TECHNISCHE UNIVERSITÄT MÜNCHEN

Lehrstuhl für Angewandte Geometrie und Diskrete Mathematik

Discrete Optimisation in Machine Learning - Learning of Bayesian Network Structures and Conditional Independence Implication

Silvia Lindner

Vollständiger Abdruck der von der Fakultät für Mathematik
der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender: Univ.-Prof. Dr. Dr. Jürgen Richter-Gebert

Prüfer der Dissertation:

1. Univ.-Prof. Dr. Raymond Hemmecke
2. Univ.-Prof. Dr. Rüdiger Schultz, Universität Duisburg-Essen
3. Prof. Ruriko Yoshida, University of Kentucky, USA
   (nur schriftliche Beurteilung)

Die Dissertation wurde am 26.10.2011 bei der Technischen Universität München
eingereicht und durch die Fakultät für Mathematik am 06.02.2012 angenommen.

Acknowledgement

First and foremost I would like to thank Prof. Raymond Hemmecke for his cooperation over the many years. I am especially indebted to his permanent encouragement, motivation and patience.

I would like to thank Prof. Weismantel and Prof. Gritzmann for the possibility to work in this interesting field of research both in Magdeburg and in Munich. Moreover I would like to thank Prof. Weismantel for introducing me to the field of "discrete optimisation" and giving me the opportunity to start working on various interesting topics already during my study.

Furthermore I would like to thank my coauthors and my cooperation partners Remco R. Bouckaert, David Haws, Raymond Hemmecke and Milan Studený for the nice team work, the numerous helpful discussions and emails and for their permission to use results of our joint research in this thesis.

I would like to thank both groups in Magdeburg and in Munich for the daily support, the encouragement, the kind working atmosphere and for the helpful discussions during the tea time in the morning and while waiting for the train to depart.

Moreover I owe Utz Uwe Haus und Michael Ritter a dept of gratitude for their technical support.

I would like to thank Andreas Alpers, Steffen Borgwardt, Melanie Herzog, Michel Ritter and Felix Schmiedl whose helpful comments contributed a lot to finalise this thesis and to make it a little more readable.

I am indebted to all the proofreaders who had the difficult task to bring my english spelling and style in due form.

Contents

List of Tables

List of Figures

English abstract

In this thesis we present an application of methods from discrete optimisation in the field of Machine Learning.

Learning Bayesian network structures is an $\mathcal{NP}$-hard nonlinear combinatorial optimisation problem. By means of algebraic representatives like standard imsets, this problem can be transformed to a linear program but with exponentially many variables. The underlying polyhedron (the convex hull of the standard imsets) is not yet well-understood and subject to ongoing research.

In this thesis we introduce new representatives of the underlying structures of the Bayesian networks. These so-called characteristic imsets (exponential size integer vectors) can be derived by an affine transformation from standard imsets and are therefore also unique representatives of structures of Bayesian networks. Analogous to standard imsets, the use of characteristic imsets turns the initial non-linear problem into a linear problem. But characteristic imsets are 0/1-vectors and hence enable us to derive on the one hand theoretical results and on the other hand to use existing optimisation software, which is especially fitted to 0/1-problems, to solve the corresponding learning task. Likewise, a restricted classes of structures, reduction onto certain sub-structures, can be considered. For these restricted classes, characteristic imsets enable us to simplify proofs of known results and easily to derive new complexity results for learning among these restricted structures.

Up to now, it is not possible to compute an explicit inequality description for the polytope of standard imsets for larger problem sizes. But now it is possible to obtain a LP-relaxation for the characteristic imset polytope such that first computational results of learning (restricted) Bayesian network structures via characteristic imsets using optimisation software can be derived. These computations gain from the simple binary representation and from the state-of-the-art solvers.

Bayesian network structures are defined by statements of conditional independence. Therein an important problem is to infer additional valid statements on the base of a fixed set of valid statements. This inference problem is known as the conditional independence implication problem. By the use of standard and characteristic imsets for the representation of Bayesian network structures the implication problem can be geometrically described and solved with linear programming techniques, as well. We present corresponding computational experiments and derive further structural simplifications.

German abstract

In dieser Arbeit stellen wir eine Anwendung von Methoden der diskreten Optimierung im Gebiet des maschinellen Lernens vor.

Das Strukturlernen von Bayes'schen Netzen ist ein $\mathcal{NP}$-schweres nichtlineares kombinatorisches Optimierungsproblem. Mit Hilfe von algebraischen Repräsentanten wie Standardimsets kann dieses nichtlineare Problem in ein lineares Problem, jedoch mit exponentiell vielen Variablen, transformiert werden. Das zu Grunde liegende Polyeder (die konvexe Hülle der Standardimsets) ist bisher wenig untersucht bzw. verstanden und daher Gegenstand aktueller Forschung.

In dieser Arbeit stellen wir neue Repräsentanten der zu Grunde liegenden Bayes'schen Netzstrukturen vor. Diese so genannten charakteristischen Imsets können durch eine affin-lineare Transformation aus den Standardimsets abgeleitet werden und sind dementsprechend ebenfalls eindeutige Repräsentanten von Strukturen Bayes'scher Netze. Ebenso wie Standardimsets lassen sie eine Transformation des nichtlinearen Problems in ein lineares Optimierungsproblem zu. Andererseits handelt es sich bei charakteristischen Imsets um 0/1-Vektoren. Sie ermöglichen es daher, auf eine einfache Art und Weise einerseits theoretische Resultate zu erhalten und andererseits existierende Optimierunssoftware, die insbesondere 0/1-Probleme sehr gut lösen kann, zum Strukturlernen zu nutzen. Auch können mit charakteristischen Imsets eingeschränkte Strukturen, d.h. das Limitieren auf bestimmte Teilstrukturen, betrachtet werden. So können z.B. bekannte Komplexitätsresultate über das Lernen von eingeschränkten Strukturen vereinfacht, zum Teil auch verschärft werden und neue Resultate hergeleitet werden.

