the $\mathbf{x}$-space, then requirement $n \geq 3$ is enough to guarantee the order of the inequalities.

The results of this paper provide new, essential information concerning polyhedral structure of up-peak traffic patterns in the EDP. Although most of the results relate to theoretical situations, which are very rare in real life, they might help us to better understand the structure of the EDP polytope of a general case. For example, if we discover that certain inequalities are facet-defining for up-peak traffic patterns, it is presumable their addition to the general EDP model strengthens the formulation. Hence, we believe that by using the results of this thesis, we are more capable of discovering ways to improve the EDP formulation, which will help in designing EDP solving algorithms.

Table 8.1: The number of solutions and arcs in the TSP and EDP variants represented in terms of the number of transporation requests n, the capacity of elevators m, and the number of elevators l. In the table $|H|$ is the number of solutions to the problem in the $\mathbf{x}$-space, $|A|$ the number of arcs in the reduced graph, $S(n,k)$ the Stirling number of the second kind, $L(k,l)$ the Lah number, and B_n the Bell number.

MILP		$\lvert H(MILP)\rvert$	$\lvert A(MILP)\rvert$
TSP_{2n}		$(2n)!/2$	$2n(2n-1)/2$
$TSPPD_n$		$(2n)!/2^n$	$2n^2+n+1$
$D-EDP_n^\infty$		$\sum_{l=1}^{n}\sum_{k=l}^{n} S(n,k)L(k,l)$	$2n^2+n$
EDP_n^∞		B_n	$3n(n+1)/2$
EDP_n^m	$m=1$	B_n	$n^2/2+5n/2$
	$m=2$	B_n	$2n^2+1$
	$m\geq 3$	B_n	$2n^2+2n+m(m-2n-1)/2$
$EDP_{n,l}^\infty$	$l=1$	1	$2n+1$
	$l\geq 2$	$\sum_{k=1}^{l} S(n,k)$	$3n(n+1)/2$

Table 8.2: The dimensions of two TSP polytopes represented in terms of requests n, and the dimensions of different EDP polytopes represented in terms of the number of transportation requests n, the capacity of elevators m, and the number of elevators l. The dimensions of TSP and EDP polytopes are determined in the $\mathbf{x}$-space and the $(\mathbf{x},\mathbf{t})$-space, respectively.

MILP		$\dim(P_{MILP})$
TSP_{2n}		$2n(2n-3)/2$
$TSPPD_n$		$2n^2-n-2$
$D-EDP_n^\infty$		$(3n^2+n+4)/2$
EDP_n^∞		n^2+3
EDP_n^m	$m=1$	$n^2/2+3n/2+2$
	$m=2$	n^2+3
	$m\geq 3$	$n^2+3+(n-m)(n-m+1)/2$
$EDP_{n,l}^\infty$	$l=1$	$2n+2$
	$l\geq 2$	See (6.18) and (6.19)

Bibliography

[1] Alexandris, N. *Statistical models in lift systems.* PhD thesis, University of Manchester Institute of Science and Technology, 1977.

[2] Balas, E., Fischetti, M., and Pulleyblank, W. R. The precedence-constrained asymmetric traveling salesman polytope. *Mathematical Programming 68*, 1 (1995), 241–265.

[3] Bertsimas, D., and Tsitsiklis, J. *Introduction to Linear Optimization*, 1st ed. Athena Scientific, 1997.

[4] Cameron, P. *Combinatorics: Topics, Techniques, Algorithms.* Cambridge University Press, 1994.

[5] Chvatal, V. *Linear Programming.* Series of books in the mathematical sciences. W. H. Freeman, 1983.

[6] Cortés, C. E., Matamala, M., and Contardo, C. The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method. *European Journal of Operational Research 200*, 3 (2010), 711 – 724.

[7] Dahl, G. An introduction to convexity, polyhedral theory and combinatorial optimization. *University of Oslo, Department of Informatics* (1997), 44–45.

[8] Dantzig, G. B., Fulkerson, D. R., and Johnson, S. M. Solution of a large-scale traveling-salesman problem. *Operations Research 3* (1954), 393–410.

[9] Dantzig, G. B., and Ramser, J. H. The truck dispatching problem. *Management Science 6* (1959), 80–91.

