DISSERTATIONES
INFORMATICAE
UNIVERSITATIS
TARTUENSIS
1

ABDULLAH MAKKEH

Applications of Optimization in Some Complex Systems

DISSERTATIONES INFORMATICAE UNIVERSITATIS TARTUENSIS

1

DISSERTATIONES INFORMATICAE UNIVERSITATIS TARTUENSIS

1

ABDULLAH MAKKEH

Applications of Optimization in Some Complex Systems

UNIVERSITY OF TARTU
Press

Institute of Computer Science, Faculty of Science and Technology, University of Tartu, Estonia.

Dissertation has been accepted for the commencement of the degree of Doctor of Philosophy (PhD) in informatics on June 4, 2018 by the Council of the Institute of Computer Science, University of Tartu.

Supervisor

Assoc. Prof. Dirk Oliver Theis
Institute of Computer Science
University of Tartu
Tartu, Estonia

Opponents

Assoc. Prof. Nicolas Gillis
Department of Mathematics and Operational Research
University of Mons
Mons, Belgium

Dr. Matthias Walter
Chair of Operations Research
RWTH Aachen University
Aachen, Germany

The public defense will take place on August 27, 2018 at 16:15 in Liivi 2-405.

The publication of this dissertation was financed by the Institute of Computer Science, University of Tartu.

European Union
European Regional
Development Fund

Investing
in your future

Eesti Teadusagentuur
Estonian Research Council

ISSN 2613-5906
ISBN 978-9949-77-780-8 (print)
ISBN 978-9949-77-781-5 (PDF)

University of Tartu Press
www.tyk.ee

To my family,

ABSTRACT

Mathematical Optimization is the process of finding an optimal solution among others, with respect to a certain criterion. In other words, it is finding the optimal solution of an optimization problem, i.e., the maximum or minimum value of a real function (objective function) subjected to a set of constraints. This process has been applied to several fields such as physics, biology, chemistry, social sciences, computer science, and their intersections. Mathematical optimization is divided into classes depending on the nature of the objective function as well as the constraints. These classes vary in their computational complexity, for example, Integer Programming is NP-hard whereas Linear Programming has a polynomial complexity.

In practice, robust and fast processes are needed to tackle the problems, classes with polynomial complexity are used directly. Several efficient algorithms exist which solve optimization problems in those classes. But, in some cases, even these efficient algorithms face difficulties in solving optimization problems such as taking immensely long time to find the optimal solution or halting even when the returned solution is still very far from the optimal solution. In this case, studying the optimization, in-depth, leads to figuring out the reasons behind these pitfalls and so either adjust the problem to avoid them or recommend the suitable algorithm to be used. The problems which belong to classes of high complexity can also be used in practice. They are not used directly but rather as an assist or control to the quality of solution retrieved by heuristics.

This thesis applies optimization of polynomial and high complexity to some complex systems. In one part, it studies the solution of a Convex Program which was used to obtain partial information decomposition, a tool used recently to analyze a particular complex system. Many difficulties arise in solving the Convex Program and so the study done in this thesis resulted in a fast and robust algorithm to tackle the problem. This part is concluded by studying the situation, appeared recently in neuroscience, when the partial information decomposition obtained is optimized subject to some constraints. In the other part, it studies a fundamental optimization problem in the control of automated systems that is APX-hard. The problem is modeled into IP and CNF which can be used, in future, to design good heuristics which tackle the problem in reasonable time.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ORIGINAL PUBLICATIONS

Publications included in the thesis

1. Makkeh, A., Theis, D.O. and Vicent, R. Bivariate Partial Information Decomposition: The Optimization Perspective. Entropy, 19(10):530, 2017.
2. Makkeh, A. and Theis, D.O. Comparison of IP and CNF Models for Control of Automated Valet Parking Systems. In Springer Proceedings in Mathematics and Statistics, Volume 217, pages 233–241, 2018.
3. Makkeh, A., Theis, D.O. and Vicente, R. BROJA_2PID: A roboust estimator for bivariate partial information decomposition. Entropy, 20(4):271,2018.
4. Makkeh, A. and Theis, D.O. Optimizing Bivariate Partial Information Decomposition (preprint, arXiv:1802.03947).

Publications not included in the thesis

1. Makkeh, A., Pourmoradnasseri, M. and Theis, D.O.: The Graph of the Pedigree Polytope is Asymptotically Almost Complete (Extended Abstract). In Proceedings of The International Conference on Algorithms and Discrete Applied Mathematics, CALDAM 2017.
2. Makkeh, A., Pourmoradnasseri, M. and Theis, D.O.: On the Graph of the Pedigree Polytope (preprint, arXiv:1611.08431).

1. INTRODUCTION

Optimization has applications in a broad spectrum of disciplines such as transportation networks, circuit design, and economics. The theory of optimization has been well developed since the 20th century. Many efficient algorithms have been designed even for large-scale optimization problems. Thus making it a powerful tool for tackling problems.

Due to the huge developments in optimization techniques, scientist started utilizing optimization in analyzing and studying complex systems. A system is said to be *complex* when its components interact in serpentine ways and reveal dependencies at all scales. For example, earth climate, the human brain, living cells, and biological networks are all complex systems in which analyzing them is vital but rather complicated.

In particular, using optimization in analyzing complex system can be done either by modeling aspects of the system as optimization problems or by using analytical techniques that rely on optimization. For example, automation systems are complex systems where optimization can be used to design efficient algorithms to tackle some problems involved in the control of these systems. Also studying the information that the components of a complex system exchange is a way to analyze these systems. Partial information decomposition is a tool which decomposes this information exchange and uses optimization to achieve such decomposition.

This thesis deals with using optimization for applications which analyzes and studies complex systems. The thesis can be seen as two parts. One part studies an optimization problem related to decomposing the information which the components of complex systems exchange. The other part tackles a fundamental optimization problem involved in the control of an automated car parking system.

1.1. Partial information Decomposition

This section aims to give an overview of partial information decomposition. This analytic tool is being used extensively in neuroscience. Before diving into the formal definition of partial information decomposition in the next subsection, the following is a motivating example describing partial information decomposition on a conceptional level.

Let $\mathbf{Y}$ and $\mathbf{Z}$ be the source signals about a target signal $\mathbf{X}$ in a complex system. E.g., two $\mathbf{Y}$ and $\mathbf{Z}$ are excitations by two different stimuli to a neuron $\mathbf{X}$. Studying the response of the neuron and the dependencies to characterize the relation between the response and the stimuli is called *neural coding*. In this case, a neurologist wants to investigate whether redundant information about the response of $\mathbf{X}$ is provided by both $\mathbf{Y}$ and $\mathbf{Z}$ or they behave synergistically. Also, it is useful to figure out whether any of the sources (stimuli) provides unique information about the response of $\mathbf{X}$, i.e., information that can be obtained only from that specific source. Thus, in such system, determining the quantities of shared, unique, and synergistic information that the components reveal can be used to analyze the interaction between them.

1.1.1. Definition of Partial Information Decomposition

The neural coding example raises three different measures of information shared, unique, and synergistic. Consider the random variables $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ sampled from the finite sets X, Y, Z which represents the signals $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$. Suppose that $\mathbb{P}$ is the joint probability distribution of $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$.

So, the amount of shared information that $\mathbf{Y}$ and $\mathbf{Z}$ hold about $\mathbf{X}$ is $SI(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$. The amount of unique information about $\mathbf{X}$ is split into two quantities, one contained only in $\mathbf{Y}$ that is $\mathrm{UI}(\mathbf{X}; \mathbf{Y} \backslash \mathbf{Z})$ and the other contained only in $\mathbf{Z}$ that is $\mathrm{UI}(\mathbf{X}; \mathbf{Z} \backslash \mathbf{Y})$. The amount of synergistic information about $\mathbf{X}$ that can be measured by knowing both $\mathbf{Y}$ and $\mathbf{Z}$ is $CI(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$. All the quantities $\mathrm{SI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$, $\mathrm{UI}(\mathbf{X}; \mathbf{Y} \backslash \mathbf{Z})$, $\mathrm{UI}(\mathbf{X}; \mathbf{Z} \backslash \mathbf{Y})$, and $\mathrm{CI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$ are nonnegative functions that take as argument the joint distribution $\mathbb{P}$ of $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$. When confusion may arise, $\mathrm{SI}_{\mathbb{P}}(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$ will refer to the shared information where $\mathbb{P}$ is the joint probability distribution of $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$. Similar referring will be used for the other PID quantities.

Recall that the total information about $\mathbf{X}$ that $(\mathbf{Y}, \mathbf{Z})$ possesses can be quantified by the mutual information $\mathrm{MI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$. In fact, shared, unique, and synergistic information decompose the amount of total information that $\mathbf{Y}$ and $\mathbf{Z}$ hold about $\mathbf{X}$, i.e., $\mathrm{MI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$. So, partial information decomposition (PID) aims to divide $\mathrm{MI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$ into more interpretable parts shared, unique, and synergistic information

$$\mathrm{MI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z}) = \mathrm{SI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z}) + \mathrm{UI}(\mathbf{X}; \mathbf{Y} \backslash \mathbf{Z}) + \mathrm{UI}(\mathbf{X}; \mathbf{Z} \backslash \mathbf{Y}) + \mathrm{CI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z}), \quad \text{(PID)}$$

rather than estimating $\mathrm{MI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$. Moreover, the quantity of total information that $\mathbf{Y}$ has about $\mathbf{X}$, i.e. $\mathrm{MI}(\mathbf{X}; \mathbf{Y})$, is decomposed into the quantity of unique information that $\mathbf{Y}$ has about $\mathbf{X}$ and $\mathbf{Y}$ shares with $\mathbf{Z}$ about $\mathbf{X}$. Similarly for $\mathrm{MI}(\mathbf{X}; \mathbf{Z})$, the quantity of total information that $\mathbf{Z}$ has about $\mathbf{X}$, thus leading to the following identities:

$$\begin{aligned} \mathrm{MI}(\mathbf{X}; \mathbf{Y}) &= \mathrm{SI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z}) + \mathrm{UI}(\mathbf{X}; \mathbf{Y} \backslash \mathbf{Z}), \\ \mathrm{MI}(\mathbf{X}; \mathbf{Z}) &= \mathrm{SI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z}) + \mathrm{UI}(\mathbf{X}; \mathbf{Z} \backslash \mathbf{Y}). \end{aligned} \tag{1.1}$$

To sum up, (PID) defines partial information decomposition. This decomposition must satisfy the two identities in (1.1). This decomposition is unique meaning that give a joint distribution $\mathbb{P}$ of $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$ there is one and only one PID of $\mathrm{MI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$. Furthermore, from the identities (1.1) and the definition (PID), it is clear that there is (at most) one degree of freedom in defining the PID, whenever

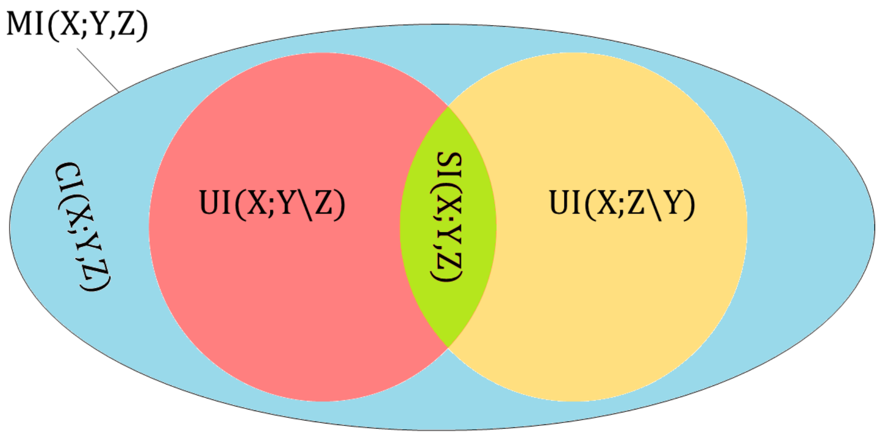


Figure 1. $MI(\mathbf{X};\mathbf{Y},\mathbf{Z})$ decomposed into the four information quantities.

the joint distribution of $(\mathbf{X},\mathbf{Y},\mathbf{Z})$ is known. In other words, defining the value of one of the information quantities defines the PID. From now on, PID measure refers to defining the value of one of the information quantities.

Unlike $\mathrm{MI}(\mathbf{X};\mathbf{Y},\mathbf{Z})$, the other quantities $\mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z}), \mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z}), \mathrm{UI}(\mathbf{X};\mathbf{Z}\backslash\mathbf{Y})$, and $\mathrm{CI}(\mathbf{X};\mathbf{Y},\mathbf{Z})$ are qualitative, i.e., when introduced there was no closed form to compute them – see Subsection 1.1.2 for details. This prompted scientists to define meaningful ways to compute them.

Applications. The need for PID has been addressed long before Williams and Beer breakthrough – see Subsection 1.1.2, for example, neural coding [13, 53] where the interest was mainly in studying whether two neurons (or neural populations) are correlated or not, i.e., exhibit synergy and whether ignoring this synergy affect the success of stimulus decoding. Another addressing was in biological networks [8, 9, 30] where they needed to compute shared and synergistic information empirically on neurophysiological datasets.

Recently, after being computable, scientists started utilizing PID in various complex systems and some related fields, for example, PID was used to reevaluate and define neural goals functions that help in understanding how information is processed in certain neural structure [72, 73]. Complex networks utilized PID like neural [68, 73] where PID was used in quantifying how information is modified within developing neural networks or biological [71] where PID helped in identifying algorithms run by these systems or dynamical [5, 64, 66] where analyzing information distribution in such systems was done by PID. In addition, PID was used to directly study human brain [44] as PID gave new insights on the pattern of information transfer in a network of scalp electrodes of EEG (Electroencephalography).

1.1.2. Different Partial Information Measures

Williams and Beer in [74] were the first to give a concrete decomposition of mutual information into nonnegative quantities. They introduced the so-called *Williams-Beer axioms* [74] which are natural properties of shared information – see Chapter 3 for the definition.

From these axioms, they proposed the *partial information lattice* framework for partial information decomposition. They proposed $\mathrm{I}_{\min}$ which is the quantity of minimum information that any of the random variables $\mathbf{Y}$ or $\mathbf{Z}$ can obtain about each outcome of $\mathbf{X}$, averaged over all the possible outcomes, to be the quantity of shared information, $\mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z})$.

Unlike their framework, $\mathrm{I}_{\min}$ does not always give reliable values [6]. $\mathrm{I}_{\min}$ results in a counterintuitive PID for the COPY gate. The COPY gate is defined as $\mathbf{X} = (\mathbf{Y},\mathbf{Z})$ where $\mathbf{Y}$ and $\mathbf{Z}$ are sampled. The intuitive decomposition of COPY gate is the following:

$$\begin{aligned}
\mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) &= \mathrm{MI}(\mathbf{Y};\mathbf{Z})\\
\mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z}) &= \mathrm{H}(\mathbf{Y}\mid\mathbf{Z})\\
\mathrm{UI}(\mathbf{X};\mathbf{Z}\backslash\mathbf{Y}) &= \mathrm{H}(\mathbf{Y}\mid\mathbf{Z})\\
\mathrm{CI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) &= 0
\end{aligned}$$

But $\mathrm{I}_{\min}$ quantifies shared information to be 1 bit even when $\mathbf{Y}$ and $\mathbf{Z}$ are sampled independently, i.e., $\mathrm{MI}(\mathbf{Y};\mathbf{Z}) = 0$. The problem is that the estimated shared information by $\mathrm{I}_{\min}$ is very large.

Harder, Salge, and Polani's [34] claimed that $\mathrm{I}_{\min}$ overestimates shared and synergistic information. In order to solve the COPY gate issue of $\mathrm{I}_{\min}$, Harder, Salge, and Polani's suggested adding the so-called identity axiom to Williams-beer lattice – see Chapter 3 for the definition. Then, they defined the decomposition based on projections in the probability simplex.

However, several arguments were formulated against adding the identity axiom. In fact, Bertschinger et al. [6] studied the case when the so-called strong symmetry – see Chapter 3 for the definition – is added to Williams-beer lattice. They proved [6, Theorem 1] that there exists no shared information measure that simultaneously satisfies the original Williams-beer axioms, strong symmetry, and has all other measures nonnegative.

The latter motivated their BROJA measure – see Chapter 3 for details – with a restriction made by dropping the strong symmetry axiom. Following this, Griffith and Koch [32] proposed that synergy comes from information which is not necessarily present given the marginal distributions $(\mathbf{X},\mathbf{Y})$ and $(\mathbf{X},\mathbf{Z})$ which coincides with the BROJA measure.

BROJA PID measure. This thesis deals with the PID measure introduced by Bertschinger, Rauh, Olbrich, Jost, and Ay [7]. They defined their measure based on ideas from decision theory. They proposed $\mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z})$ to be the minimum information that $\mathbf{Y}$ holds on $\mathbf{X}$ given the marginal distributions of $(\mathbf{X},\mathbf{Y})$ and $(\mathbf{X},\mathbf{Z})$

which will be referred to as the BROJA PID measure. The thesis chose to deal with this measure among others for its desirable properties which makes it suitable and currently used for several problems in neuroscience, for more details see [73, Section 4.1].

1.1.3. Contributions to Computing Partial Information Decomposition

The supervisor, Dirk Oliver Theis, had conjectured that the difficulties of state-of-the-art solvers resulted from smoothness problems of the objective function at the boundary of the feasible region. This view stood in contrast with other opinions, which suggested that the shape of the feasible region itself was problematic.

In our paper [42], *Bivariate partial information decomposition: The optimization perspective* authored jointly with Dirk Oliver Theis and Raul Vicente, the author of this thesis studied the behavior of the objective function near the boundary of the feasible region. He found that while the optimum is obtained on the boundary for some problems, the boundary region where that can happen is a low dimensional manifold (i.e., a subset of) the boundary region such that all surrounding boundary points are infinitely repellent (have infinite directional derivatives). This strongly advocated that the difficulties in solving the problem result from the characteristics of the objective function.

In the search for a robust approach for computing BROJA PID, the author also implemented 6 different approaches to solving the convex program, including Geometric Programming and several Cone Programming models, and performed extensive computational experiments.

The conclusion was that the MOSEK [1, 2] solver was robust and very fast, while one of the Cone Programming models was the most robust, with satisfying speed. These computations and comparisons were coded in the scientific computing programming language JULIA [39], and is available on GITHUB[1].

The contents of the paper [42] are discussed in Chapter 4 of this thesis; the whole paper [42] is included in the thesis. The author's contribution among others in [42] is working out the Cone Programming model and Geometric Programming model along with proof of optimality conditions (Proposition 4.2.1) and other proofs. The author also performed all the computational experiments and produced the code needed.

BROJA_2PID. Based on the results of [42], the author of this thesis proceeded to develop a production-quality software for the robust computation of the BROJA PID. The result was a Python module, dubbed BROJA_2PID, whose user-friendly interface makes decomposing mutual information easy: If `pdf` is a Python dictionary containing the joint probability distribution, then the following two lines are all that is needed to print the shared information.

[1] `https://github.com/Abzinger/BROJA-Bivariate-Partial_Information_Decomposition`

```
pid = BROJA_2PID.pid( pdf )
print("Shared information: ", pid['SI'])
```

The user interface for the Python module (there are more technical details for the expert user), along with in-depth discussion of the Cone Programming model based on the so-called Exponential Cone, are discussed in the paper [43], *A robust estimator for Bertschinger et al.'s bivariate partial information decomposition*, authored jointly with Dirk Oliver Theis and Raul Vicente. The estimator is available on GITHUB[2].

The paper [43] also discusses the formulation of a multivariate partial information decomposition measure as Exponential Cone Programs in a way that allows BROJA_2PID, with some modifications, to estimate a multivariate partial information decomposition.

The contents of the paper [43] are discussed in Chapter 5 and included in the thesis. The author's contribution among others in [43] is implementing the solver along with working out proofs that the theoretical model solves the problem to optimality and the modeling of a multivariate partial information decomposition measure as Cone Programming.

1.2. Optimizing Partial Decomposition Measures

Consider the problem in neural networks that aims to measure where and when the information does not only pass through the channel but gets modified. In this case, $\mathbf{Y},\mathbf{Z}$ are two sources of the channel and $\mathbf{X}$ is the target of the channel. The information from both $\mathbf{Y}$ and $\mathbf{Z}$ are getting modified via a mechanism and then outputted within $\mathbf{X}$. This information modification of $\mathbf{Y}$ and $\mathbf{Z}$ can be identified as the synergistic information.

Thus $\mathrm{CI}(\mathbf{X};\mathbf{Y},\mathbf{Z})$ quantifies how much information modification the channel mechanism performed. But this contribution does not inform about the potential of the channel mechanism to modify information. Wibral et al. [73] formulated the question of the capability of the mechanism to modify information as follows:

$$\max_{Q\in\Delta_{(\mathbf{Y},\mathbf{Z})}} \mathrm{CI}_{\mathbb{P}}(\mathbf{X};\mathbf{Y},\mathbf{Z}),$$

where $\Delta_{(\mathbf{Y},\mathbf{Z})}$ is the set of all joint distributions of $(\mathbf{Y},\mathbf{Z})$ and $\mathbb{P}$ is a joint probability distribution of $(\mathbf{X},\mathbf{Y},\mathbf{Z})$ such that Q is its $\mathbf{YZ}$-marginal distribution. The set $\Delta_{(\mathbf{Y},\mathbf{Z})}$ can be expressed as constraints and so hints the need to maximize or minimize any of the information decomposition measures subject to constraints.

Other problems, see extractable shared information problem [60], that can be formulated as maximizing or minimizing one of the information decomposition measures subject to different sets of constraints. In this case, one must understand how the objective value changes when parameters of an optimization problem are

[2]`https://github.com/Abzinger/BROJA_2PID`

changed, in order to formulate a general approach which can be used for tackling this type of problems. The latter is known as Sensitivity Analysis in Convex Optimization. Primal-dual methods are useful when applying sensitivity analysis since they provide the so-called optimal dual solution which plays the main role in sensitivity analysis, for example, Gradient descent does not give access to the optimal dual solution and thus does not offer a clue about sensitivity analysis.

Unfortunately, the functions that are minimized inside the BROJA Convex Program (variants of conditional entropy) are not convex on the whole probability simplex (they require some marginals to be fixed to be convex). Hence most likely[3] the most powerful part of the theory of sensitivity analysis is not applicable.

Fortunately, the author of this thesis was able to give what we believe are useful data in this situation. In generic situations (optimum is not on the boundary), we can produce the gradient of the information decomposition measure, and in the degenerate cases, we can still obtain a "local" sub-/super-gradient. In future research, this will allow us to use gradient descent or ascent approaches[4] for optimizing information decomposition measures subject to constraints.

1.2.1. Contributions to Optimizing Partial Decomposition Measures

The author of this thesis was able to prove formulas for the gradients and sub-/super-gradients for the information decomposition measures conjectured by Dirk Oliver Theis. The results are in the preprint [41], which is discussed in Chapter 6 of this thesis. The author of this thesis also developed Julia code for the computation of the PID and simultaneous extraction of the (sub-/super-) gradients from the Karush-Kuhn-Tucker optimality conditions returned by the MOSEK solver. The software is available on GITHUB[5]. In future work, the author of this thesis plans to use it to develop heuristics for computing extractable shared information and for other applications of optimizing information decomposition measures.

1.3. Automation Systems

An automation system is a system where machines can carry out designated tasks from start to finish, without human assistance. This section explains a specific automation system, the automated valet parking system. Subsection 1.3.1 presents the automated valet parking system. Subsection 1.3.2 presents the contributions regarding automated valet parking system.

[3]Rare cases of non-convex optimization have the required conditions in which sensitivity analysis can be fully applied, for more details see [11]

[4]which which will only provide local minima or maxima, since the information decomposition measures are not convex or concave

[5]`https://github.com/Abzinger/Opt_PID`

1.3.1. Automated Valet Parking Systems

In May 2015, the Miami Herald [46] printed an article about a fancy condominium tower in Miami. That building was equipped with a state-of-the-art automated valet parking system. In these systems, a central computer controls a small army of robots moving in the parking area. The robots have the capability to move in two directions, move under cars, lift a car up, carry it to another parking slot, and drop it. A human driver drops off his car at a *vehicle transfer station,* where the car is picked up by a robot. The robots store away the car in the parking lot until the driver requests to retrieve it. Systems like these are installed in parking lots of apartment complexes, airports, and malls in many countries.

According to the Miami Herald, in that apartment complex, the following was happening. In the morning, when the inhabitants of the apartments wanted to drive to work, the system was overloaded: Its algorithms were not able to steer the robots to satisfy a large number of requests within a small time window. Car owners experienced long waiting times (20–30 minutes) between the time when they requested their cars, and when they were delivered to the vehicle transfer stations. According to the Miami Herald, the car owners took Ubers to work, instead of waiting for the sluggish machines to drag out their cars. The operator promised improvement of the algorithms [46], but ultimately had to shut down the project [45], went into bankruptcy [47], and was sued by the residents [37].

This example illustrates nothing else than the need for optimization in the robot operated car parking systems: according to the New York Times, “the technology is there” [61].

This thesis also deals with a fundamental optimization problem involved in the control of an automated car parking system. It studies the theoretical throughput limitations of these systems: Given a car park layout, an initial configuration of a car park (location of cars, robots), into a desired, terminal configuration, what is the optimal set of control instructions for the robots to reorganize the initial configuration into the terminal configuration. The notion “optimal” in this problem, means fastest, in terms of clock-on-the-wall waiting time until the robots have completed their tasks.

1.3.2. Contributions to Automated Valet Parking Systems

In our paper [40], *Comparison of IP and CNF Models for Control of Automated Valet Parking Systems*, authored jointly with Dirk Oliver Theis, the author studied exact methods to tackle the involved optimization problem. The results showed the need for heuristic algorithms to tackle the control of an automated car parking system.

The author’s contribution’s among others in [40] is partly formulating the Integer Programming model and fully the CNF-based Constraint Programming model. In addition, the author implemented in the general-purpose programming language C++ [35] the two models and performed the computational experiments, and is available on GITHUB[6].

[6]`https://github.com/Abzinger/crobots`

2. CONVEX OPTIMIZATION

This chapter reviews the relevant definitions and facts about convex optimization which are needed for this thesis. It starts with reviewing some definitions needed for optimality conditions and then proceed to state optimality conditions for a specific Convex Optimization Problem. Later it explains in a nutshell Interior Point methods, a class of algorithms which solves convex optimization problems, in particular, the Barrier method. The chapter is a collection of known results taken from the literature on convexity (e.g. [11] and [51]), and on interior point methods (e.g. [48], [52], and [62]).

2.1. Basics

The section aims to shed light on an important issue in Convex Optimization, namely, optimality conditions. First, it recalls some notions needed for developing the optimality conditions. Then derives the optimality conditions for a specific constrained Convex Optimization problem which is used in this thesis.

2.1.1. Useful Definitions

This subsection aims to review some of the notions needed to develop the so-called "optimality conditions". It states the notions of directional derivative, sub-gradient, and some of their needed properties.

The functions are considered to take the special values: $-\infty$ and $+\infty$. So, $\mathbb{R}$ will be augmented with these values and denoted by $\overline{\mathbb{R}}$. For every function $f : \mathbb{R}^n \to \overline{\mathbb{R}}$, the *domain* of f is the set:

$$\text{dom}(f) := \{x \mid f(x) < +\infty\}.$$

Definition 2.1.1. Let $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ be a convex function and $x \in \text{dom}(f)$. Then for every $d \in \mathbb{R}^n$ the quantity

$$f'(x;d) = \lim_{\varepsilon \searrow 0} \frac{f(x+\varepsilon d) - f(x)}{\varepsilon}, \tag{2.1}$$

is called the directional derivative of f at x in the direction d.

Remark 2.1.1 ([62, Lemma 2.71])**.** The limit in (2.1) exists in $[-\infty, \infty]$, by the convexity of f.

Definition 2.1.2. Let $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ be a convex function and let $x \in \text{dom}(f)$. A vector $g \in \mathbb{R}^n$ such that

$$f(y) \geq f(x) + g^T(y-x) \text{ for all } y \in \mathbb{R}^n,$$

is called a sub-gradient of f at x.

Remark 2.1.2. Let $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ be a convex function. Then the following hold.

1. There can be many sub-gradients of a convex function f at x.
2. The set of all sub-gradients of f is called the subdifferential of f at x and is denoted by $\partial f(x)$.
3. If f is differentiable at x, then $\nabla f(x)$ is the only sub-gradient of f at x.

Lemma 2.1.1 ([62, Lemma 2.73])**.** *Let f be a convex function and $x \in \operatorname{dom}(f)$. If g is a sub-gradient of f, then $g^T(y-x) \leq f'(x; y-x)$.*

This subsection is concluded by the following lemma for convenience (it is used in the proof of Proposition 4.2.1). It is a simplification of the Moreau-Rockafellar Theorem ([62, Theorem 2.85]).

Lemma 2.1.2. *Let $C_i \subset \mathbb{R}^\ell, i = 1, \ldots, k$ be closed convex sets, $f_i : C_i \to \mathbb{R}$ continuous convex functions, and*

$$f = \sum_{i=1}^{k} f_i(x_i),$$

for $x = (x_1, \ldots, x_k) \in \prod_{i=1}^{k} C_i \subset (\mathbb{R}^\ell)^k$. If, for $i = 1, \ldots, k, g_i$ is a sub-gradient of f_i at x_i, then $g := (g_1, \ldots, g_i)$ is a sub-gradient of f at x. Moreover, all sub-gradients of f at x are of this form.

2.1.2. Optimality Conditions

This subsection discusses the optimality conditions for a specific Convex Optimization. Recall the standard form of the constrained convex optimization problem. Let $f_i : C_i \subseteq \mathbb{R}^n \to \mathbb{R},\ i = 1, \ldots, p$ be convex functions and $h_i : \mathbb{R}^n \to \mathbb{R},\ i = 1 \ldots q$ be affine functions, then a constrained Convex Optimization problem is of the form

$$\begin{array}{lll} \text{minimize} & f_0(x) & \\ \text{subject to} & f_i(x) \leq 0 & \text{for all } i = 1, \ldots, p \\ & h_i(x) = 0 & \text{for all } i = 1, \ldots, q. \end{array}$$

The remainder of this subsection deals with the following Convex Optimization

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \\ & x \geq 0. \end{array} \tag{P}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function, $A \in \mathbb{R}^{m,n}$, and $b \in \mathbb{R}^m$.

Definition 2.1.3. Consider the problem (P) with it feasible region $C = \{x \in \mathbb{R}^n \mid Ax = b, x \geq 0\}$ and let $d \in \mathbb{R}^n$ be a vector. Then d is said to be a

(a) *feasible direction* of f at x, if for some $\varepsilon > 0$, $x + \varepsilon d \in C$.

(b) *descent direction* of f at x, if $f'(x; d) < 0$.

Remark 2.1.3. Consider the problem (P). If there exists a feasible descent direction of f at x, then x is not a minimum of f over C.

Now the optimality conditions of (P) are ready to be given. Throughout the literature, these conditions have been referred to as the Karush-Kuhn-Tucker conditions (KKT). The following theorem is a direct consequence of [62, Theorem 3.34] applied to (P).

Theorem 2.1.1 (Karush-Kuhn-Tucker). *Let $C = \{x \in \mathbb{R}^n \mid Ax = b, x \geq 0\}$. Suppose that for every $j = 1, \dots, n$, there is an $x \in C$ with $x_j > 0$. The function f attains a minimum over C at a point $x \in C$ if, and only if, there exist*

- *a sub-gradient g of f at x*
- *and an m-vector $\lambda \in \mathbb{R}^m$*

such that $A^\top \lambda \leq g$, and for all $j = 1, \dots, n$ with $x_j > 0$, equality holds: $(A^\top \lambda)_j = g_j$.

The condition "$x_j = 0$ or $(A^\top \lambda)_j = g_j$" is called *complementarity.*

2.2. Interior point Theory

This section discusses a subclass of Interior Point methods called the Barrier method. Interior Point methods are class of algorithms which solves linear and nonlinear Convex Optimization problems. The featuring property of these methods is that while the iterates can converge to a point on the boundary, none of the iterates actually lie on the boundary. In 1986, Narendra Karmarkar presented an algorithm for solving Linear Programming which falls into the class of Interior Point methods. The importance of Karmarkar's algorithm is not being the first to address an Interior Point method but rather in its complexity which was polynomial. Form that point on, scientists started developing new Interior Point methods one of which is the Barrier method.

2.2.1. Self-concordant functions

Self-concordant barriers play an important role in having a polynomial time Interior Point method, namely, Barrier method. The subsection starts by defining the self-concordant functions, then the ν-self-concordant barriers, and concluding by a theorem about the existence of self-concordant barriers for closed convex sets.

Definition 2.2.1. Let $C \subseteq \mathbb{R}^n$ be a convex set with non-empty interior. A point x is an interior point of C, denoted by $x \in \text{int}(C)$, if there exists $\varepsilon > 0$ such that for any $y \in \mathbb{R}^n$, we have $y \in C$ whenever $\|x - y\| \leq \varepsilon$.

Definition 2.2.2. Let $C \subseteq \mathbb{R}^n$ be a convex set with non-empty interior, and let $f : \mathbb{R}^n \to \mathbb{R}$ be a three times continuously differentiable convex function defined on the $\text{int}(C)$. Then f is self-concordant with constant M if for all $x \in \text{int}(C), h \in \mathbb{R}^n$,

$$D^3 f(x)[h,h,h] \leq M \cdot \left(h^\top Hf(x)h\right)^{3/2},$$

where $Hf(x)$ is the Hessian of f at point x and D^3 denotes the tensor of third derivatives. Moreover, if $M = 2$ then f is called a standard self-concordant function.

Definition 2.2.3. Let $C \subseteq \mathbb{R}^n$ be a convex set with non-empty interior. Then the function $F : \text{int}(C) \to \mathbb{R}$ is a barrier of C if

$$F(x) \to +\infty, \quad \text{as } x \to \partial C,$$

where ∂C is the boundary of the set C.

Remark 2.2.1. This section only considers strictly convex barriers. Also the domain of definition of F is extended to $\mathbb{R}^n$ where $F(x) = +\infty$ for $x \notin \text{int}(C)$.

Definition 2.2.4. A barrier function F is a ν-self-concordant barrier for a closed convex set $C \in \mathbb{R}^n$ with non-empty interior if it is a standard self-concordant function and

$$HF(x) \geq \frac{1}{\nu} \nabla F(x)^T \nabla F(x), \quad \text{for all } x \in \text{int}(C). \tag{2.2}$$

Example 2.2.1. $F : \mathbb{R}^n_{++} \to \mathbb{R}, F(x) = -\sum_{i=1}^{n} \log(x_i)$ is an n-self-concordant barrier for $C = R^n_+$.

Similar to convex sets there are some operations which conserve the self-concordance of a barrier. This following proposition is a consequence of [52, Proposition 2.3.1].