Eine allgemeine Beschreibung des Polyeders aus Standardimsets kann bereits für kleine Problemgrößen nicht mehr rechnerisch ermittelt werden. Durch ihre einfache graphische Interpretation kann nun aber eine LP-Relaxierung für das Polytop der charakteristischen Imsets gefunden werden, so dass erste Berechnungen zum Strukturlernen (und eingeschränktem Strukturlernen) durchgeführt werden können. Diese profitieren vor allem von der Verwendung aktueller Optimierungsoftware.

Strukturen Bayes'scher Netze werden durch Aussagen bedingter Unabhängigkeit definiert. Dabei ist ein wichtiges Problem, von einer bestimmten Menge von zulässigen Aussagen auf weitere zulässige Aussagen zu schließen. Dieses Problem ist bekannt als das Problem der Implikation bedingter Unabhängigkeit. Durch die Repräsentation mittels Standardimsets und charakteristischen Imsets lässt sich dieses Problem geometrisch beschreiben und ebenfalls mit Methoden der linearen Optimierung lösen. Wir stellen entsprechende Laufzeitvergleiche vor und leiten weitere strukturelle Vereinfachungen her.

Introduction

Opening words

Various questions arising in Machine Learning, data analysis and reasoning under uncertainty are nowadays solved with techniques based on discrete mathematics and especially on discrete optimisation. Such methods are for example based on boolean satisfiability problems to answer the feasibility of networks [29] or integral representations of relations to reconstruct valid minimal networks obtaining these relations [28]. The reason for the possibility to apply discrete mathematics is caused by the discrete representation of reasoning via networks, i.e. graphs, applicable to many different objectives within statistical dominated fields of research.

Other discrete representations are introduced by algebraic statistics providing a geometric view of statistical concepts [35], which graphs not always can do. On the base of this algebraic and geometric view discrete optimisation can be applied.

These discrete and geometric representations are valid for reasoning with Bayesian networks, as well. Inheriting a probability distribution together with its factorisation a graph can be used to represent this factorisation and therefore the structure of the Bayesian network. In real life the structure is often not given and only observations can be used to infer relations among variables under consideration.

In this thesis we introduce methods from discrete optimisation to obtain the structure of a Bayesian network from data. Of course, due to a limit in observations this cannot represent the real life relationships for sure. But we can derive networks best representing the given data without major assumptions on the data and their distribution. We provide the optimal solutions together with optimality guaranties. Our method can be also adapted to incorporate structural information available in advance. We present computational experiments for several instances of learning Bayesian network structures with and without additional structural information.

On the other hand important properties of Bayesian networks arise from the equivalence of different ways of representing them. The network structure implies a graphical representation, whose various ways of characterisation are analysed in a general way in [42] and in a more specific way in [1], [2], [21] and [66]. But Bayesian networks or, more precisely, the probability distributions they describe, can also be represented by statements of conditionally independence. These statements uniquely determine a set of variables of the network to be conditional independent and thus can be used to overall describe the network. The lack of the pure graphical representation is its inability to represent all conditional independence statements. However, both ways of representation, via graphs and via conditional independence statements, yield to questions of minimal representation. Let a Bayesian network be defined by a particular set of conditional independence statements or by a graph. Several conditional independence statements are additionally valid for the Bayesian network but need not to be explicitly given in the definition of the Bayesian network. These statements are not included in a minimal representation of the Bayesian network. The problem of deciding statements to be additional valid or not is known as the conditional independence problem which can be translated into a geometric problem.

In this thesis we introduce methods from discrete optimisation based on a geometric translation solving the conditional independence problem approximately. The geometric translation is based on already known algebraic representatives. Additionally, we introduce a transformation of this geometric translation leading into an equivalent linear programming formulation. This formulation we show to be solvable far more quickly in practice, because simple necessary conditions can be obtained by analysing the simpler structure.

Both problems, the derivation of the structure of a Bayesian network and the implication of conditional independence statements, can be solved by the help of algebraic representatives of networks and conditional independence statements.

The major contribution of this thesis is the introduction of new combinatorial representatives of Bayesian network structures which are closer to the graphical representation than other algebraic representations. With these combinatorial representatives we can derive new theoretical results, e.g. for the inclusion of independence models or for the complexity of the structure learning task. Moreover, we can provide polyhedral descriptions of both problems mentioned above which then can be solved with linear integer programming techniques.

Summary of main contributions

We use the presented combinatorial representatives to answer the following questions appearing in the field of Bayesian networks.

- New theoretical results can be obtained and already known results can be concluded easily.

  We derive results in Chapter 2, page 37: e.g. Theorem 2.1.2, Lemma 2.1.3, Corollary 2.2.1, Corollary 2.2.2, Corollary 2.2.3, Corollary 2.2.4, Theorem 2.2.5, Corollary 2.2.6 and Corollary 2.3.1, pages 38-47. Moreover, we reproduce e.g. Lemma 2.1.4, page 40, and Lemma 2.1.6, page 40, in a simple way based on the combinatorial representatives.

- Polyhedral descriptions of the set of valid structures with and without several additional structural restrictions can be computed and used to calculate provable optimal network structures. Former approaches have not been able to derive problem descriptions of structures with additional restrictions simultaneously in a simple, because graphical, way.

  In Chapter 3, page 51, we present an inequality description and provide results: e.g. Theorem 3.1.1, Theorem 3.2.3, Theorem 3.2.4, Conclusion 3.3.14, Theorem 3.4.3, Theorem 3.4.4 and Lemma 3.5.1, pages 52-84.

- Complexity results of learning restricted Bayesian network structures can be both extended and derived in a simple, graphical way.

  In Chapter 4, page 89, we reproduce and tighten results e.g. Lemma 4.2.1, Theorem 4.2.5 and Theorem 4.2.7, pages 89-92.

- We show linear programming to be an appropriate tool for solving conditional independence implication. Additionally, using a problem description based on inverses of our combinatorial representatives enables us to derive necessary conditions on the validity of the implication, which speed up the computations tremendously.

Related references

This thesis builds on the work by Studený [60] who introduced an algebraic representation of the conditionally independence statements and of the network structures, which we use to obtain our new combinatorial representatives. We therefore extend the presented methods to solve the conditional independence implication problem from [9] and use the representation of the structure learning problem from [65]. We answer questions related to those presented in [64] and [65].