[10] Dumitrescu, I., Ropke, S., Cordeau, J.-F., and Laporte, G. The traveling salesman problem with pickup and delivery: polyhedral

results and a branch-and-cut algorithm. *Mathematical Programming 121*, 2 (2008), 269–305.

[11] Fiorini, S., Massar, S., Pokutta, S., Tiwary, H. R., and Wolf, R. D. Exponential lower bounds for polytopes in combinatorial optimization. *J. ACM 62*, 2 (May 2015), 17:1–17:23.

[12] Gomory, R. E. Outline of an algorithm for integer solutions to linear programs. *Bulletin of the American Society 64* (1958), 275–278.

[13] Grötschel, M., and Padberg, M. W. On the symmetric travelling salesman problem: Theory and computation. In *Optimization and operations research, Proc. Workshop, Bonn 1977* (1978), R. Henn, B. Korte, and W. Oettli, Eds., vol. 157 of *Lecture Notes in Economics and Mathematical Systems*, Springer, pp. 105–115.

[14] Gutin, G., and Punnen, A. P., Eds. *The traveling salesman problem and its variations*. Combinatorial optimization. Kluwer Academic, Dordrecht, London, 2002.

[15] Hiller, B., and Tuchscherer, A. Real-time destination-call elevator group control on embedded microcontrollers. In *Operations Research Proceedings 2007* (2008), pp. 357 – 362.

[16] Hirasawa, K., Eguchi, T., Zhou, J., Yu, L., Hu, J., and Markon, S. A double-deck elevator group supervisory control system using genetic network programming. *IEEE Transactions on Systems, Man, and Cybernetics, Part C 38*, 4 (2008), 535–550.

[17] Ikeda, K., Suzuki, H., Markon, S., and Kita, H. Evolutionary optimization of a controller for multi-car elevators. In *IEEE Congress on Evolutionary Computation* (2006), Industrial Technology, 2006. ICIT 2006. IEEE International Conference on, pp. 2474–2479.

[18] Koehler, J., and Ottiger, D. An AI-based approach to destination control in elevators. *AI Magazine* (Sept. 2002).

[19] Kumar, S., and Panneerselvam, R. A survey on the vehicle routing problem and its variants. *Intelligent Information Management 4*, 3 (2012), 66–74.

[20] Laporte, G., and Nobert, Y. Exact algorithms for the vehicle routing problem. *Annals of Discrete Mathematics 31* (1987), 147–184.

[21] Laporte, G., and Osman, I. H. Routing problems: A bibliography. *Annals of Operations Research 61*, 1 (1995), 227–262.

[22] Luh, P. B., Xiong, B., and chung Chang, S. Group elevator scheduling with advance information for normal and emergency modes, 2008.

[23] Markon, S., Kise, H., Kita, H., and Bartz-Beielstein, T. Control of traffic systems in buildings. Springer London, 2006, ch. Elevator group control by neural networks and stochastic approximation, pp. 163–186.

[24] Mosheiov, G. The Travelling Salesman Problem with pick-up and delivery. *European Journal of Operational Research 79*, 2 (December 1994), 299–310.

[25] Naddef, D., and Rinaldi, G. The vehicle routing problem. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001, ch. Branch-and-cut Algorithms for the Capacitated VRP, pp. 53–84.

[26] Pisinger, D., and Ropke, S. A general heuristic for vehicle routing problems. *Computers & Operations Research 34* (2007), 2403–2435.

[27] Roshier, N., and Kaakinen, M. New formulae for elevator round trip calculation. *Supplement to Elevator World fo ACIST Members* (1978), 189–197.

[28] Ruokokoski, M., Ehtamo, H., and Pardalos, P. M. Elevator dispatching problem: A mixed integer linear programming formulation and polyhedral results. *J. Comb. Optim. 29*, 4 (May 2015), 750–780.

[29] Ruokokoski, M., Sorsa, J., Siikonen, M.-L., and Ehtamo, H. Assignment formulation for the elevator dispatching problem with destination control and its performance analysis. *European Journal of Operational Research 252*, 2 (2016), 397 – 406.

[30] Savelsberg, M. W. P., and Sol, M. The general pickup and delivery problem. *Transportation Science 29* (1995), 17–29.

[31] Siikonen, M.-L. Elevator group control with artificial intelligence. Tech. rep., Helsinki University of Technology, 1997.