Proposition 2.2.1. *Let $\{C_j\}_{j \in J}$ be a family of convex sets for some finite index set J. Let F_j be ν_j-self-concordant barriers C_j respectively. Then*

- ***Intersection.*** *$F(x) = \sum_{j \in J} F_j(x)$ is a $\left(\sum_{j \in J} \nu_j\right)$-self-concordant barrier for the intersection $C = C_1 \cap \cdots \cap C_{|J|}$.*
- ***Direct Product.*** *$F(x) = \sum_{j \in J} F_j(x_j)$ is a $\left(\sum_{j \in J} \nu_j\right)$-self-concordant barrier for the product $C = C_1 \times \cdots \times C_{|J|}$.*

Theorem 2.2.1 ([14, Theorem 5.5])**.** *Let $C \subseteq \mathbb{R}^n$ be a convex set with non-empty interior. There exists F which is a $(c \cdot n)$-self-concordant barrier for C (where c is some universal constant)*

2.2.2. Barrier Method

This section will explain the barrier method specialized to the following Convex Optimization problem:

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & Ax = b \\ & x \geq 0. \end{aligned} \tag{P}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function, $A \in \mathbb{R}^{m,n}$, and $b \in \mathbb{R}^m$. It starts with sketching a pseudo-code of the Barrier method. Then, it explains the algorithm and the role that the barriers play. Finally, it concludes with the maximum number of iterations of the barrier method.

Let F be a ν-self concordant barrier for $C := \{x \mid Ax = b, x \geq 0\}$. If $f(x)$ is not linear, then the problem (P) is written in the *epigraph form*

$$\begin{aligned} \text{minimize} \quad & s \\ \text{subject to} \quad & f(x) \leq s \\ & Ax = b \\ & -x \leq 0, \end{aligned}$$

where its feasible region

$$C' := \{(x,s) \in \mathbb{R}^n \times \mathbb{R} \mid f(x) \leq s, Ax = b, x \geq 0\} \tag{2.3}$$

is still convex. Then, suppose that $f(x) = c^T x$ where $c \in \mathbb{R}^n$. The *barrier method* or sometimes called the *path-following method* is described in the following algorithm.

Algorithm 1: Barrier method

```
1 Given $t := t_0 > 0$ and tolerance $\varepsilon > 0$.
2 Initialize x to a point in the interior of the feasible region (strictly feasible).
3 repeat
4     Compute $x^*(t)$ by minimizing $t \cdot c^T x + F(x)$, subjected to $Ax = b$ starting
      at $x$.
5     Update $x := x^*(t)$
6     increase $t$ such that $t := \left(1 + \frac{1}{4\sqrt{\nu}}\right) t$.
7 until $\frac{n}{t} < \varepsilon$
```

The iteration of the barrier method is referred to as the outer iteration. In any outer iteration k, the algorithm has to solve an equality-constrained minimization

$$x^*(t_k) := \underset{\substack{x \in \mathbb{R}^n \\ Ax = b}}{\operatorname{argmin}} \left(t_k \cdot c^T x + F(x)\right). \tag{2.4}$$

The equality-constrained problem is solved by a variant of Newton's method. Recall that Newton's method, which is itself iterative, is used to solve the unconstrained optimization problem

$$x^*(t_k) := \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \Big(t_k \cdot c^\top x + F(x)\Big), \tag{2.5}$$

as follows. If x is the current iterate, Newton's method finds the minimum to the second order Taylor expansion about x of the objective function $g \colon x \mapsto t_k \cdot c^\top x + F(x)$:

$$\begin{aligned} &\underset{y \in \mathbb{R}^n}{\operatorname{argmin}} \Big(g(x) + \nabla g(x)^\top (y - x) + \tfrac{1}{2}(y - x) Hg(x)(y - x)\Big) \\ &= x - Hg(x)^{-1} \nabla g(x). \end{aligned}$$

where $Hg(x)$ denotes the Hessian of g at x.

Thus in the Barrier method, if x is the current iterate, then Newton's method finds the minimum to the second order Taylor expansion about x of the function $g\colon x \mapsto t_k c^T x + F(x)$ *subject to* the equations $Ax = b$, using Lagrange multipliers, i.e., it solves the system

$$\begin{aligned} Hg(x)y + A^{\top}\lambda &= -\nabla g(x) \\ Ax &= b. \end{aligned} \tag{2.6}$$

Recall that if $Hg(x) = HF(x)$ satisfies certain conditions, other than being invertible, the convergence of Newton's method is quadratic. The importance of $F(x)$ being a ν-self-concordant barrier for the feasible region of (2.2.2) aligns in ensuring that those conditions are satisfied in any outer iteration. Precisely, the standard self-concordant property allows finding t_0 and the initial point x_0 ensuring that the convergence of Newton's method for $x^*(t_0)$, is quadratic. The condition that $F(x)$ satisfies (2.2) ensures that by choosing $t_k = \left(1 + \frac{1}{4\sqrt{\nu}}\right) t_{k-1}$, in any future outer iteration k, the convergence of Newton's method for $x^*(t_k)$ is quadratic.

The following is an example of a barrier which can be used for (P). Note that the barrier used in the example is not necessarily a self-concordant barrier for $\{x \mid Ax = b, x \geq 0\}$.

Example 2.2.2. One of the most common barriers is the *logarithmic barrier*. Using this barrier method for (2.2.2), in every outer iteration k, Newton's method solves (2.6) where $g\colon x \mapsto t_k f_0(x) - \sum \ln(x_j)$.

In this case, Newton's method solves the system (2.6) with $Hg(x) = t_k Hf(x) + \mathrm{Diag}_j(1/x_j^2)$ and $\nabla g(x) = t_k \nabla f(x) - \mathrm{Diag}_j(1/x_j)$, where $Diag(\cdot)$ denotes a diagonal matrix of appropriate size with the given diagonal.

By convexity of f, $Hf(x)$ is positive semidefinite, so that adding the diagonal matrix results in a (strictly) positive definite matrix. Hence, the system of linear equations always has a solution.

Complexity. The section is concluded by the following theorem which gives an upper bound on the number of iterations that Algorithm 1 needs to return an ε-optimal solution of (P), i.e., $\left\|\hat{x} - \hat{x^*}\right\|_2 \leq \varepsilon$ where $\hat{x}$ is the optimal solution of (P) and x^* is th returned solution of Algorithm 1. The theorem can be seen as a consequence of [14, Theorem 5.6].

Theorem 2.2.2. *The number of iterations that Algorithm 1 performs before it returns an ε-optimal solution of* (P)

$$O\left(\sqrt{\nu}\log\frac{\nu}{t_0\varepsilon}\right).$$

3. PARTIAL INFORMATION DECOMPOSITION

The introduction gave an overview of partial information decomposition. This chapter discusses in-depth Williams-Beer axioms which led to the first definition of a partial information decomposition measure. Then it goes through the modifications of Williams-Beer axioms which motivated the introduction of BROJA PID measure. Finally, it explains in details the BROJA PID measure that is modeled as an optimization problem.

3.1. Preliminaries

The section reviews definitions and facts about information theory that will be used in the partial information decomposition framework. All information-theoretic measures will be defined in nats instead of bits. Consider the random variables $\mathbf{X},\mathbf{Y},\mathbf{Z}$ sampled from the finite sets X,Y,Z. Let $\mathbb{P}$ be a joint probability distribution of $\mathbf{X},\mathbf{Y},\mathbf{Z}$ where

$$\begin{aligned} p(x,y,z) &:= \mathbb{P}(\mathbf{X}=x,\mathbf{Y}=y,\mathbf{Z}=z) \\ p(x,y) &= \mathbb{P}(\mathbf{X}=x,\mathbf{Y}=y) \\ p(x) &= \mathbb{P}(\mathbf{X}=x) \\ p(x\mid y) &= \mathbb{P}(\mathbf{X}=x\mid \mathbf{Y}=y). \end{aligned}$$

3.1.1. Classical Information Theory Measures

Shannon entropy or *entropy* of $\mathbf{X}$ defined as,

$$\mathrm{H}(\mathbf{X}) = \mathrm{H}_{\mathbb{P}}(\mathbf{X}) := -\sum_{x\in X} p(x)\ln p(x),$$

is the measure of uncertainty of the random variable $\mathbf{X}$, i.e., the amount of information the random variable $\mathbf{X}$ holds. *Conditional entropy* of $\mathbf{X}$ given $\mathbf{Y}$ defined as,

$$\mathrm{H}(\mathbf{X}\mid\mathbf{Y}) = \mathrm{H}_{\mathbb{P}}(\mathbf{X}\mid\mathbf{Y}) := -\sum_{(x,y)\in X\times Y} p(x,y)\ln p(y\mid x),$$

is a measure of the uncertainty of the random variable $\mathbf{X}$ given that the value of $\mathbf{Y}$ is known. *Mutual information* of $\mathbf{X}$ and $\mathbf{Y}$ defined as,

$$\mathrm{MI}(\mathbf{X};\mathbf{Y}) = \mathrm{MI}_{\mathbb{P}}(\mathbf{X};\mathbf{Y}) := \sum_{(x,y)\in X\times Y} p(x,y)\ln\frac{p(x,y)}{p(x)p(y)},$$

is a measure of the amount of information that $\mathbf{Y}$ contains about $\mathbf{X}$ or that $\mathbf{X}$ contains about $\mathbf{Y}$. Also, $\mathrm{MI}(\mathbf{X};\mathbf{Y})$ can be seen as the reduction in uncertainty of $\mathbf{X}$ due to the knowledge of $\mathbf{Y}$ or conversely and so it can be equivalently expressed by the following identities:

$$\begin{aligned} \mathrm{MI}(\mathbf{X};\mathbf{Y}) &= \mathrm{H}(\mathbf{X}) - \mathrm{H}(\mathbf{Y}\mid\mathbf{X}) \\ \mathrm{MI}(\mathbf{X};\mathbf{Y}) &= \mathrm{H}(\mathbf{Y}) - \mathrm{H}(\mathbf{X}\mid\mathbf{Y}). \end{aligned}$$

Conditional mutual information of $\mathbf{X}$ and $\mathbf{Y}$ given $\mathbf{Z}$ defined as

$$\mathrm{MI}(\mathbf{X};\mathbf{Y} \mid \mathbf{Z}) = \mathrm{MI}_{\mathbb{P}}(\mathbf{X};\mathbf{Y} \mid \mathbf{Z}) := \sum_{(x,y,z)\in X\times Y\times Z} p(x,y,z) \ln \frac{p(x,y \mid z)}{p(x \mid z)p(y \mid z)}$$

is the measure of the amount of information that $\mathbf{Y}$ contains about $\mathbf{X}$ or $\mathbf{X}$ contains about $\mathbf{Y}$ given $\mathbf{Z}$. Finally, the chain rule of entropy and mutual information is as follows:

$$\begin{aligned} \mathrm{H}(\mathbf{X},\mathbf{Y}) &= \mathrm{H}(\mathbf{X}) + \mathrm{H}(\mathbf{Y} \mid \mathbf{X}) \\ \mathrm{H}(\mathbf{X},\mathbf{Y}) &= \mathrm{H}(\mathbf{Y}) + \mathrm{H}(\mathbf{X} \mid \mathbf{Y}) \\ \mathrm{MI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) &= \mathrm{MI}(\mathbf{X},\mathbf{Y}) + \mathrm{MI}(\mathbf{X};\mathbf{Z} \mid \mathbf{Y}) \\ \mathrm{MI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) &= \mathrm{MI}(\mathbf{X},\mathbf{Z}) + \mathrm{MI}(\mathbf{X};\mathbf{Y} \mid \mathbf{Z}). \end{aligned} \tag{3.1}$$

3.1.2. Partial Information Decomposition Extension

The goal of partial information decomposition is to partition $\mathrm{MI}(\mathbf{X};\mathbf{Y},\mathbf{Z})$, the amount of total information that $\mathbf{Y}$ and $\mathbf{Z}$ hold about $\mathbf{X}$, into more interpretable parts, namely, shared, unique, and synergistic information,

$$\mathrm{MI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) = \mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) + \mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z}) + \mathrm{UI}(\mathbf{X};\mathbf{Z}\backslash\mathbf{Y}) + \mathrm{CI}(\mathbf{X};\mathbf{Y},\mathbf{Z}). \tag{PID}$$

The partitions of PID are all nonnegative functions in $\mathbb{R}$ that take as argument the joint distribution $\mathbb{P}$ of $(\mathbf{X},\mathbf{Y},\mathbf{Z})$. In addition, this PID must also fulfill the following two identities:

$$\begin{aligned} \mathrm{MI}(\mathbf{X};\mathbf{Y}) &= \mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) + \mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z}), \\ \mathrm{MI}(\mathbf{X};\mathbf{Z}) &= \mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) + \mathrm{UI}(\mathbf{X};\mathbf{Z}\backslash\mathbf{Y}). \end{aligned} \tag{3.2}$$

using the chain rule of mutual information (3.1) and (PID), the two identities in (3.2) can be written as follows:

$$\begin{aligned} \mathrm{MI}(\mathbf{X};\mathbf{Y} \mid \mathbf{Z}) &= \mathrm{CI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) - \mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z}) \\ \mathrm{MI}(\mathbf{X};\mathbf{Z} \mid \mathbf{Y}) &= \mathrm{CI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) - \mathrm{UI}(\mathbf{X};\mathbf{Z}\backslash\mathbf{Y}) \end{aligned}$$

Hence, PID doesn't only decompose the $\mathrm{MI}(\mathbf{X};\mathbf{Y};\mathbf{Z})$ into more interpretable parts but also all the other classical partitions of it.

Finally, recall some properties of PID which were mentioned in Section 1.1. This decomposition is unique meaning that given a joint distribution $\mathbb{P}$ of $(\mathbf{X},\mathbf{Y},\mathbf{Z})$ there is one and only one PID of $\mathrm{MI}(\mathbf{X};\mathbf{Y},\mathbf{Z})$. In this decomposition, defining the value of one of the information quantities defines the whole PID. So, PID measure refers to defining the value of one of the information quantities. There are no known closed forms which quantify $\mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z}), \mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z}), \mathrm{UI}(\mathbf{X};\mathbf{Z}\backslash\mathbf{Y})$, or $\mathrm{CI}(\mathbf{X};\mathbf{Y},\mathbf{Z})$.

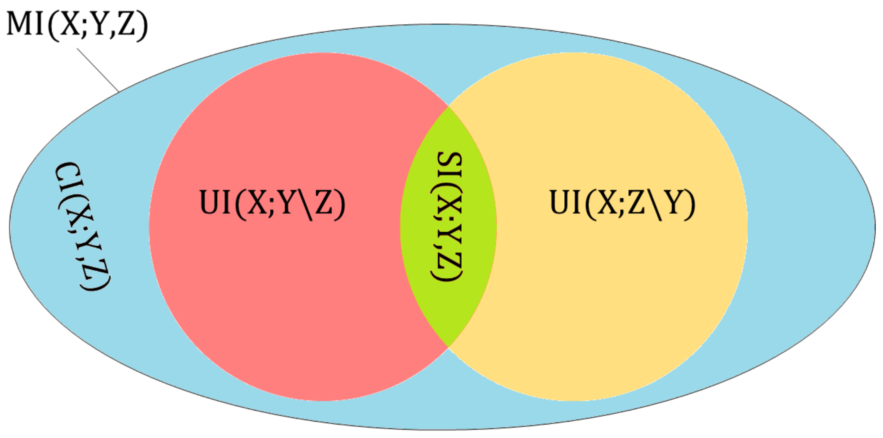


Figure 2. $MI(\mathbf{X};\mathbf{Y},\mathbf{Z})$ decomposed into the four information quantities.

3.2. Evaluating Partial Information Decomposition

This section gives the definition of Williams-Beer axioms, the main result that yielded the first PID measure. Then, it discusses several enhancements of these axioms and the different resulted PID measures from these enhancements. Finally, it gives an in-depth overview of the BROJA measure which is used in this thesis.

3.2.1. Williams-Beer Axioms and its Variations

During their work to define a PID measure with nonnegative quantities, Williams and Beer claimed some natural properties of shared information which will be later called *Williams-Beer axioms* [74]. These axioms are the following:

(S) $\mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) = \mathrm{SI}(\mathbf{X};\mathbf{Z},\mathbf{Y})$, *(Symmetry)*

(SR) $\mathrm{SI}(\mathbf{X};\mathbf{Y}) = \mathrm{MI}(\mathbf{X};\mathbf{Y})$ and $\mathrm{SI}(\mathbf{X};\mathbf{Z}) = \mathrm{MI}(\mathbf{X};\mathbf{Z})$, *(Self-redundancy)*

(M) $\mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) \leq \mathrm{SI}(\mathbf{X};\mathbf{Y})$ and $\mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) \leq \mathrm{SI}(\mathbf{X};\mathbf{Z})$,
with equality if $\mathbf{Y} = f(\mathbf{Z})$ and $\mathbf{Z} = f(\mathbf{Y})$ respectively
for some function f. *(Monotonicity)*

These axioms were the starting point for their PID measure $\mathrm{I}_{\mathrm{min}}$ which evaluated the quantity of shared information $\mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z})$ as the quantity of minimum information that any of the random variables $\mathbf{Y}$ or $\mathbf{Z}$ can obtain about each outcome of $\mathbf{X}$ averaged over all the possible outcomes.

Unlike their framework, $\mathrm{I}_{\mathrm{min}}$ does not always give reliable values [6]. $\mathrm{I}_{\mathrm{min}}$ results in a counterintuitive PID for the COPY gate. The COPY gate is defined as $\mathbf{X} = (\mathbf{Y},\mathbf{Z})$ and intuitively $\mathrm{SI}((\mathbf{Y},\mathbf{Z});\mathbf{Y},\mathbf{Z})\,\mathrm{MI}(\mathbf{Y};\mathbf{Z})$, but $\mathrm{I}_{\mathrm{min}} = \ln 2$ even when $\mathrm{MI}(\mathbf{Y};\mathbf{Z}) = 0$.

Harder, Salge, and Polani's [34], following this misevaluation of $\mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z})$ by $\mathrm{I}_{\min}$, suggested to enforce the COPY gate shared information as an additional axiom to Williams-beer lattice:

(Id) $\mathrm{SI}((\mathbf{Y},\mathbf{Z});\mathbf{Y},\mathbf{Z}) = \mathrm{MI}(\mathbf{Y};\mathbf{Z})$. *(Identity)*

However, adding **(Id)** to the Williams-Beer axiom was highly criticized. For example, Bertschinger et al. [6] presented a game theoretic analysis resulted in a non-trivial shared information in the COPY gate when $\mathbf{Y}$ and $\mathbf{Z}$ are independent random variables. Their conclusion was that $\mathrm{I}_{\min}$ overestimated this non-trivial shared information and that in this case the value should be shared information should be positive but much smaller than $\ln 2$.

Bertschinger et al. [6] studied the consequence of adding to Williams-Beer axioms another natural property of shared information called strong symmetry:

(SS) $\mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) = \mathrm{SI}(\mathbf{Y};\mathbf{X},\mathbf{Z}) = \mathrm{SI}(\mathbf{Z};\mathbf{X},\mathbf{Y})$. *(Strong Symmetry)*

Their argument was that **(SS)** is a natural extension of **(SR)** as $\mathrm{SI}(\mathbf{X};\mathbf{Y}) = \mathrm{SI}(\mathbf{Y};\mathbf{X})$ which the strong symmetry property in the case of $(\mathbf{X},\mathbf{Y})$. But they found out that **(SS)** cannot be added to the Williams-Beer lattice and proved [6, Theorem 1] that there exists no shared information measure that simultaneously satisfies the original Williams-beer axioms, the strong symmetry, and has all other measures nonnegative. This motivated their BROJA measure–see next subsection for full details– with a restriction made by dropping **(SS)**.

3.2.2. BROJA Partial Information Decomposition

This subsection explains BROJA PID measure introduced by Bertschinger, Rauh, Olbrich, Jost, and Ay [7] which is used in this thesis. The definition of the BROJA measure is based on ideas from decision theory. They proposed $\mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z})$ to be the minimum information that $\mathbf{Y}$ holds on $\mathbf{X}$ given the marginal distributions of $(\mathbf{X},\mathbf{Y})$ and $(\mathbf{X},\mathbf{Z})$.

Motivation. Let $\mathbb{P} \in \Delta$ where Δ is the set of all joint probability distributions of $(\mathbf{X},\mathbf{Y},\mathbf{Z})$. Bertschinger et al. core idea to construct the BROJA PID measure is that if $\mathbf{Y}$ has a unique information about $\mathbf{X}$ means that it is possible for $\mathbf{Y}$ to exploit this unique information in its favor against $\mathbf{Z}$ – at least given a suitable situation. The suitable situation was formulated in terms of a decision problem – see [7, Section 2] for details. The analysis of the decision problem resulted in the following:

$$\begin{aligned} \mathrm{UI}_{\mathbb{P}}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z}) &= \mathrm{UI}_{Q}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z}) \quad \text{for all } Q \in \Delta_{\mathbb{P}} \\ &\text{and} \\ \mathrm{UI}_{\mathbb{P}}(\mathbf{X};\mathbf{Z}\backslash\mathbf{Y}) &= \mathrm{UI}_{Q}(\mathbf{X};\mathbf{Z}\backslash\mathbf{Y}) \quad \text{for all } Q \in \Delta_{\mathbb{P}}. \end{aligned} \tag{UC}$$

where

$$\begin{aligned} \Delta_{\mathbb{P}} := \{Q \in \Delta \mid & Q(\mathbf{X}=x,\mathbf{Y}=y) = \mathbb{P}(\mathbf{X}=x,\mathbf{Y}=y), \\ & Q(\mathbf{X}=x,\mathbf{Z}=z) = \mathbb{P}(\mathbf{X}=x,\mathbf{Z}=z) \\ & \text{for all } (x,y,z) \in X \times Y \times Z\}. \end{aligned}$$

So, (UC) states that unique information is constant over $\Delta_{\mathbb{P}}$ and by exploiting this result a definition of unique information can be formulated.

Defining Unique Information. From (UC) and the identities (3.2), it follows that shared information is also constant over $\Delta_{\mathbb{P}}$, i.e., $\mathrm{SI}_{\mathbb{P}}(\mathbf{X};\mathbf{Y},\mathbf{Z}) = \mathrm{SI}_{Q}(\mathbf{X};\mathbf{Y},\mathbf{Z})$ for all $Q \in \Delta_{\mathbb{P}}$. But $\mathrm{MI}_{Q}(\mathbf{X};\mathbf{Y},\mathbf{Z})$ is not constant on $\Delta_{\mathbb{P}}$ and so $\mathrm{CI}(\mathbf{X};\mathbf{Y},\mathbf{Z})$ varies on $\Delta_{\mathbb{P}}$ by (PID). Bertschinger et al. noticed that if there exists $Q_0 \in \Delta_{\mathbb{P}}$ such that $\mathrm{CI}_{Q_0}(\mathbf{X};\mathbf{Y},\mathbf{Z}) = 0$, then

$$\mathrm{UI}_{\mathbb{P}}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z}) = \mathrm{UI}_{Q_0}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z}) = \mathrm{MI}_{Q_0}(\mathbf{X};\mathbf{Y} \mid \mathbf{Z})$$

leading to an evaluation of unique information. Unfortunately, such Q_0 doesn't necessarily exist for all definitions of SI, UI, and CI. Thus, they introduced a definition of unique information that guarantees the existence of this special Q_0 [7, Lemma 3]:

$$\begin{aligned}\mathrm{UI}_{\mathbb{P}}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z}) &= \min_{Q\in\Delta_{\mathbb{P}}} \mathrm{MI}_{Q}(\mathbf{X};\mathbf{Y} \mid \mathbf{Z})\\ \mathrm{UI}_{\mathbb{P}}(\mathbf{X};\mathbf{Z}\backslash\mathbf{Y}) &= \min_{Q\in\Delta_{\mathbb{P}}} \mathrm{MI}_{Q}(\mathbf{X};\mathbf{Z} \mid \mathbf{Y}).\end{aligned} \tag{BROJA I}$$

So, from (PID) and (3.2),

$$\begin{aligned}\mathrm{SI}_{\mathbb{P}}(\mathbf{X};\mathbf{Y},\mathbf{Z}) &= \max_{Q\in\Delta_{\mathbb{P}}} \mathrm{CoI}_{Q}(\mathbf{X};\mathbf{Y};\mathbf{Z})\\ \mathrm{CI}_{\mathbb{P}}(\mathbf{X};\mathbf{Y},\mathbf{Z}) &= \mathrm{MI}_{\mathbb{P}}(\mathbf{X};(\mathbf{Y},\mathbf{Z})) - \min_{Q\in\Delta_{\mathbb{P}}} \mathrm{MI}_{Q}(\mathbf{X};(\mathbf{Y},\mathbf{Z})).\end{aligned} \tag{BROJA II}$$

where $\mathrm{CoI}_{Q}(\mathbf{X};\mathbf{Y};\mathbf{Z}) = \mathrm{MI}_{Q}(\mathbf{X};\mathbf{Y}) - \mathrm{MI}_{Q}(\mathbf{X};\mathbf{Y} \mid \mathbf{Z})$. In addition, they proved that the four optimization problems in (BROJA I) and (BROJA II) are equivalent [7, Lemma 4].

Convex Program. To computing BROJA PID measure it is enough to evaluate $\mathrm{CI}_{\mathbb{P}}(\mathbf{X};\mathbf{Y},\mathbf{Z})$ which requires computing $\min\limits_{Q\in\Delta_{\mathbb{P}}} \mathrm{MI}_{Q}(\mathbf{X};(\mathbf{Y},\mathbf{Z}))$. So, the BROJA PID measure can be obtained by solving the following optimization problem which is convex – see Chapter 4.

$$\begin{array}{lll}\text{minimize } & \mathrm{MI}_{(x,y,z)\sim q}(x;y,z) & \text{over } q \in \mathbb{R}^{X\times Y\times Z}\\ \text{subject to} & q_{x,y,*} = b^{\mathbf{y}}_{x,y} & \text{for all } (x,y) \in X\times Y\\ & q_{x,*,z} = b^{\mathbf{z}}_{x,z} & \text{for all } (x,z) \in X\times Z\\ & q_{x,y,z} \geq 0 & \text{for all } (x,y,z) \in X\times Y\times Z,\end{array} \tag{CP}$$

where

$$\begin{aligned} q_{x,y,z} &:= Q(\mathbf{X}=x,\mathbf{Y}=y,\mathbf{Z}=z) && \text{for all } (x,y,z) \in X\times Y\times Z\\ q_{x,y,*} &:= \sum_{w} q_{x,y,w} && \text{for all } (x,y) \in X\times Y\\ q_{x,*,z} &:= \sum_{w} q_{x,w,z} && \text{for all } (x,z) \in X\times Z\\ b^{\mathbf{y}}_{x,y} &:= \mathbb{P}\big(\mathbf{X}=x,\mathbf{Y}=y\big) && \text{for all } x\in X,\ y\in Y\\ b^{\mathbf{z}}_{x,z} &:= \mathbb{P}\big(\mathbf{X}=x,\mathbf{Z}=z\big) && \text{for all } x\in X,\ z\in Z.\end{aligned}$$

4. COMPUTATION OF A BIVARIATE PARTIAL INFORMATION DECOMPOSITION MEASURE

Chapter 3 defines partial information decomposition (PID), explains what is meant by PID measure, and presents different measures for PID, in particular, focuses on Bertschinger et al. (BROJA) PID measure.

This chapter looks into the Bertschinger et al. (BROJA) measure of bivariate PID [7]. In particular, it studies the solution of the convex program resulting from this measure. This chapter is a collection of the results obtained in [42].

Recall from Chapter 3 Subsection 3.2.2, that the core of the BROJA PID measure is solving the following Convex Optimization problem,

$$\begin{array}{lll}
\text{minimize} & \operatorname{MI}_{(x,y,z)\sim q}(x;y,z) & \text{over } q \in \mathbb{R}^{X\times Y\times Z} \\
\text{subject to} & q_{x,y,*} = b^{\mathsf{y}}_{x,y} & \text{for all } (x,y) \in X\times Y \\
 & q_{x,*,z} = b^{\mathsf{z}}_{x,z} & \text{for all } (x,z) \in X\times Z \\
 & q_{x,y,z} \geq 0 & \text{for all } (x,y,z) \in X\times Y\times Z,
\end{array} \tag{CP}$$

where

$$\begin{aligned}
q_{x,y,*} &:= \sum_w q_{x,y,w} && \text{for all } (x,y) \in X\times Y \\
q_{x,*,z} &:= \sum_w q_{x,w,z} && \text{for all } (x,z) \in X\times Z \\
b^{\mathsf{y}}_{x,y} &:= \mathbb{P}\big(\mathbf{X}=x, \mathbf{Y}=y\big) && \text{for all } x\in X,\, y\in Y \\
b^{\mathsf{z}}_{x,z} &:= \mathbb{P}\big(\mathbf{X}=x, \mathbf{Z}=z\big) && \text{for all } x\in X,\, z\in Z.
\end{aligned}$$

Before indulging into discussing the problem above and its solution, readers who do not recall convexity well are invited to check Appendix A that serves as a background on convex sets and convex functions.

4.1. Feasible Region and Optimality

This section starts by defining the feasible region of (CP),

$$\P(b) := \Big\{ q \in \mathbb{R}_+^{X\times Y\times Z} \;\Big|\; \forall x,y,z\colon\;\; q_{x,y,*} = b^{\mathsf{y}}_{x,y},\; q_{x,*,z} = b^{\mathsf{z}}_{x,z} \Big\}.$$

Note that q lies on the boundary[1] of the feasible region if $q_{x,y,z} = 0$ for at least one triplet (x,y,z). The following assumptions hold for the feasible region.

1. b^{y} and b^{z} are probability distributions.

[1]In fact q lies on the relative boundary since $\P(b)$ is lower-dimensional. The differentiate between the two concepts will be explicit when needed.

2. No element of X is “redundant”, i.e., for every $x \in X$ we have both $b^{\mathbf{y}}_{x,*} > 0$ and $b^{\mathbf{z}}_{x,*} > 0$.

First of all, consider the vectors $\bar{d} \in \mathbb{R}^{X \times Y \times Z}$ from Appendix A.1 of [7], defined by

$$\bar{d}^{x,y,z,y',z'} := 1^{x,y,z} + 1^{x,y',z'} - 1^{x,y,z'} - 1^{x,y',z}, \tag{4.1}$$

(where $1^{\cdots}$ is the vector with exactly one non-zero entry in the given position, and 0 otherwise) satisfy $\bar{d}_{x,y,*} = \bar{d}_{x,*,z} = 0$ for all x, y, z (we omit the superscripts for convenience, when possible). Recall the equations (PID) and (3.2) which define the information decomposition, it is obvious that there will always exist a partial information decomposition for $\mathrm{MI}(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$. In the following proposition we show that the optimization problem (CP) is always feasible, i.e., $\P(b)$ is non-empty for any b.

Proposition 4.1.1. $\P(b)$ *is non-empty for any b.*

Proof. If $\P(b)$ is empty, then $q_{x,y,*} \neq b^{y}_{x,y}$ or $q_{x,*,z} \neq b^{z}_{x,z}$ for all q. It is easy to see that q can be constructed such that it satisfies one of the two equalities. Without loss of generality, assume that $q_{x,y,*} = b^{y}_{x,y}$ and $q_{x,*,z} \neq b^{z}_{x,z}$, then

$$\begin{aligned}
q_{x,*,z} \neq b^{\mathbf{z}}_{x,z} \iff & \quad \sum_y q_{x,y,z} \neq \mathbb{P}\big(\mathbf{X} = x \text{ and } \mathbf{Z} = z\big) \\
\iff & \quad \sum_z \sum_y q_{x,y,z} \neq \sum_z \mathbb{P}\big(\mathbf{X} = x \text{ and } \mathbf{Z} = z\big) \\
\iff & \quad \sum_y \sum_z q_{x,y,z} \neq \sum_z \mathbb{P}\big(\mathbf{X} = x \text{ and } \mathbf{Z} = z\big) \\
\iff & \quad \sum_y q_{x,y,*} \neq \sum_z \mathbb{P}\big(\mathbf{X} = x \text{ and } \mathbf{Z} = z\big) \\
\iff & \quad \sum_y \mathbb{P}\big(\mathbf{X} = x \text{ and } \mathbf{Y} = y\big) \neq \sum_z \mathbb{P}\big(\mathbf{X} = x \text{ and } \mathbf{Z} = z\big) \\
\iff & \quad b^{\mathbf{y}}_{x,*} \neq b^{\mathbf{z}}_{x,*}.
\end{aligned}$$

Thus $\P(b)$ is non-empty is equivalent to $b^{\mathbf{y}}_{x,*} = b^{\mathbf{z}}_{x,*}$ for all $x \in X$. ☐

The first attempt to simplify the problem is by reducing the feasible region, i.e., fixing variables when possible. This will help in diminishing the complexity of computing the objective function and its derivatives. The evident candidates are the variables that they are implied equations, i.e., $q_{x,y,z} = 0$ holds for all $q \in \P(b)$. Tracing the latter candidates, Proposition 4.1.2 restrict the set of variables of the Convex Program (CP) to those triples

$$(x, y, z) \text{ for which a feasible } q \text{ exists with } q_{x,y,z} \neq 0.$$

Proposition 4.1.2 ([42, Proposition 1])**.** *If, for some* $(x, y, z) \in X \times Y \times Z$, $q_{x,y,z} = 0$ *holds for all* $q \in \P(b)$*, then* $b^{\mathbf{y}}_{x,y} = 0$ *or* $b^{\mathbf{z}}_{x,z} = 0$.

Denote the restricted set of variables by

$$\mathscr{J}(b) := \{(x,y,z) \in X \times Y \times Z \mid \quad b^{\mathbf{y}}_{x,y} > 0,\; b^{\mathbf{z}}_{x,z} > 0 \quad \}, \tag{4.2}$$

then the dimension of the feasible region (with respect to the number of variables, $|\mathscr{J}(b)|$) is given by Corollary 4.1.1. The term "(b)" is omitted whenever no confusion can arise.

Corollary 4.1.1 ([42, Corollary 1])**.** *The dimension of* $\P(b)$ *is*

$$|\mathscr{J}(b)| + |X| - \big|\{(x,y) \mid b^{\mathbf{y}}_{x,y} > 0\}\big| - \big|\{(x,z) \mid b^{\mathbf{z}}_{x,z} > 0\}\big|.$$

The rest of the chapter discusses several possibilities for finding the optimal solution of the convex program (CP): Gradient descent, interior point methods, and geometric programming.

4.2. Direct Methods

This section starts with the direct methods for finding the optimal solution, namely, gradient descent and interior point method. First, it analyzes the objective function, gradient, and Hessian. Then, accordingly, it discusses whether each method is appropriate or not for tackling the problem.