Introducing the concept of restricted and unrestricted learning applied to our representation we can reconstruct known results and solution algorithms from e.g. [15] and add our methods to those of [3], [12], [14], [16], [18] and [40]. Moreover, we can supplement complexity results, e.g. [11], [13], [32], [46] and [56], by introducing our new approach based on combinatorial optimisation and give optimality guarantees for the obtained solution.

In addition to the work this thesis is based on, there exist various other references of authors also dealing with algebraic representations of structures and statements either enabling them to solve the conditional independence implication problem with linear programming methods ([48] and [50]) or enabling them to solve the structure learning task to optimality ([17], [37] and [55]). We give an overview of the respective representatives and methods in the corresponding sections of this thesis and relate our results to these approaches.

Several basic textbooks introduce Bayesian networks and learning Bayesian networks (and also learning Bayesian network structures) in an easy understandable manner without any required major knowledge of statistical backgrounds. Such books are for example [31], [38], [39] and [47]. Other aspects of Bayesian networks are presented in e.g. [7], [26], [42], [51] and [52].

Importance of Bayesian networks

Bayesian networks are a widely used concept to visualise relationships of variables with random or unknown value. These realisations can only be estimated based on observations. The major power of Bayesian networks is to quantify the probabilistic influence of the variables without background knowledge about their relationships and by the fact that only observations on these variables are possible. This task is known as the Learning of Bayesian networks.

Bayesian networks can be applied, whenever real-life situations can be visualised by variables and reasoning among them is only under uncertainty. This is for example medical diagnosis, autonomous vehicle navigation and information filtering just to name some of them. An overview of the details of possible applications is presented in [26].

Example 1. A famous Bayesian network is the "Alarm network" (an acronym of "A logical alarm reduction mechanism") given in [5] and [32]. This network is used in the medical attendance of patients to help making decisions based on symptoms. The Bayesian network is represented by a directed graph (see Figure 1, next page) in which 37 variables are connected with arcs to visualise their dependence. The variables are divided into diagnostic (blue nodes), intermediate (green nodes) and measurement variables (red nodes) with respect to their medical interpretation.

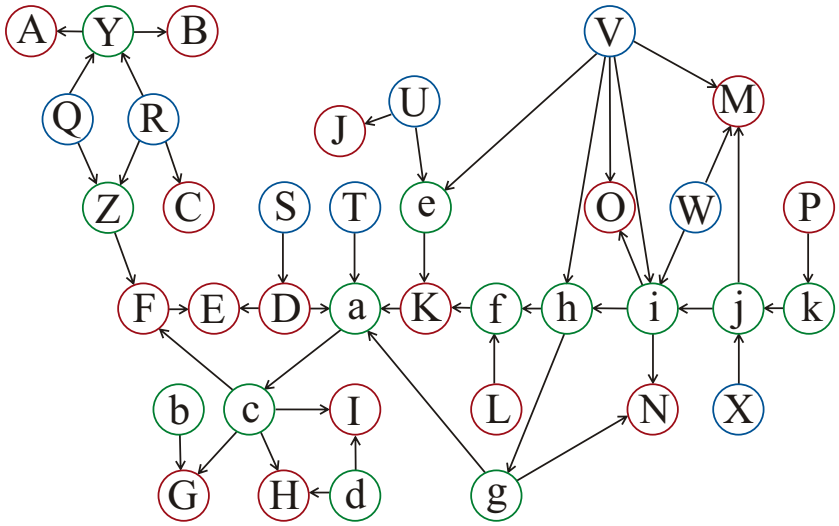

Figure 1: The Alarm network as presented in [5] and [32].

We highlight a particular detail of the Alarm network (Figure 2 below) taken from the online repository "Bayes Net Library", http://www.norsys.com/netlibrary/index.htm. This rearranged detail particularly demonstrates the combination of different variables each representing different aspects of a medical diagnosis. The tables shown in Figure 2 present the conditional probabilities, whose conditions are given in the graph. For example, the node "Y" represents the "Left Ventricular End-diastolic volume" which is a discrete variable with realisations "**l**ow", "**n**ormal" and "**h**igh". The variable is directly dependent on variables "Q" and "R" which both have binary values "**t**rue" and "**f**alse". The first number of the probabilities given for "Y" in the table reads as follows: The probability that "Y" is "low" given that "Q" and "R" are "true" is 0.95.

Combined with the probabilities of all conditional dependencies of the variables, the Alarm network is a Bayesian network. ⋆

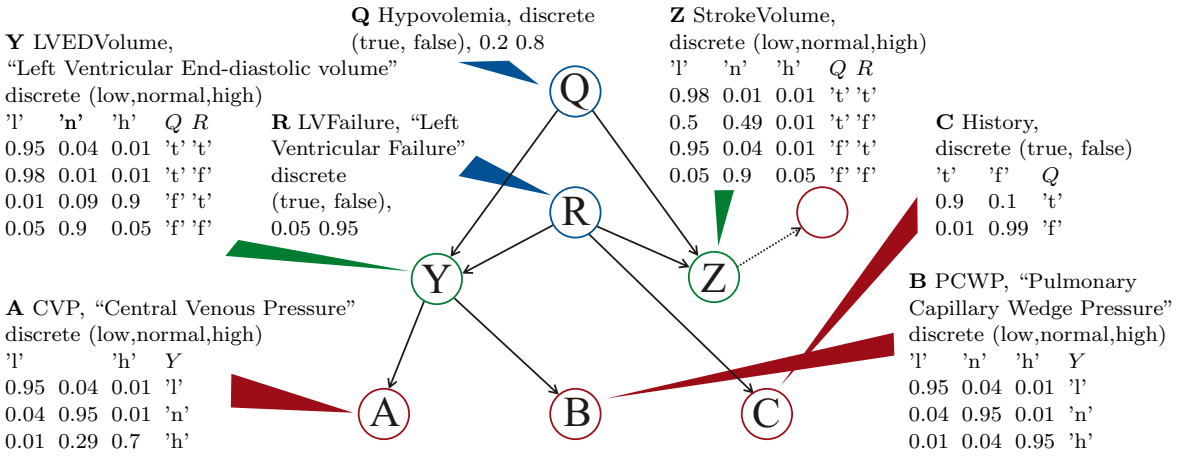

Figure 2: A detail of the Alarm network.