[32] Sorsa, J., Siikonen, M.-L., and Ehtamo, H. Optimal control of double-deck elevator group using genetic algorithm. *International Transactions in Operational Research 10* (2003), 103–114.

[33] Tanaka, S., Uraguchi, Y., and Araki, M. Dynamic optimization of the operation of single-car elevator systems with destination hall call registration: Part i. formulation and simulations. *European Journal of Operational Research 167*, 2 (2005), 550–573.

[34] Yannakakis, M. Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences 43*, 3 (1991), 441 – 466.

Appendix A

Proof of $LP^{cut}_{TSPPD} = LP^{sub}_{TSPPD}$

Let $TSPPD^{sub}$ and $TSPPD^{cut}$ denote the two different formulations of the travelling salesman problem with pickup and delivery such that

$$TSPPD^{sub} = \big\{\mathbf{x} \in \{0,1\}^{|E|} : \mathbf{x} \text{ satisfies } (4.41) - (4.44)\big\} \text{ and}$$
$$TSPPD^{cut} = \big\{\mathbf{x} \in \{0,1\}^{|E|} : \mathbf{x} \text{ satisfies } (4.41), (4.43) - (4.46)\big\}.$$

Let LP^{cut}_{TSPPD} and LP^{sub}_{TSPPD} be the feasible sets of the corresponding linear programming relaxations.

Assume $\mathbf{x} \in LP^{sub}_{TSPPD}$. The set of edges can be written as a union of three disjoint sets:

$$E = \big\{\{i,j\} \in E : i,j \in S \text{ or } i \in \overline{S}, j \in S \text{ or } i,j \in \overline{S}\big\} = \delta(S) \cup \rho(S) \cup \rho(\overline{S}),$$

where S is a subset of V, $S \neq \emptyset, V$. Now, because $\sum_{e \in \rho(S)} x_e \leq |S| - 1$, $\sum_{e \in \rho(\overline{S})} x_e \leq |\overline{S}| - 1$, and $\sum_{e \in E} x_e = \frac{1}{2} \sum_{i \in V} \sum_{e \in \delta(i)} x_e = \frac{1}{2} \sum_{i \in V} 2 = |V|$, it follows that

$$\sum_{e \in \delta(S)} x_e = \sum_{e \in E \setminus \{\rho(S) \cup \rho(\overline{S})\}} x_e = \sum_{e \in E} x_e - \sum_{e \in \rho(S)} x_e - \sum_{e \in \rho(\overline{S})} x_e \geq |V| - |S| - |\overline{S}| + 2 = 2$$

for any $S \subset V$, $S \neq \emptyset, V$. Hence, $\mathbf{x} \in LP^{cut}_{TSPPD}$, and $LP^{sub}_{TSPPD} \subset LP^{cut}_{TSPPD}$. Assume now that $\mathbf{x} \in LP^{cut}_{TSPPD}$ and let S be a subset of V, $S \neq \emptyset, V$. We can write $\rho(S)$ as a function of cutsets such that

$$\rho(S) = \bigcup_{i \in S} \delta(i) \setminus \delta(S),$$

where $\delta(S) \subset \bigcup_{i \in S} \delta(i)$. According to the principle of inclusion and exclusion, the cardinality of the union of $\delta(i)$'s is

$$\Big|\bigcup_{i \in S} \delta(i)\Big| = \sum_{i \in S} |\delta(i)| - \sum_{i,j \in S : i \neq j} |\delta(i) \cap \delta(j)| = \sum_{i \in S} |\delta(i)| - |\rho(S)|$$

from which we obtain the desired inequality

$$\begin{aligned}\sum_{e\in\rho(S)} x_e &= \sum_{e\in\bigcup_{i\in S}\delta(i)} x_e - \sum_{e\in\delta(S)} x_e \\ &= \sum_{i\in S}\sum_{e\in\delta(i)} x_e - \sum_{e\in\rho(S)} x_e - \sum_{e\in\delta(S)} x_e \leq 2|S| - 2 - \sum_{e\in\rho(S)} x_e \\ &\Rightarrow \sum_{e\in\rho(S)} x_e \leq |S| - 1.\end{aligned}$$

Since $LP^{sub}_{TSPPD} \subset LP^{cut}_{TSPPD}$ and $LP^{cut}_{TSPPD} \subset LP^{sub}_{TSPPD}$, it holds that $LP^{cut}_{TSPPD} = LP^{sub}_{TSPPD}$.