4.2.1. Preliminaries

The objective function to be minimized is

$$\mathrm{MI}_{(x,y,z)\sim q}(x; y,z) = H_{(x,y,z)\sim q}(x) - H_{(x,y,z)\sim q}(x \mid y,z). \tag{4.3}$$

Since the distribution of x is fixed by the marginal equations, the first term in the sum is a constant, and what is left is minimizing the negative conditional entropy. The negative conditional entropy is defined as:

$$\begin{aligned} f\colon \mathbb{R}^{\mathscr{J}}_{+} &\to \mathbb{R} \\ q &\to \sum_{(x,y,z)\in\mathscr{J}} q_{x,y,z} \ln\left(\frac{q_{x,y,z}}{q_{*,y,z}}\right). \end{aligned} \tag{4.4}$$

From now on, the Convex Program which is dealt with is

$$\begin{array}{lll} \text{minimize} & -H_{(x,y,z)\sim q}(x \mid y,z) & \text{over } q \in \mathbb{R}^{\mathscr{J}} \\ \text{subject to} & q_{x,y,*} = b^{\mathbf{y}}_{x,y} & \text{for all } (x,y) \in \mathscr{J}_{x,y} \\ & q_{x,*,z} = b^{\mathbf{z}}_{x,z} & \text{for all } (x,z) \in \mathscr{J}_{x,z} \\ & q_{x,y,z} \geq 0 & \text{for all } (x,y,z) \in \mathscr{J}. \end{array} \tag{CP'}$$

where $\mathscr{J}_{x,y} = \{(x,y) \mid (x,y,z) \in \mathscr{J}\}$ and $\mathscr{J}_{x,z} = \{(x,z) \mid (x,y,z) \in \mathscr{J}\}$. The function f is continuous on its domain, and it is smooth on $]0,\infty[^{\mathscr{J}}$. The gradient is

$$\Big(\nabla f(q)\Big)_{x,y,z} = \partial_{x,y,z} f(q) = \ln\Big(\frac{q_{x,y,z}}{q_{*,y,z}}\Big), \tag{4.5}$$

and the Hessian

$$\Big(Hf(q)\Big)_{(x,y,z),(x',y',z')} = \partial_{x,y,z}\partial_{x',y',z'} f(q) = \begin{cases} 0, & \text{if } (y',z') \neq (y,z) \\ \dfrac{-1}{q_{*,y,z}}, & \text{if } (y',z') = (y,z), x \neq x' \\ \dfrac{q_{*,y,z} - q_{x,y,z}}{q_{x,y,z}\, q_{*,y,z}}, & \text{if } (x',y',z') = (x,y,z). \end{cases} \tag{4.6}$$

It is worth pointing out that the Hessian, while positive semidefinite, is not positive definite, and, more pertinently, is not in general positive definite on the tangent space of the feasible region, i.e., $r^\top Hf(q)r = 0$ is possible for $r \in \mathbb{R}^{\mathscr{J}}$ with $r_{x,y,*} = r_{x,*,z} = 0$ for all (x,y,z). Indeed, if, e.g., $\mathscr{J} = X \times Y \times Z$, it is easy to see that the kernel of $Hf(q)$ has dimension $|Y \times Z|$, whereas the feasible region has dimension $|X|(|Y|-1)(|Z|-1) = |X \times Y \times Z| - |X||Y| - |X||Z| + 1$. Hence, if $|Y \times Z| > |X||Y| + |X||Z| - 1$, then *for every* q(!) the kernel of $Hf(q)$ must have a non-empty intersection with the tangent space of the feasible region.

Optimality condition and boundary issues. In the case of points q which lie on the boundary of the domain, i.e., $q_{x,y,z} = 0$ for at least one triplet (x,y,z), some partial derivatives do not exist. For y,z, denote $X_{yz} := \{x \mid (x,y,z) \in \mathscr{J}\}$. Proposition 4.2.1 describes the situation when the optimal solution lies on the boundary.

Definition 4.2.1 (Section 2.3 [23])**.** The *relative entropy* or *Kullback–Leibler distance* between two probability mass functions $p(x)$ and $q(x)$ is defined as

$$D_{\text{KL}}(p(x)\|q(x)) = \sum_{x\in X} p(x)\log(\frac{p(x)}{q(x)})$$

Moreover, if there is any symbol $x \in X$ such that $p(x) > 0$ and $q(x) = 0$, then

$$D_{\text{KL}}(p(x)\|q(x)) = \infty.$$

Otherwise, if there is any symbol $x \in X$ such that $p(x) = 0$ and $q(x) \geq 0$, then by convention $D_{\text{KL}}(p(x)\|q(x)) = 0$.

Proposition 4.2.1. *Let $q \in ¶$.*

(a) If there is an $(x,y,z) \in \mathscr{J}$ with $q_{x,y,z} = 0$ but $q_{*,y,z} > 0$, then f does not have a sub-gradient at q. Indeed, there is a feasible descent direction of f at q with directional derivative $-\infty$.

(b) Otherwise — i.e., for all $(x,y,z) \in \mathscr{J}$, $q_{x,y,z} = 0$ only if $q_{*,y,z} = 0$ — sub-gradients exist. For all y,z, let $\rho^{y,z} \in]0,1]^{X_{yz}}$ be a probability distribution on X_{yz}. Suppose that, for all y,z with $q_{*,y,z} > 0$,

$$\rho^{y,z}_x = \frac{q_{x,y,z}}{q_{*,y,z}} \qquad \textit{for all } x \in X_{yz}. \tag{4.7}$$

Then g defined by $g_{x,y,z} := \ln(\rho_x^{y,z})$, for all $(x,y,z) \in \mathscr{I}$, is a sub-gradient of f at q.

Moreover, g' is a sub-gradient iff there exists such a g with

- $g'_{x,y,z} \le g$ *for all* $(x,y,z) \in \mathscr{I}$ *with* $q_{x,y,z} = 0$;
- $g'_{x,y,z} = g$ *for all* $(x,y,z) \in \mathscr{I}$ *with* $q_{x,y,z} > 0$.

Proof. For (a), let $(y,z) \in Y \times Z$ with $q_{*,y,z} > 0$, and $x \in X_{yz}$ with $q_{x,y,z} = 0$. There exist y', z' such that $q_{x,y',z}, q_{x,y,z'} > 0$. This means that $\bar{d} := \bar{d}^{x,y,z,y',z'}$ as defined in (4.1) is a feasible direction. Now let's show that $f'(q;\bar{d}) = -\infty$.

$$\begin{aligned} f'(q;\bar{d}) &= \ln\Big(\frac{q_{x,y,z}}{q_{*,y,z}}\Big) + \ln\Big(\frac{q_{x,y',z'}}{q_{*,y',z'}}\Big) - \ln\Big(\frac{q_{x,y,z'}}{q_{*,y,z'}}\Big) - \ln\Big(\frac{q_{x,y',z}}{q_{*,y',z}}\Big) \\ &= \ln\Big(\frac{q_{x,y,z} \cdot q_{x,y',z'} \cdot q_{*,y,z'} \cdot q_{*,y',z}}{q_{x,y,z'} \cdot q_{x,y',z} \cdot q_{*,y,z} \cdot q_{*,y',z'}}\Big), \end{aligned}$$

evaluating all the variables, accordingly, we get that $f'(q;\bar{d}) = -\infty$. Invoking Lemma 2.1.1 yields non-existence of the sub-gradient.

As to (b), we prove the statement for every pair $(y,z) \in Y \times Z$ for the function

$$\begin{aligned} f_{yz} \colon \mathbb{R}_+^{X_{yz}} &\to \mathbb{R} \\ q &\to \sum_{x \in X_{yz}} q_x \ln(q_x/q_*), \end{aligned}$$

and then use Lemma 2.1.2.

Let us fix one pair (y,z). If $q_{x,y,z} > 0$ for all $x \in X_{yz}$ holds, then $f_{y,z}$ is differentiable at q, so we simply apply Remark 2.1.1.

Now assume $q_{*,y,z} = 0$. A vector $g \in \mathbb{R}^{X_{yz}}$ is a sub-gradient of f_{yz}, iff

$$\sum_{x \in X_{yz}} r_{x,y,z} \ln(r_{x,y,z}/r_{*,y,z}) = f_{yz}(q+r) \overset{!}{\ge} f_{yz}(q) + g^\top r = f_{yz}(q) + \sum_{x \in X_{yz}} r_x g_x, \quad (4.8)$$

holds for all $r \in \mathbb{R}^{\mathscr{I}}$ with $r_{x,y,z} \ge$ for all $(x,y,z) \in \mathscr{I}$, and $r_* > 0$.

We immediately deduce $g_x \le 0$ for all x. Let $\rho'_x := e^{g_x}$ for all x, and $\rho_x := \rho'_x/C$, with $C := \rho'_*$. Clearly, ρ is a probability distribution on X_{yz}. Moreover, the difference LHS $-$ RHS of (4.8) is equal to

$$D_{\mathrm{KL}}(r/r_* \| \rho) - \ln C,$$

with D_{KL} denoting Kullback-Leibler divergence. From the usual properties of the Kullback-Leibler divergence, we see that this expression is greater-than-or-equal to 0 for all r, if and only if $C \le 1$, which translates to

$$\sum_x e^{g_x} \le 1.$$

From this, the statements in (b) follow. $\square$

From the proposition, the following corollary is derived.

Corollary 4.2.1 ([42, Corollary 2]). *Suppose a minimum of f over $\P(b)$ is attained in a point q with $q_{x,y,z} = 0$ for a triple (x,y,z) with $b^{\mathbf{y}}_{x,y} > 0$ and $b^{\mathbf{z}}_{x,z} > 0$. Then $q_{u,y,z} = 0$ for all $u \in X$.*

Now the optimality conditions can be written down based on Proposition 4.2.1 and the KKT conditions.

Corollary 4.2.2 ([42, Corollary 3]). *Let $q \in \P(b)$. The minimum of f over $\P(b)$ is attained in q if, and only if, (a) $q_{*,y,z} = 0$ holds whenever there is an $(x,y,z) \in \mathscr{J}(b)$ with $q_{x,y,z} = 0$; and (b) there exist*

- *$\lambda_{x,y} \in \mathbb{R}$, for each (x,y) with $b^{\mathbf{y}}_{x,y} > 0$;*
- *$\mu_{x,z} \in \mathbb{R}$, for each (x,z) with $b^{\mathbf{z}}_{x,z} > 0$;*

satisfying the following: For all y,z with $q_{*,y,z} > 0$,

$$\lambda_{x,y} + \mu_{x,z} = \ln\Big(\frac{q_{x,y,z}}{q_{*,y,z}}\Big) \qquad \textit{holds for all } x \in X_{yz};$$

for all y,z with $q_{*,y,z} = 0$ (but $X_{yz} \neq \emptyset$), there is a probability distribution $\rho \in]0,1]^{X_{yz}}$ on X_{yz} such that

$$\lambda_{x,y} + \mu_{x,z} \leq \ln(\rho^{y,z}_x) \qquad \textit{holds for all } x \in X_{yz}.$$

4.2.2. Gradient Descent

It is clear from Proposition 4.2.1 and Corollary 4.2.1 that the boundary is "problematic". On the one hand, the optimal point can sometimes be on the boundary (e.g. the `AND`-gate, see Appendix A.2 of [7].) On the other hand, by Corollary 4.2.1, optimal boundary points lie in a lower dimensional subset inside the boundary (codimension $\approx |X|$), and the optimal points on the boundary are "squeezed in" between boundary regions which are "infinitely strongly repellent".

From the perspective of choosing an algorithm, it is pertinent that sub-gradients do not exist everywhere on the boundary. This rules out the use of algorithms which rely on evaluating (sub-)gradients on the boundary, such as projected (sub-)gradient descent as well as generic active set and sequential quadratic programming methods[2].

Due to the huge popularity of gradient descent, the decision has been made to present at least one version of it. Thus, a designed ad-hoc quick-and-dirty gradient descent algorithm which does its best to avoid the pitfalls of the feasible region: its boundary. We now describe this algorithm.

Denote by A the matrix representing the LHS of the equations in (CP); also reduce A by removing rows which are linear combinations of other rows. Now, multiplication by the matrix $P := A^{\top}(A^{\top}A)^{-1}A$ amounts to projection onto the

[2]We refer the interested reader to [54] for background on these optimization methods.

tangent space $\{d \mid Ad = 0\}$ of the feasible region, and $P\nabla f(q)$ is the gradient of f in the tangent space, taken at the point q.

The strategy by which we try to avoid the dangers of approaching the boundary of the feasible region in the "wrong" way is by never reducing the smallest entry of the current iterate q by more than 10%. Here is the algorithm.

Algorithm 2: Gradient Descent

```
1  Construct the matrix $A$
2  Compute $P := A^\top(A^\top A)^{-1}A$
3  Initialize $q$ to a point in the interior of the feasible region
4  repeat
5      Compute $f(q)$ if $f(q)$ better than all previous solutions then
6          store $q$
7      Compute the gradient $\nabla f(q)$
8      Compute the projection of the gradient $g := P\nabla f(q)$
9      Determine a step size $\eta$, ensuring $q_{x,y,z} > \eta g_{x,y,z}$ for all $x,y,z$
10     Update $q = q - \eta g$
11 until stopping criterion is reached
```

There are lots of challenges with this approach, not the least of which is deciding on the step size η. Generally, a good step size for gradient descent is 1 over the largest eigenvalue of the Hessian—but the eigenvalues of the Hessian tend to infinity.

The stopping criterion is also not obvious: we use a combination of the norm of the projected gradient, the distance to the boundary, and a maximum of 1000 iterations. (None of these decisions are motivated by careful thought.) It is worth to note that this algorithm produced solutions which are far from being optimal, see [42]. The latter might be due to that 2 is not designed well, but another implementation based on the Frank-Wolfe algorithm, provided by the PYTHON package DIT, used to solve (CP) and the tests in [43] showed that Frank-Wolfe algorithm produces solutions which are far from being optimal.

4.2.3. Interior Point Methods

Using Interior Point Methods (IPMs) appears to be the natural approach: While the iterations can converge to a point on the boundary, none of the iterations actually lie on the boundary, and that is an inherent property of the method. Consequently, problems with gradients, or even non-existing sub-gradients, never occur.

Even here, however, there are caveats involving the boundary. Chapter 2 Subsection 2.2.2 discusses the convergence of a particular Interior Point method, Barrier method. The complexity of this method depends mainly on computing the gradient and the Hessian of the barrier rather than those of the objective function.

Recall that the analysis of this method requires that the function

$$F\colon q \mapsto tf_0(q) - \sum_{i=1}^{m} \phi(f_i(q)),$$

be *standard self-concordant,* which means that, for some constant C, for all q, h

$$D^3F(q)[h,h,h] \le C \cdot \left(h^{\top} HF(q) h\right)^{3/2}, \tag{4.9}$$

where f_0 denotes the objective function of the convex problem, f_i for all $1 \le i \le m$ denote the inequality constraints, ϕ denotes the barrier function, and D^3 denotes the tensor of third derivatives. If the function $F\colon q \mapsto tf_0(q) - \sum_{i=1}^{m} \phi(f_i(q))$ is a standard self-concordant and

$$HF(x) \ge \frac{1}{\nu} \nabla F(x)^T \nabla F(x), \quad \text{for all } x \in \text{int}(C). \tag{4.10}$$

then the feasible region has a ν-self-concordant barrier, F. Theorem 2.2.1 shows that for any closed convex set, which is the case for (CP'), there exists a self-concordant barrier. But as Nemirovski has stated

> The aforementioned theorem says that such a barrier always exists, and thus gives us certain encouragement. At the same time, the "universal" barrier given by the theorem usually is too complicated numerically, since straightforward computation of a multidimensional integral involved into the construction is, typically, an untractable task.

So, the goal is to find a computable barrier for (CP') where the gradient and inverse of the Hessian for the barrier are easier to compute than those of the objective function. Consider the function $F\colon q \mapsto tf(q) - \sum_{xyz} \ln(q_{xyz})$, which is needed for the upcoming analysis, with its Hessian

$$\left(D^2F(q)\right)_{(x,y,z),(x',y',z')} = \partial_{(x',y',z')}\partial_{(x,y,z)}F(q) \tag{4.11}$$

$$= \begin{cases} \dfrac{-t}{q_{*,y,z}}, & \text{if } x' \neq x; (y,z) = (y',z') \\[2ex] \dfrac{t}{q_{x,y,z}} + \dfrac{1}{q^2_{x,y,z}} - \dfrac{t}{q_{x,y,z}}, & \text{if } (x,y,z) = (x',y',z') \\[2ex] 0, & \text{otherwise,} \end{cases} \tag{4.12}$$

its tensor of third derivative

$$\left(D^3F(q)\right)_{(x,y,z),(x',y',z'),(x'',y'',z'')} = \partial_{(x'',y'',z'')} \tag{4.13}$$

$$\partial_{(x',y',z')}\partial_{(x,y,z)}F(q) = \begin{cases} \dfrac{t}{q^2_{*,y,z}}, & \text{if } {}^{x''\neq x'\neq x,}_{(y,z)=(y',z')=(y'',z'')} \\ \dfrac{-t}{q^2_{x,y,z}} - \dfrac{2}{q^3_{x,y,z}} + \dfrac{t}{q^2_{*,y,z}}, & \text{if } {}^{\theta=y,}_{\gamma=\delta=x} \\ 0, & \text{otherwise,} \end{cases} \tag{4.14}$$

and their evaluations at the point $h \in \mathbb{R}^{X\times Y\times Z}$

$$\begin{aligned} D^3F(q)[h,h,h] = &\sum_{x,y,z}\left(-\frac{t}{q^2_{x,y,z}} - \frac{2}{q^3_{x,y,z}} + \frac{t}{q^2_{*,y,z}}\right)h^3_{x,y,z} \\ &+3\sum_{\substack{x,x',y,z\\ x\neq x'}}\frac{t}{q^2_{*,y,z}}\left(h^2_{x,y,z}\cdot h_{x',y,z} + h_{x,y,z}\cdot h^2_{x',y,z}\right) \\ &+6\sum_{\substack{x,x',x'',y,z\\ x\neq x'\neq x''}}\frac{t}{q^2_{*,y,z}}\left(h_{x,y,z}\cdot h_{x',y,z}\cdot h_{x'',y,z}\right), \end{aligned} \tag{4.15}$$

$$D^2F(q)[h,h] = \sum_{x,y,z}\left(\frac{t}{q_{x,y,z}} + \frac{1}{q^2_{x,y,z}} - \frac{t}{q_{*,y,z}}\right)h^3_{x,y,z} - 2\sum_{\substack{x,x',y,z\\ x\neq x'}}\frac{t}{q_{*,y,z}}\left(h_{x,y,z}\cdot h_{x',y,z}\right), \tag{4.16}$$

where f is the negative conditional entropy function.

The first try is to use the logarithmic barrier, i.e., $\phi(f_i) = \ln(f_i)$. Proposition 4.2.2 shows that the logarithmic barrier is not self-concordant when the optimal solution is on the boundary. It explains why, even for some IPMs, approaching the boundary can be problematic.

Proposition 4.2.2. *Let $n \geq 2$, and consider the function*

$$F\colon q \mapsto t\sum_{x=1}^{n} q_x \ln(q_x/q_*) - \sum_x \ln(q_x).$$

There is no t such that (4.9) *holds for all $q \in]0,\infty[^n$ and all h.*

Proof. Let $X = \{0,1,\ldots,n-1\}, Y = \{0\}$, and $Z = \{0\}$. Set q to be

$$q_{x,y,z} = \begin{cases} \dfrac{1}{2} & \text{if } x=y=z=0 \\ \dfrac{1}{2\cdot(n-1)} & \text{if } y=z=0 \text{ and } x\neq 0, \end{cases} \tag{4.17}$$

$h = (h_0, h_1, \dots, h_1)$ where $\dfrac{h_0}{h_1} = \dfrac{n-1}{2}$ and $h_1 = \dfrac{1}{(n-1)^2}$.

The terms of $D^3F(q)[h,h,h]$ sums up to more than wh^3 and those of the Hessian $D^2F(q)[h,h,h]$ to less than $w'h^2$ where $w \geq 2(w')^{3/2}$. Hence

$$D^3F(q)[h,h,h] - 2(DF(q)[h,h])^{3/2} \geq 0.$$

□

Thus, for the Convex Program (CP') the log barrier does not work. However, (CP') can be written as

$$\begin{aligned}
&\text{minimize} && \sum_{(x,y,z)\in X\times Y\times Z} -r_{x,y,z} && \\
&\text{subject to} && q_{x,y,*} = b^{\mathbf{y}}_{x,y} && \text{for all } (x,y) \in \mathscr{I}_{x,y} \\
& && q_{x,*,z} = b^{\mathbf{z}}_{x,z} && \text{for all } (x,z) \in \mathscr{I}_{x,z} \\
& && q_{x,y,z} \geq 0 && \text{for all } (x,y,z) \in \mathscr{I} && \text{(4.18a)} \\
& && p_{x,y,z} = q_{*,y,z} && \text{for all } (x,y,z) \in \mathscr{I} && \text{(4.18b)} \\
& && r_{x,y,z} \leq q_{x,y,z} \ln\left({}^{p_{x,y,z}}/_{q_{x,y,z}}\right) && \text{for all } (x,y,z) \in \mathscr{I}. && \text{(4.18c)}
\end{aligned}$$

For (4.18), the function

$$\begin{aligned}
F : \mathbb{R}^{\mathscr{I}} \times \mathbb{R}^{\mathscr{I}} \times \mathbb{R}^{\mathscr{I}} &\to \mathbb{R} \\
(r,q,p) &\to -\sum_{x,y,z} \ln(q_{x,y,z}\ln(p_{x,y,z}/q_{x,y,z}) - r_{x,y,z}) - \ln(q_{x,y,z}) - \ln(p_{x,y,z})
\end{aligned}$$

is a $|\mathscr{I}|$-self concordant barrier for $\mathscr{K}_{\exp}$. The next chapter shows that using the barrier method the number of iterations needed until it converges to the ε-optimal solution (defined in Section 2.2.2) of the Convex Program (4.18) is

$$O\left(\sqrt{|\mathscr{I}|}\log\left(\frac{|\mathscr{I}|}{t_0\varepsilon}\right)\right).$$

4.3. Geometric Program

Geometric programming forms a sub-class of Convex Programs; they are considered to be easier to solve than general Convex Programs: Specialized algorithms for solving Geometric Programs have been around for a half-century (or more). The so-called "Lagrange dual" of (CP') can be written as Geometric Program. This section briefly reviews Geometric programming and writes down the Geometric program corresponding to (CP').

4.3.1. Background

Geometric Programming (GP) is introduced by Duffin, Peterson, and Zener in [27, 1967]. GP is a famous sub-class of Convex Optimization since it models numerous real-world applications in particularly integrated circuit designing [10,12] and communication systems [20]. GP naturally are not convex but can be easily transformed into Convex Programs. GP can be solved using an Interior Point method, first described by Nesterov and Nemirovsky [52], which has a polynomial time complexity. Recently, new solution methods emerged which solves Geometric Programs (GP), even large-scale ones, efficiently and reliably (see e.g. [1]). Here only the convex formulation of GPs is discussed. The GP can be written as follows,

$$\begin{aligned} &\text{minimize} \quad \ln\big[\sum_{k=1}^{K_0} \exp\big(a_{0,k}^T y + c_{0,k}\big)\big] \\ &\text{subject to} \quad \ln\big[\sum_{k=1}^{K_i} \exp\big(a_{i,k}^T y + c_{i,k}\big)\big] \leq 0 \quad i = 1,\dots,t, \end{aligned} \tag{GP}$$

over the variables $y \in \mathbb{R}^m$, where $n > 0$ such that $a_{i,k}^T$, $i = 1,\dots,t$ is the kth row of a matrix $A_i \in \mathbb{R}^{K_i \times n}$, and $K_0 + \dots + K_t = n$. It can be easily noted that length of $a_{i,k}^T$ is m. In what follows, let $a_{0,k}^T = -b$ and $c_j = (c_{j,1},\dots,c_{j,K_j}) = (\ln v_{j,1},\dots,\ln v_{j,K_j})$, $j = 0,1,\dots,t$.

For every (GP), there exists the following so-called dual GP which was called a linear logarithmic program in 1960s (see e.g. [22]),

$$\begin{aligned} &\text{maximize} \quad c_0^T x_0 - \sum_{j=1}^{K_0} x_{0j} \ln x_{0j} + \sum_{i=1}^{t} \big(c_i^T \cdot x_i - \sum_{j=1}^{K_i} x_{ij} \ln \frac{x_{ij}}{\mathbf{1}^T x_i}\big) \\ &\text{subject to} \quad \sum_{i=0}^{t} A_i^T x_i = 0 \\ &\qquad \mathbf{1}^T x_0 = 1 \\ &\qquad x_i \geq 0 \qquad\qquad i = 1,\dots,t, \end{aligned} \tag{Dual GP}$$

over the variable $x \in \mathbb{R}^n$. For more details on the construction of the dual of GP we refer to [11]. The next subsection shows that (CP') is the dual of a primal GP. Note that Chapter 5 expresses the GP which is a primal of (CP') as a “Cone Program” and discuss its duality.

4.3.2. (CP) is a Dual GP

From Equation (4.3), the BROJA measure can be seen as $\max_{(x,y,z)\in\mathscr{J}} H(\mathbf{X} \mid \mathbf{Y}, \mathbf{Z})$ subjected to some corresponding constraints.

Proposition 4.3.1. (CP') *is a dual Geometric Program.*

Proof. Let $A = [\mathbb{0}^T_{K_0,n}, A_1^T, A_2^T, \dots, A_t^T] \in \mathbb{R}^{m\times n}$ and $A_0 = (b^{\mathbf{y}}_{x,y}, b^{\mathbf{z}}_{x,z})$. Then we can write (Dual GP) as

$$\begin{aligned} \text{maximize} \quad & \sum_{i=0}^{t}\sum_{j=1}^{K_i} x_{ij} \ln \frac{v_{i,j}}{x_{i,j}} + \sum_{i=1}^{t}\sum_{j=1}^{K_i} x_{i,j} \ln\big(\sum_{j=1}^{K_i} x_{i,j}\big) \\ \text{subject to} \quad & Ax = b \\ & x \geq 0. \end{aligned} \tag{4.19}$$

Now set $t = |Y \times Z|$ and $m = |X \times Y| + |X \times Z|$. For each $(y_j, z_j) \in Y \times Z$, let $K_j = \sum_{x\in X} I_{q_x \neq 0}$ when $j \neq 0$, $K_0 = 0$, and $(v_{j,1}, \dots, v_{j,K_j}) = \mathbf{1}^T$ for all $j = 0, \dots, t$. Finally choosing A accordingly, we see that (CP') is the dual of the Geometric Program. □

Using the proof of Proposition 4.3.1, the primal of Problem (CP') is

$$\begin{aligned} \text{minimize} \quad & \sum_{(x,y)\in X\times Y} b^{\mathbf{y}}_{x,y} \cdot \lambda_{x,y} + \sum_{(x,z)\in X\times Z} b^{\mathbf{z}}_{x,z} \cdot \mu_{x,z} \\ \text{subject to} \quad & \ln\Bigg(\sum_{x\in X} \exp\big(\lambda_{xy} + \mu_{xz}\big)\Bigg) \leq 0 \qquad \text{for all } (y,z) \in Y \times Z. \end{aligned} \tag{GP}$$

over the variables $\lambda \in \mathbb{R}^{X\times Y}$, $\mu \in \mathbb{R}^{X\times Y}$.

4.4. Computational Results

This section serves to mention the computational results obtained by solving BROJA Convex Program (CP') on a large number of instances, using state-of-art optimization toolkits. The author of this thesis performed numerous experiments in Julia [39] and used different optimization solvers like MOSEK [1, 2], IPOPT [69], ARTELYS KNITRO [15], CVXOPT [65], ECOS [25], and SCS [55,56].

The summary of the results is that the commercial convex optimization solver MOSEK which uses interior point methods was the fastest in computing BROJA PID measure for most of the instances, but MOSEK was not robust. ECOS which is a Cone Programming solver and uses interior point methods was very robust and comparably as fast as MOSEK except for some large instances.

As for the rest of the solvers, the results were inadequate. The best solvers among this group were the ones which employ interior point methods, but even these solvers were not able to compute BROJA PID measure for more than 40% of the instances. The ones which use first-order methods were unable to compute the BROJA PID measure even for very easy and small instances such as $\mathbf{X} = \mathbf{Y}\,\text{XOR}\,\mathbf{Z}$ where $\mathbf{Y}$ and $\mathbf{Z}$ are uniformly distributed. For the full details of the computational results, the reader is invited to check [42].

5. ESTIMATOR OF BIVARIATE PARTIAL INFORMATION DECOMPOSITION

Chapter 4 discusses the solution of the convex program (CP) and concludes that interior point methods are the most effective and reliable in solving the problem. Here, rather than dealing with (CP) as a general Convex Programming problem, the chapter models it as a Cone Programming problem, a subclass of Convex Optimization. Furthermore, the chapter discusses a software package to compute the bivariate partial information decomposition based on the Cone Programming model.

Section 5.1 explains Cone Programming, in essence, and discusses its duality properties. Section 5.2 gives the Conic formulation of the BROJA PID measure. Section 5.4 presents some of the computational results of the solver BROJA_2PID [43] to compute the BROJA PID measure which is based one of the Cone Programming models presented in Section 5.2.

5.1. Background on Cone Programming

This section reviews the mathematical definitions to the point in which they are necessary to understand our model. Conic optimization is the largest proper subclass in convex optimization. Many subclasses of convex optimization like Semidefinite, Geometric, Quadratic, and Linear are in fact Cone Programming. Before defining Cone Programming, recall the definition of closed convex cones and their duals. The section is to a certain extent a collection of known results taken from the literature on Cone Programming (e.g. [11,29,51,62]).

5.1.1. Closed Convex Cones

A nonempty *closed convex cone* $\mathscr{K} \subseteq \mathbb{R}^m$ is a closed set which is *convex*, i.e., for any $x, y \in \mathscr{K}$ and $0 \leq \theta \leq 1$ we have

$$\theta x + (1-\theta) y \in \mathscr{K},$$

and is a *cone*, i.e., for any $x \in \mathscr{K}$ and $\theta \geq 0$ we have

$$\theta x \in \mathscr{K}.$$

Example 5.1.1. Examples of closed convex cones.

- $\{0\}$ and $\mathbb{R}^m$.
- The nonnegative orthant $\mathbb{R}^m_+$.
- The set of positive semidefinite matrices $\mathscr{S}^m_+$.
- The Lorentz (second order) cone

$$\mathscr{L}^m = \{(x_1, \dots, x_{m-1}, x_m)^T \in \mathbb{R}^m \mid x_m \geq \sqrt{\sum_{i=1}^{m-1} x_i^2}\}.$$

Remark 5.1.1. Let $\mathcal{K}, \mathcal{L} \subseteq \mathbb{R}^m$ be two closed convex cones. Then the direct sum of $\mathcal{K}$ and $\mathcal{L}$

$$\mathcal{K} \oplus \mathcal{L} = \{(x,y) \in \mathbb{R}^m \oplus \mathbb{R}^m \mid x \in \mathcal{K}, y \in \mathcal{L}\} \tag{5.1}$$

is a closed convex cone.

The proof of Remark 5.1.1 is easy and left to the reader. (Closure can be attained using a continuous bilinear mapping.) Note that since the direct sum will be always done on finite indices we are going to use the terms direct sum and direct product interchangeably.

Generalized inequalities. When working with closed convex cones, the definition of inequalities is generalized into the so-called "generalized inequalities". The *generalized inequality* is a partial ordering on $\mathbb{R}^m$ that has many properties of the standard ordering on $\mathbb{R}$. A closed convex cone $\mathcal{K}$ is associated with a partial ordering on $\mathbb{R}^m$ defined by

$$x \leq_{\mathcal{K}} y \Leftrightarrow y - x \in \mathcal{K}.$$

Similarly, the *strict generalized inequality* is defined when a closed convex cone $\mathcal{K}$ is associated with a strict partial ordering on $\mathbb{R}^m$

$$x <_{\mathcal{K}} y \Leftrightarrow y - x \in \operatorname{int}(\mathcal{K}).$$

Dual Cones. The *dual* of a closed convex cone $\mathcal{K}$ is the set $\mathcal{K}^*$ defined as

$$\mathcal{K}^* = \{x \in \mathbb{R}^m \mid y^T x \geq 0 \text{ for all } y \in \mathcal{K}\}.$$

Example 5.1.2. Examples of duals of closed convex cones.

- The dual of $\{0\}$ is $\mathbb{R}^m$.
- The dual of $\mathbb{R}^m$ is $\{0\}$.
- The dual of nonnegative orthant is itself.
- The dual of $\mathcal{S}_+^m$ is itself.
- The dual of the Lorentz cone is itself.

Remark 5.1.2. The dual of a closed convex cone is again a closed convex cone.

The notion of cone duality can be applied to direct sums and so

Lemma 5.1.1. *Let $\mathcal{K}, \mathcal{L} \subseteq \mathbb{R}^m$ be two closed convex cones. Then*

$$(\mathcal{K} \oplus \mathcal{L})^* = \mathcal{K}^* \oplus \mathcal{L}^*.$$

The eloquence of the notion of duality can be expressed by the following fact: The dual of the dual is the original object.

Lemma 5.1.2. *Let $\mathcal{K} \subseteq \mathbb{R}^m$ be a closed convex cone. The dual of the cone $\mathcal{K}^*$ is the cone $\mathcal{K}$ itself.*

It is relevant to mention that the proof of the above lemma needs the so-called "Separation Theorem". Different styles of proofs can be found in [29, 49, 51, 62]. This subsection is concluded by a special cone, namely, exponential cone, $\mathcal{K}_{\text{exp}}$. The cone $\mathcal{K}_{\text{exp}}$ will be the cornerstone of our model and shall be used throughout the rest of the chapter.

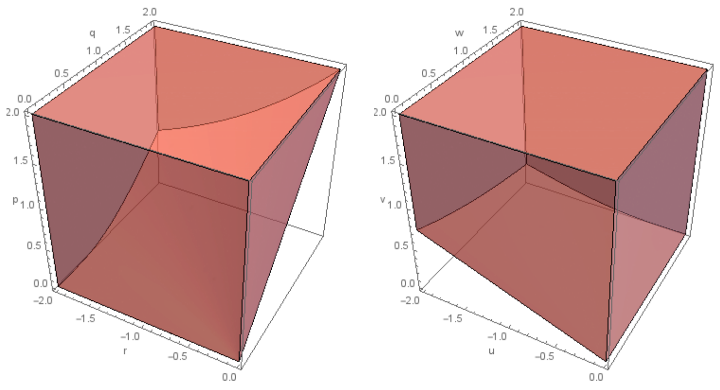


Figure 3. Left - the cone $\mathscr{K}_{\exp}$ for $-2 \le r \le 0$ and $0 \le q, p \le 2$. Right - the cone $\mathscr{K}_{\exp}^*$ for $-2 \le u \le 0$ and $0 \le w, v \le 2$.

Exponential Cone. Before proceeding to construct the exponential cone which is defined in [18], recall the following facts about convex functions.

Proposition 5.1.1. *A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if its epigraph*

$$\operatorname{epi}(f) := \{(x,t) \in \mathbb{R}^n \times \mathbb{R} \mid f(x) \le t\}$$

is a convex set. Moreover, f is said to be concave if $-f$ is convex.