Having constructed a Bayesian network, various questions especially interesting for practical application can be answered by evaluating the Bayesian network under different objectives. This evaluation is known as inference in Bayesian networks, which itself is a highly complex (i.e. $\mathcal{NP}$-hard) problem. Several of the possible answerable questions are presented in e.g. [38] and [47]. We summarise some of them and use the medical application to visualise their power.

- If a Bayesian network is defined and some evidence of its variables is given, then the

state of other variables can be estimated. Based on some first symptoms, the states of some other symptoms can be estimated.

- In a Bayesian network, the joint probability of some set of its variables can be estimated. This estimation answers the questions of the probability of several symptoms to occur at the same time.

- In a Bayesian network, the most probable configuration (which is known as abductive inference) of states of variables can be estimated. This is one of the questions whose power is intuitively clear. Given some symptoms, the Bayesian network can be evaluated to get the most probable development.

- Bayesian networks can also be used to visualise complex relationships, called influence diagrams, containing decisions and actions. Nodes for treatments and nodes for symptoms can be incorporated together into one Bayesian network with which questions directly related to treatments and their effects can be answered. An example of this type of Bayesian networks is the Alarm network which combines particularly variables for diagnoses and symptoms given by measurements.

- Bayesian networks can only visualise relationships that cause no feedback. But so-called dynamic Bayesian networks can incorporate different points in time. Instead of introducing single variables, there is now one variable within a dynamic Bayesian network for each point in time.

The visualisation of relationships based on absolutely no knowledge of any causes is not only of interest for medical diagnosis but for general research. However sometimes, certain background knowledge is given as side effects are already well-analysed and well-understood. This knowledge can easily be incorporated into Bayesian networks and the questions above can be answered given some supposed knowledge. Last but not least, this supposed knowledge can be validated by the Bayesian network structure in contrast to some observed data.

Structure of this thesis

Chapter 1 introduces the statistical background of the topic this thesis deals with. Due to the focus on the application of discrete optimisation, we only present the necessary definitions and results enabling us to present our results in a clear and comprehensible way. We introduce the notion of Bayesian networks together with the Bayesian decision theory which is an application of Bayesian networks. Based on the definition of a Bayesian network we analyse possible graphical and algebraic representatives of the networks. Moreover we define the learning of Bayesian networks and the learning of Bayesian network structures. With the definition of learning we present known algorithms and complexity results. The following chapters will contribute to and deal with several of the questions appearing in Chapter 1. Therefore, we use Chapter 1 as a summary of already known results and the central theme of this thesis.

Based on already known algebraic representatives of structures of Bayesian networks we introduce new combinatorial representatives in Chapter 2. The analysis of their properties leads to some theoretical results and the property of a closer graphical relation. Additionally, theoretical results of the algebraic representatives can be added using their transformation. Chapter 2 supplements new combinatorial representatives to those presented in Chapter 1.

With respect to the presented polyhedral approach in Chapter 1 we introduce a LP-relaxation of the polytope of the new representatives of Chapter 2 in Chapter 3. This problem description enables us to solve the discrete optimisation problem, to compute the globally best structure,

presented in Chapter 1. Moreover to the introduction of a LP-relaxation we study several aspects concerning the geometric properties of the polytope of algebraic and combinatorial representatives. Using both representatives (presented in Chapter 1 and 2) the linear integer program to be solved can be described by, first, deriving general inequalities and by, second, evaluating the combinatorial structure of edges or of the complete polytope. We use the new combinatorial representatives of Chapter 2 to derive complete inequality descriptions of various subclasses of Bayesian network structures.

By the help of the new combinatorial representatives, we define in Chapter 4 the learning of Bayesian network structures to be a combinatorial problem. Furthermore we analyse the complexity of this problem and several related problems. With this analysis we reproduce several known results in a direct way and moreover derive new complexity results. These results can be added directly to those presented in Chapter 1.

In Chapter 5 we use the derived linear integer programs to compute provably optimal Bayesian network structures. Therein, we divide the learning task into restricted and unrestricted structures according to those presented in Chapter 3. The presented computational results are interpreted according to the ability of linear programming methods to solve the learning problem. Several additional simplifications are presented, as well. In this chapter we relate our computational results to other approaches and show the discrete optimisation based approach, already presented in Chapter 1, to be a useful tool to solve the learning problem to optimality.

The last Chapter 6 deals with the second topic of this thesis - conditional independence implication. We use the presented algebraic representatives to derive a new linear programming based method to solve the implication problem exactly and the conditional independence implication approximately. In the second part of this chapter, the presented linear transformation of Chapter 2 is used to reformulate the problem leading into a structurally easier problem. Computational experiments are done, whose results, first, prove the linear programming based method to be more effective than known methods and, second, show the transformation to simplify the problem tremendously.

This thesis closes in the conclusion with a summary of the presented results. Open problems which have appeared during the development of this thesis and in literature are collected and future work is outlined. To close this summary we again relate the title of this thesis and the presented results to validate the contributions of this work.

For an easier look-up of notions and definitions of terms we have added a list of symbols and an index of definitions especially useful for readers who are not familiar with this particular statistical background.

Target group and basic notation

We situate this thesis in the field of discrete optimisation. Presented statistical background and special terms from mathematical optimisation are fitted to the corresponding target group. In spite of this, the reader requires the knowledge of at least some basic definitions from probability theory, which can be found in any basic textbook introducing probability theory and statistics. We define them only informally according to [47] and [19].

A **sample space** Ω is a set of distinct **outcomes** of an experiment. An **event** $A \subseteq \Omega$ is a subset of a finite sample space and hence consists of several outcomes. If an event consists of one single outcome, $A = \{e_i\}$ and $e_i \in \Omega = \{e_1, \ldots, e_n\}$, it is called an **elementary event**. Events that constitute a partition of the sample space, e.g. events A and B with $A \cup B = \Omega$ and $A \cap B = \emptyset$, are called **mutually exclusive and exhaustive**. A **probability**

function P is a function on the power set of the sample space to the 0/1-interval with the following properties. First, the probability of an event sums up over the probabilities of all contained elementary events, $P(A) = \sum_{e_i \in A} P(\{e_i\})$. Second, the sum over the probability of all elementary events is one, i.e. $P(\{e_1\}) + \ldots + P(\{e_n\}) = 1$ for Ω above. The tuple of the sample space and the probability function (Ω, P) is called **probability space**. With this the probability of each event $E \subseteq \Omega$ is in the 0/1-interval, $P(E) \in [0, 1]$, as well, with value one if it is the **complete event**, $E = \Omega$. Moreover, for two disjoint events the probability of the union of the two is the sum of the single probabilities. Let E and F be two disjoint events, then $P(E \cup F) = P(E) + P(F)$.

We also use the notion of the **conditional probability** of one event E given another event F denoted with $P(E|F)$. This is the probability of the intersection of both events divided by the probability of event F, supposed it is non-zero: $P(E|F) = P(E \cap F)/P(F)$. With a **total probability** we denote a probability of an event F given other mutually exclusive and exhaustive events $E_1, \ldots, E_n$, which is the sum of the single probabilities $P(F \cap E_i)$, $i = 1, \ldots, n$, and $P(F) = \sum_{i=1}^{n} P(F \cap E_i)$. If $P(E_i) \neq \emptyset$, then this total probability can also be written in terms of the conditional probabilities of event F given events E_i and $P(F) = \sum_{i=1}^{n} P(F|E_i)P(E_i)$. With **Bayes' theorem** is denoted the fact, that we can compute the conditional probability of event E given event F using the conditional probability of F given E and the single probabilities. In formulas, assumed $P(E)$ and $P(F)$ are non-zero, then $P(E|F) = P(F|E)P(E)/P(F)$. This is the central conclusion Bayesian networks are based on.

A **random variable** X usually denoted with capital letters is a function of the sample space given a probability space. The random variable assigns a value to each outcome of the sample space. These values x are also called **states** of the variable X and they are typically denoted with small letters. The set of the possible states is called the **space** of the random variable. If this space is finite or countable, we call the corresponding random variable **discrete**. The corresponding probability distribution of the random variable X is the probability of the variable to have a certain state x, which is $P(X = x)$. For abbreviation, we sometimes only denote this with the probability of a state $P(x)$ or with $P(X)$ and $P(\neg X)$ if X has only binary states. The **joint probability** of two random variables X and Y is the probability of the corresponding joint event $P(X = x, Y = y)$.

Given some observations, the **relative frequency** of an event given an observation is the amount of its observed occurrences in the set of observations divided by the total number of observations. In other words, the relative frequency is the number of observed successes divided by the number of trials. In [19] the **maximum likelihood principle** (ML) is defined as the search for the parameters of the distributions for which the given data is most likely. Given a parameter, the **maximum log-likelihood estimator** of that parameter is then the value maximising the corresponding log-likelihood function which is the logarithm of the likelihood function. For example, the relative frequency is the ML-estimate of the probability of a single trial.

As random variables are functions on the sample space, we can assign the probability of a corresponding event to every probability of a random variable that has a certain state. Hence, probabilities of events and probabilities of states are in a close connection and rules for calculation probabilities of events can be used to calculate probabilities of states.

Additionally, the reader should be familiar with basic notation of linear and linear integer programming, such as the definition of a polyhedron, a polytope, a face, a facet, a LP-relaxation, the Simplex and the branch and bound method. Moreover we use the terms cone in its inner and outer description, extreme rays, monoid and Hilbert base. Furthermore, we suppose knowledge from basic complexity theory that is, particularly, the classes $\mathcal{P}$, $\mathcal{NP}$, co$\mathcal{NP}$, $\mathcal{NP}$-complete and $\mathcal{NP}$-hard and from basic graph and matroid theory.

1 Fundamentals of Bayesian networks

1.1 Bayesian networks

To understand Bayesian networks in their aim and formalism we introduce the fundamentals of Bayesian networks defined in [38], namely rule-based systems. These systems consist of a set of extended conditioned rules (an event with certainty x implies another event with certainty $p(x)$) and an "inference system combining the rules with observations to come up with a conclusion" [38]. Making this conclusion can be used to visualise personal reasoning and rational decisions enabling us to define the Bayesian decision theory.

From this we derive definitions yielding into a characterisation of Bayesian networks, which is, from a polyhedral point of view, sufficient to understand the context we are dealing with and to gain new insights and results. The results and definitions in this chapter, if not explicitly marked, are mainly based on [60] but also on [38], [39], [47], [51] and [52] and the references therein.

1.1.1 Bayesian decision theory

We follow the introduction of Jensen [38], Haddawy [26] and Savage [52] (which [26] is based on) and summarise them. We use the rule-based system introduced in Haddawy [26] to present the Bayesian decision theory which is an application of Bayesian networks. In general, the fundamental task of any kind of decision theory is a description or, if possible, a prediction of rational behaviour of an individual. This behaviour can be represented by a set of respective **actions**. For this, we have to distinguish between rational and non-rational behaviour. Obviously, rational behaviour is, besides its definition, behaviour that can be reproduced under identical circumstances, whereas non-rational behaviour has some random influence. Thus, to define rational behaviour consisting of rational actions, we have to define the given circumstances. These circumstances are represented by a set of objects (random variables) with certain values (states). The actual realisations among the possible states of the variables are unknown. Actions must be performed only on the base of these possibly uncertain or guessed states.

In general we can distinguish two types of actions. The first type consists of actions influencing other variables, called **intervening actions**, i.e. they are able to change the actual state of variables. Contrarily, the other type consists of actions called **non-intervening actions** not influencing the state of variables.