The perspective of a function $f : \mathbb{R}^n \to \mathbb{R}$ is the function $g : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ such that

$$g(x,t) = tf(x/t) \text{ and } \mathbf{dom}(g) = \{(x,t) \mid x/t \in \mathbf{dom}(f), t > 0\}. \tag{5.2}$$

Proposition 5.1.2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function then the perspective of f is a convex function.*

The epigraph of the perspective of e^x is a cone

$$\mathscr{K} = \{(r,q,p) \in \mathbb{R}^3 \mid qe^{r/q} \le p, q > 0\},$$

and so the *exponential cone* is the closure of $\mathscr{K}$, i.e.,

$$\mathscr{K}_{\exp} := K \cup \{(r,0,p) \mid r \le 0, p \ge 0\}.$$

From the previous propositions and the construction of $\mathscr{K}_{\exp}$, it is clear that $\mathscr{K}_{\exp}$ is a closed convex cone. Finally, the dual cone of $\mathscr{K}_{\exp}$ is written down in the following lemma – for the proof see [18, Section 4.3].

Lemma 5.1.3. *The dual cone of $\mathscr{K}_{\exp}$ is*

$$\mathscr{K}_{\exp}^* = \{(u,v,w) \in \mathbb{R}^3 \mid u < 0, -ue^{v/u} \le ew\} \cup \{(0,v,w) \mid v \ge 0, w \ge 0\}. \tag{5.3}$$

5.1.2. Cone Programs

Cone Programming is a far-reaching generalization of Linear Programming, which may contain generalized inequalities. This subsection gives the standard form Cone Program and discusses important properties regarding the optimality.

Standard form Cone Program. Let $\mathscr{L} \subseteq \mathbb{R}^m, \mathscr{K} \subseteq \mathbb{R}^n$ be two closed convex cones, $A \in \mathbb{R}^{m,n}$ be a matrix, and $c \in \mathbb{R}^n, b \in \mathbb{R}^m$ be two vectors. The *primal cone program* is of the form

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & Ax \leq_{\mathscr{L}} b \\ & x \in \mathscr{K}. \end{aligned} \tag{P}$$

Example 5.1.3. The simplest Cone Program is the Linear Program which can be defined using the following cones: non-negative orthant, $\{0\}$, and $\mathbb{R}^n$. The standard form Linear program is

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & Ax = b \\ & Gx \leq h. \end{aligned} \tag{5.4}$$

over the variable $x \in \mathbb{R}^n$ where $G \in \mathbb{R}^{p,n}$ and $h \in \mathbb{R}^p$. Thus the Cone Program model of a Linear Program is

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & Ax \leq_{\{0\}} b \\ & Gx \leq_{\mathbb{R}^n_+} h \\ & x \geq_{\mathbb{R}^n} 0. \end{aligned} \tag{5.5}$$

Note that from now on the inequality $x \geq_{\mathbb{R}^n} 0$ is expressed by the statement "x is a free variable".

The Dual Cone Program. Before writing down the dual of a cone program, the concept of duality is explained in a nutshell. Let $f_0, f_1, \ldots, f_m$ be convex functions from $\mathbb{R}^n$ to $\mathbb{R}$ and $G \in \mathbb{R}^{p,n}$ be a matrix. Then the general constrained convex optimization problem can be written as

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq_{\mathscr{L}} 0 \quad \text{for all } 1 \leq i \leq m \\ & Gx = h. \end{aligned} \tag{5.6}$$

Every problem of the form (5.6) is associated with the following *Lagrangian* function

$$\begin{aligned} L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \quad & \to \quad \mathbb{R} \\ (x, \eta, \theta) \quad & \to \quad L(x, \eta, \theta) = f_0(x) + \sum_{i=1}^{m} \eta_i f_i(x) + \sum_{i=1}^{p} \theta_i (Gx - h)_i. \end{aligned} \tag{5.7}$$

From the Lagrangian function, the *Lagrangian dual function* is defined as

$$\begin{aligned} g : \mathbb{R}^m \times \mathbb{R}^p \quad & \to \quad \mathbb{R} \\ (\eta, \theta) \quad & \to \quad \inf_{x \in D} L(x, \eta, \theta) = \inf_{x \in D} \left(f_0(x) + \sum_{i=1}^{m} \eta_i f_i(x) + \sum_{i=1}^{p} \theta_i (Gx - h)_i \right), \end{aligned} \tag{5.8}$$

where $D := \{x \mid x \in \cap_{i=0}^{m} \operatorname{dom}(f_i), Gx = h\}$ and g is the objective function of the dual problem.

Proposition 5.1.3 ([62], Lemma 4.4)**.** *The Lagrangian dual function $g(\eta, \theta)$ is concave. For any $\eta \geq_{\mathscr{L}^*} 0$ and any θ,*

$$g(\eta, \theta) \leq f_0(x^*) \tag{5.9}$$

where x^* is the optimal solution of $f_0(x)$.

Proposition 5.1.3 indicates that when maximizing $g(\eta, \theta)$ subjected to $\eta \geq_{\mathscr{L}^*} 0$ the result is a convex optimization problem which lower bounds the optimal value of the primal problem. Thus the *dual problem* is

$$\begin{aligned} \text{maximize} \quad & g(\eta, \theta) \\ \text{subject to} \quad & \eta \geq_{\mathscr{L}^*} 0. \end{aligned} \tag{5.10}$$

Remark 5.1.3. The problem of (5.10) can also be written as

$$\begin{aligned} \text{maximize} \quad & g'(\eta, \theta) \\ \text{subject to} \quad & \eta \geq 0. \end{aligned} \tag{5.11}$$

Proof. In Problem (5.6), $f_i \leq_{\mathscr{L}} 0,\ i = 1, \ldots, m$ can be written as

$$f_i' \leq 0,\ i = 1, \ldots, m,$$

where each f_i' explicitly represents the cone $\mathscr{L}$. So, f_i' will replace f_i in $g(\eta, \theta)$. The primal problem (5.6) becomes

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & f_i'(x) \leq_{\mathbb{R}^n_+} 0 \quad \text{for all } 1 \leq i \leq m \\ & Gx = h. \end{aligned} \tag{5.12}$$

In the new dual problem, the constraint $\eta \geq 0$ follows from the fact $(\mathbb{R}^n_+)^* = \mathbb{R}^n_+$. □

Proposition 5.1.4. *The dual cone problem which is the dual problem of the primal cone program* (P) *is*

$$\begin{aligned} \textit{maximize} \quad & -b^T \cdot \eta \\ \textit{subject to} \quad & -A^T \eta \leq_{\mathscr{K}^*} c \\ & -\eta \in \mathscr{L}^*. \end{aligned} \tag{D}$$

Proof. The Lagrangian of problem (P) is

$$\begin{aligned} L(x,\eta') &= x^T c + (b - Ax)^T \eta' \\ &= x^T (c - A^T \eta') + b^T \eta', \end{aligned}$$

and so the dual objective function is

$$g(\eta') = \begin{cases} b^T \eta' & \text{if } -A^T \eta' + c \in \mathscr{K}^* \\ -\infty & \text{otherwise.} \end{cases}$$

Finally, define $\eta = -\eta'$ and we write the implicit constraint $-\eta \in \operatorname{dom}(g)$ explicitly as $-A^T \eta \leq_{\mathscr{K}^*} c$. □

The following example writes down the dual of two special Cone Programs that this section is using later.

Example 5.1.4. When $\mathscr{L} = \{0\}$ which is sometimes referred to as the *equational form* of Cone Programs, the dual problem can be written as

$$\begin{aligned} \text{maximize} \quad & -b^T \eta \\ \text{subject to} \quad & -A^T \eta \leq_{\mathscr{K}^*} c \\ & \eta \text{ is free.} \end{aligned} \tag{D'}$$

When the variables are free, i.e., $\mathscr{K} = \{\mathbb{R}^n\}$ the dual problem can be written as

$$\begin{aligned} \text{maximize} \quad & -b^T \eta \\ \text{subject to} \quad & -A^T \eta = c \\ & -\eta \in \mathscr{L}^*. \end{aligned} \tag{D''}$$

Weak and Strong Duality. From Proposition 5.1.3, the optimal value of the dual problem is at most that of the primal problem. This property is referred to as the weak duality. The primal-dual pair (P), (D) always have weak duality. In what follows, a stating of another duality property for the Cone Program pair namely strong duality which the primal-dual pair may not necessarily have.

Definition 5.1.1. Consider the primal-dual pair (P), (D). Then the following is defined,

1. A vector $x \in \mathbb{R}^n$ (resp. $\eta \in \mathbb{R}^{m_1}$) is said to be a *feasible solution* of (P) (resp. (D)) if $b - Ax \in \mathscr{L}$ and $x \in \mathscr{K}$ (resp. $-\eta \in \operatorname{dom}(g) \cap \mathscr{L}^*$).
2. We say that (P) and (D) *satisfy weak duality* if for any x and any η feasible solutions of (P) and (D) respectively,
$$c^T x + b^T \eta \geq 0.$$
3. If x is a feasible solution of (P) and η is a feasible solution of (D), then the *duality gap* d is
$$d := c^T x + b^T \eta.$$

4. We say that (P) and (D) *satisfy strong duality* if and only if d is zero.

Remark. If strong duality holds for (P) and (D), then x^* and η^* in which $c^T x^* + b^T \eta^* = 0$ are the optimal solutions of (P) and (D) respectively.

As mentioned before, strong duality does not necessarily hold for the Cone Program pair. In what follows, the definition of the interior point of a Cone Program (P) is stated which is sufficient for the strong duality of the Cone program pair to hold.

Definition 5.1.2 (Definition 4.6.4 [29])**.** An *interior point* of the cone program (P) is a point $\tilde{x}$ such that

$$A\tilde{x} \leq_{\mathscr{L}} b \text{ and } \tilde{x} \in \mathscr{K},$$

and the following additional requirement holds:

$$\begin{aligned} \tilde{x} \in \operatorname{int}(\mathscr{K}) & \quad \text{if } \mathscr{L} = \{0\}, \text{and} \\ b - A\tilde{x} \in \operatorname{int}(\mathscr{L}) & \quad \text{otherwise.} \end{aligned}$$

Remark. Let $\mathscr{K}$ be a closed convex cone. Recall that, $x \in \operatorname{int}(\mathscr{K})$ is equivalent to the following: there exists $\varepsilon > 0$ such that for any $y \in \mathbb{R}^n$, we have $y \in \mathscr{K}$ whenever $\|x - y\| \leq \varepsilon$.

Theorem 5.1.1 (Theorem 4.7.1 [29])**.** *If the Primal program* (P) *is feasible, has a finite value γ, and an interior point $\tilde{x}$, then the dual program* (D) *is also feasible and has the same value γ.*

***Conic formulation of Geometric Program*.** The previous chapter mentions that our Convex program is a dual of a Geometric Program and that such duality can be obtained using a conic formulation of a special Geometric Program. The remainder of this section discusses the duality of that GP which will serve as a primary example of our conic formulation of (CP). From now on, the triplets (x,y,z) for which $b^{\mathsf{y}}_{x,y} = 0$ or $b^{\mathsf{z}}_{x,y} = 0$ are ignored. The latter is valid since Proposition 4.1.2 allows tracing their corresponding values in the optimal solution.

Let $\mathscr{J}_{x,y} := \{(x,y) \mid \exists\, z, (x,y,z) \in \mathscr{J}\}$ and similarly define $\mathscr{J}_{x,z}$ and $\mathscr{J}_{y,z}$. Recall the following GP which will be shown to be a primal of (CP),

$$\begin{aligned} \text{minimize} \quad & \sum_{(x,y)\in\mathscr{J}_{x,y}} b^{\mathsf{y}}_{x,y}\lambda_{x,y} + \sum_{(x,z)\in\mathscr{J}_{x,z}} b^{\mathsf{z}}_{x,z}\lambda_{x,z} \\ \text{subject to} \quad & \ln\left(\sum_{x\in X} \exp\left(\lambda_{xy} + \lambda_{xz}\right)\right) \leq 0 \qquad \text{for all } (y,z) \in \mathscr{J}_{y,z}. \end{aligned} \tag{BGP}$$

where $\lambda_{x,y}, \lambda_{x,z} \in \mathbb{R}$ for all $(x,y,z) \in \mathscr{J}$. The cone which is used in modeling (BGP) is called the *geometric cone*, $\mathscr{G}^n$, defined as

$$\mathscr{G}^n = \{(s,v) \in \mathbb{R}^n_+ \times \mathbb{R}_+ \mid \sum_{i=1}^{n} e^{-s_i/v} \leq 1\}. \tag{5.13}$$

Note that by convention $e^{-s_i/v} = 0$ whenever $v = 0$. The author refers to [31] for the proof as well as the construction of the geometric cone and its following dual.

Proposition 5.1.5. *$\mathscr{G}^n$ is a closed convex cone with a nonempty interior. Moreover, its corresponding dual cone is $(\mathscr{G}^n)^*$ is*

$$(\mathscr{G}^n)^* = \{(q,z) \in \mathbb{R}^n_+ \times \mathbb{R} \mid z \geq \sum_{q_i>0} q_i \ln \frac{q_i}{q_1 + \cdots + q_n}\}.$$

Proposition 5.1.6. *The following Convex Program is the dual of* (BGP)

$$\begin{array}{lll} \textit{maximize} & \displaystyle\sum_{(x,y,z)\in\mathscr{J}} q_{x,y,z} \ln\left(\frac{q_{*,y,z}}{q_{x,y,z}}\right) & \\ \textit{subject to} & q_{x,y,*} = b^{\mathbf{y}}_{x,y} & \textit{for all } (x,y) \in \mathscr{J}_{x,y} \\ & q_{x,*,z} = b^{\mathbf{z}}_{x,z} & \textit{for all } (x,z) \in \mathscr{J}_{x,z} \\ & q_{x,y,z} \geq 0 & \textit{for all } (x,y,z) \in \mathscr{J}. \end{array} \tag{5.14}$$

Proof. Let A be the accordingly chosen matrix in Proposition 4.3.1, $\mathbb{1}_{|Y\times Z|}$ be the all ones vector of size $|Y \times Z|$, and $\ell_{y,z} = \{(x,y,z) \mid \text{for some } (y,z) \in \mathscr{J}_{y,z}\}$, $K_{y,z} = |\ell_{y,z}|$. The problem (BGP) can be written as

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{(x,y)\in X\times Y} b^{\mathbf{y}}_{x,y}\lambda_{x,y} + \sum_{(x,z)\in X\times Z} b^{\mathbf{z}}_{x,z}\lambda_{x,z} \\ \text{subject to} & A^T\lambda + s = 0 \\ & \nu = \mathbb{1}_{|Y\times Z|} \\ & (\lambda, s_{\ell_{y,z}}, \nu_{y,z}) \in \mathbb{R}^{|X\times Y|+|X\times Z|} \times \mathscr{G}^{K_{y,z}} \quad \text{for all } (y,z) \in \mathscr{J}_{y,z}. \end{array} \tag{GP-Cone}$$

Using the dual of the equational form of a Cone Program, the dual of (GP-Cone) is

$$\begin{array}{ll} \text{maximize} & -\mathbb{1}^T_{|Y\times Z|} \cdot z \\ \text{subject to} & Aq = b \\ & (q_{\ell_{y,z}}, z_{y,z}) \in (\mathscr{G}^n)^* \quad \text{for all } (y,z) \in \mathscr{J}_{y,z}. \end{array} \tag{5.15}$$

Now by writing the generalized constraints explicitly

$$z_{y,z} \geq \sum_{i\in\ell_{y,z}} q_i \ln \frac{q_i}{q_{(x_0,y,z)} + \cdots + q_{(x_{K_{y,z}},y,z)}}$$

$$q_i \geq 0 \quad \text{for all } i \in \ell_{y,z},$$

and replacing z by their tails in the objective function, the desired optimization problem is

$$\begin{array}{lll} \text{maximize} & \displaystyle\sum_{(x,y,z)\in\mathscr{J}} q_{x,y,z} \ln \frac{q_{*,y,z}}{q_{x,y,z}} & \\ \text{subject to} & Aq = b & \\ & q_{x,y,z} \geq 0 & \text{for all } (x,y,z) \in \mathscr{J}. \end{array} \tag{5.16}$$

□

Remark. The dual of the Geometric Program obtained has a linear objective function. Furthermore, if looking at the primal cone $\mathbb{R}^{|X\times Y|+|X\times Z|}\times\mathscr{G}^n$ of this GP, it is clear that for $s=n$, i.e., $\lambda_{x,y}+\lambda_{x,z}=-n$ the point $(\lambda,s,1)\in\operatorname{int}(\mathbb{R}^{|X\times Y|+|X\times Z|}\times\mathscr{G}^n)$. The latter means that strong duality always holds for this GP.

5.2. Exponential Cone Program for Computing (CP)

This section presents two models for computing the bivariate BROJA PID based on Cone Programming. The corresponding Cone Programs use the exponential cone and thus will refer to the models as "Exponential Cone Programming". For each model, the duality properties are discussed.

5.2.1. Exponential Cone programming I

Before introducing the first conic formalization of (CP) using an exponential cone, recall from Chapter 4 some of the notations which are used here. The Convex Program which yields a partial information decomposition is

$$\begin{array}{lll l}
\text{minimize} & \mathrm{MI}_{(x,y,z)\sim q}(x;y,z) & \text{over } q\in\mathbb{R}^{X\times Y\times Z} & \\
\text{subject to} & q_{x,y,*}=b^{\mathtt{y}}_{x,y} & \text{for all } (x,y)\in X\times Y & (5.17a)\\
 & q_{x,*,z}=b^{\mathtt{z}}_{x,z} & \text{for all } (x,z)\in X\times Z & (5.17b)\\
 & q_{x,y,z}\geq 0 & \text{for all } (x,y,z)\in X\times Y\times Z. &
\end{array}$$

As discussed in Subsection 4.2.1, the objective function is reduced to the negative conditional entropy (in nats)

$$-H(\mathbf{X}\mid\mathbf{Y},\mathbf{Z})=\sum_{x,y,z}q_{x,y,z}\ln\frac{q_{x,y,z}}{q_{*,y,z}}. \tag{5.18}$$

The set of restricted variables that (5.17) is optimized over is

$$\mathscr{J}(b):=\left\{(x,y,z)\in X\times Y\times Z\;\middle|\;b^{\mathtt{y}}_{x,y}>0,b^{\mathtt{z}}_{x,z}>0\right\}.$$

Consider the following Exponential Cone Program where $r,q\in\mathbb{R}^{\mathscr{J}}$ and $p\in\mathbb{R}^{\mathscr{J}_{y,z}}$ are the variables,

$$\begin{array}{lll l}
\text{minimize} & -\sum_{(x,y,z)\in\mathscr{J}} r_{x,y,z} & & \\
\text{subject to} & q_{x,y,*}=b^{\mathtt{y}}_{x,y} & \text{for all } (x,y)\in\mathscr{J}_{x,y} & \\
 & q_{x,*,z}=b^{\mathtt{z}}_{x,z} & \text{for all } (x,z)\in\mathscr{J}_{x,z} & \text{(EXP-I)}\\
 & q_{*,y,z}-p_{y,z}=0 & \text{for all } (y,z)\in\mathscr{J}_{y,z} & \\
 & r_{x,y,z}\leq q_{x,y,z}\ln({}^{p_{y,z}}/_{q_{x,y,z}}) & \text{for all } (x,y,z)\in\mathscr{J} & \\
 & (r_{x,y,z},q_{x,y,z},p_{y,z}) \text{ are free} & \text{for all } (x,y,z)\in\mathscr{J}. &
\end{array}$$

The first two constraints represent the marginal equations (5.17a) and (5.17b). The third type of equations connects the p-variables with the q-variables. Finally, the inequalities connect the r-variables, p-variables, and q-variables forming the objective function (5.18). The following proposition shows that the Cone Program (EXP-I) computes the BROJA bivariate PID.

Proposition 5.2.1. *The exponential cone program* (EXP-I) *is equivalent to* (CP).

Proof. Let $\P_{\mathrm{CP}}$ and $\P_{\exp}$ be the feasible regions of (CP) and (EXP-I) respectively. Denote by $g_{\mathrm{CP}}(q)$ and $g_{\exp}(r,q,p)$ the objective functions of (CP) and (EXP-I) respectively. Set $|X_{y,z}| = n_1, |Y_{x,z}| = n_2$, and $|Z_{x,y}| = n_3$ where

$$\begin{aligned} X_{y,z} :=& \{x \mid (x,y,z) \in \mathscr{J}\} \\ Y_{x,z} :=& \{y \mid (x,y,z) \in \mathscr{J}\} \\ Z_{x,y} :=& \{z \mid (x,y,z) \in \mathscr{J}\}. \end{aligned}$$

Define the following function

$$\begin{aligned} f : \P_{\mathrm{CP}} \to& \quad \P_{\exp} \\ \mathbf{q} \to& \quad f(\mathbf{q}) := (\mathbf{r}, \mathbf{q}, \mathbf{p}), \end{aligned}$$

where

$$\begin{aligned} \mathbf{r} :=& (r_{x_1,y_1,z_1}, \dots, r_{x_{n_1},y_{n_2},z_{n_3}}) \\ \mathbf{q} :=& (q_{x_1,y_1,z_1}, \dots, q_{x_{n_1},y_{n_2},z_{n_3}}) \\ \mathbf{p} :=& (p_{y_1,z_1}, \dots, p_{y_{n_2},z_{n_3}}), \end{aligned}$$

and for all $x,y,z \in \mathscr{J}$

$$\begin{aligned} r_{x,y,z} :=& \begin{cases} q_{x,y,z} \ln \dfrac{q_{*,y,z}}{q_{x,y,z}} & \text{if } q_{x,y,z} > 0 \\ 0 & \text{if } q_{x,y,z} = 0 \end{cases} \\ p_{y,z} :=& q_{*,y,z}. \end{aligned}$$

For $q \in \P_{\mathrm{CP}}$,

$$g_{\exp}(f(q)) = -\sum_{x,y,z} r_{x,y,z} = (-\mathbb{1}_{n_1}, \mathbb{0}_{n_2}, \mathbb{0}_{n_3})^T \cdot f(q),$$

where $\mathbb{1}, \mathbb{0}$ are the vectors all ones and zeros respectively. Since conditional entropy at $q_{x,y,z} = 0$ vanishes, then

$$g_{\mathrm{CP}}(q) = g_{\exp}(f(q)).$$

In addition, when $\mathrm{Im}(f) \subsetneq \P_{\exp}$, i.e., there exists a triplet (x,y,z) such that $r_{x,y,z} < q_{x,y,z} \ln(p_{y,z}/q_{x,y,z})$, the objective function of (EXP-I) is

$$g_{\exp}(r,q,p) = -\sum_{x,y,z} r_{x,y,z} > \sum_{x,y,z} q_{x,y,z} \ln \frac{q_{x,y,z}}{p_{y,z}} = g_{\exp}(f(q)).$$

Hence the two optimization problems are equivalent. $\square$

Duality. In what follows, the dual of (EXP-I) is written down and whether this primal-dual pair has strong duality holds is checked.

Proposition 5.2.2. *The dual of* (EXP-I) *is*

$$\begin{array}{lll} \textit{maximize} & -\sum_{(x,y)\in\mathscr{J}_{x,y}} \lambda_{x,y} b^{y}_{x,y} - \sum_{(x,z)\in\mathscr{J}_{x,z}} \lambda_{x,z} b^{z}_{x,z} & \\ \textit{subject to} & \nu^1_{x,y,z} = 1 & \textit{for all } (x,y,z) \in \mathscr{J} \\ & -\lambda_{x,y} - \lambda_{x,z} - \mu_{xz} - \nu^1_{x,y,z} = 0 & \textit{for all } (x,y,z) \in \mathscr{J} \\ & -\mu_{y,z} - \nu^3_{*,y,z} = 0 & \textit{for all } (y,z) \in \mathscr{J}_{y,z} \\ & -(\nu^1_{x,y,z}, \nu^2_{x,y,z}, \nu^3_{x,y,z}) \in \mathscr{K}^*_{\exp} & \textit{for all } (x,y,z) \in \mathscr{J} \\ & (\lambda_{x,y}, \lambda_{x,z}, \mu_{y,z}) \ \textit{are free} & \textit{for all } (x,y,z) \in \mathscr{J}. \end{array} \tag{DEXP-I}$$

Proof. The Primal Cone program (EXP-I) has $\mathscr{K} = \{\mathbb{R}^n\}$ and $\mathscr{L}$ which connects the variables $\nu^1_{x,y,z}$, $\nu^2_{x,y,z}$, and $\nu^3_{x,y,z}$ for all $(x,y,z) \in \mathscr{J}$ such that $\mathscr{L} = \prod_{x,y,z} \mathscr{L}_{x,y,z}$ where

$$\mathscr{L}_{x,y,z} := \{0\} \times \{0\} \times \{0\} \times \mathscr{K}_{\exp}.$$

So the dual is of the form (D”) where $\eta = (\lambda_{x,y}, \lambda_{x,z}, \mu_{y,z}, \nu^1_{x,y,z}, \nu^2_{x,y,z}, \nu^3_{x,y,z})$. The objective function and the first three affine constraints of (EXP-I) can be obtained by direct computations of A and b. The last constraint in (D”) is $-\eta \in \mathscr{L}^*$. Using Lemma 5.1.1, we get $\mathscr{L}^* = \prod_{x,y,z} \mathscr{L}^*_{x,y,z}$ where

$$\mathscr{L}^*_{x,y,z} := \{\mathbb{R}^{|\mathscr{J}_{x,y}|}\} \times \{\mathbb{R}^{|\mathscr{J}_{x,z}|}\} \times \{\mathbb{R}^{|\mathscr{J}_{y,z}|}\} \times \mathscr{K}^*_{\exp}.$$

So, $-(\nu^1_{x,y,z}, \nu^2_{x,y,z}, \nu^3_{x,y,z}) \in \mathscr{K}^*_{\exp}$ and the remaining variables are free. □

This subsection is concluded by showing that strong duality does not always hold for (EXP-I) and (DEXP-I). And so (DEXP-I) can only be used to obtain lower bounds on the actual optimal solution. For (EXP-I), $\mathscr{K} = \{0\}$ and $\mathscr{L} = \prod_{x,y,z} \mathscr{L}_{x,y,z}$ where

$$\mathscr{L}_{x,y,z} := \{0\} \times \{0\} \times \{0\} \times \mathscr{K}_{\exp}.$$

It is clear that the cone $\mathscr{L}$ does not have an interior point and using Theorem 5.1.1 the primal-dual pair (EXP-I) and (DEXP-I) doesn’t satisfy strong duality. Therefore, the next subsection discusses a modified Cone Program where strong duality holds.

5.2.2. Exponential Cone programming II

This subsection slightly modifies the latter formalization so it guarantees strong duality and then obtains an upper bound on the number of iterations that the barrier method needs to get the optimal solution of the Convex Program 4.18 as promised in Chapter 4 Section 4.2.3. The primal cone program becomes

$$\begin{array}{lll} \text{minimize} & -\sum_{(x,y,z)\in\mathscr{J}} r_{x,y,z} & \\ \text{subject to} & q_{x,y,*} = b^{\mathrm{y}}_{x,y} & \text{for all } (x,y)\in\mathscr{J}_{x,y} \\ & q_{x,*,z} = b^{\mathrm{z}}_{x,z} & \text{for all } (x,z)\in\mathscr{J}_{x,z} \\ & q_{*,y,z} - p_{x,y,z} = 0 & \text{for all } (x,y,z)\in\mathscr{J} \\ & (r_{x,y,z}, q_{x,y,z}, p_{x,y,z}) \in \mathscr{K}_{\exp} & \text{for all } (x,y,z)\in\mathscr{J}. \end{array} \tag{EXP-II}$$

Again the above Cone Program is shown that it computes the BROJA PID measure.

Proposition 5.2.3. *The exponential cone program* (EXP-II) *is equivalent to* (CP).

Proof. Let $\P_{\mathrm{CP}}(b)$ and $\P_{\exp}(b)$ be the feasible region of (CP) and (EXP-II) respectively. Define the following

$$\begin{aligned} f : \P_{\mathrm{CP}}(b) \to &\quad \P_{\exp}(b) \\ q_{x,y,z} \to &\quad f(q_{x,y,z}) := \begin{cases} (q_{x,y,z}\ln\frac{q_{*,y,z}}{q_{x,y,z}}, q_{x,y,z}, q_{*,y,z}) & \text{if } q_{xyz} > 0 \\ (0, q_{x,y,z}, q_{*,y,z}) & \text{if } q_{x,y,z} = 0. \end{cases} \end{aligned} \tag{5.19}$$

Using f and following the same scheme as in the proof of proposition 5.2.1, the two problems (EXP-II) and (CP) are equivalent. □

Duality. In what follows, the dual of (EXP-II) is written down and it is shown that the obtained primal-dual pair has strong duality.

Proposition 5.2.4. *The dual of* (EXP-II) *is*

$$\begin{array}{lll} \textit{maximize} & -\sum_{(x,y)\in\mathscr{J}_{x,y}} b^{y}_{x,y}\lambda_{x,y} - \sum_{(x,z)\in\mathscr{J}_{x,z}} b^{z}x,z\cdot\lambda_{x,z} & \\ \textit{subject to} & 0 \le_{\mathscr{K}^*_{\exp}} -1 & \textit{for all } (x,y,z)\in\mathscr{J} \quad (5.20a) \\ & -\lambda_{xy} - \lambda_{xz} - \mu_{*,y,z} \le_{\mathscr{K}^*_{\exp}} 0 & \textit{for all } (x,y,z)\in\mathscr{J} \quad (5.20b) \\ & \mu_{x,y,z} \le_{\mathscr{K}^*_{\exp}} 0 & \textit{for all } (x,y,z)\in\mathscr{J} \quad (5.20c) \\ & \lambda_{x,y}, \lambda_{x,z}, \mu_{x,y,z} \textit{ are free} & \textit{for all } (x,y,z)\in\mathscr{J}. \quad (5.20d) \end{array}$$

Proof. The Primal Cone program (EXP-II) has $\mathscr{L} = \{0\}$ and $\mathscr{K} = \prod_{x,y,z}\mathscr{K}_{\exp}$. So the dual is of the form (D') where $\eta = (\lambda_{x,y}, \lambda_{x,z}, \mu_{y,z})$. Using Lemma 5.1.1, we get $\mathscr{K}^* = \prod_{x,y,z} \mathscr{K}^*_{\exp}$. So, the objective function and the three inequality constraints of (EXP-II) can be obtained by direct computations of A and b. □

Using the definition of $\mathscr{K}^*_{\exp}$ the system consisting of (5.20a), (5.20b), (5.20c), and (5.20d) is equivalent to

$$\lambda_{x,y} + \lambda_{x,z} + \mu_{*,y,z} + 1 + \ln(-\mu_{x,y,z}) \geq 0 \quad \text{for all } (x,y,z) \in \mathscr{J},$$

and so the dual problem of (EXP-II) can be formulated as

$$\begin{aligned} &\text{maximize} \quad -\sum_{(x,y)\in\mathscr{J}_{x,y}} \lambda_{x,y} b^{\mathbf{y}}_{x,y} - \sum_{(x,z)\in\mathscr{J}_{x,z}} \lambda_{x,z} b^{\mathbf{z}}_{x,z} \\ &\text{subject to} \quad \lambda_{x,y} + \lambda_{x,z} + \mu_{*,y,z} + 1 + \ln(-\mu_{x,y,z}) \geq 0 \quad \text{for all } (x,y,z) \in \mathscr{J}. \end{aligned} \tag{DEXP-II}$$

The following proposition shows that strong duality holds for (EXP-II) and (DEXP-II).

Proposition 5.2.5. *Strong duality holds for the primal-dual pair* (EXP-II) *and* (DEXP-II).

Proof. Without loss of generality assume that $b^{\mathbf{y}}_{x,y}, b^{\mathbf{z}}_{x,z} > 0$. Consider the point $\tilde{t}$ with $\tilde{t}_{x,y,z} = (\tilde{r}_{x,y,z}, \tilde{q}_{x,y,z}, \tilde{p}_{x,y,z})$ such that

$$\begin{aligned} \tilde{r}_{x,y,z} &:= \tilde{q}_{x,y,z} \log \frac{\tilde{p}_{x,y,z}}{\tilde{q}_{x,y,z}} - 100 \\ \tilde{q}_{x,y,z} &:= \frac{b^{\mathbf{y}}_{x,y} b^{\mathbf{z}}_{x,z}}{b^{\mathbf{y}}_{x,*}} \\ \tilde{p}_{x,y,z} &:= \tilde{q}_{*,y,z}. \end{aligned} \tag{5.21}$$

In what follows it is shown that $\tilde{t}$ is an interior point of (EXP-II). First, $\tilde{t}$ is a feasible point of (EXP-II) since it can be easily verified that

$$\tilde{q}_{x,y,*} = b^{\mathbf{y}}_{x,y}, \tilde{q}_{x,*,z} = b^{\mathbf{z}}_{x,z}, \text{ and } \tilde{q}_{x,y,z} \log(\frac{\tilde{p}_{x,y,z}}{\tilde{q}_{x,y,z}}) \geq \tilde{r}_{x,y,z}.$$

Now since $\mathscr{L} = \{0\}$, it is sufficient for $\tilde{t}$ to be an interior point of (EXP-II) that $\tilde{t} \in \operatorname{int}(\mathscr{K})$ where

$$\mathscr{K} = \prod_{x,y,z} \mathscr{K}_{\exp}.$$

This means that it is enough to prove for any $s \in \mathbb{R}^n$ such that the entry $s_{x,y,z} = (u_{x,y,z}, v_{x,y,z}, w_{x,y,z})$ and $\|s - \tilde{t}\|_2 \leq \varepsilon$ that $s_{x,y,z} \in \mathscr{K}_{\exp}$ for all $(x,y,z) \in X \times Y \times Z$ holds. If ε is sufficiently small, then

$$u_{x,y,z} \leq v_{x,y,z} \ln({w_{x,y,z}}/{v_{x,y,z}}) \text{ for all } (x,y,z) \in \mathscr{J}.$$

Hence $\tilde{t} \in \operatorname{int}(\mathscr{K})$ and by Theorem 5.1.1, strong duality holds for the primal-dual pair (EXP-II), (DEXP-II). □

Complexity. Efficient algorithms for Cone Programming exist for some closed convex cones; in particular for the exponential cone. Interior point methods are widely used to solve Exponential Cone Programming. In practice, cone solvers, e.g. ECOS, use the so-called primal-dual interior point method which outperforms the barrier method. The remainder of this subsection is dedicated to providing a proof for the upper bound on the number of iterations that barrier method needs in order to find an optimal solution of the following Convex Program (4.18).

$$\begin{array}{lll} \text{maximize} & \sum\limits_{(x,y,z)\in X\times Y\times Z} -r_{x,y,z} & \\ \text{subject to} & q_{x,y,*} = b^{\mathbf{y}}_{x,y} & \text{for all } (x,y)\in \mathscr{J}_{x,y} \\ & q_{x,*,z} = b^{\mathbf{z}}_{x,z} & \text{for all } (x,z)\in \mathscr{J}_{x,z} \\ & q_{x,y,z} \geq 0 & \text{for all } (x,y,z)\in \mathscr{J} \\ & p_{x,y,z} = q_{*,y,z} & \text{for all } (x,y,z)\in \mathscr{J} \\ & r_{x,y,z} \leq q_{x,y,z}\ln\left({}^{p_{x,y,z}}/_{q_{x,y,z}}\right) & \text{for all } (x,y,z)\in \mathscr{J}. \end{array} \tag{5.22}$$

The latter Convex Program can be seen as a Cone Program of the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & A^T x = b \\ & x \in \mathscr{K}. \end{array} \tag{5.23}$$

where $\mathscr{K}$ is a direct product of $\mathscr{K}_{\exp}$. Recall from Chapter 2 that one of the important factors in the convergence of barrier method is the existence of a ν-self-concordant barrier for $\mathscr{K}$. In this case, there exists a ν-self-concordant barrier for $\mathscr{K}_{\exp}$, which can be used to formulate a ν'-self-concordant barrier for $\mathscr{K}$. The barrier was derived by Yurii Nesterov and the author refers to the proof of the following theorem to [50].