Example 2. Consider the following situation: We want to cross a street at a junction with a pedestrian light, but we cannot see whether the signal is red or green, e.g. because there is a big tree hiding it. The variables and states are defined by these circumstances and their actual realisations are unknown. We introduce variable L for the pedestrian light with states "green" and "red". Among others we have the choice to cross the street or not. This defines the set of actions G consisting of a single action with possible states "go" and "not go". ⋆

Following one intervening action will bring the variables into another set of states. The action to choose depends on what a person prefers individually as an outcome and how likely the signal is red or green. Obviously, if we know the signal to be always green, then a rational decision would be to always cross the street. A rational decision indicates to act only based on consistent orderings of preferences such that those actions are most likely whose outcome is preferred. But actions may have influence on several variables and states with possibly independent results and different gain as well.

Example 3. *Example 2, page 1, continued.* We introduce additional variables, namely an appointment A, and a car C heading for the junction. The appointment can be missed or not, because we are already late, and the car can cross the junction or wait. Following the definitions above, G is an intervening action with influence on the appointment.

If we choose not to go, then, independent on the actual colour of the traffic light, we miss the appointment. Otherwise, the car can cause us to have an accident and we again miss the appointment. This shows both the action G and the car C to have an influence on the appointment A.

But if the colour is red and the car does not stop, then waiting would prevent us from having an accident. Likewise, if we would know the car to come, then we would not cross the street, when the light shows red. This shows the action to be dependent on both the pedestrian light L and the car.

Last but not least, if the pedestrian light shows green, then the car is ought to stop and the pedestrian light has an influence directly on the car, as well.

Figure 1.1 below shows a corresponding graphical visualisation. The red node in this figure illustrates the decision (implying an action) we are faced with.

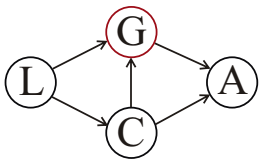

Figure 1.1: A graphical representation of the dependencies between variables of the "junction" example.

The gain of the action is determined by the strength of our wish not to miss the appointment. Since the action, as a random variable, is (conditional) dependent on other variables, the gain is also dependent on the corresponding probabilities. ⋆

This gain is called the **utility** of an action which can be seen as a not-further specified numerical and personal rating of actions given the variables which they are directly influenced by. These variables are called **determining variables**. In our running example - "junction" - above, L and C are determining variables for action G. The problem of which action is attained can now be seen as a maximisation of the utility given the determining variables. For this, we have to consider the probabilities of all determining variables and the utility of each resulting state and compute the expected utility. To bring this problem into a formalism we introduce the Bayesian decision theory.

Definition 1.1.1 (Bayesian decision theory) *Let there be a set of possible actions A, a set of possible states S of variables and a probability distribution P over S. Then $a(s)$ is the outcome of an action $a \in A$ in state $s \in S$ and $u(a)$ the attained utility of an action a. We define*

$$EU(a) = \sum_{s \in S} p(s)u(a(s))$$

to be the expected utility of the action a. An optimal action or decision is the action attaining the maximum of expected utility.

In general, $p(s)$ reflects the conditional probability of a state s given other states of variables and the action.

The network representing the Bayesian decision theory now satisfies the properties of a Bayesian network, which we define in the next section.

1.1.2 Bayesian networks - definition

Definition 1.1.2 (Bayesian network) *A **Bayesian network** (short BN) is a directed acyclic graph (short DAG) $G = (N, A)$ representing a certain discrete probability distribution P that is Markovian with respect to G and its factorisation. With the set of nodes N we identify the set of random variables $X_1, \ldots, X_{|N|}$ and the arcs A are correlations between these variables. We denote with DAGs(N) the collection of all directed acyclic graphs over the same set of random variables N and with DAGs all directed acyclic graphs with the same but not specified node set.*

Thus BNs are a graphical representation of Bayesian decision theory visualising causal influences among random variables and actions. We use this bijection of nodes in the graph and the random variables to sometimes replace random variables X_i with the name i of the corresponding node. The arcs of the graph are the direct dependencies of the variables, weighted with the conditional probabilities. For the definition of a Markovian probability distribution, see below the next example or the subsequent sections.

Example 4. The following network (Figure 1.2 below) satisfies the properties of a BN. It consists of four random variables with binary states. The BN is given by the factorisation of the corresponding probability distribution, which defines the DAG, together with the corresponding values. ⋆

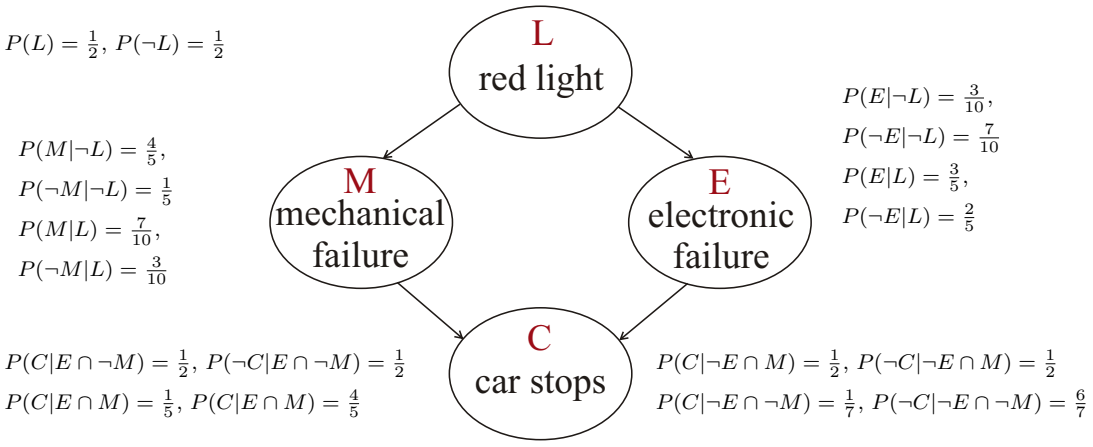

Figure 1.2: A Bayesian network with four variables.