Theorem 5.2.1 (Theorem 3 [50])**.** *The function*

$$\begin{array}{rl} F:\mathbb{R}\times\mathbb{R}^2_+ \to & \mathbb{R} \\ (r,q,p) \to & -\ln(q\ln(p/q)-r)-\ln(q)-ln(p) \end{array} \tag{5.24}$$

is a 3-self-concordant barrier for the exponential cone.

Now the upper bound on the number of iterations in which the barrier method do to find the optimal solution of (4.18) follows by applying Theorem 5.2.1 to the Cone Program (EXP-II).

Corollary 5.2.1. *Let $\varepsilon > 0$ be the optimality gap, and $t_0 > 0$ a factor in the initial descent step. Then, the number of iterations for which the barrier method is ε-away from the optimal solution of* (EXP-II) *is*

$$O\left(\sqrt{|\mathscr{J}|}\log\left(\frac{|\mathscr{J}|}{t_0\varepsilon}\right)\right).$$

Proof. The inequality constraints in the problem (EXP-II) form the following closed convex cone

$$\mathscr{K} = \prod_{(x,y,z)\in\mathscr{J}} \mathscr{K}_{\exp},$$

where $\mathscr{K}_{\exp}$ is the exponential cone. The function (5.24) is a 3-self concordant barrier for $\mathscr{K}_{\exp}$. By Proposition 2.2.1,

$$\begin{array}{lll} F : \mathbb{R}^{\mathscr{J}} \times \mathbb{R}^{\mathscr{J}} \times \mathbb{R}^{\mathscr{J}} & \to & \mathbb{R} \\ \quad (r,q,p) & \to & -\sum\limits_{x,y,z} \ln(q_{x,y,z}\ln(p_{x,y,z}/q_{x,y,z}) - r_{x,y,z}) - \ln(q_{x,y,z}) - ln(p_{x,y,z}) \end{array}$$

is a $|\mathscr{J}|$-self concordant barrier for $\mathscr{K}$. Now using Proposition 2.2.2, the number of iterations is

$$O\left(\sqrt{|\mathscr{J}|}\log\left(\frac{|\mathscr{J}|}{t_0\varepsilon}\right)\right).$$

□

The author of the thesis and others in [43] developed BROJA_2PID solver that implement the Cone Program (EXP-II) to compute BROJA bivariate PID and tested it against some datasets. The solver used to solve the Exponential Cone Programming is ECOS. ECOS [26] is a lightweight numerical software for solving convex cone programs [25], using an Interior Point approach. In BROJA_2PID, the version from Nov 8, 2016 of ECOS is used. The next section discusses the results of these tests with respect to the quality of the solution and computational time of the solver. More details on how to use the BROJA_2PID solver can be found in [43].

5.3. Cone Programming Model for a Multivariate PID

Danial Chicharro [21] has introduced a multivariate PID measure using the so-called tree-base decompositions. The measure determines multivariate shared information within the framework of maximum entropy. BROJA PID measure is a special case of the Chicharro measure in the bivariate case. This section shows how to model the trivariate PID quantities using the exponential cone which enable to obtain a trivariate PID. First, Chicharro trivriate PID is defined and then the exponential formulations of each PID quantity are presented.

5.3.1. Chicharro Trivariate PID

Recall that the trivariate PID is decomposing $\mathrm{MI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3})$, the mutual information, into shared, unique, and synergistic information where $\mathbf{S},\mathbf{X_1},\mathbf{X_2},\mathbf{X_3}$ are random variables with finite range. Let Δ be the set of all joint distributions of

$(\mathbf{S},\mathbf{X_1},\mathbf{X_2},\mathbf{X_3})$. Suppose that $(\mathbf{S};\mathbf{X_1},\mathbf{X_2};\mathbf{X_3})$ are sampled from the joint distribution $\mathbb{P}$, then the following convex set can be defined:

$$\begin{aligned}\Delta_{\mathbb{P}} := \{Q \in \Delta \,|Q(\mathbf{S}=s,\mathbf{X_1}=x_1) &= \mathbb{P}(\mathbf{S}=s,\mathbf{X_1}=x_1),\\ Q(\mathbf{S}=s,\mathbf{X_2}=x_2) &= \mathbb{P}(\mathbf{S}=s,\mathbf{X_2}=x_2),\\ Q(\mathbf{S}=s,\mathbf{X_3}=x_3) &= \mathbb{P}(\mathbf{S}=s,\mathbf{X_3}=x_3),\\ \text{for all } (s,x_1,x_2,x_3) &\in S\times X_1\times X_2\times X_3\},\end{aligned} \tag{5.25}$$

For all $i,j,k \in \{1,2,3\}$, the co-information of $\mathbf{S},\mathbf{X_i},\mathbf{X_j}$ namely $\operatorname{CoI}(\mathbf{S};\mathbf{X_i};\mathbf{X_j})$ and the co-information of $\mathbf{S},\mathbf{X_i},\mathbf{X_j}$ given $\mathbf{X_k}$ namely $\operatorname{CoI}(\mathbf{S};\mathbf{X_i};\mathbf{X_j} \mid \mathbf{X_k})$ are defined as follows

$$\begin{aligned}\operatorname{CoI}(\mathbf{S};\mathbf{X_i};\mathbf{X_j}) &:= \operatorname{MI}(\mathbf{S};\mathbf{X_i}) - \operatorname{MI}(\mathbf{S};\mathbf{X_i}\mid\mathbf{X_j})\\ \operatorname{CoI}(\mathbf{S};\mathbf{X_i};\mathbf{X_j}\mid\mathbf{X_k}) &:= \operatorname{MI}(\mathbf{S};\mathbf{X_i}\mid\mathbf{X_k}) - \operatorname{MI}(\mathbf{S};\mathbf{X_i}\mid\mathbf{X_j},\mathbf{X_k}).\end{aligned} \tag{5.26}$$

Chicharro [21] defines the trivariate PID of $\operatorname{MI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3})$ as follows:

$$\operatorname{CI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3}) = \operatorname{MI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3}) - \min_{Q\in\Delta_{\mathbb{P}}} \operatorname{MI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3}) \tag{5.27a}$$

$$\operatorname{SI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3}) = \min_{\substack{Q\in\Delta_{\mathbb{P}},\operatorname{CoI}(S;X_1;X_2)=0,\\ \operatorname{CoI}(S;X_1;X_2|X_3)=0w(Q)}} \operatorname{MI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3}) - \min_{\substack{Q\in\Delta_{\mathbb{P}},w(Q),\\ \operatorname{CoI}(S;X_1;X_2|X_3)=0}} \operatorname{MI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3}), \tag{5.27b}$$

$$\operatorname{UI}(\mathbf{S};\mathbf{X_i}\backslash\mathbf{X_j},\mathbf{X_k}) = \min_{Q\in\Delta_{\mathbb{P}}} \operatorname{MI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j},\mathbf{X_k}) - \min_{Q\in\Delta_{\mathbb{P}}} \operatorname{MI}(\mathbf{S};\mathbf{X_j},\mathbf{X_k}) \tag{5.27c}$$

$$\operatorname{UI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j}\backslash\mathbf{X_k}) = \min_{Q\in\Delta_{\mathbb{P}},\operatorname{CoI}(S;X_i;X_j|X_k)=0} \operatorname{MI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j},\mathbf{X_k}) - \min_{Q\in\Delta_{\mathbb{P}}} \operatorname{MI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j},\mathbf{X_k}), \tag{5.27d}$$

for all $i,j,k \in \{1,2,3\}$ and where $w(Q)$ evaluates $\operatorname{SI}(\mathbf{S};\mathbf{X_3},(\mathbf{X_1},\mathbf{X_2}))$ as follows, such that Q is its input distribution,

$$\begin{array}{lll}\text{minimize} & \operatorname{CoI}_{(s,x_1,x_2,x_3)\sim q'}(s;(x_1,x_2);x_3) & \text{over } q' \in \mathbb{R}_+^{S\times X_1\times X_2\times X_3}\\ \text{subject to} & q'_{s,*,*,x_3} = Q(\mathbf{S}=s,\mathbf{X_3}=x_3) & \text{for all } (s,x_3)\in S\times X_3\\ & q'_{s,x_1,x_2,*} = Q(\mathbf{S}=s,\mathbf{X_1}=x_1,\mathbf{X_2}=x_2) & \text{for all } (s,x_1,x_2)\in S\times X_1\times X_2.\end{array}$$

Note that the optimization problems in (5.27a) and (5.27c) are convex problems whereas those in (5.27d) and (5.27b) are nonconvex since the some of the equality constraints of the problems in (5.27d) and (5.27b) are not affine. Thus if it is possible to make the non-affine equality constraints of the optimization problems in (5.27d) implicit, then this will allow finding a trivariate PID since all components of the PID will be evaluated except for the shared information which can be computed using the following identity:

$$\begin{aligned}\operatorname{MI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3}) = \operatorname{CI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3}) &+ \sum_{\substack{i,j,k\in\{1,2,3\}\\ i\neq j, i\neq k, j<k}} \operatorname{UI}(\mathbf{S};\mathbf{X_i}\backslash\mathbf{X_j},\mathbf{X_k})\\ &+ \sum_{\substack{i,j,k\in\{1,2,3\}\\ k\neq i, k\neq j, i<j}} \operatorname{UI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j}\backslash\mathbf{X_k}) + \operatorname{SI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3}).\end{aligned}$$

5.3.2. Exponential Formulation of the Trivariate Synergistic Information

The objective function of (5.27a), given the marginal conditions, is equal, up to a constant $\mathrm{H}(\mathbf{S})$ to the negative conditional entropy $\mathrm{H}(\mathbf{S} \mid \mathbf{X_1}, \mathbf{X_2}, \mathbf{X_3})$. So, the optimization problem (5.27a) is equivalent to the following Convex Program:

$$\begin{array}{lll}
\text{minimize} & \displaystyle\sum_{s,x_1,x_2,x_3} q_{s,x_1,x_2,x_3} \ln \frac{q_{s,x_1,x_2,x_3}}{q_{*,x_1,x_2,x_3}} & \text{over } q \in \mathbb{R}^{S\times X_1\times X_2\times X_3} \\
\text{subject to} & q_{s,x_1,*,*} = b^{\mathtt{x}_1}_{s,x_1} & \text{for all } (s,x_1) \in S\times X_1 \\
& q_{s,*,x_2,*} = b^{\mathtt{x}_2}_{s,x_2} & \text{for all } (s,x_2) \in S\times X_2 \\
& q_{s,*,*,x_3} = b^{\mathtt{x}_3}_{s,x_3} & \text{for all } (s,x_3) \in S\times X_3 \\
& q_{s,x_1,x_2,x_3} \geq 0 & \text{for all } (s,x_1,x_2,x_3) \in S\times X_1\times X_2\times X_3.
\end{array} \tag{5.28}$$

where

$$\begin{array}{ll}
b^{\mathtt{x}_1}_{s,x_1} = \mathbb{P}(\mathbf{S} = s, \mathbf{X_1} = x_1) & \text{for all } (s,x_1) \in S\times X_1 \\
b^{\mathtt{x}_2}_{s,x_2} = \mathbb{P}(\mathbf{S} = s, \mathbf{X_2} = x_2) & \text{for all } (s,x_2) \in S\times X_2 \\
b^{\mathtt{x}_3}_{s,x_3} = \mathbb{P}(\mathbf{S} = s, \mathbf{X_3} = x_3) & \text{for all } (s,x_3) \in S\times X_3
\end{array}$$

The exponential formulation of (5.28) is expressed via the following Exponential Cone Program where the variables are $r, p, q \in \mathbb{R}^{S\times X_1\times X_2\times X_3}$.

$$\begin{array}{lll}
\text{minimize} & -\displaystyle\sum_{s,x_1,x_2,x_3} r_{s,x_1,x_2,x_3} & \\
\text{subject to} & q_{s,x_1,*,*} = b^{\mathtt{x}_1}_{s,x_1} & \text{for all } (s,x_1) \in S\times X_1 \\
& q_{s,*,x_2,*} = b^{\mathtt{x}_2}_{s,x_2} & \text{for all } (s,x_2) \in S\times X_2 \\
& q_{s,*,*,x_3} = b^{\mathtt{x}_3}_{s,x_3} & \text{for all } (s,x_3) \in S\times X_3 \\
& q_{*,x_1,x_2,x_3} - p_{s,x_1,x_2,x_3} = 0 & \text{for all } (s,x_1,x_2,x_3) \in S\times X_1\times X_2\times X_3 \\
& (r_{s,x_1,x_2,x_3}, q_{s,x_1,x_2,x_3}, p_{s,x_1,x_2,x_3}) \in \mathscr{K}_{\exp} & \text{for all } (s,x_1,x_2,x_3) \in S\times X_1\times X_2\times X_3.
\end{array} \tag{5.29}$$

Using the same approach as in Proposition 5.2.3, the exponential cone program (5.29) is equivalent to the Convex Program (5.28). The dual problem of (5.29) can be formulated as

$$\begin{array}{ll}
\text{maximize} & -\displaystyle\sum_{(s,x_1)\in S\times X_1} \lambda_{s,x_1} b^{\mathtt{x}_1}_{s,x_1} - \sum_{(s,x_2)\in S\times X_2} \lambda_{s,x_2} b^{\mathtt{x}_2}_{s,x_2} - \sum_{(s,x_3)\in S\times X_3} \lambda_{s,x_3} b^{\mathtt{x}_3}_{s,x_3} \\
\text{subject to} & \lambda_{s,x_1} + \lambda_{s,x_2} + \lambda_{s,x_3} + \mu_{*,x_1,x_2,x_3} + 1 + \ln(-\mu_{s,x_1,x_2,x_3}) \geq 0 \\
& \quad \text{for all } (s,x_1,x_2,x_3) \in S\times X_1\times X_2\times X_3.
\end{array} \tag{5.30}$$

Using the same approach as in Proposition 5.2.5, strong duality holds for the primal-dual pair (5.29), (5.30).

5.3.3. Exponential Formulation of the Trivariate Unique Information

In the trivariate PID there are two types of unique information: $\mathrm{UI}(\mathbf{S};\mathbf{X_i}\backslash\mathbf{X_j},\mathbf{X_k})$, the information that any variable $\mathbf{X_i}$ holds unique about $\mathbf{S}$, and $\mathrm{UI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j}\backslash\mathbf{X_k})$, the information that any two variables $\mathbf{X_i}$ and $\mathbf{X_j}$ share uniquely about $\mathbf{S}$. First, the exponential formulation of the first type of unique information, i.e., (5.27c) will be given and then the exponential formulation of the other type, i.e., (5.27d) will be given.

In (5.27c) there are two optimization problems,

$$\min_{Q\in\Delta_{\mathbb{P}}} \mathrm{MI}(\mathbf{S};\mathbf{X_1},\mathbf{X_2},\mathbf{X_3}) \qquad \text{and} \qquad \min_{Q\in\Delta_{\mathbb{P}}} \mathrm{MI}(\mathbf{S};\mathbf{X_j},\mathbf{X_k}).$$

The first problem is the same as (5.28) and it has been discussed before. Using the chain rule of mutual information:

$$\mathrm{MI}(\mathbf{S};\mathbf{X_j},\mathbf{X_k}) = \mathrm{H}(\mathbf{S}) - \mathrm{H}(\mathbf{S}\mid\mathbf{X_j},\mathbf{X_k}).$$

But, $\mathrm{H}(\mathbf{S})$ is constant on $\Delta_{\mathbb{P}}$ and so the second problem is written as the Convex Program:

$$\begin{array}{lll}
\text{minimize} & \displaystyle\sum_{s,x_2,x_3} q_{s,*,x_2,x_3} \ln \frac{q_{s,*,x_2,x_3}}{q_{*,*,x_2,x_3}} & \text{over } q\in\mathbb{R}^{S\times X_1\times X_2\times X_3} \\
\text{subject to} & q_{s,x_1,*,*} = b^{\mathbf{x_1}}_{s,x_1} & \text{for all } (s,x_1)\in S\times X_1 \\
& q_{s,*,x_2,*} = b^{\mathbf{x_2}}_{s,x_2} & \text{for all } (s,x_2)\in S\times X_2 \\
& q_{s,*,*,x_3} = b^{\mathbf{x_3}}_{s,x_3} & \text{for all } (s,x_3)\in S\times X_3 \\
& q_{s,x_1,x_2,x_3} \geq 0 & \text{for all } (s,x_1,x_2,x_3)\in S\times X_1\times X_2\times X_3.
\end{array} \tag{5.31}$$

The exponential formulation of (5.31) is expressed via the following Exponential Cone Program where the variables are $r,p\in\mathbb{R}^{S\times X_2\times X_3}$ and $q\in\mathbb{R}^{S\times X_1\times X_2\times X_3}$.

$$\begin{array}{lll}
\text{minimize} & \displaystyle -\sum_{s,x_2,x_3} r_{s,x_2,x_3} & \\
\text{subject to} & q_{s,x_1,*,*} = b^{\mathbf{x_1}}_{s,x_1} & \text{for all } (s,x_1)\in S\times X_1 \\
& q_{s,*,x_2,*} = b^{\mathbf{x_2}}_{s,x_2} & \text{for all } (s,x_2)\in S\times X_2 \\
& q_{s,*,*,x_3} = b^{\mathbf{x_3}}_{s,x_3} & \text{for all } (s,x_3)\in S\times X_3 \\
& q_{*,*,x_2,x_3} - t_{s,x_2,x_3} = 0 & \text{for all } (s,x_2,x_3)\in S\times X_2\times X_3 \\
& (-r_{s,x_2,x_3}, -t_{s,x_2,x_3}, -q_{s,*,x_2,x_3}) \in \mathscr{K}_{\exp} & \text{for all } (s,x_2,x_3)\in S\times X_2\times X_3.
\end{array} \tag{5.32}$$

Using the same approach as in Proposition 5.2.3, the exponential cone program (5.32) is equivalent to the Convex Program (5.31). Using Proposition 5.2.5 it can be shown that strong duality hold for (5.32) and its dual.

In (5.27d) there are two optimization problems:

$$\min_{Q\in\Delta_{\mathbb{P}},\mathrm{CoI}(S;X_i;X_j|X_k)=0} \mathrm{MI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j},\mathbf{X_k}) \qquad \text{and} \qquad \min_{Q\in\Delta_{\mathbb{P}}} \mathrm{MI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j},\mathbf{X_k}).$$

The second optimization problem is the same as the one in (5.27a) and so its exponential formulation is (5.29). For any $i,j,k \in \{1,2,3\}$, the optimization problem

$$\min_{Q\in\Delta_{\mathbb{P}},\mathrm{CoI}(S;X_i;X_j|X_k)=0} \mathrm{MI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j},\mathbf{X_k})$$

is nonconvex since the constraint $\mathrm{CoI}(\mathbf{S};\mathbf{X_i};\mathbf{X_j} \mid \mathbf{X_k}) = 0$ is not affine. But using the chain rule of mutual information (3.1),

$$\begin{aligned}\mathrm{CoI}(\mathbf{S};\mathbf{X_i};\mathbf{X_j} \mid \mathbf{X_k}) &= \mathrm{MI}(\mathbf{S};\mathbf{X_i} \mid \mathbf{X_k}) + \mathrm{MI}(\mathbf{S};\mathbf{X_j} \mid \mathbf{X_k}) - \mathrm{MI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j} \mid \mathbf{X_k}) \\ \mathrm{MI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j},\mathbf{X_k}) &= \mathrm{MI}(\mathbf{S};\mathbf{X_k}) + \mathrm{MI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j} \mid \mathbf{X_k}).\end{aligned} \tag{5.33}$$

Then, the constraint $\mathrm{CoI}(\mathbf{S};\mathbf{X_i};\mathbf{X_j} \mid \mathbf{X_3}) = 0$ implies that

$$\mathrm{MI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j} \mid \mathbf{X_k}) = \mathrm{MI}(\mathbf{S};\mathbf{X_i} \mid \mathbf{X_k}) + \mathrm{MI}(\mathbf{S};\mathbf{X_j} \mid \mathbf{X_k}).$$

So, the optimization problems

$$\min_{Q\in\Delta_{\mathbb{P}},\mathrm{CoI}(S;X_i;X_j|X_k)=0} \mathrm{MI}(\mathbf{S};\mathbf{X_i},\mathbf{X_j},\mathbf{X_k}) \quad \text{and} \quad \min_{Q\in\Delta_{\mathbb{P}}} (\mathrm{MI}(\mathbf{S};\mathbf{X_i} \mid \mathbf{X_k}) + \mathrm{MI}(\mathbf{S};\mathbf{X_j} \mid \mathbf{X_k}))$$

are equivalent.

Hence what is left to formulate (5.27d) as Exponential Cone Programming is to formulate $\min_{Q\in\Delta_{\mathbb{P}}} (\mathrm{MI}(\mathbf{S};\mathbf{X_i} \mid \mathbf{X_k}) + \mathrm{MI}(\mathbf{S};\mathbf{X_j} \mid \mathbf{X_k}))$ as Exponential Cone Programming. Using the chain rule for mutual information:

$$\mathrm{MI}(\mathbf{S};\mathbf{X_i} \mid \mathbf{X_k}) = \mathrm{H}(\mathbf{S} \mid \mathbf{X_i}) - \mathrm{H}(\mathbf{S} \mid \mathbf{X_i},\mathbf{X_k}).$$

Since $\mathrm{H}(\mathbf{S} \mid \mathbf{X_i})$ is constant on $\Delta_{\mathbb{P}}$, it is left to solve the following,

$$\min_{Q\in\Delta_{\mathbb{P}}} -\mathrm{H}(\mathbf{S} \mid \mathbf{X_i},\mathbf{X_k})$$

which is exponentially formulated as (5.31). Similarly for $\mathrm{MI}(\mathbf{S};\mathbf{X_j} \mid \mathbf{X_k})$.

Hence the Chicharro trivariate PID measure can be computed using Exponential Cone Programming. Due to the similarity of constraints and objective functions of the required Exponential Cone Programs for this trivariate PID measure and BROJA PID measure, BROJA_2PID can be extended to compute trivariate PID.

5.4. Computational Results

Due to the need in the scientific computing community to have a reliable easily usable software for computing the BROJA bivariate PID, BROJA_2PID was made available on GITHUB[1] as a PYTHON implementation of the Exponential Cone Programming that is explained in Section 5.2.2.

[1] github.com/Abzinger/BROJA_2PID/.

5.4.1. Data

In what follows, the main set of instances for which testing the efficiency of the solver is presented. It has three subsets of instances where each one of them is useful for an aspect of efficiency when the solver is used against large systems. It has been used in testing other solver which computes BROJA PID measure [4]. This set of instances had many hard distributions in the sense that the solution lies on the boundary of the feasible region of (CP).

This set of instances is based on the joint distributions of $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ sampled uniformly at random over the probability simplex, i.e., each joint distribution is a random vector p of size $|X||Y||Z|$ such that

$$0 \leq p_{x,y,z} \leq 1 \text{ and } \sum_{x,y,z \in X \times Y \times Z} p_{x,y,z} = 1$$

where $p_{x,y,z}$ is the probability of obtaining $(\mathbf{X} = x, \mathbf{Y} = y, \mathbf{Z} = z)$. In particular, in the experiments, the random variables $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ are sampled from the sets $X := \mathbb{Z}_{n_x}, Y := \mathbb{Z}_{n_y}, Z := \mathbb{Z}_{n_z}$ respectively. The distributions are divided into three different sets of the joint distributions depending on the size of X, Y, and Z as follows.

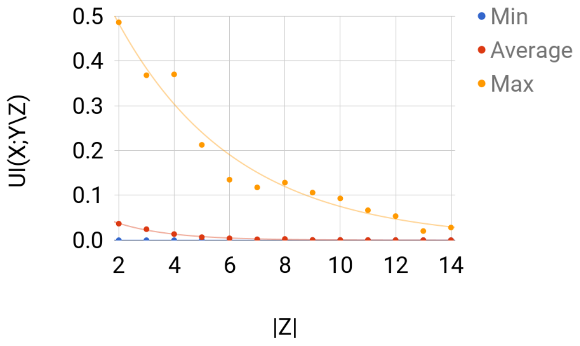
a) For set 1, $n_x = n_y = 2$, i.e., $X = Y = \{0, 1\}$ and n_z varies in $\{2, 3, \ldots, 14\}$. Then, for each n_z, 500 joint distribution of $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ are sampled uniformly at random over the probability simplex.
b) For set 2, $n_x = n_z = 2$ and n_y varies in $\{2, 3, \ldots, 14\}$. Then, for each n_y, 500 joint distribution of $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ are sampled uniformly at random over the probability simplex.
c) For set 3, $n_x = n_y = n_z = s$ where $s \in \{8, 9, \ldots, 18\}$. Then, for each s, 500 joint distribution of $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ are sampled uniformly at random over the probability simplex.

Note that in each set, instances are grouped according to the varying value, i.e., n_y, n_z, and s respectively.

5.4.2. Results

In what follows for each of the sets, $\mathrm{UI}(\mathbf{X}; \mathbf{Y} \backslash \mathbf{Z})$ is used to validate the solution, in addition, feasibility gap and duality gap are used to examine the quality of the solution–see details below, and the running time is used to analyze the efficiency of the solver. The feasibility shows how much the constraints are violated, whereas, the duality gap tells how far each of the obtained PID quantities from the actual PID quantities. Recall from Chapter 3 that $\mathrm{UI}(\mathbf{X}; \mathbf{Y} \backslash \mathbf{Z})$ is the quantity of unique information that a random variable $\mathbf{Y}$ has about $\mathbf{X}$ in some complex system $(\mathbf{X}; \mathbf{Y}, \mathbf{Z})$.

Validation. Sets 1 and 2 are mainly used to validate the solution of the estimator BROJA_2PID. For set 1, when $|Z|$ is considerably larger than $|Y|$, the amount of unique information that $\mathbf{Y}$ has about $\mathbf{X}$ is more likely to be small for any sampled joint distribution. So for set 1, the average $\mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z})$ is expected to decrease as the size of Z increases. Whereas for set 2, $\mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z})$ is expected to increase as the size of Y increases, i.e., when $|Y|$ is considerably larger that $|Z|$. BROJA_2PID shows such behavior of $\mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z})$ on the instances of sets 1 and 2 see Figures 4.



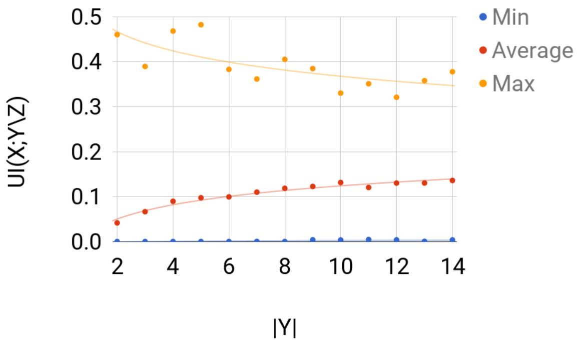


Figure 4. For each group of instances in set 1 (left) and set 2 (right), the figure shows: the instance with the largest $\mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z})$, the average value of $\mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z})$ for the instances, and the instance with the smallest $\mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z})$.

Quality. Before discussing the quality aspect, what is meant by quality of the solution is explained. In Definition 5.1.1, for strong duality, the solution needed to be feasible and duality gap to be zero. But the solver uses primal-dual interior point methods which find the solution that is ε-far from the actual optimal solution. In addition, the variables are in $\mathbb{R}$, i.e., computations have to deal with real numbers. Since computers represent real numbers up to floating precisions, the feasibility (primal and dual) or the value of the duality gap is going to depend on ε and the floating precision. As mentioned before, the duality gap can be used as a reference for the PID precision.

For the remainder of the section the maximum violation of, primal feasibility, dual feasibility, and duality gap is referred to as the maximum numerical error.

The solver did well in most of the instances. The percentage of solved instances to ε-optimality was at least 99.8% for each size in any set of instances. The term ε-optimality means that solution returned is ε-far form the actual opti-

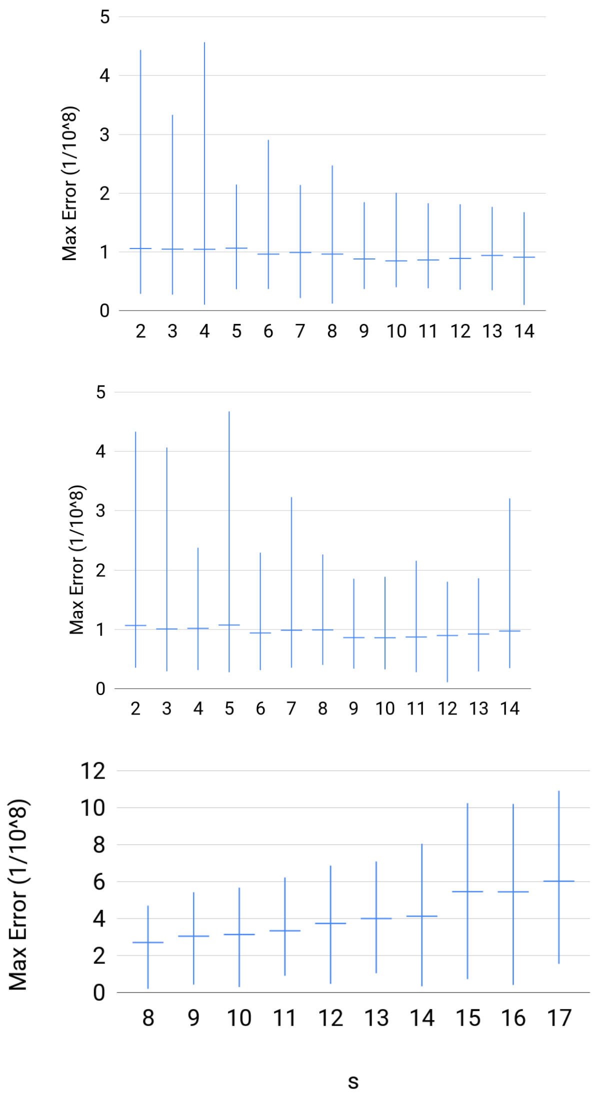


Figure 5. For each group of instances in set 1 (left), set 2 (right), and 3 (bottom center) the figure shows: the instance with the largest ε, the average value of ε for the instances, and the instance with the smallest ε; where ε is the maximum numerical error.

mal solution. In BROJA_2PID, ε was of order 10^{-8} as well as the ratio of ε to the primal and dual objective values should be of order 10^{-8}.

Figures 5 plot the successfully solved instances against the maximum numerical error. On one hand, these plots show that whenever an instance is solved successfully the quality of the solution is good. On the other hand, it is noticed that, for any instance for which the Cone Programming solver failed to find an optimal solution, the duality gap was very large and the Cone Programming solver terminated since the search direction became too small. This suggests that the solution of the system that Newton's method solves was not changing. In this case, since not much can be done with respect to the optimization, an ad-hoc fix is to the pick an input distribution close to the original one. Note that the author tried this ad-hoc fix on the unsuccessful instances and was able to either retrieve the optimal solution or at least a feasible solution with better duality gap.

Thus, these results reflect the reliability of the solution returned by the estimator BROJA_2PID. Note that even when BROJA_2PID fails to solve an instance to ε-optimality, it will return a solution[2] which is based on the last triplet (r, p, q) before the solver halted. This means that even when BROJA_2PID fails to solve an instance to ε-optimality, it provides the users with a piece of information which is the solution in this case. For example, such a solution can be regarded as an inaccurate solution if the duality gap was large.

Efficiency. Set 3 is used in order to test the efficiency of BROJA_2PID in the sense of running time. The reason is that set 1 and 2 are small-scale systems. Whereas, set 3 has a large input size mimicking large-scale systems. Testing set 3 instances also reveals how the solver empirically scales with the size of the input. Figure 6 shows that the running time for BROJA_2PID solver against large instances was below 50 minutes. Furthermore, the solver has a scaling of $(|X| \times |Y| \times |Z|)^2$, so on set 3, it scales as N^2 where N is the size of input for the sampled distributions such that $n_x = n_y = n_z = s = \sqrt[3]{N}$.

[2]BROJA_2PID raise an exception if the conic optimization solver fails to return a solution.

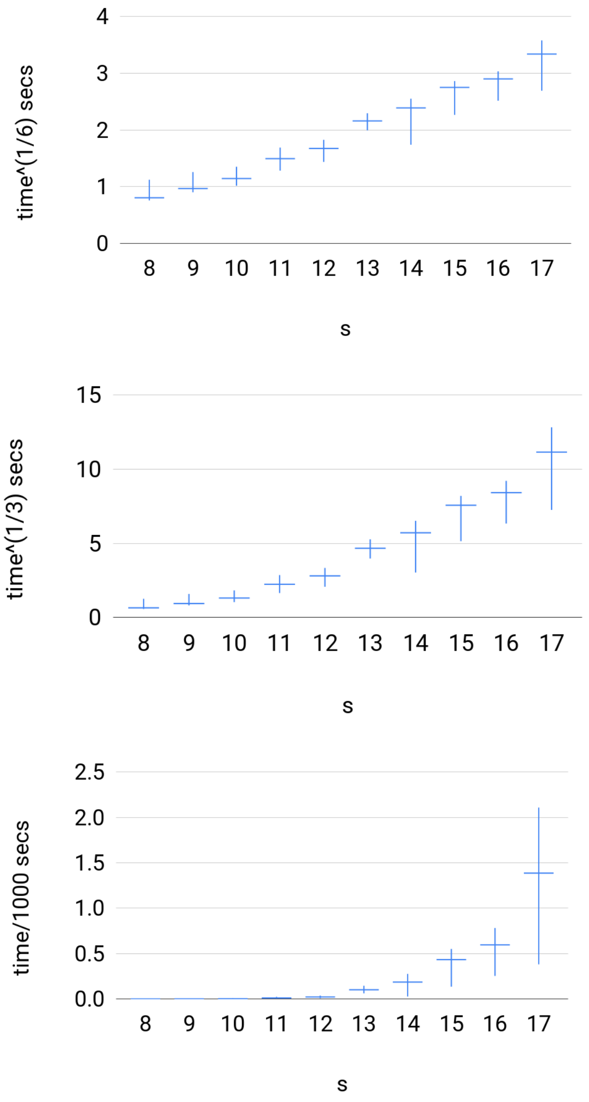


Figure 6. For each group of instances in set 3 the figure shows: the slowest instance, the average value of running times, and the fastest instance; where the running time of BROJA_2PID, t (secs), is scaled to $t^{1/6}$ (top left), $t^{1/3}$, (top right) and non-scaled (bottom center).

6. OPTIMIZING PARTIAL INFORMATION MEASURES

Chapter 1 Section 1.2 describes a problem where optimization of several information decomposition measures over a constraint set of probability distribution is needed. Unfortunately, the functions which are minimized inside the BROJA Convex Program are not convex on the whole probability simplex. This hints that sensitivity analysis of the above optimization problems will not be as informative as it would have been if the problems were convex optimization problems.

This chapter presents to some extent useful data to understand how the objective value changes when parameters of the above optimization problems are changed. It derives the sub-/super-gradients (and local sub-/super-gradients when differentiable) of information decomposition measures when optimized over a constrained set of probability distributions and presents a standard optimization problem to compute the extractable shared information.

6.1. BROJA PID Quantities: Functions of Distributions

This section studies the properties of BROJA PID quantities: shared, unique, and synergistic quantities when considered as functions of probability distributions. It derives important results which are needed whenever these functions are optimized over a constrained set of probability distributions. It starts off by describing the notation and terminology used in the remainder of the chapter.

6.1.1. Terminology and notation

Consider the random variables $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ with the joint probability distribution p. Since the BROJA PID quantities are treated directly as functions of p rather than $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$, the following shorthand will be conveniently used:

$$\begin{aligned}
\mathrm{SI}(p) &= \mathrm{SI}(\mathbf{X};\mathbf{Y},\mathbf{Z})\\
\mathrm{CI}(p) &= \mathrm{CI}(\mathbf{X};\mathbf{Y},\mathbf{Z})\\
\mathrm{UIy}(p) &= \mathrm{UI}(\mathbf{X};\mathbf{Y}\backslash\mathbf{Z})\\
\mathrm{UIz}(p) &= \mathrm{UI}(\mathbf{X};\mathbf{Z}\backslash\mathbf{Y})
\end{aligned}$$

and the common shorthand $[n] := \{1,\dots,n\}$ as well. For a (finite) set X of size n, the *probability simplex* is denoted by

$$\mathbb{\Delta}^X := \{p \in \mathbb{R}^n_+ \mid p_* = 1\}$$

and so, a probability distribution on a set X, is a vector in $\mathbb{\Delta}^X$.

Definition 6.1.1. Let $\mathbf{X}$ be a random variable. The range $\mathrm{Rg}(\mathbf{X})$ of a random variable $\mathbf{X}$ is the set

$$\mathrm{Rg}(\mathbf{X}) := \{x \mid \mathbb{P}(\mathbf{X} = x) > 0\}.$$

If a range of a random variable exists, then it is unique. If the range of a random variable exists and is finite, then the random variable is said to have finite range. All random variables considered in this chapter have finite range (unless explicitly stated otherwise). If $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ are random variables with the joint distribution p, then let

$$\mathrm{Rg}_0(p) := \{x \in X \mid p_{x,*,*} > 0\}.$$

such that X is the domain of the random variable $\mathbf{X}$ and so $\mathrm{Rg}(\mathbf{X}) = \mathrm{Rg}_0(p)$.

6.1.2. Derivatives of PID Quantities

The Convex Program (CP) which computes the BROJA PID measure can be written as a function of the input distribution $\mathbb{P}$. The function $M(p)$ is defined as follows:

$$\begin{aligned}
&\text{maximum} && h(q) && \text{over } q \in \mathbb{R}^{X \times Y \times Z} && \text{(6.1a)}\\
&\text{subject to} && q_{x,y,*} = p_{x,y,*} && \text{for all } (x,y) \in X \times Y && \text{(6.1b)}\\
& && q_{x,*,z} = p_{x,*,z} && \text{for all } (x,z) \in X \times Z && \text{(6.1c)}\\
& && q_{x,y,z} \geq 0 && \text{for all } (x,y,z) \in X \times Y \times Z. && \text{(6.1d)}
\end{aligned}$$

where $h(q)$ is the conditional entropy $H_Q(\mathbf{X} \mid \mathbf{Y}, \mathbf{Z})$ such that

$$\begin{aligned}
p_{x,y,z} &= \mathbb{P}(\mathbf{X} = x, \mathbf{Y} = y, \mathbf{Z} = z)\\
q_{x,y,z} &= Q(\mathbf{X} = x, \mathbf{Y} = y, \mathbf{Z} = z).
\end{aligned}$$

Recall the optimality conditions of (CP) derived in Chapter 4 – see Corollary 4.2.2,

Proposition 6.1.1. *A feasible point q is an optimal solution to* (6.1)*, if and only if there exist $\lambda \in \mathbb{R}^{X \times Y}$ and $\mu \in \mathbb{R}^{X \times Z}$ satisfying the following:*

(a) For all $(y,z) \in Y \times Z$ with $q_{*,y,z} > 0$:

$$\lambda_{x,y} + \mu_{x,z} = \ln\left(\frac{q_{x,y,z}}{q_{*,y,z}}\right) \qquad \textit{holds for all } x \in X;$$

(b) For all $(y,z) \in Y \times Z$ with $q_{*,y,z} = 0$, there is a probability distribution ρ with support X such that

$$\lambda_{x,y} + \mu_{x,z} \leq \ln(\rho_x^{y,z}) \qquad \textit{holds for all } x \in X.$$

If q, λ, μ are as in the proposition, then λ, μ are called the Lagrange multipliers certifying optimality. The following lemma uses the Lagrange multipliers certifying optimality to formulate the gradient of $M(p)$ when it is differentiable and derives its super-gradients otherwise.

Lemma 6.1.1. *Suppose p has full support*[1]*. Let q be an optimal solution of* (6.1)*, and let λ, μ be Lagrange multipliers certifying optimality.*

[1] $\mathrm{Rg}(\mathbf{X}, \mathbf{Y}, \mathbf{Z}) = |X \times Y \times Z|$.

(a) *If $q_{x,y,z} > 0$ for all $(x,y,z) \in X \times Y \times Z$, then M is differentiable in p, and we have*

$$\partial_{x,y,z} M\big(p \big) = -\lambda_{x,y} - \mu_{x,z} \tag{6.2}$$

(b) *In any case, the vector defined by*

$$\mathfrak{g}(p)_{x,y,z} := -\lambda_{x,y} - \mu_{x,z} \tag{6.3}$$

is a super-gradient on M in the point p.

Proof. Since q is the optimal solution of (6.1), then

$$M(p) = \max_q h(q) = -\min_q -h(q) = h(q)$$

where $h(q) = H(\mathbf{X} \mid \mathbf{Y}, \mathbf{Z})$. So, the gradient of M in p is

$$\nabla M(p) = -\left(\ln \left(\frac{q_{x,y,z}}{q_{*,y,z}} \right) \right)_{x,y,z} \tag{6.4}$$

If $q_{x,y,z} > 0$ for all $(x,y,z) \in X \times Y \times Z$, then $q_{*,y,z} > 0$ for all $(y,z) \in Y \times Z$ and so inverse function theorem shows that M is differentiable as a function of p. Moreover, Equation (6.2) follows from the fact that q, λ, μ are as in Proposition 6.1.1 and the gradient defined in (6.4).

From Proposition 4.2.1 and Proposition 6.1.1, it follows that $\lambda_{x,y} + \mu_{x,z}$ is a sub-gradient to $W(p) := \min_q -h(q)$ subject to the constraints (6.1b), (6.1c), and (6.1d) in the point p. Hence $-\lambda_{x,y} - \mu_{x,z}$ is a super-gradient on M in the point p. □

It is important to emphasize that, in this lemma as well as in the following results, the condition that p has full support is only there to simplify notation, and can be readily abandoned. The following theorem uses the (super-) gradient of $M(p)$ to provide the gradients of the BROJA PID quantities along with their corresponding sub-gradients or super-gradients when they are not smooth.

Theorem 6.1.1. *Suppose p has full support. Let q be an optimal solution of* (6.1), *and let λ, μ be Lagrange multipliers certifying optimality.*

(a) *If $q_{x,y,z} > 0$ for all $(x,y,z) \in X \times Y \times Z$, then* CI, SI, UIy, UIz *are all differentiable in p, and we have*

$$\partial_{x,y,z} \mathrm{CI}(p) = \ln \left(\frac{p_{*,y,z}}{p_{x,y,z}} \right) - \lambda_{x,y} - \mu_{x,z} \tag{6.5a}$$

$$\partial_{x,y,z} \mathrm{SI}(p) = -1 + \ln \left(\frac{p_{x,y,*} p_{x,*,z}}{p_{x,*,*} p_{*,y,*} p_{*,*,z}} \right) - \lambda_{x,y} - \mu_{x,z} \tag{6.5b}$$

$$\partial_{x,y,z} \mathrm{UIy}(p) = \ln \left(\frac{p_{*,*,z}}{p_{x,*,z}} \right) + \lambda_{x,y} + \mu_{x,z} \tag{6.5c}$$

$$\partial_{x,y,z} \mathrm{UIz}(p) = \ln \left(\frac{p_{*,y,*}}{p_{x,y,*}} \right) + \lambda_{x,y} + \mu_{x,z} \tag{6.5d}$$

(b) *In any case, the vectors defined by*

$$\mathfrak{g}_{\mathrm{CI}}(p)_{x,y,z} = \ln\left(\frac{p_{*,y,z}}{p_{x,y,z}}\right) - \lambda_{x,y} - \mu_{x,z} \tag{6.6a}$$

$$\mathfrak{g}_{\mathrm{SI}}(p)_{x,y,z} = -1 + \ln\left(\frac{p_{x,y,*}p_{x,*,z}}{p_{x,*,*}p_{*,y,*}p_{*,*,z}}\right) - \lambda_{x,y} - \mu_{x,z} \tag{6.6b}$$

are local super-gradients of CI *and* SI *respectively and the vectors defined by*

$$\mathfrak{g}_{\mathrm{UIy}}(p)_{x,y,z} = \ln\left(\frac{p_{*,*,z}}{p_{x,*,z}}\right) + \lambda_{x,y} + \mu_{x,z} \tag{6.7a}$$

$$\mathfrak{g}_{\mathrm{UIz}}(p)_{x,y,z} = \ln\left(\frac{p_{*,y,*}}{p_{x,y,*}}\right) + \lambda_{x,y} + \mu_{x,z} \tag{6.7b}$$

are local sub-gradients of UIy *and* UIz *in the point p respectively.*

Proof. For 6.1.1, Bertschinger et al. in [7] defined the partial information decomposition as follows:

$$\begin{aligned}
\mathrm{CI}(p) &= \mathrm{MI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) - \min_q \mathrm{MI}(\mathbf{X};\mathbf{Y},\mathbf{Z})\\
\mathrm{SI}(p) &= \max_q \mathrm{CoI}(\mathbf{X};\mathbf{Y},\mathbf{Z})\\
\mathrm{UIy}(p) &= \min_q \mathrm{MI}(\mathbf{X};\mathbf{Y} \mid \mathbf{Z})\\
\mathrm{UIz}(p) &= \min_q (\mathbf{X};\mathbf{Z} \mid \mathbf{Y})
\end{aligned}$$

where the optimization is subject to the constraints (6.1b), (6.1c), and (6.1d). Using the definition of $\mathrm{MI}(\mathbf{X};\mathbf{Y},\mathbf{Z})$ and the chain rule of entropy and mutual information, the following holds:

$$\begin{aligned}
\mathrm{CI}(p) &= \mathrm{MI}(\mathbf{X};\mathbf{Y},\mathbf{Z}) - \min_q \mathrm{MI}(\mathbf{X};\mathbf{Y},\mathbf{Z})\\
&= -H(\mathbf{X} \mid \mathbf{Y},\mathbf{Z}) + M(p).\\
\mathrm{SI}(p) &= \max_q \mathrm{CoI}(\mathbf{X};\mathbf{Y};\mathbf{Z})\\
&= \mathrm{MI}(\mathbf{X};\mathbf{Y}) - H(\mathbf{X} \mid \mathbf{Z}) + M(p)\\
\mathrm{UIy}(p) &= \min_q \mathrm{MI}(\mathbf{X};\mathbf{Y} \mid \mathbf{Z}).\\
&= H(\mathbf{X} \mid \mathbf{Z}) - M(p).\\
\mathrm{UIz}(p) &= \min_q \mathrm{MI}(\mathbf{X};\mathbf{Z} \mid \mathbf{Y})\\
&= H(\mathbf{X} \mid \mathbf{Y}) - M(p).
\end{aligned}$$

By direct computations, the equations in 6.1.1 follow from Lemma 6.1.1:

$$\partial_{x,y,z} M(p) = -\lambda_{x,y} - \mu_{x,z}.$$

For (b), let

$$\begin{aligned} g_{\mathrm{CI}}(p) &= H(\mathbf{X} \mid \mathbf{Y},\mathbf{Z}) \\ g_{\mathrm{SI}}(p) &= \mathrm{MI}(\mathbf{X};\mathbf{Y}) - H(\mathbf{X} \mid \mathbf{Z}) \\ g_{\mathrm{UIy}}(p) &= \mathrm{MI}(\mathbf{X};\mathbf{Z}) + H(\mathbf{X}) \\ g_{\mathrm{UIz}}(p) &= \mathrm{MI}(\mathbf{X};\mathbf{Y}) + H(\mathbf{X}). \end{aligned} \tag{6.8}$$

Since p has a full support then all the functions in (6.8) are differentiable and

$$\begin{aligned} g'_{\mathrm{CI}}(p,d) &= \sum_{x,y,z} \ln\left(\frac{p_{*,y,z}}{p_{x,y,z}}\right) d_{x,y,z} \\ g'_{\mathrm{SI}}(p,d) &= \sum_{x,y,z} \left(\ln\left(\frac{p_{x,y,*}p_{x,*,z}}{p_{x,*,*}p_{*,y,*}p_{*,*,z}}\right) - 1\right) d_{x,y,z} \\ g'_{\mathrm{UIy}}(p,d) &= \sum_{x,y,z} \ln\left(\frac{p_{*,*,z}}{p_{x,*,z}}\right) d_{x,y,z} \\ g'_{\mathrm{UIz}}(p,d) &= \sum_{x,y,z} \ln\left(\frac{p_{*,y,*}}{p_{x,y,*}}\right) d_{x,y,z} \end{aligned} \tag{6.9}$$

where $d \in \mathbb{R}^{X\times Y\times Z}$. From Lemma 6.1.1 and 2.1.1, $\mathfrak{g}(p)$ is a super-gradient of M at p and for any $d \in \mathbb{R}^{X\times Y\times Z}$, we have $-M'(p,d) \geq -\mathfrak{g}^T d$. Hence, the vectors defined by (6.6a) and (6.6b) are super-gradients of CI and SI respectively and the vectors defined by (6.7a) and (6.7b) are local sub-gradients of UIy and UIz in the point p respectively. □

Corollary 6.1.1. *Let I be any of* CI, SI, UIy, UIz. *At the points where I is not smooth it is*

(a) concave, in the case of $I =$ CI, SI*;*

(b) convex, in the case of $I =$ UIy, UIz.

Proof. Using the result (a) of Theorem 6.1.1, the vectors $\mathfrak{g}_{\mathrm{CI}}(p)$ and $\mathfrak{g}_{\mathrm{SI}}(p)$ are local super-gradients of CI and SI and the vectors $\mathfrak{g}_{\mathrm{UIy}}(p)$ and $\mathfrak{g}_{\mathrm{UIz}}(p)$ are local sub-gradients of UIy and UIz in the point p. From this, the statements in this Corollary follow. □

6.2. Application: Extractable Shared Information

The concept of extractable shared information was introduced by Rauh et al. [60]. They studied the properties of extractable shared information measure aiming to check whether it can be considered as a new measure for shared information. This section is focused on how to compute this extractable shared information rather than discussing it as a measure of shared information.

6.2.1. Extractable Shared Information Measures

Let $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ be random variables with distribution $p \in \mathbb{A}^{X \times Y \times Z}$ Rauh et al. [60] define *extractable shared information* of $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ as

$$\mathrm{SI}^{\mathrm{ext}}(p) := \sup_f \mathrm{SI}(f(\mathbf{X}); \mathbf{Y}, \mathbf{Z}) \tag{6.10}$$

where the supremum is taken over all functions $f\colon X \to T$ such that X is the range of $\mathbf{X}$ and T is an arbitrary finite set.

They generalized the definition of extractable shared information by looking at "probabilistic extractability" rather than "deterministic extractability", i.e., replacing f by a stochastic matrix. So, the *probabilistically extractable shared information* is defined as

$$\mathrm{SI}^{\mathrm{prext}}(p) := \sup_{\mathbf{T}} \mathrm{SI}(\mathbf{T}; \mathbf{Y}, \mathbf{Z}) \tag{6.11}$$

where the supremum is taken over all random variables $\mathbf{T}$ (with finite range) which are conditionally independent of $\mathbf{Y}, \mathbf{Z}$ given $\mathbf{X}$.

6.2.2. Reformulation of $\mathrm{SI}^{\mathrm{ext}}$ and $\mathrm{SI}^{\mathrm{prext}}$

The optimization problems (6.10) and (6.11) are not concave as well as (6.11) being an infinite dimensional problem. This subsection aims to reformulate the two problems so that they are optimized over the set of stochastic matrices.

For a set R and a $m \in \mathbb{N}$, a *column stochastic* $([m] \times R)$*-matrix* is a matrix Π with m rows (indexed $1, \ldots, m$ as usual) and columns indexed by the elements of R, whose entries are nonnegative reals such that $\Pi_{*,r} = 1$ for all r. Let p be a probability distribution on $X \times Y \times Z$, and Π be a stochastic $([m] \times X)$-matrix. Then define the probability distribution $\Pi(p)$ as follows:

$$\Pi(p)_{t,y,z} := \sum_{x \in \mathrm{Rg}_0(p)} \Pi_{t,x} p_{x,y,z}, \text{ for all } t \in [m] \text{ and } (y,z) \in Y \times Z.$$

Reformulation of $\mathrm{SI}^{\mathit{ext}}$. Consider the random variables $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ with distribution p. Forcing some *integrality constraints*, i.e,

$$\Pi_{t,x} \in \mathbb{Z} \qquad \text{for all } (t,x) \in [m] \times X$$

together with the nonnegativity inequalities, i.e.,

$$\Pi_{t,x} \geq 0 \qquad \text{for all } (t,X) \in [m] \times X$$

and restricting to

$$\Pi_{*,x} = 1 \qquad \text{for all } x \in X$$

have precisely the effect of ensuring that for every x in the range of $\mathbf{X}$ there exists a unique $t \in [m]$ with $\Pi_{t,x} = 1$. In other words, Π defines a mapping from $\mathrm{Rg}(\mathbf{X})$ to

$[m]$. Finally, since m is the size of the range of $\mathbf{X}$, then restricting to m to $|X|$ simply optimizes over all functions defined on the range of $\mathbf{X}$, which is exactly (6.10).

Hence, it is straightforward that the extractable shared information of p is the value of the following optimization problem:

$$\begin{aligned}
&\text{With } m := |X| : \\
&\quad \mathrm{SI}^{\mathrm{ext}}(p) := \max \mathrm{SI}(\Pi(p)) && (6.12a)\\
&\qquad \text{over } \Pi \in \mathbb{R}^{[m]\times X} \\
&\quad \text{subject to} \\
&\qquad\quad \Pi_{*,x} = 1 \qquad \text{for all } x \in X && (6.12b)\\
&\qquad\quad \Pi_{t,x} \geq 0 \qquad \text{for all } (t,X) \in [m]\times X && (6.12c)\\
&\qquad\quad \Pi_{t,x} \in \mathbb{Z} \qquad \text{for all } (t,x) \in [m]\times X. && (6.12d)
\end{aligned}$$

The optimization problem (6.12) has only a finite number of solutions which is equal to the number of partitions of a finite set X, i.e., the Bell number of X. Thus, it is possible to solve the optimization problem by enumerating all solution (in a clever way). But since the Bell numbers increase super-exponentially, for large sets X, the latter might not be feasible.

Reformulation of $\mathrm{SI}^{\boldsymbol{prext}}$. The probabilistically extractable shared information is the value of the following optimization problem:

$$\begin{aligned}
&\qquad \sup \mathrm{SI}(\Pi(p)) && (6.13a)\\
&\text{over } m \geq |X| && (6.13b)\\
&\qquad \Pi \in \mathbb{R}^{[m]\times X} \\
&\text{subject to} \\
&\qquad \Pi_{*,x} = 1 \qquad \text{for all } x \in X && (6.13c)\\
&\qquad \Pi_{t,x} \geq 0 \qquad \text{for all } (t,x) \in [m]\times X. && (6.13d)\\
& && (6.13e)
\end{aligned}$$

Lemma 6.2.1. *The optimization problem* (6.13) *is equivalent to the definition* (6.11).

Proof. Let $\Pi \in \mathbb{R}^{[m]\times X}$ such that it satisfies (6.13c) and (6.13d). Consider the relation

$$\Pi_{t,x} = \mathbb{P}(\mathbf{T} = t \mid \mathbf{X} = x). \qquad (6.14)$$

Let $\Pi(p)$ be the probability distribution of $(\mathbf{T},\mathbf{Y},\mathbf{Z})$ such that

$$\Pi(p)_{t,y,z} := \sum_{x\in \mathrm{Rg}_0(p)} \Pi_{t,x} p_{x,y,z}, \text{ for all } t \in [m] \text{ and } (y,z) \in Y \times Z.$$

Now when condition on $\mathbf{X} = x$,

$$\Pi(p)_{t,y,z} := \mathbb{P}(\mathbf{T} = t) p_{*,y,z}, \text{ for all } t \in [m] \text{ and } (y,z) \in Y \times Z.$$

which means that $\mathbf{T}$ is conditionally independent of $\mathbf{Y}$ and $\mathbf{Z}$ given $\mathbf{X}$. Since m is the size of the domain of $\mathbf{T}$, the optimization problem (6.13) computes (6.11) where the supremum is taken over all random variables $\mathbf{T}$ (with finite range) which are conditionally independent of $\mathbf{Y},\mathbf{Z}$ given $\mathbf{X}$.

On the other hand, given a random variable $\mathbf{T}$ conditionally independent of $\mathbf{X}_1,\dots,\mathbf{X}_k$ given $\mathbf{X}$ and set $m := \max |X|$. Then relation (6.14) implies

$$\begin{aligned}\mathbb{P}(\mathbf{T}=t,\mathbf{Y}=y,\mathbf{Z}=z) &= \sum_{x\in X}\mathbb{P}(\mathbf{T}=t,\mathbf{Y}=y,\mathbf{Z}=z \mid \mathbf{X}=x)\cdot\mathbb{P}(\mathbf{X}=x)\\ &= \sum_{x\in X}\mathbb{P}(\mathbf{T}=t \mid \mathbf{X}=x)\cdot\mathbb{P}(\mathbf{Y}=y,\mathbf{Z}=z \mid \mathbf{X}=x)\cdot\mathbb{P}(\mathbf{X}=x)\\ &= \sum_{x\in X}\mathbb{P}(\mathbf{T}=t \mid \mathbf{X}=x)\cdot\mathbb{P}(\mathbf{X}=x,\mathbf{Y}=y,\mathbf{Z}=z).\end{aligned}$$

Then any matrix $\Pi\in\mathbb{R}^{[m]\times X}$ with the conditions (6.13c) and (6.13d) defines a probability distribution $\Pi(p)$ of $(\mathbf{T},\mathbf{Y},\mathbf{Z})$. Hence, the optimization problem (6.11) where the supremum is taken over all random variables $\mathbf{T}$ (with finite range) which are conditionally independent of $\mathbf{Y},\mathbf{Z}$ given $\mathbf{X}$ computes (6.13). □

6.2.3. Bounds on $\mathrm{SI}^{\mathbf{ext}}$ and $\mathrm{SI}^{\mathbf{prext}}$

This subsection provides a standard optimization problem which is an upper bound of (6.10) and a lower bound of (6.11).

There are two significant differences between the (6.12) and (6.13). Firstly, (6.13) lacks the integrality constraints, making it a continuous optimization problem. Secondly, the dimension, m, is a variable, making the optimization problem infinite dimensional (as observed in [60]), and thus basically intractable from an algorithmic point of view. (The lower bound $m\geq|X|$ is redundant, see Lemma 6.2.2 below).

The following optimization problem, however, is a standard continuous optimization problem to which we can apply our results: For a fixed value of $m\in\mathbb{N}$, define

$$\begin{aligned}&\mathrm{SI}^{\clubsuit}_m(p) := \max \mathrm{SI}(\Pi(p)) &&(6.15a)\\ &\qquad\text{over } \Pi\in\mathbb{R}^{[m]\times X}\\ &\text{subject to}\\ &\qquad\qquad \Pi_{*,x}=1 \qquad \text{for all } x\in X &&(6.15b)\\ &\qquad\qquad \Pi_{t,x}\geq 0 \qquad \text{for all } (t,x)\in[m]\times X. &&(6.15c)\\ & &&(6.15d)\end{aligned}$$

Lemma 6.2.2. *The sequence $m\mapsto \mathrm{SI}^{\clubsuit}(m)$ is non-decreasing and for every fixed $m_0\geq|X|$,*

$$\mathrm{SI}^{ext}(p)\leq \mathrm{SI}^{\clubsuit}_{m_0}(p)\leq \sup_{m\geq 0}\mathrm{SI}^{\clubsuit}_m(p) = \mathrm{SI}^{prext}(p).$$

Proof. For any positive integers $m_1 \leq m_2$, $\mathbb{R}^{[m_1]\times X} \subset \mathbb{R}^{[m_2]\times X}$ and so the feasible region of (6.15) with fixed value m_1 is contained in that of (6.15) with fixed value m_2. Thus, the sequence $m \mapsto \mathrm{SI}^{\clubsuit}_m(p)$ is nondecreasing and for every fixed value $m_0 \geq |X|$

$$\mathrm{SI}^{\mathrm{ext}}(p) \leq \mathrm{SI}^{\clubsuit}_{m_0}(p) \leq \sup_{m\geq 0} \mathrm{SI}^{\clubsuit}_m(p) = \mathrm{SI}^{\mathrm{prext}}(p),$$

where the first inequality holds since the set

$$\{\Pi \in \mathbb{R}^{[m]\times X} \mid \Pi_{*,x} = 1, \Pi_{t,x} \in \mathbb{Z}_+\} \subset \{\Pi \in \mathbb{R}^{[m_0]\times X} \mid \Pi_{*,x} = 1, \Pi_{t,x} \geq 0\}.$$

□

Note that the results obtained in the previous section can be used (with minor modifications) to apply an appropriate gradient descent to compute the solution of (6.15).

7. CONTROL OF AUTOMATED VALET PARKING SYSTEMS

This chapter studies a fundamental optimization problem involved in the control of an automated car parking system. It studies the theoretical throughput limitations of these systems: Given a car park layout, an initial configuration of a car park (location of cars, robots), into a desired, terminal configuration, what is the optimal set of control instructions for the robots to reorganize the initial configuration into the terminal configuration. The notion *optimal* in this chapter means fastest in terms of clock-on-the-wall waiting time until the robots have completed their tasks.

With the ultimate goal of heuristic algorithms for the control of the robots, this chapter studies exact algorithms. The importance of the modeling approach in this chapter is that it took into consideration the physical properties of the robots. It proposes an Integer Programming (IP) model and a Constraint Programming model (Boolean variables with constraints in conjunctive normal form, CNF), and compares the two approaches by testing them on different parking lot scenarios.

The Chapter is based on the results obtained in [40]. Section 7.1.1, presents the problem and the basic approach to solve it and review literature relevant to the problem. Section 7.2, describes the variables used for both the IP- and CNF-model. Section 7.3 gives an overview of the IP clauses and objective function. Section 7.4 describes, in brief, the CNF clauses. Section 7.5 describes the data, the computational experiments that are run and discusses the obtained results. Section 7.6 draws some conclusions regarding the problem.

7.1. Basic Approach and Related work

This section describes the basic approach which is used in modeling the problem. Then the section is concluded by reviewing proposed simplified versions of the problem from the literature which were shown to be NP-hard.

7.1.1. Problem Data and Basic Modeling Approach

The technical details of the problem were the result of a collaboration with a company which considered entering the market of automated valet parking installations. In the considered parking lots, all slots have the same width (3m) and length (6m), and all slots are parallel: the width is in East-West direction, the length is in North-South direction. The parking lots have a rectangular bounding box: e.g., if the bounding box is 300m in wide and 600m long, the parking lot can contain up to $10\times10=100$ slots. Correspondingly, slots are identified by x (East-to-West) and y (South-to-North) coordinates. We allow for slots to be unusable, due to either obstacle (walls, pillars, broken down robots) or simply parked cars which, for some reason, we currently do not want to move.

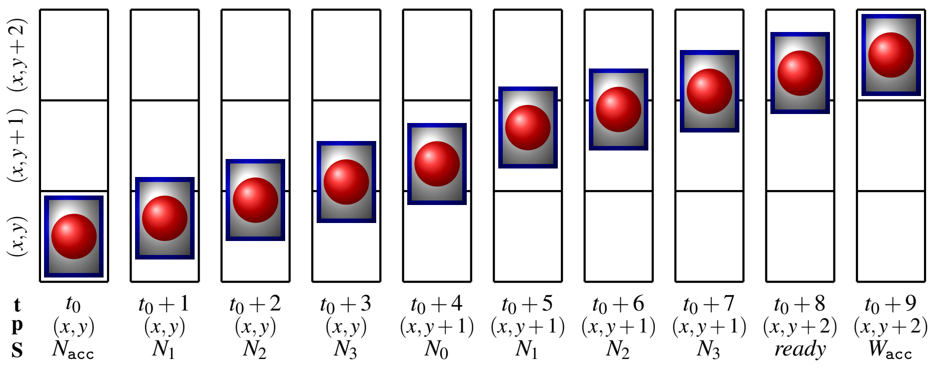

(x,y)
$(x,y+1)$
$(x,y+2)$
t t_0 t_0+1 t_0+2 t_0+3 t_0+4 t_0+5 t_0+6 t_0+7 t_0+8 t_0+9
p (x,y) (x,y) (x,y) (x,y) $(x,y+1)$ $(x,y+1)$ $(x,y+1)$ $(x,y+1)$ $(x,y+2)$ $(x,y+2)$
S N_{acc} N_1 N_2 N_3 N_0 N_1 N_2 N_3 *ready* W_{acc}

Figure 7. *North* movement of a loaded robot. **t** = time interval; **p** = position associated and **S** = stage of the move. Robots are drawn in the physical location which it occupies at the beginning of the time interval.

The robots can move either in North-South or in East-West direction (but not, e.g., diagonally); also they do not need to "rotate". A robot must come to a complete standstill before changing directions. The robots' maximum speed is 3m/s when empty, and 1.5m/s when carrying a car. In real life, robots do not reach their maximum speed from standstill in no time, they need to accelerate. The best way to model this acceleration is via probability distribution but this will make the modeling complicated. The acceleration modeling which is used is a compromise between satisfying the physical properties and the complexity of the model. The robots require 1s to accelerate to maximum speed or decelerate from maximum speed to standing still[1]. (We do not allow for a robot to standstill between two slots.) The robot needs 6s to lift a car and 2s to drop it.

This data suggests to discretize time into intervals of .25s. E.g., Fig. 7 shows a complete sequence of a North-movement by a robot carrying a car from a slot (x,y) to a slot $(x,y+2)$ and then accelerate West. In time interval t_0, the robot picks up speed and moves 1/8 of the length of a slot. In time intervals $t_0+1,\ldots,t_0+7$, the robot moves at full speed. In time interval t_0+8, it slows down and comes to a stand two places north of where it started. The control instruction "ready" causes the robot to initiate stopping. In time interval t_0+9, the robot could be executing a new control instruction; in the figure, it is accelerating West.

The discrete optimal control problem we aim to solve is now the following: Given two configurations of locations of robots and cars on the parking lot, an "initial configuration" and a "terminal configuration", determine a feasible set of control instructions for the robots which transforms the initial configuration into the terminal configuration; minimize the time the reconfiguration takes. Clearly equivalent to a set of control instructions is a sequence of configurations, each one of which can be derived from the previous one by a feasible move of the robot.

[1]It was suggested by Karl Tarbe [67].

Note that in practice, whenever a customer wants to retrieve his or her car, a new set of initial and terminal configurations are generated and the above discrete optimal control problem should be solved.

Also, in a practical application, there is usually no need to fix initial and terminal locations of each individual car. It is, e.g., not relevant whether car #47 goes to vehicle transfer station #23 and car number #54 goes to vehicle transfer station #22 — or vice versa. Hence, the model deals with "colored" cars, and the initial and terminal configurations specify only whether a slot is occupied by a car and if so, what the "color" of that car should be. In the implemented computer code, IP and CNF models are formulated and solved where these models have feasible solutions which are sequences of configurations, for 3 colors. The number of colors was fixed to three in order to distinguish between the cars which just arrived, the ones just requested, and the dormant ones.

7.1.2. Simplifications and literature review

Simplified versions of the problem have been studied theoretically. Most importantly, disregarding the role of robots (i.e., assuming that the cars move by themselves) and the speeds give variants of pebble motion and reconfiguration problems in grids, or, more generally graphs: Vertices represent parking slots, and edges represent slots sharing an edge. We now review what is known.

The most famous problem in this group is the *15-Puzzle:* on a 4×4 grid, 15 vertices are occupied by labeled pebbles. *Labeled* means that each pebble has a color of its own. The decision version of the problem is whether a given initial configuration can be transformed into a given terminal configuration through a sequence of moving pebbles to adjacent vertices; the optimization version asks for a reconfiguration with the smallest number of total moves. The decision problem can be solved in polynomial time [75]. The optimization problem on grids is NP-hard [59], but constant factor approximation algorithms exist [58].

For graphs in general, Papadimitriou et al. [57] proved that with two labels, one pebble has a unique label, the problem is NP-hard. They also gave a polynomial time algorithm, using flow techniques, for the optimal solution on trees. The result was later improved to $O(n^5)$ by Auletta et al. [3]. Pach et al. [16] showed that the optimization problem on graphs is APX-hard. Moreover, when allowing more pebbles to move at the same time, J. Yu and S. La Valle [76] proved that the optimization problem on graphs is NP-hard. It follows from [16] and [76] that there is no polynomial time approximation scheme for the simplified version of our problem, unless $\mathrm{P} = \mathrm{NP}$.