In Definition 1.1.2 and throughout this thesis we limit ourselves on discrete random variables and on discrete probability distributions and suppose the distribution only to be Markovian. Graphically interpreted, a Markovian probability distribution is a distribution determined by direct conditionals of variables only. These direct conditionals are represented with direct connections of the corresponding nodes. Nodes that are not connected directly are not conditionally dependent. With these direct links the Markov property can also be defined in a classic way using “future”, “current” and “past” states. Let there be given a random variable

together with entering and outgoing arcs and the corresponding variables. We call the states of this particular variable the current states. The future states correspond to variables which the outgoing arcs head to and the past states correspond to variables where the entering arcs come from. A probability distribution satisfies the Markov property if the probability of future states, given all other states, is only dependent on the current states and not on the past states. The formal definition of a Markovian probability distribution requires more insight of the definitions of the underlying probability based models and is skipped to the next sections.

From this graphical representation BNs inherit both qualitative and quantitative information. The existence of the arcs determines dependence correlations and with this the qualitative information. This qualitative information is known as the **structure of the BN** (short BNS). If there is an arc between two nodes, then the corresponding variables are conditionally dependent. The quantitative information is determined by the strength of this influence visible in the corresponding conditional probabilities. These conditional probabilities are called **parameters of the BN**. Let x_i be a realisation of a random variable X_i, then the strength of all possible influences on $X_i = x_i$ is determined via **local probability distributions** $P(X_i = x_i | \bigcup_{j \neq i} (X_j = x_j))$ for all choices of sets of realisations x_j. If there is no conditional influence, then there is no corresponding set of arcs, either. Using only the existing conditional probabilities the set of possible influences on X_i reduces to the set of its parents $\mathrm{pa}_G(i)$ in the graph. This can be used to reduce likewise the number of parameters of the BN from all possible conditional probability to only the existing ones. The joint distribution $P(X_N)$ is then defined by the product of the remaining local distributions given by the underlying DAG G,

$$P(X_N) = \prod_{i \in N} P(X_i | \mathrm{pa}_G(i)).$$

For notational details, see Section 1.2.1, page 8, and the beginning of Section 1.4.1, page 19.

This joint probability of all nodes can also be used to compute the joint probability of certain subsets of nodes. Let $X_1, \ldots, X_k$ be a subset of nodes, then

$$P(X_1, \ldots, X_k) = \prod_{i=1}^{k} P(X_i | \mathrm{pa}_G(i)) \tag{1.1.1}$$

is called the **Bayesian chain rule** (compare with [39]).

Example 5. *Example 4, page 3, continued.* The product of the local distributions of the Bayesian network in Figure 1.2 results in the joint distribution, which is

$$P(X_L, X_M, X_E, X_C) = P(X_L) \cdot P(X_M | X_L) \cdot P(X_E | X_L) \cdot P(X_C | X_M, X_E).$$

The local distributions are defined by all combinations of the states of the corresponding variables. For example, the local distribution in node M is defined by the conditional probabilities $P(M|L)$ or $P(\neg M|L)$ and $P(M|\neg L)$ or $P(\neg M|\neg L)$, respectively. Since both are distributions they each sum up to one. ⋆

1.1.3 Conditional independence statements

The aim of the previous sections is to provide a straightforward graphical intuition of the topic of BNs. In this section, we derive formal definitions and give proofs of known results, if they contribute insights into used definitions and notions. On the other hand, we skip statistical formalisms whenever it is possible to gain from a simple and short introduction.

The illustration of BNs in the previous sections defines probabilistic models via dependence structures, because this is an easier way for introductory purpose. In contrast to this, we define the described probabilistic model equivalently via independence structures in this section (see e.g. [7] for a comparison and further information).

Definition 1.1.3 (conditional independence (statement)) *We assume the existence of a probability measure P over N and pairwise disjoint subsets $A, B, C \subseteq N$. A is **conditionally independent** (short CI) of B given C with respect to P if and only if*

$$P(A = a|B = b, C = c) = P(A = a|C = c).$$

*We refer to it with the **conditional independence statement** $A \perp\!\!\!\perp B \,|\, C \; [P]$.*

The set of all pairwise disjoint triplets of $A, B, C \subseteq N$, denoted with $\langle A, B|C\rangle$, is called $\mathcal{T}(N)$. We can now define a BN to be a formal **conditional independence model**. We call this shortly a CI model which is

$$\mathcal{M}_P := \{\langle A, B|C\rangle \in \mathcal{T}(N) : A \perp\!\!\!\perp B \,|\, C \; [P]\}$$

defined by the set of all CI statements valid for the corresponding probability distribution P. The corresponding factorisation of the distribution into local distributions defines the corresponding **CI structure**.

CI models can be generalised to sets of triplets of CI statements $\mathcal{M} \subseteq \mathcal{T}(N)$ not necessarily restricted to those arising as $\mathcal{M}_P$ for a probability distribution P. These sets $\mathcal{M}$ can be analysed in order to obtain properties of the CI models as well.

Definition 1.1.4 (disjoint semi-graphoid over N) *A subset $\mathcal{M} \subseteq \mathcal{T}(N)$ is called a **disjoint semi-graphoid**, if for pairwise disjoint sets $A, B, C, D \subseteq N$ the following holds:*

1. ***triviality*** $A \perp\!\!\!\perp \emptyset \,|\, C \; [\mathcal{M}]$,
2. ***symmetry*** $A \perp\!\!\!\perp B \,|\, C \; [\mathcal{M}] \Longrightarrow B \perp\!\!\!\perp A \,|\, C \; [\mathcal{M}]$,
3. ***decomposition*** $A \perp\!\!\!\perp (B \cup D) \,|\, C \; [\mathcal{M}] \Longrightarrow A \perp\!\!\!\perp D \,|\, C \; [\mathcal{M}]$,
4. ***weak union*** $A \perp\!\!\!\perp (B \cup D) \,|\, C \; [\mathcal{M}] \Longrightarrow A \perp\!\!\!\perp B \,|\, (D \cup C) \; [\mathcal{M}]$,
5. ***contraction*** $A \perp\!\!\!\perp B \,|\, (D \cup C) \; [\mathcal{M}] \wedge A \perp\!\!\!\perp D \,|\, C \; [\mathcal{M}] \Longrightarrow A \perp\!\!\!\perp (B \cup D) \,|\, C \; [\mathcal{M}]$.

We use the short notation $[\mathcal{M}]$ in the CI statements to abbreviate $\langle A, B|C\rangle \in \mathcal{M}$.