As for the practical aspect of the problem, several robot control systems problems have been modeled using Integer Programming [36] and Mixed Integer Programming [28, 38]. But their simulations were tested with small inputs, e.g. no more than 4 robots, where all robots have the same color. Similar simplified problems have been studied in path scheduling problems [17, 19, 24, 70] using Mixed

Integer Programming models. These simplifications arise in the path topology (studied only free-obstacle grids or other simpler topologies) and in the loose time limit of the problem. To the author's knowledge, there is no CNF formulation of the problem or any simplification of it even though there were some Boolean logic formulations in [38] which were transformed into inequalities.

Sections 7.2, 7.3, and 7.4 describe the variables of the IP- and CNF-model that were introduced in [40], the constraints of that IP-model, and the clauses of that CNF-model.

7.2. Variables

This section describes the variables of the IP model and CNF model. Both IP- and CNF-models have the same variables. The two models are completely new models formulated by the author among others in [40].

Let $C := \{0, 1, \ldots, |C| - 1\}$ denote the set of colors of cars. The models require an upper bound on the number of time intervals: Each time interval is identified by an element of $T = \{0, 1, \ldots, t_{\max}\}$. The configuration at $t = 0$ is the initial configuration, the configuration at $t = t_{\max}$ is the terminal configuration. A car is called *stationary* if it is not carried by a robot.

For every time interval and slot, there are four types of variables to determine what is happening there and then: *slot occupation status* (whether there is a stationary car, and if so, of what color), *slot robot-status* (whether there is a robot, and if so, what it is carrying and doing), *robot vertical process* (if there is a robot lifting or dropping a car, which phase of that process is it executing), *robot horizontal movements* (if there is a moving robot, which phase of the movement is it executing). All variables are Boolean. At most one variable in each group is active, i.e., it has value 1. In what follows, a description for each group of variables and its role is presented.

7.2.1. Slot Occupation Status

These variables inform whether a car of a certain color is occupying the slot or if it is empty. These variables are of the form $x_{v,t,c}$. For every slot $v = (x, y)$, every time interval $t = 0, \ldots, t_{\max}$, and every $c \in \{\phi\} \cup C$, we have a variable $x_{v,t,c}$. The symbol ϕ stands for "no car". Example: $x_{v,t,\phi} = 1$ iff during time interval t, there is no stationary car at the slot with coordinates v; $x_{v,t,2} = 1$ iff during time interval t, there is a stationary car of color 2 at the slot with coordinates v. Note that the case whether a slot has a robot or not is not recognized by this group of variables.

7.2.2. Slot Robot-Status (*SRS*)

These variables inform whether a certain slot is occupied by a robot, and if so, reports what the robot is carrying or doing. These variables are vital since they help in determining whether a future action is valid, but if so, they do not make the robot do the action, i.e., they are acting as consultants. These variables are of the form $s_{v,t,c,d}$. For every slot v, every time interval t, every $c \in \{\phi\} \cup C$, and every $d \in \{\varepsilon, \rho, \mu, \zeta\}$, there is a variable $s_{v,t,c,d}$. The symbol ε stands for "no robot", ρ stands for "ready", μ stands for "robot moving", ζ stands for "vertical process". Example: $s_{v,t,\phi,\rho} = 1$ iff at the slot with coordinates v, during time interval t, there is a robot executing the "ready" control instruction, i.e., it is doing one of the following: (1) nothing (being idle); (2) ending a movement (decelerating); (3) ending lifting or dropping a car. Note that the being idle case in the "ready" instructions is important since, as mentioned before, robots are not allowed to change direction unless they become standstill. Another example: $s_{v,t,2,\zeta} = 1$, iff at the slot with coordinates v, during time interval t, there which is in the process of either lifting or dropping a car of type 2. In the table below *SRS* variables are listed with description.

Variable	Description
$s_{v,t,\phi,\varepsilon}$	v has no robot occupying it.
$s_{v,t,\phi,\rho}$	v has an unloaded robot executing the "ready" control instruction.
$s_{v,t,i,\rho}$	v has a loaded robot with car of type i executing the "ready" control instruction.
$s_{v,t,\phi,\mu}$	v has an unloaded robot moving horizontally.
$s_{v,t,i,\mu}$	v has a loaded robot with car of type i which is moving horizontally.
$s_{v,t,i,\zeta}$	v has a loaded robot is in the process of either lifting or dropping a car of type i.

Table 1. List of *SRS* variables at the slot with coordinates v during time interval t.

7.2.3. Robot Vertical Movements (*RVert*)

These variables determine which phase of the lifting or dropping process the robot is executing. These variables are of the form $z_{v,t,p}$. For every slot v, every time interval t, and every $p \in \{\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \delta\}$, there is a variable $z_{v,t,p}$. The symbols λ_i for $0 \leq i \leq 4$ stand for the phases of "lifting", δ stands for "dropping". Example: $z_{v,t,\lambda_2} = 1$ iff during time interval t, the robot is in the third phase of lifting at the slot with coordinates v.

Note that the assigned phases of lifting (resp. dropping) took into consideration that the robot needs 6s (resp. 2s) to lift (resp. drop) a car. For example, lifting the process has 4 phases each takes 1s and the two remaining seconds are counted by forcing the robot to be ready before and after it finishes the lift. The following diagram shows the interconnection between x-, s-, and z-variables:

$t = t_0$	$x_{v,t,3} = 1$, $s_{v,t,\phi,\rho} = 1$, $z_{v,t,*} = 0$	a car of color 3, a robot
$t = t_0 + 1$	$x_{v,t,\phi} = 1$, $s_{v,t,3,\zeta} = 1$, $z_{v,t,\lambda_0} = 1$	no car; a robot lifting a car of color 3
$\vdots$	$\vdots$	$\vdots$
$t = t_0 + 5$	$x_{v,t,\phi} = 1$, $s_{v,t,3,\zeta} = 1$, $z_{v,t,\lambda_4} = 1$	no car; a robot lifting a car of color 3
$t = t_0 + 6$	$x_{v,t,\phi} = 1$, $s_{v,t,3,\rho} = 1$, $z_{v,t,*} = 0$	no car; robot executing "ready"

7.2.4. Robot Horizontal movements (*RMove*)

These variables determine the displacement movements of the robot. These variables are of the form $m_{v,t,c,e}$. For every slot v, every time interval t, every $c \in \{\phi\} \cup \{C\}$, and every

$$e \in \{N_{\texttt{acc}}\} \cup \{N_i | 0 \leq i \leq 3\} \cup \{S_{\texttt{acc}}\} \cup \{S_i | 0 \leq i \leq 3\} \cup \{E_{\texttt{acc}}\} \cup \{E_i | 0 \leq i \leq 1\} \cup \{W_{\texttt{acc}}\} \cup \{W_i | 0 \leq i \leq 1\},$$

there is a variable $m_{v,t,c,e}$. The symbol $N_{\texttt{acc}}$ stands for " acceleration in the north direction" and N_i for $0 \leq i \leq 3$ stands for the phases of "northern move". Similarly, the others stand for the other directions. Example: $m_{v,t,2,E_1} = 1$ iff during time interval t, the robot moving E_1 at the slot with coordinates v towards its east.

Similar to the (*Rvert*), these variables take into consideration the time to traverse the slot North-South is double that of East-West, the difference between the speed of the loaded and unloaded robot, and the acceleration times. The interconnection between the x-, the $s-$, and the m-variables is similar to the *RVert* case. *RVert* and *RMove* can be seen as the decision variables for they are the actions that the model should perform.

7.3. IP Model

This section discusses the IP-model constraints and objective function. The setup of the variables allows formulating the IP-model by linking only consecutive time intervals $t, t+1$. The main focus is to allow all possible feasible movements. Similarly, the objective function pushes robot into reaching the terminal configuration as quickly as possible – see below for details.

7.3.1. Constraints

In this subsection, we present an overview of the IP model constraints. As was mentioned before, no constraint links more than two-time intervals, and if so, these time intervals should be consecutive. For every time interval, there are two types of constraints, namely, *stationary* and *time link*.

Stationary Constraints. At a given time interval and on a certain slot, these constraints encode the feasible movements that can occur on this slot and its neighbors (adjacent slots). It is clear that a slot can have up to four neighbors. The stationary constraints set basic rules like: (1) at a given time interval, on any slot there can be at most one car (2) a robot can move in at most one direction (3) a robot can be in at most one phase of movement (resp. vertical process) and so on. Most importantly, by including the feasible movements on the adjacent slots, this permit simultaneous motion of robots while avoiding collisions (see Figure 7.3.1). The following constraints describe a loaded robot on a slot of coordinate $v = (x, y)$ moving *north* during the time interval t. These constraints will just be concerned with the feasible movements of the robot at only during time t.

$$
\begin{aligned}
\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_0} &\leq s_{(x,y+1),t,\phi,\varepsilon} + \sum_{i=0}^{|C|-1} \big(m_{(x,y+1),t,i,N_0} + m_{(x,y+1),t,i,N_{\text{acc}}} \\
&\quad + m_{(x,y+1),t,i,1} + m_{(x,y+1),t,i,N_2} + m_{(x,y+1),t,i,N_3}\big) \\
&\quad + m_{(x,y+1),t,\phi,N_{\text{acc}}} + m_{(x,y+1),t,\phi,N_1} && (7.1) \\
\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_{\text{acc}}} &\leq s_{(x,y+1),t,\phi,\varepsilon} + \sum_{i=0}^{|C|-1} \big(m_{(x,y+1),t,i,N_{\text{acc}}} + m_{(x,y+1),t,i,N_1} \\
&\quad + m_{(x,y+1),t,i,N_2} + m_{(x,y+1),t,i,N_3}\big) + m_{(x,y+1),t,\phi,N_{\text{acc}}} \\
&\quad + m_{(x,y+1),t,\phi,N_1} && (7.2) \\
\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_1} &\leq s_{(x,y+1),t,\phi,\varepsilon} + \sum_{i=0}^{|C|-1} \big(m_{(x,y+1),t,i,N_1} + m_{(x,y+1),t,i,N_2} \\
&\quad + m_{(x,y+1),t,i,N_3}\big) + m_{(x,y+1),t,\phi,N_1} && (7.3) \\
\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_2} &\leq s_{(x,y+1),t,\phi,\varepsilon} + \sum_{i=0}^{|C|-1} \big(m_{(x,y+1),t,i,N_2} + m_{(x,y+1),t,i,N_3}\big) \\
& && (7.4) \\
\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_3} &\leq s_{(x,y+1),t,\phi,\varepsilon} + \sum_{i=0}^{|C|-1} m_{(x,y+1),t,i,N_3} && (7.5) \\
\sum_{i=0}^{|C|-1} m_{(x,y),t,i,N_0} &\leq s_{(x,y-1),t,\phi,\varepsilon} + \sum_{i=0}^{|C|-1} m_{(x,y-1),t,i,N_0} && (7.6)
\end{aligned}
$$

The inequality (7.1) states that if a robot is in the northern move phase N_0 then *north* of it can be nothing or a robot accelerating in the same direction as it and so on. E.g., when a robot is the northern move phase N_1 from (7.3) there cannot be a robot to the *north* of it accelerating since a collision will occur. Moreover, since N_0 is the northern move phase in which the robot arrives at (x,y) and is almost occupying all of (x,y) slot, then by (7.6) $(x,y-1)$ be either unoccupied or there can be a robot in the northern phase N_0 at $(x,y-1)$.

Time Link Constraints. For any two consecutive time intervals $t,t+1$ and any slot v, these constraints encode the feasible decisions (movements or processes) from time t to time $t+1$ for a robot. Similarly to the stationary constraints they also take into consideration the neighboring slots to decide whether a decision is feasible. The following constraints describe the future feasible movements of a loaded robot on (x,y) moving east.

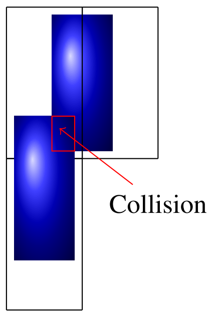


Figure 8. *North east* collision

$$
\begin{aligned}
& m_{(x,y),t+1,i,E_{\text{acc}}} && && \leq s_{(x,y),t,i,\rho} && && (7.7)\\
& m_{(x,y),t,i,E_{\text{acc}}} && + \; m_{(x,y),t,i,E_0} && \leq m_{(x,y),t+1,i,E_1} && && (7.8)\\
& m_{(x,y),t+1,i,E_1} && && \leq m_{(x,y),t,i,E_{\text{acc}}} && + \; m_{(x,y),t,i,E_0} && (7.9)\\
& m_{(x,y),t+1,i,E_0} && && \leq m_{(x-1,y),t,i,E_1} && && (7.10)\\
& m_{(x,y),t,i,E_1} && && \leq s_{(x+1,y),t+1,i,\rho} && + \; m_{(x+1,y),t+1,i,E_0} && (7.11)\\
& m_{(x,y),t,i,E_1} && && \leq s_{(x,y),t+1,\phi,\varepsilon} && + \; m_{(x,y),t+1,\phi,E_0} && (7.12)
\end{aligned}
$$

Before starting to explain this group of constraints, it is important to understand how the phases of movements work. An example of a robot moving East is utilized to explain the phases. Recall that E_{acc} and E_0 on (x,y) refer to start moving (accelerating) towards $(x+1,y)$ and continue the movement towards $(x+2,y)$ respectively.

On one hand, during t and on the slot (x,y), if the robot accelerates, then during $t+1$ it will either continue moving towards $(x+2,y)$ or prepare to stop at $(x+1,y)$ (starts decelerating). Since the robot is moving with half its speed, then either it continues moving towards $(x+2,y)$ but will not reach $(x+2,y)$ during $t+2$ or it decelerates but still cover parts of (x,y) during $t+1$.

On the other hand, during $t+1$ and on the slot (x,y), if the robot continues moving towards $(x+2,y)$, then during t and on slot $(x-1,y)$ it had the choice of either continue moving towards $(x+1,y)$ or stopping at (x,y). Similarly, the robot either will not reach $(x+2,y)$ during $t+2$ or decelerate but still cover parts of $(x-1,y)$ during $t+1$.

In order to encode the above, transition phases

- after the robot accelerates,
- before and after the robot continues moving,
- and before the robot decelerates

are added where the robot undergoes these phases. In the case of East (resp. West) loaded robot movement, there is a need for one transition phase, namely, E_1 (resp. W_1). Whereas there is no need for it in East (resp. West) unloaded robot movement since robots move at double speed, so after acceleration during t, robots will be halfway between (x,y) and $(x+1,y)$ and during $t+1$ robots will be fully on $(x+1,y)$ when decelerating and halfway on $(x+2,y)$ if it continues moving.

Remark. Since the length in the North-South direction is double the width, there is a need for three transition phases for the North and South loaded robot movements and one transition phase for the unloaded robots.

Now back to the above set of constraints, as mentioned when the robot starts accelerating towards a certain direction it has to standstill. Inequality (7.7) obliges the robot to standstill, i.e., the *SRS* variable $s_{(x,y),t,i,\rho} = 1$ when the robot accelerates east. Inequalities (7.8), (7.9), (7.10), and (7.11) organize the eastern move phases.

Inequalities (7.8) and (7.9) encode the following case. Either after the robot accelerates during t and on (x,y), i.e., $m_{(x,y),t,i,E_{\text{acc}}} = 1$ or the robot plans to continue moving towards $(x+2,y)$, i.e., $m_{(x,y),t,i,E_0} = 1$ there should be a transition phase during $t+1$, i.e., $m_{(x,y),t+1,i,E_1} = 1$. Conversely, if during $t+1$ there is any transition phase, i.e., $m_{(x,y),t+1,i,E_1} = 1$, then there should be either during t accelerating on (x,y), i.e., $m_{(x,y),t,i,E_{\text{acc}}}$ or continue moving towards $(x+2,y)$, i.e., $m_{(x,y),t,i,E_0} = 1$.

Inequality (7.10) encodes the following case. During $t+1$ when the robot is planning to continue moving towards $(x+1,y)$, i.e., $m_{(x,y),t+1,i,E_0} = 1$ there should be a transition on $(x-1,y)$ during t, i.e., $m_{(x-1,y),t,i,E_1} = 1$.

Inequality (7.11) encodes the following case. During t if the robot is in transition phase on (x,y), i.e., $m_{(x,y),t,i,E_1} = 1$, then the robot should either stop decelerate on $(x+1,y)$ during t, i.e., $s_{(x+1,y),t,i,\rho} = 1$ or is planing to move toward $(x+2,y)$, i.e., $m_{(x+1,y),t+1,i,E_0} = 1$.

Finally, inequality (7.12) encodes the following case. During t if the robot is in transition phase on (x,y) then the slot is no longer occupied by a robot during $t+1$, i.e., $s_{(x,y),t+1,\phi,\varepsilon} = 1$ unless some unloaded robot was planning to move towards (x,y) during $t+1$, i.e., $m_{(x,y),t+1,\phi,E_0} = 1$. Now we state the following result that can be obtained by counting.

Proposition 7.3.1. *The IP-formulation using the x-, s-, z-, and m-variables on an $n \times n$ grid has $O(|C| t_{\max} n^2)$ inequalities and variables. Moreover, in the computations done, the number of colors was fixed to three and so there were $O(t_{\max} n^2)$ inequalities and variables.*

7.3.2. Objective Function

Numerous experiments were made with different objective functions. The best objective function was the one that aimed to "nudge" the robots into movement while trying to support reaching the terminal configuration as quickly as possible. For each time interval, $f(t) = \dfrac{t}{t_{max}}$ was the cost of the *SRS* excluding $s_{v,t,\phi,\rho}$ and

$s_{v,t,\phi,\varepsilon}$. The objective function is as follows,

$$\sum_{\forall v \in V, \forall t \in T, \forall i \in C} f(t) \cdot (s_{v,t,i,\rho} + s_{v,t,\phi,\mu} + s_{v,t,i,\mu} + s_{v,t,\phi,\zeta}) \tag{7.13}$$

7.4. SAT Model

The SAT-model follows the same basic approach as the IP-model. Indeed, many of the clauses have a direct counterpart in an inequality and vice versa. Overall, deriving the CNF clauses from the control requirements and the variables is an exercise in logical thinking. The following clauses determine the situation of a slot v at time t after lifting or before dropping.

$$s_{v,t+1,i,\rho} \quad \bigvee_{\substack{j \in C \\ j \neq i}} \quad s_{v,t+1,j,\rho} \quad \vee \quad x_{v,t,\phi} \quad \vee \quad z_{v,t,\lambda_4} \qquad \text{for all } i \in C \tag{7.14}$$

$$s_{v,t,i,\rho} \quad \bigvee_{\substack{j \in C \\ j \neq i}} \quad s_{v,t,j,\rho} \quad \vee \quad x_{v,t,\phi} \quad \vee \quad z_{v,t+1,\delta} \qquad \text{for all } i,j \in C \tag{7.15}$$

$$\neg s_{v,t+1,i,\rho} \quad \vee \quad \neg s_{v,t+1,j,\rho} \qquad \text{for all } i,j \in C \tag{7.16}$$

$$\neg s_{v,t,i,\rho} \quad \vee \quad \neg s_{v,t,j,\rho} \qquad \text{for all } i,j \in C \tag{7.17}$$

By (7.14) and (7.16), in $t+1$, if a loaded robot (car of type i) will be ready on v, then, in t, v has no cars or the robot is done with lifting a car of type i. Similarly (7.15) and (7.17), if in t, the loaded robot is ready on v, then, in $t+1$, v will have no cars or the robot will drop a car of type i.

7.5. Computational Results

This section aims to compare and analyze the performance of IP- and CNF-model on different instances. The next subsection describes the set of instances in which some of them are based on existing parking lots in Tallinn, Estonia. Subsection 7.5.2 explains the software which was used to solve the IP and CNF-model and the platform where the computations are done. Subsection 7.5.3 is a summary of the performance of both model solvers on different instances.

7.5.1. The data

There are two sets of parking lots. Five parking lots (a,b,c,d, and e) are derived from an existing parking area (Marsi-3) in Tallinn, Estonia (e.g. Figure 9 and 10). That parking area has been split up into parts, based on requirements such as having vehicle transfer stations accessible from elevators for pedestrians. Humans and robots cannot use the same floor area for operational safety. E.g. the name of the third parking lot is Marsi-3c. Then there is a set of 5 parking lots generated randomly: Starting from an $m \times n$ grid, a "walls" is placed between parking slots with a given probability (e.g. Figure 11). The randomly generated parking lots are referred to as rnd-γ where $\gamma \in \{1, 2, 8, 7, 19\}$

Each parking lot gives rise to several instances to be solved by choosing the initial and terminal configurations of the cars and robots and the number of robots and cars. Each instance is solved several times, for varying $t_{\max}$. (The difficulty is that no good upper bound on $t_{\max}$ is known.) The size of C in all the instances is 3. All configurations were chosen randomly (discarding those with no reconfiguration of initial to terminal configurations.) An instance is refer to as either Marsi-3$\alpha_{\text{rnd-}\beta}$ or rnd-$\gamma_{\text{rnd-}\beta}$ where $\alpha \in \{a,b,c,d,e\}$, β varies from 1 to 4, and $\gamma \in \{1,2,8,7,19\}$.

An important feature of any instance is the number of free slots, i.e., the slots with no cars or robots on and the robots can move along them (not obstacles). Another feature which plays a role in the complexity is the *bottleneck slots* (see Figure 11) which is the only slot that the robot has to traverse when passing from one area of the parking lot to another. The number of bottleneck slots is assumed to make the instance harder–see Subsection 7.5.3.

The total number of instances is 30. The largest instances were 13×10 with only 41 free spots (e.g. Figure 10). For these instances, 7 cars of 3 different colors and 5 robots were used in the generation of the initial and terminal configurations. The smallest instances 4×4 with 9 free spots (see Figure 9) have 6 cars of 3 different colors and 2 robots. Compared to real-world problems, the instances are small in terms of the number of cars, but the sizes and number of colors are compatible with real-world ones. The full description of all parking lots and instances is presented in Table 3.

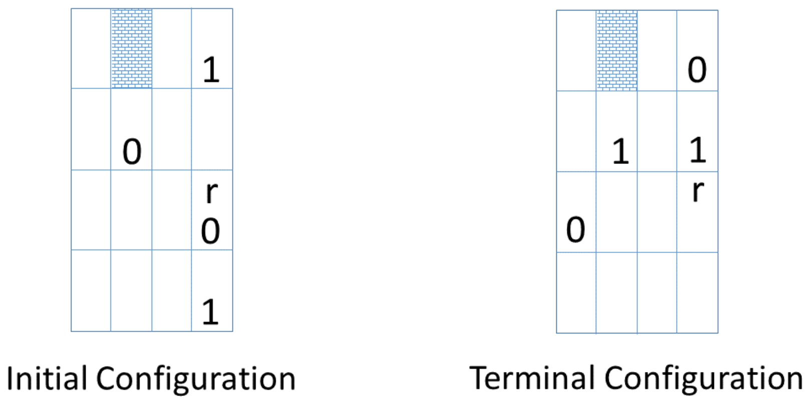


Figure 9. The instance Marsi-3$b_{\text{rnd-}1}$ with 9 free spots and the tiled spot is an obstacle.

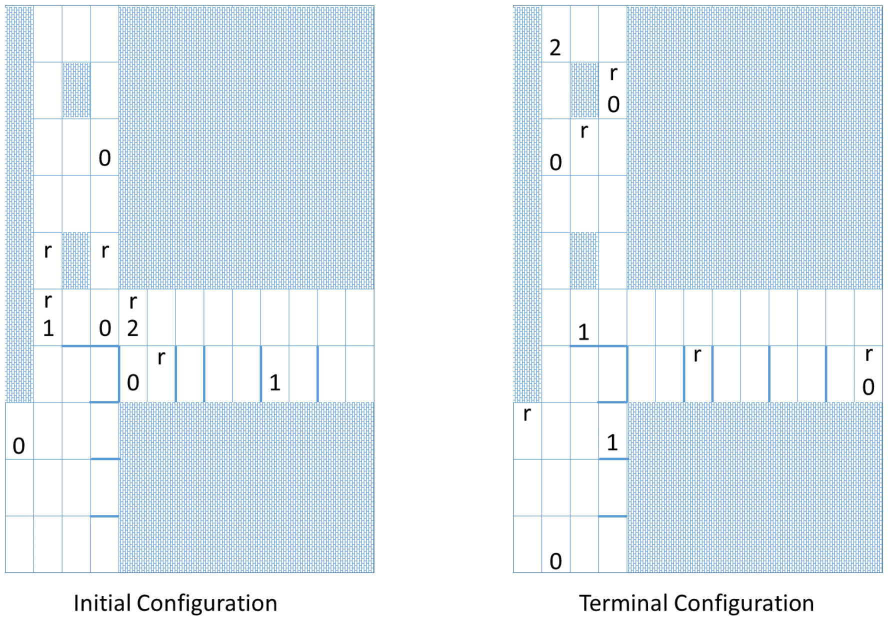


Figure 10. The instance Marsi-3$e_{\text{rnd-4}}$ with 41 free slots. The tiled spots are obstacles, and the bold borders are uncrossable too.

7.5.2. Hardware and Software

The GUROBI optimizer [33] in version 7.0 is used to solve the IP-model, and CRYPTOMINISAT 5 [63] is used as SAT solver. The times were taken on a computer server with Intel(R) Core(TM) i7-4790K CPU (4 cores) and 16GB of RAM. (Both solvers were run with one thread.) GUROBI is run with the parameters `MIPFocus`$= 2$, `Presolve`$= 2$. CRYPTOMINISAT 5 is run with `preproc`$= 1$. Both solvers are given a time limit of 10000sec. The time to read in the data from a file and construct the models is negligible.

7.5.3. Results

Since the CNF-model does not optimize an objective function, for the comparison, GUROBI is asked to give only a feasible solution, a solution where the terminal configuration is matched. CRYPTOMINISAT 5 very significantly outperforms GUROBI especially when the number of robots is more than two.

GUROBI is able to solve only one of the IPs within the time limit. For that instance, GUROBI returns the solution in 90 sec, however, CRYPTOMINISAT 5 returns the same solution in 76 sec.

CRYPTOMINISAT 5, on the other hand, gives relatively satisfactory results.

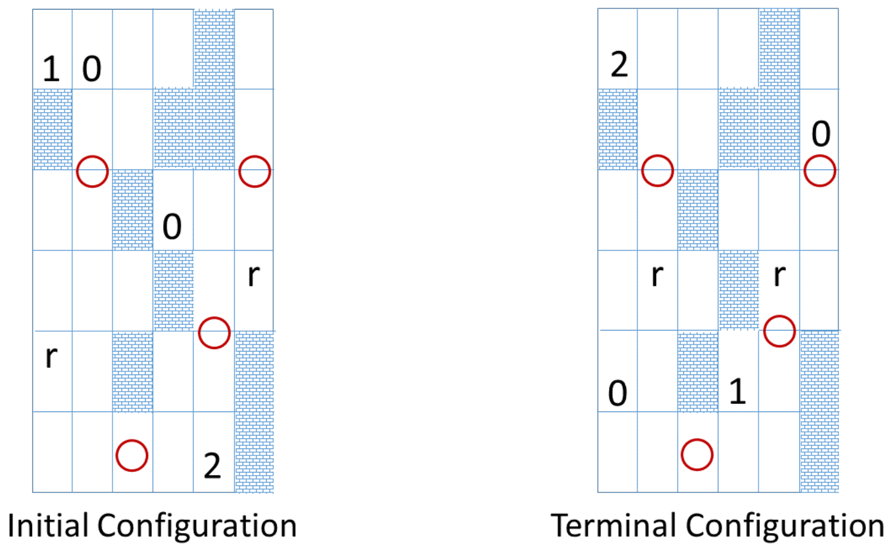


Figure 11. The instance rnd-$8_{\text{rnd-}1}$ with bottlenecks. The tiled spots are obstacles, and the red circles are place on the bottlenecks

CRYPTOMINISAT 5 terminates within the given time limit for 85% of the CNF-models (yielding either a feasible solution or the information that there was none). In 60% of the instances, the running time is below 3600 sec. In only 20% of the instances, the running time is larger than 6000 sec.

The experiments reveal that the complexity of the instance depends on the number of robots versus cars (e.g. Marsi-3$e_{\text{rnd-}2}$ and Marsi-3$e_{\text{rnd-}5}$), the size of the parking lot (e.g. Marsi-3e), and the number of bottlenecks in the parking lot (rnd-8). In addition, the complexity highly depends on the choice of $t_{\max}$ (large $t_{\max}$ or small $t_{\max}$ are not good). The full computational results are presented in Tables 4, 5, and 6. The code, instances, and solutions are in the github `https://github.com/Abzinger/crobots`.

Before developing the CNF-model, another IP-model, TH, for the same problem was developed within Algorithms and Theory research group for the master thesis [67]. The idea is to create a model with fewer variables and different model than this IP-model. Indeed, the constraints will become more complicated (constraints use variables from more than consecutive time-steps).

The TH model considered the parking lot as a graph $G(V,E)$ where V is set of all parking spaces and there is an edge between two parking lots if they are reachable from one another. Cars are supposed to be moved by robots on the graph. So, in this case, there cannot be a robot occupying part of a vertex instead it is going to occupy the edge. Therefore, instead of having a variable for each step of the move in a certain direction, e.g., m_{v,t,i,N_j} for all $j \in \{0,\ldots,3\}$ there is one variable indicating that the robot is still moving in this direction $m_{v,t,i,N}$ and when a robot is moving along an edge $e = (u,v)$, namely, $\mathrm{occ}_{e,t}$, and the edge should be marked occupied until the move is done, i.e., $\mathrm{occ}_{e,j}$ for all $i \in \{t,\dots,t+\eta\}$ where η is the time for a move to be completed. Also, the node status of nodes u and v should stay the same until the move is done. This forces to have constraints linking time t and $t+\eta$, e.g., if at time t a robot is doing a move that needs η time steps to finish along an edge $e = (u,v)$ in order to check whether the edge e can be occupied during the move there will be a constraint connecting the variable of status of u at time t with the variables of edge occupation for all the time steps t till $t+\eta$. Table 2 summarizes the main difference between TH and this IP model.

Thesis IP	**TH IP**
Encodes directly the parking lot	Encodes the parking lot via a graph $G(V,E)$
$O(\|C\|t_{\max}n^2)$ variables	$O(\|C\|t_{\max}\max(n,\|E\|))$ variables
$O(\|C\|t_{\max}n^2)$ constraints	$O(t_{\max}\max(n\|C\|,\|E\|\|C\|^3))$
Slot Occupation variables	Edge Occupation variables
Variables representing each step of a move	One variable for all steps of a move
Constraints connect only consecutive times	Constraints can connect nonconsecutive times

Table 2. Summary of the main difference between TH and the thesis IP model.

TH was not tested with any of these instances since already its performance is as good as the IP-model on the instances tested in [67]. More details about the TH model and how it compares with the current IP-model can be found in [67].

7.6. Conclusion

Difficult discrete optimal control problems arise in the control of automated valet parking systems. For the study of exact methods for the problems, a time-expanded model is devised, and its solution via Integer Programming and CNF-based Constraint Programming is compared. Using state of the art software for the two approaches, Integer Programming is found useless.

The CNF-model, however, proved to be usable. Now, the next goal is to (design and) compare the solution qualities (amount of time needed to reach a terminal configuration from a current configuration) of heuristic algorithms to optimal solutions, which can be found or at least bounded using our CNF-model.

Parking Lot				Robots & Cars						
Name	Area	Spaces	Bottleneck(?)	**Name**	#Robots	#Cars/0	#Cars/1	#Cars/2	Free Spaces	#Avg. Dist.
Marsi-3a	04x05	17	no	rnd-1	2	2	2	1	10	50
			no	rnd-2	2	2	2	1	11	57
			no	rnd-3	2	2	2	2	9	54
Marsi-3b	04x04	16	no	rnd-1	2	2	2	1	10	68
			no	rnd-2	2	2	2	1	10	52
Marsi-3c	11x04	30	yes	rnd-1	3	2	2	2	22	58
Marsi-3d	10x07	37	yes	rnd-1	4	4	3	0	23	64
				rnd-2	3	2	2	2	29	56
				rnd-3	5	4	2	1	27	67
				rnd-4	5	4	2	1	25	55
				rnd-5	5	4	2	1	26	53
Marsi-3e	13x10	48	yes	rnd-1	3	2	2	1	41	63
				rnd-2	3	2	2	2	39	72
				rnd-3	3	2	2	2	39	69
				rnd-4	5	4	2	1	38	51
				rnd-5	5	4	2	1	36	66
rnd-1	04x04	15	no	rnd-1	1	2	2	0	11	87
rnd-2	05x05	19	yes	rnd-1	1	2	2	0	14	62
				rnd-2	1	2	2	0	16	65
				rnd-3	1	2	2	0	15	68
rnd-7	05x05	20	no	rnd-1	2	2	1	1	14	59
				rnd-2	2	2	2	1	13	73
rnd-8	06x06	27	yes	rnd-1	2	2	1	1	21	86
				rnd-2	2	2	1	1	21	78
				rnd-3	2	2	1	1	22	83
				rnd-4	2	2	2	1	21	92
rnd-19	05x10	39	yes	rnd-1	4	4	2	1	30	62
				rnd-2	4	4	2	1	30	58
				rnd-3	4	4	2	1	29	54
				rnd-4	3	4	3	0	30	70

Table 3. List of instances. The *Parking Lot* columns describe the parking lot in general such as, the name of the grid, its area, number of cells it contains which are not obstacles, and whether it has bottlenecks or not. The *Robot & Cars* columns describe the configurations of cars and robots on a certain grid. For most of the grids, there were multiple configurations and so *Name* here refers to the configuration. *#Robots*, *#Cars/0*, *#Cars/1*, *#Cars/2* are the number of robots and cars of type 0, 1, and 2 respectively. *Free spaces* represents the number of empty parking spaces, i.e., no cars or robots. The *Avg. Dist.* is computed by finding a weighted perfect matching between the initial and final configurations then evenly distributing the result over the robots.

Instance	CNF			IP	
	t_{max}	solve time (sec)	feasible	solve time (sec)	feasible
Marsi-3a rnd-1	90	814	y	∞	n/a
	78	268	y	∞	n/a
	65	∞	n/a	∞	n/a
	53	∞	n/a	∞	n/a
	40	30	n	1861	n
rnd-2	90	8280	y	∞	n/a
	78	2088	y	∞	n/a
	65	∞	n/a	∞	n/a
	53	∞	n/a	∞	n/a
	40	13	n	893	n
rnd-3	90	2554	y	∞	n/a
	85	9604	y	∞	n/a
	70	∞	n/a	∞	n/a
	55	∞	n/a	∞	n/a
	40	29	n	1520	n
Marsi-3b rnd-1	90	1585	y	∞	n/a
	80	1039	y	∞	n/a
	70	∞	n/a	∞	n/a
	60	∞	n/a	∞	n/a
	50	576	n	∞	n/a
rnd-2	90	2113	y	∞	n/a
	80	2023	y	∞	n/a
	70	6271	y	∞	n/a
	60	∞	n/a	∞	n/a
	50	5010	n	∞	n/a
Marsi-3c rnd-1	120	7999	y	∞	n/a
	100	4054	y	∞	n/a
	80	101	y	∞	n/a
	60	∞	n/a	∞	n/a
	40	10	n	5	n
Marsi-3d rnd-1	90	1906	y	∞	n/a
	80	6223	y	∞	n/a
	70	∞	n/a	∞	n/a
	60	3165	n	∞	n/a
	50	19	n	∞	n/a
rnd-2	90	1400	y	∞	n/a
	80	6694	y	∞	n/a
	70	∞	n/a	∞	n/a
	60	305	n	∞	n/a
	50	19	n	∞	n/a
rnd-3	90	1857	y	∞	n/a
	80	1539	y	∞	n/a
	70	1841	y	∞	n/a
	60	93	y	∞	n/a
	50	15	n	17	n
rnd-4	90	1331	y	∞	n/a
	80	994	y	∞	n/a
	70	∞	n/a	∞	n/a
	60	∞	n/a	∞	n/a
	50	39	n	∞	n/a
rnd-5	90	4891	y	∞	n/a
	80	3085	y	∞	n/a
	70	9706	y	∞	n/a
	60	∞	n/a	∞	n/a
	50	21	n	∞	n/a

Table 4. Computational results I

Instance	CNF			IP	
	t_{max}	solve time (sec)	feasible	solve time (sec)	feasible
Marsi-3e rnd-1	100	2080	y	∞	n/a
	90	6584	y	∞	n/a
	80	∞	n/a	∞	n/a
	70	213	n	∞	n/a
Marsi-3e rnd-1	60	22	n	96	n
	50	18	n	31	n
rnd-2	90	2811	y	∞	n/a
	80	∞	n/a	∞	n/a
	70	∞	n/a	∞	n/a
	60	46	n	∞	n/a
	50	19	n	62	n
rnd-3	90	3246	y	∞	n/a
	80	∞	n/a	∞	n/a
	70	∞	n/a	∞	n/a
	60	731	n	∞	n/a
	50	18	n	113	n
rnd-4	90	∞	n/a	∞	n/a
	80	2479	y	∞	n/a
	70	2807	y	∞	n/a
	60	4009	y	∞	n/a
	50	94	n	∞	n/a
rnd-5	100	1587	y	∞	n/a
	90	2173	y	∞	n/a
	80	896	y	∞	n/a
	70	4330	y	∞	n/a
	60	22	n	820	n
	50	21	n	24	n
rnd-1 rnd-1	110	2334	y	∞	n/a
	100	2124	y	∞	n/a
	95	5261	n	∞	n/a
	90	3082	n	∞	n/a
	80	1019	n	∞	n/a
rnd-2 rnd-1	150	2761	y	∞	n/a
	120	6260	y	∞	n/a
	100	∞	n/a	∞	n/a
	80	1489	n	∞	n/a
	60	61	n	∞	n/a
rnd-2	150	3797	y	∞	n/a
	120	2516	y	∞	n/a
	100	7269	y	∞	n/a
	80	2922	n	∞	n/a
	60	123	n	∞	n/a
rnd-3	150	3683	y	∞	n/a
	120	9781	y	∞	n/a
	100	9008	n	∞	n/a
	80	1764	n	∞	n/a
	60	38	n	∞	n/a
rnd-7 rnd-1	90	76	y	90	y
	70	21	y	∞	n/a
	65	1141	y	∞	n/a
	60	∞	n/a	∞	n/a
	50	133	n	∞	n/a

Table 5. Computational results II

Instance	CNF			IP	
	t_{max}	solve time (sec)	feasible	solve time (sec)	feasible
rnd-2	100	3726	y	∞	n/a
	88	1532	y	∞	n/a
	75	∞	n/a	∞	n/a
	62	∞	n/a	∞	n/a
	50	1105	n	∞	n/a
rnd-8 rnd-1	140	496	y	∞	n/a
	120	1024	y	∞	n/a
	100	5970	y	∞	n/a
	80	182	n	∞	n/a
	60	15	n	50	n
rnd-8 rnd-2	140	2927	y	∞	n/a
	120	2581	y	∞	n/a
	100	623	y	∞	n/a
	80	∞	n/a	∞	n/a
	60	16	n	∞	n/a
rnd-3	140	3040	y	∞	n/a
	120	1319	y	∞	n/a
	100	2333	y	∞	n/a
	80	5847	n	∞	n/a
	60	17	n	∞	n/a
rnd-4	150	2350	y	∞	n/a
	133	1585	y	∞	n/a
	115	6740	y	∞	n/a
	98	∞	n/a	∞	n/a
	80	922	n	∞	n/a
rnd-19 rnd-1	90	5537	y	∞	n/a
	80	525	y	∞	n/a
	70	8902	y	∞	n/a
	60	∞	n/a	∞	n/a
	50	19	n	67	n
rnd-2	90	1488	y	∞	n/a
	80	3126	y	∞	n/a
	70	1654	y	∞	n/a
	60	732	y	∞	n/a
	50	19	n	15	n
rnd-3	90	7821	y	∞	n/a
	80	3929	y	∞	n/a
	70	2239	y	∞	n/a
	60	3617	y	∞	n/a
	50	775	n	∞	n/a
rnd-4	90	2012	y	∞	n/a
	80	4780	y	∞	n/a
	70	∞	n/a	∞	n/a
	60	∞	n/a	∞	n/a
	50	78	n	∞	n/a

Table 6. Computational results III

8. CONCLUSION

Many fundamental problems in several disciplines can be modeled as Optimization problems. One significant class of these problems which has been widely applied is Convex Optimization. Efficient algorithms exist to find the solutions of Convex optimizations. But sometimes these problems express some nontrivial difficulties when being solved. In this case, studying the solution of the problem can lead to a better understanding of how to avoid these caveats by choosing the suitable model and algorithm. Also, optimization can be used to model hard problems in order to study its theoretical limitations and use these in designing heuristics and controlling their solutions qualities.

This thesis applies optimization to some complex systems. In one part, it studies a convex optimization problem to obtain partial information decomposition that serves as an analytical tool for some complex systems such as biological networks. In the other part, it designs exact algorithms for an optimization problem in the control of automated valet parking systems to study the theoretical limitation throughputs of these systems. The thesis is a collection of results obtained in [40–43].

In [42], the author among others studied the solution of the Convex Program involved in computing the bivariate BROJA PID measure. The author among others found that if the optimal solution of the problem is obtained on the boundary of the feasible region, then the boundary where the solution lies have an infinite directional derivative, i.e., all the surrounding points are infinitely repellent. The latter advocates the opinion of the supervisor of this thesis Dirk Oliver Theis that the difficulties which arise when solving the BROJA Convex Program are caused by the smoothness problems of the objective function on the boundary of the feasible region and not the shape of the feasible region itself.

The extensive study of the BROJA Convex Program gave strong insights on which optimization methods are used to achieve an efficient solver for BROJA PID. The author coded 6 different approaches, including Geometric Programming and other Cone Programming models, in order to have a robust solver for BROJA PID measure. He found out that MOSEK, a commercial Convex Optimization solver, was robust and fast, however, one of the Cone Programming models was the most robust, with satisfying speed.

In [43], based on the results in [42], the author among others developed a production-quality software for the robust computations of BROJA PID that is based on the reliable Cone Programming model of [42]. Then, the author among others discussed several Cone Programming models of BROJA PID based on the exponential cone and their properties and explained the technical details of the software. In addition, the author among others formulated Exponential Cone Programming to compute a multivariate PID.

The author also studied some properties of problems when optimizing partial information measures subjected to some constraints. The latter problem arises in neural coding, for example, when studying the capability of a mechanism independently from the input distribution to modify the information of the output [73].

The author found that such optimization is not convex or concave revealing the need for heuristic approaches to solve it. He also proved formulas for the gradients and sub-/super-gradients for the information decomposition measures conjectured by Dirk Oliver Theis. Using these gradients, the future plan is to utilize it in designing heuristic algorithms to compute the extractable shared information [60] and the maximum capacity of a mechanism for modifying information [73].

In [40], the author among others studied the theoretical throughput limitations of an optimization problem in the control of automated parking systems. The study was done via designing exact methods to solve the following problem: Given a car park layout, an initial configuration of a car park (location of cars, robots), into a desired, terminal configuration, what is the optimal set of control instructions for the robots to reorganize the initial configuration into the terminal configuration. The notion "optimal" in this problem, means fastest, in terms of clock-on-the-wall waiting time until the robots have completed their tasks.

The author among others designed an IP- and a CNF-model to solve the problem. The IP-model was not able to produce results most of the time, even when directed to only find one feasible solution, making the current version useless for the study. However, the CNF-model was found to be useful. Note that improving the IP model using, e.g., cutting planes should be an option. But, the future work is mainly focused on designing heuristics to solve the problem and utilizing the CNF-model to compare the solution qualities (amount of time needed to reach a terminal configuration from a current configuration) of heuristic algorithms to optimal solutions, which can be found or at least bounded using our CNF model.

BIBLIOGRAPHY

[1] E. D. Andersen and Y. Ye. A computational study of the homogeneous algorithm for large-scale convex optimization. *Computational Optimization and Applications*, 10(3):243–269, 1998.

[2] E. D. Andersen and Y. Ye. On a homogeneous algorithm for the monotone complementarity problem. *Mathematical Programming*, 84(2):375–399, 1999.

[3] V. Auletta, A. Monti, M. Parente, and P. Persiano. A linear-time algorithm for the feasibility of pebble motion on trees. *Algorithmica*, 23(3):223–245, 1999.

[4] P. K. Banerjee, J. Rauh, and G. Montúfar. Computing the unique information. *arXiv preprint arXiv:1709.07487*, 2017.

[5] A. B. Barrett. Exploration of synergistic and redundant information sharing in static and dynamical gaussian systems. *Physical Review E*, 91(5):052802, 2015.

[6] N. Bertschinger, J. Rauh, E. Olbrich, and J. Jost. Shared information—new insights and problems in decomposing information in complex systems. In *Proceedings of the European Conference on Complex Systems 2012*, pages 251–269. Springer, 2013.

[7] N. Bertschinger, J. Rauh, E. Olbrich, J. Jost, and N. Ay. Quantifying unique information. *Entropy*, 16(4):2161–2183, 2014.

[8] L. M. Bettencourt, V. Gintautas, and M. I. Ham. Identification of functional information subgraphs in complex networks. *Physical review letters*, 100(23):238701, 2008.

[9] L. M. Bettencourt, G. J. Stephens, M. I. Ham, and G. W. Gross. Functional structure of cortical neuronal networks grown in vitro. *Physical Review E*, 75(2):021915, 2007.

[10] S. Boyd, S.-J. Kim, L. Vandenberghe, and A. Hassibi. A tutorial on geometric programming. *Optimization and engineering*, 8(1):67, 2007.

[11] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[12] S. P. Boyd, T. H. Lee, et al. Optimal design of a cmos op-amp via geometric programming. *IEEE Transactions on Computer-aided design of integrated circuits and systems*, 20(1):1–21, 2001.

[13] N. Brenner, S. P. Strong, R. Koberle, W. Bialek, and R. R. d. R. v. Steveninck. Synergy in a neural code. *Neural computation*, 12(7):1531–1552, 2000.

[14] S. Bubeck and R. Eldan. The entropic barrier: a simple and optimal universal self-concordant barrier. *arXiv preprint arXiv:1412.1587*, 2014.

[15] R. Byrd, J. Nocedal, and R. Waltz. K nitro: An integrated package for nonlinear optimization. *Large-scale nonlinear optimization*, pages 35–59, 2006.

[16] G. Călinescu, A. Dumitrescu, and J. Pach. Reconfigurations in graphs and grids. *SIAM Journal on Discrete Mathematics*, 22(1):124–138, 2008.

[17] A. Chabrier. Vehicle routing problem with elementary shortest path based column generation. *Computers & Operations Research*, 33(10):2972–2990, 2006.

[18] R. Chares. *Cones and interior-point algorithms for structured convex optimization involving powers andexponentials.* PhD thesis, UCL-Université Catholique de Louvain, 2009.

[19] K. Chen and P. Ji. A mixed integer programming model for advanced planning and scheduling (aps). *European journal of operational research*, 181(1):515–522, 2007.

[20] M. Chiang et al. Geometric programming for communication systems. *Foundations and Trends® in Communications and Information Theory*, 2(1–2):1–154, 2005.

[21] D. Chicharro. Quantifying multivariate redundancy with maximum entropy decompositions of mutual information. *arXiv preprint arXiv:1708.03845*, 2017.

[22] R. J. Clasen. The linear-logarithmic programming problem. Technical report, RAND CORP SANTA MONICA CA, 1963.

[23] T. M. Cover and J. A. Thomas. *Elements of information theory.* John Wiley & Sons, 2012.

[24] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations research*, 40(2):342–354, 1992.

[25] A. Domahidi, E. Chu, and S. Boyd. ECOS: An SOCP solver for embedded systems. In *European Control Conference (ECC)*, pages 3071–3076, 2013.

[26] A. Domahidi, E. Chu, and S. Boyd. ECOS: An SOCP solver for embedded systems. In *European Control Conference (ECC)*, pages 3071–3076, 2013.

[27] R. J. Duffin, E. L. Peterson, and C. M. Zener. Geometric programming: theory and application. 1967.

[28] M. G. Earl and R. D'Andrea. Modeling and control of a multi-agent system using mixed integer linear programming. In *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, volume 1, pages 107–111. IEEE, 2002.

[29] B. Gärtner and J. Matousek. *Approximation algorithms and semidefinite programming.* Springer Science & Business Media, 2012.

[30] Q. Gaucher, C. Huetz, B. Gourévitch, and J.-M. Edeline. Cortical inhibition reduces information redundancy at presentation of communication sounds in

the primary auditory cortex. *Journal of Neuroscience*, 33(26):10713–10728, 2013.

[31] F. Glineur. Proving strong duality for geometric optimization using a conic formulation. *Annals of Operations Research*, 105(1):155–184, 2001.

[32] V. Griffith and C. Koch. Quantifying synergistic mutual information. In *Guided Self-Organization: Inception*, pages 159–190. Springer, 2014.

[33] I. Gurobi Optimization. Gurobi optimizer reference manual, 2016.

[34] M. Harder, C. Salge, and D. Polani. Bivariate measure of redundant information. *Physical Review E*, 87(1):012130, 2013.

[35] ISO. *ISO/IEC 14882:2011 Information technology — Programming languages — C++*. International Organization for Standardization, Geneva, Switzerland, Feb. 2012.

[36] J. Li, M. Zhou, T. Guo, Y. Gan, and X. Dai. Robust control reconfiguration of resource allocation systems with petri nets and integer programming. *Automatica*, 50(3):915–923, 2014.

[37] D. Lima. Brickell condo residents sue developer over faulty robotic garage. Miami Herald, April 20 2016.

[38] Z. Liu, J. Dai, B. Wu, and H. Lin. Communication-aware motion planning for multi-agent systems from signal temporal logic specifications. In *American Control Conference (ACC), 2017*, pages 2516–2521. IEEE, 2017.

[39] M. Lubin and I. Dunning. Computing in operations research using julia. *INFORMS Journal on Computing*, 27(2):238–248, 2015.

[40] A. Makkeh and D. O. Theis. Comparison of ip and cnf models for control of automated valet parking systems. In *International Conference on Optimization and Decision Science*, pages 233–241. Springer, 2017.

[41] A. Makkeh and D. O. Theis. Optimizing information decomposition measures. *arXiv preprint*, arXiv:1802.03947, 2018.

[42] A. Makkeh, D. O. Theis, and R. Vicente. Bivariate partial information decomposition: The optimization perspective. *Entropy*, 19(10):530, 2017.

[43] A. Makkeh, D. O. Theis, and R. Vicente. Broja-2pid: A robust estimator for bivariate partial information decomposition. *Entropy*, 20(4):271, 2018.

[44] D. Marinazzo, O. Gosseries, M. Boly, D. Ledoux, M. Rosanova, M. Massimini, Q. Noirhomme, and S. Laureys. Directed information transfer in scalp electroencephalographic recordings: insights on disorders of consciousness. *Clinical EEG and neuroscience*, 45(1):33–39, 2014.

[45] N. Nehamas. Residents furious as robotic parking garage at Brickell condo shuts down. Miami Herald, November 6 2015.

[46] N. Nehamas. Robotic parking garage ruins commute, Brickell condo residents say. Miami Herald, May 15 2015.

[47] N. Nehamas. Operator pulls out of Brickell condo's disastrous robotic parking garage. Miami Herald, January 11 2016.

[48] A. Nemirovski. Efficient methods in convex programming, 1994. Lecture Notes.

[49] A. Nemirovski. Advances in convex optimization: conic programming. In *International Congress of Mathematicians*, volume 1, pages 413–444, 2006.

[50] Y. Nesterov. Constructing self-concordant barriers for convex cones. 2006.

[51] Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

[52] Y. Nesterov and A. Nemirovskii. *Interior-point polynomial algorithms in convex programming*. SIAM, 1994.

[53] S. Nirenberg and P. E. Latham. Decoding neuronal spike trains: how important are correlations? *Proceedings of the National Academy of Sciences*, 100(12):7348–7353, 2003.

[54] J. Nocedal and S. Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[55] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068, June 2016.

[56] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. SCS: Splitting conic solver, version 1.2.7. `https://github.com/cvxgrp/scs`, April 2016.

[57] C. H. Papadimitriou, P. Raghavan, M. Sudan, and H. Tamaki. Motion planning on a graph. In *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 511–520. IEEE, 1994.

[58] D. Ratner and M. Warmuth. The (n2- 1)-puzzle and related relocation problems. *Journal of Symbolic Computation*, 10(2):111–137, 1990.

[59] D. Ratner and M. K. Warmuth. Finding a shortest solution for the n$\times$ n extension of the 15-puzzle is intractable. In *AAAI*, pages 168–172, 1986.

[60] J. Rauh, P. K. Banerjee, E. Olbrich, J. Jost, and N. Bertschinger. On extractable shared information. *arXiv preprint arXiv:1701.07805*, 2017.

[61] F. Robles. Road to robotic parking is littered with faulty projects. The New York Times, November 27 2015.

[62] A. P. Ruszczyński. *Nonlinear optimization*, volume 13. Princeton university press, 2006.

[63] M. Soos. The cryptominisat 5 set of solvers at sat competition 2016. *SAT COMPETITION 2016*, page 28, 2016.

[64] S. Sootla, D. O. Theis, and R. Vicente. Analyzing information distribution in complex systems. *Entropy*, 19(12):636, 2017.

[65] S. Sra, S. Nowozin, and S. J. Wright. *Optimization for machine learning*. Mit Press, 2012.

[66] S. Stramaglia, J. M. Cortes, and D. Marinazzo. Synergy and redundancy in the granger causal analysis of dynamical networks. *New Journal of Physics*, 16(10):105003, 2014.

[67] K. Tarbe. Täisarvulise planeerimise mudel automaatparklale. Master's thesis, Tartu Ülikooli, 2016.

[68] T. Tax, P. A. Mediano, and M. Shanahan. The partial information decomposition of generative neural network models. *Entropy*, 19(9):474, 2017.

[69] R. A. Waltz, J. L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical programming*, 107(3):391–408, 2006.

[70] Q. L. H. Wen-Jing. Scheduling and routing algorithms for agvs: a survey. 1999.

[71] M. Wibral, J. T. Lizier, and V. Priesemann. Bits from brains for biologically inspired computing. *Frontiers in Robotics and AI*, 2:5, 2015.

[72] M. Wibral, W. A. Phillips, J. T. Lizier, and V. Priesemann. Partial information decomposition as a unified approach to the characterization and design of neural goal functions. *BMC neuroscience*, 16(1):P199, 2015.

[73] M. Wibral, V. Priesemann, J. W. Kay, J. T. Lizier, and W. A. Phillips. Partial information decomposition as a unified approach to the specification of neural goal functions. *Brain and cognition*, 112:25–38, 2017.

[74] P. L. Williams and R. D. Beer. Nonnegative decomposition of multivariate information. *arXiv preprint arXiv:1004.2515*, 2010.

[75] R. M. Wilson. Graph puzzles, homotopy, and the alternating group. *Journal of Combinatorial Theory, Series B*, 16(1):86 – 96, 1974.

[76] J. Yu and S. M. LaValle. Structure and intractability of optimal multi-robot path planning on graphs. In *AAAI*, 2013.

Appendix A. CONVEXITY

This short appendix aims to define and review properties of convex sets and convex functions. The appendix is to a certain extent a collection of known results taken from the literature on convexity (e.g. [11,51,62]).

A.1. Convex Sets

Recall the definition of convex sets and some of their properties.

Definition A.1.1. A set $C \in \mathbb{R}^n$ is said to be *convex* if for all $c_1, c_2 \in C$ it contains all points

$$\alpha c_1 + (1-\alpha)c_2,\ 0 < \alpha < 1.$$

Example A.1.1. Some examples of convex sets.

- The set of all n-tuples of real numbers $\mathbb{R}^n$ where $n > 0$.
- The *half space* $\{x \mid a^T x \leq b\}$ where $a \neq 0$.
- The *hyperplane* $\{x \mid a^T x = b\}$ where $a \neq 0$.
- $B(x_c, r)$, the *(Euclidean) ball* in $\mathbb{R}^n$, defined as $\{x \mid (x-x_c)^T(x-x_c) \leq r\}$ where $r > 0$ and $x_c \in \mathbb{R}^n$.

There are operations which preserve convexity. Thus one can check whether a set C is convex either by using the definition of convexity or by showing that C is the image of another convex set under a transformation which preserves convexity. Below are some examples of such transformations.

Proposition A.1.1 ([62]). *Let $\{C_j\}_{j\in J}$ be a family of convex sets for some index set J. Let $\mathscr{A} : \mathbb{R}^n \to \mathbb{R}^m$ such that $\mathscr{A}(x) = Ax + b$ for some $A \in \mathbb{R}^{m,n}$ and $b \in \mathbb{R}^m$ be an affine mapping. Then*

1. ***Intersection.*** *$\bigcap_{j\in J} C_j$ is a convex set. Note that, in this case, J can have an infinite cardinality*
2. ***Direct product.*** *$C_1 \times \cdots \times C_{|J|} = \{(x_1, \ldots, x_{|J|}) \mid x_1 \in C_1, \ldots, x_{|J|} \in C_{|J|}\}$ is a convex set ($|J| < \infty$).*
3. ***Composition with affine function.*** *The set $\mathscr{A}(C_j) := \{\mathscr{A}(x_j) \mid x_j \in C_j\}$ is a convex set for any $j \in J$.*

Example A.1.2. Using Proposition A.1.1, the set $C = \{x \in \mathbb{R}^n \mid Ax = b \wedge x \geq 0\}$ is a convex set where $A \in \mathbb{R}^{m,n}$ and $b \in \mathbb{R}^m$.

A.2. Convex Functions

Recall the definition of convex functions.

Definition A.2.1. A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if for all $x_1, x_2 \in \mathbb{R}^n$ and for all $0 \leq \alpha \leq 1$ we have

$$f(\alpha x_1 + (1-\alpha)x_2) \leq \alpha f(x_1) + (1-\alpha) f(x_2).$$

Definition A.2.2. A function f is called proper if $f(x) > -\infty$ for all x and $f(x) < +\infty$ for at least one x.

All the functions that are used in this thesis are considered to be proper unless otherwise mentioned.

Example A.2.1. Here are some examples of convex functions.

- *Exponential.* e^{ax} *is convex on* $\mathbb{R}$*, for any* $a \in \mathbb{R}$.
- *Logarithm.* $\log x$ *is convex on* $\mathbb{R}_{++}$.
- *Negative entropy.* $x \log x$ *is convex on* $\mathbb{R}_+$ *(or* $\mathbb{R}_{++}$*).*

Similar to convex sets there are some operations which conserve the convexity of a function. The proposition below lists some of the transformations which will be used later.

Proposition A.2.1 ([62]). *Let* $F_i : C_i \subseteq \mathbb{R}^n \to \mathbb{R}$*, for all* $1 \leq i \leq m$ *be convex functions where* C_i*,* $1 \leq i \leq m$ *are convex sets. Then*

1. ***Nonnegative weighted sum.*** *The function* $F(x) = \sum_i \alpha_i F_i(x_i)$ *is convex on* $C_1 \times \cdots \times C_m$ *where* $\alpha_i \geq 0$ *for all* $1 \leq i \leq m$.
2. ***Pointwise supremum.*** *For any* $1 \leq i \leq m$*, let* $h_i : C_i \times Y \subseteq \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ *be a convex function in* $x \in C_i$ *for each* $y \in Y$*. Then the function* g_i*, defined as*

$$g_i(x) = \sup_{y \in Y} \ h_i(x,y)$$

is convex in x *where the domain of* g_i *is*

$$\mathrm{dom}(g_i) = \{x \mid (x,y) \in \mathrm{dom}(h_i) \textit{ for all } y \in Y, \sup_{y \in Y} \ h_i(x,y) < \infty\}.$$

ACKNOWLEDGMENT

I am exceptionally thankful to my supervisor, Dirk Oliver Theis, for his guidance, support, and patience during my PhD studies. By all means, I cannot imagine having a better supervisor. You provided me with motivation, challenges, advice, knowledge, and an enormous amount of time to all of that I am immensely grateful. You taught me how to adeptly multitask, be a stout-hearted during the tough moments, and become a proficient researcher. Whatever I write would never amount to what you truly deserve.

I am especially grateful to my coauthor Raul Vicente for all the research, discussions, and feedback that led to interesting well-written research papers. Thanks for the support and time you have given me.

I am very thankful to the University of Tartu internal reviewer Sven Laur and my reviewers/opponents Matthias Walter and Nicolas Gillis whose insightful suggestions and feedback have tremendously improved the quality of the thesis.

I am extremely thankful to the Estonian Research Council, the Estonian IT Academy, and the Estonian ICT Doctoral studies for the financial support throughout my studies. I am also sincerely grateful for the professional cozy environment within the Institute of Computer Science and for all the support provided by its administration and employees.

I would like to thank my colleges and professors in the chair of Theoretical Computer Science for the friendly environment and interesting discussions that had broadened my knowledge.

I would like to thank my friends whom I met in Estonia. Thanks for the joyful times, the board games nights, and the trips all over Estonia. I thank my teammates in FC International Tartu for the happy enthusiastic moments we had and the awesome football we enjoyed. During stressful times you were my safety net.

I thank my childhood friends, even though these days we are spread all over the world but your support always kept me on my toes.

I have a special thanks to my second family, the Algorithms and Theory group current and previous members, for the best possible environment we had in the office and outside it, the discussions and feedback which improved this thesis and my research work in general, and for lifting up my spirit during the tough times.

Last but not least, I give my wholehearted thanks to my family, in particular, the most important beings in my life, my parents and my sister. For all your untold support, unconditional love, unswerving confidence, and endless sacrifices. For cheering me up whenever I think of you. I owe you all the achievements that I have or will accomplish during my life.

SISUKOKKUVÕTE (SUMMARY IN ESTONIAN)

Optimeerimise rakendamine keerulistes süsteemides

Paljusid erinevate ainevaldkondade probleeme saab esitada optimeerimisülesannetena. Üheks märkimisväärseks ülesannete klassiks, kus optimeerimist rakendadakse, on kumer optimeerimine (Convex Optimization). Kumera optimeerimisülesande lahendamiseks leiduvad efektiivsed algoritmid, aga mõnikord tulevad nende ülesannete lahendamisel esile mittetriviaalsed takistused. Nendel juhtudel on aga võimalik lahenduse analüüsimise teel valida sobiv lahendusmeetod, mis neid takistusi väldib. Lisaks saab optimeerimist kasutada keeruliste ülesannete modelleerimisel, et uurida nende teoreetilisi piire ning kasutada saadud teadmisi heuristikute loomiseks ja lahenduse kvaliteedi kontrollimiseks.

Antud töö rakendab optimeerimist keerulistele süsteemidele. Esiteks uuritakse BROJA kumerat optimeerimisülesannet osaliseks informatsiooni lahutuseks (partial information decomposition). Seda kasutatakse näiteks analüütilise tööriistana bioloogiliste võrkude uurimiseks. BROJA abil leiti, et kui ülesande optimaalne lahend saavutatakse lubatava piirkonna (feasible region) piiril, siis sellel piiril on lõpmatu suunatud tuletis. See tähendab, et kõik ümbritsevad punktid on lõpmatult tõrjuvad. Viimane toetab juhendaja Dirk Oliver Theisi arvamust, et raskused, mis tekivad BROJA kumera ülesande lahendamisel, ei ole mitte põhjustatud lubatava piirkonna kujust, vaid ülesande eesmärkfunktsiooni siledusega seotud probleemidest lubatava piirkonna piiril.

Peale ulatuslike arvutuslike katsete läbiviimist jõuti järeldusele, et optimeerimisülesannete lahendamise tarkvara Mosek on kiire ja robustne, aga koonusplaneerimise (cone programming) mudel on kõige robustsem rahuldava kiirusega. Tulemuste põhjal loodi koonusplaneerimise mudelil põhinev robustne BROJA PID lahendamise tarkvara.

Antud töös uuritakse samuti situatsiooni, mis tuli hiljuti esile neuroteaduse valdkonnas, kus osaline informatsiooni lahutamise ülesanne lahendatakse kitsendustega. See optimeerimine ei ole kumer ega nõgus ning seetõttu vajab lahendamiseks heuristilist lähenemist. Selleks tuletatakse antud töös informatsiooni lahutamise näitajate sub- ja supergradiendid. Edasiseks tööks on nende tulemuste kasutamine heuristilise algoritmi loomiseks.

Viimasena uuritakse antud töös optimeerimisülesandena automaatse parkimissüsteemi läbilaskevõime teoreetilist limiiti. Töös luuakse täisarvulise planeerimise mudel ja kehtestatavuse kontrollimise ehk SAT mudel ülesande lahendamiseks. Täisarvulise planeerimise formulatsioon enamus ajast lahendeid efektiivselt leida ei võimalda, seda isegi juhul kui nõuda vaid ühe lubatava lahendi otsimist, ning seega on antud mudel vaadeldava ülesande lahendamiseks kasutu. Antud töös aga leitakse, et SAT mudel on antud ülesande jaoks kasulik. Edasine töö on põhiliselt suunatud antud ülesande jaoks heuristikute loomisele ning heuristiliste algoritmide ja optimaalsete lahenduste kvaliteedi (aeg, mis kulub praegusest konfiguratsioonist lõppkonfiguratsiooni jõudmiseks) võrdlemisele, kasutades SAT mudelit, mille abil saab leida optimaalsed lahendid või tõkked nendele.

PUBLICATIONS

CURRICULUM VITAE

Personal data

Name:	Abdullah Makkeh
Birth:	April 21st, 1990
Citizenship:	Lebanese
Language:	Arabic, English
Address:	Ülikooli 17, 209 Tartu 51005, Tartumaa, Estonia
Contact:	makkeh@ut.ee

Education

2014–	University of Tartu, Ph.D. candidate in Computer Science
2013–2014	Lebanese University, Beirut, Lebanon Research Intern
2011–2013	Lebanese University, Beirut, Lebanon M.Sc. in Mathematics
2008–2011	Lebanese University, Beirut, Lebanon B.Sc. in Mathematics
1993–2008	National Evangelical School in Nabatieh, Nabateih, Lebanon

ELULOOKIRJELDUS

Isikuandmed

Nimi:	Abdullah Makkeh
Sünniaeg ja -koht:	21. aprill 1990, Liibanon
Kodakondsus:	Liibanonlane
Keelteoskus:	araabia, inglise
Aadress:	Ülikooli 17-209 Tartu
	51005, Tartumaa, Eesti
Kontaktandmed:	makkeh@ut.ee

Haridus

2014–	Tartu Ülikool, informaatika doktorant
2013–2014	Liibanoni Ülikool, Beirut, Liibanon
	Uuringu praktikant
2011–2013	Liibanoni Ülikool, Beirut, Liibanon
	M.Sc. matemaatikas
2008–2011	Liibanoni Ülikool, Beirut, Liibanon
	B.Sc. matemaatikas
1993–2008	Riiklik Evangeelne Kool Nabatiehis, Nabatieh, Liibanon

74. **Meelis Kull.** Statistical enrichment analysis in algorithms for studying gene regulation. Tartu 2011, 151 p.
77. **Bingsheng Zhang.** Efficient cryptographic protocols for secure and private remote databases. Tartu 2011, 206 p.
78. **Reina Uba.** Merging business process models. Tartu 2011, 166 p.
79. **Uuno Puus.** Structural performance as a success factor in software development projects – Estonian experience. Tartu 2012, 106 p.
81. **Georg Singer.** Web search engines and complex information needs. Tartu 2012, 218 p.
83. **Dan Bogdanov.** Sharemind: programmable secure computations with practical applications. Tartu 2013, 191 p.
84. **Jevgeni Kabanov.** Towards a more productive Java EE ecosystem. Tartu 2013, 151 p.
87. **Margus Freudenthal.** Simpl: A toolkit for Domain-Specific Language development in enterprise information systems. Tartu, 2013, 151 p.
90. **Raivo Kolde.** Methods for re-using public gene expression data. Tartu, 2014, 121 p.
91. **Vladimir Šor.** Statistical Approach for Memory Leak Detection in Java Applications. Tartu, 2014, 155 p.
92. **Naved Ahmed.** Deriving Security Requirements from Business Process Models. Tartu, 2014, 171 p.
94. **Liina Kamm.** Privacy-preserving statistical analysis using secure multi-party computation. Tartu, 2015, 201 p.
100. **Abel Armas Cervantes.** Diagnosing Behavioral Differences between Business Process Models. Tartu, 2015, 193 p.
101. **Fredrik Milani.** On Sub-Processes, Process Variation and their Interplay: An Integrated Divide-and-Conquer Method for Modeling Business Processes with Variation. Tartu, 2015, 164 p.
102. **Huber Raul Flores Macario.** Service-Oriented and Evidence-aware Mobile Cloud Computing. Tartu, 2015, 163 p.
103. **Tauno Metsalu.** Statistical analysis of multivariate data in bioinformatics. Tartu, 2016, 197 p.
104. **Riivo Talviste.** Applying Secure Multi-party Computation in Practice. Tartu, 2016, 144 p.
108. **Siim Orasmaa.** Explorations of the Problem of Broad-coverage and General Domain Event Analysis: The Estonian Experience. Tartu, 2016, 186 p.
109. **Prastudy Mungkas Fauzi.** Efficient Non-interactive Zero-knowledge Protocols in the CRS Model. Tartu, 2017, 193 p.
110. **Pelle Jakovits.** Adapting Scientific Computing Algorithms to Distributed Computing Frameworks. Tartu, 2017, 168 p.
111. **Anna Leontjeva.** Using Generative Models to Combine Static and Sequential Features for Classification. Tartu, 2017, 167 p.
112. **Mozhgan Pourmoradnasseri.** Some Problems Related to Extensions of Polytopes. Tartu, 2017, 168 p.