In this definition the use of $\mathcal{M}$ instead of $\mathcal{M}_P$ is important as not every set of disjoint triplets describes a CI model $\mathcal{M}_P$. But this statement is true for the other direction.

Lemma 1.1.5 (Lemma 2.1 in [60]) *Every CI model $\mathcal{M}_P$ induced by a probability measure P over N is a disjoint semi-graphoid over N.*

The proof of this fact is based on measurable spaces and Σ-algebras and we refer to [60].

By using the semi-graphoid properties certain special CI statements can be defined, which are sufficient and necessary for the existence of other statements. These CI statements are called elementary.

Definition 1.1.6 (elementary CI statements) *An **elementary CI statement** (or elementary statement) $A \perp\!\!\!\perp B \,|\, C\ [o]$ induced by a not specified object o is an (elementary) triplet $\langle A, B|C\rangle$, where $A = \{a\}$ and $B = \{b\}$ are single variables of N. We use CI statements of type $a \perp\!\!\!\perp b \,|\, C\ [o]$ as elementary statements.*

Due to Lemma 1.1.5 this object o can be the underlying probability distribution P or equivalently the independence model $\mathcal{M}$. Therefore, we skip this object in the statement if it is not necessary to specify.

We use the elementary CI statements to detect additional CI statements valid for the same CI model. These CI statements are also valid for the probability distribution but are not explicitly stated within the definition of the model.

Lemma 1.1.7 (Lemma 2.2 in [60]) *Suppose, $\mathcal{M}$ is a disjoint semi-graphoid over N. For any disjoint triplet $\langle A, B|C\rangle$ over N, the statements $A \perp\!\!\!\perp B \,|\, C\ [\mathcal{M}]$ are valid if and only if the following conditions hold*

$$\forall a \in A,\ \forall b \in B,\ \forall D:\ C \subseteq D \subseteq (A \cup B \cup C) \setminus \{a, b\} \text{ we have } a \perp\!\!\!\perp b \,|\, D\ [\mathcal{M}]. \tag{1.1.2}$$

Proof. We follow the proof given in [60]. Let $A \perp\!\!\!\perp B \,|\, C\ [\mathcal{M}]$ be given. We apply the Decomposition property (Property 3 in Definition 1.1.4) to the set B which can be separated into $B := B \setminus \{b_i\}$ and $D := \{b_i\}$ for any singleton $b_i \in B$. This implies the statements $A \perp\!\!\!\perp b_i \,|\, C\ [\mathcal{M}]$, $\forall b_i \in B$. Applying the Symmetry property (Property 2 in Definition 1.1.4,) yields $b_i \perp\!\!\!\perp A \,|\, C\ [\mathcal{M}]$, $\forall b_i \in B$. In a likewise way we apply the Decomposition property with singletons $a_j \in A$ to these statements in order to get $a_j \perp\!\!\!\perp b_i \,|\, C\ [\mathcal{M}]$, $\forall b_i \in B$ and $\forall a_j \in A$. With the same reason, we can apply the Weak union property (Property 4 in Definition 1.1.4) to $A \perp\!\!\!\perp B \,|\, C\ [\mathcal{M}]$ with $B := B \setminus \{b_i\}$ and $D := \{b_i\}$ and get $A \perp\!\!\!\perp B \setminus \{b_i\} \,|\, (C \cup \{b_i\})\ [\mathcal{M}]$, $\forall b_i \in B$. We get the set of Statements (1.1.2) above if we apply combinations of these arguments.

In order to show the opposite direction, we assume the set of Statements (1.1.2) to be valid. Let A, B, C be a non-trivial combination of subsets of N. The implication can now be shown by induction on $|A \cup B|$. If $|A \cup B| = 2$, then both sets are singletons and Statements (1.1.2) are identical to $a \perp\!\!\!\perp b \,|\, C$ and therefore to $A \perp\!\!\!\perp B \,|\, C\ [\mathcal{M}]$. Because of this, we suppose $|A \cup B| > 2$ and A or B to have more than one element. Let w.l.o.g. $|B| \geq 2$ be given and we set $B' := B \setminus \{b\}$. For B' we now have

$$\forall a \in A,\ \forall b' \in B',\ \forall D:\ C \subseteq D \subseteq (A \cup B' \cup C) \setminus \{a, b'\} \implies a \perp\!\!\!\perp b' \,|\, D\ [\mathcal{M}].$$

We can apply the induction hypothesis to these statements to obtain $A \perp\!\!\!\perp B' \,|\, C\ [\mathcal{M}]$. On the other hand, we define $\tilde{B} := \{b\}$ and evaluate

$$\forall a \in A,\ \forall b \in \tilde{B} = \{b\},\ \forall D:\ C \subseteq D \subseteq (A \cup B' \cup C) \setminus \{a\} \implies a \perp\!\!\!\perp b \,|\, D\ [\mathcal{M}]$$

with respect to $D := C \cup B'$ to get $A \perp\!\!\!\perp b \,|\, (C \cup B')\ [\mathcal{M}]$. The Contraction property (Property 5 in Definition 1.1.4) can be applied to $A \perp\!\!\!\perp B' \,|\, C\ [\mathcal{M}]$ and $A \perp\!\!\!\perp b \,|\, (C \cup B')\ [\mathcal{M}]$, hence $A \perp\!\!\!\perp B \,|\, C\ [\mathcal{M}]$ is obtained. □

There exists a graphical translation of CI statements, because DAGs are used to represent BNs and BNs are CI models. This translation is called the d-separation criterion to derive valid CI statements encoded within a DAG (see Section 6.2.4, page 119, for more information).

Example 6. Let $A \perp\!\!\!\perp B \,|\, C\ [G]$ be a CI statement encoded within a DAG G (Figure 1.3, next page). A and B are said to be conditionally independent given C, if every directed path

from nodes of A to nodes of B contains nodes of set C. In particular, this means that there is no direct arc connecting (any node of) A and B.

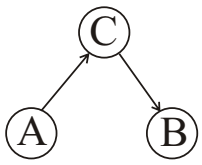

Figure 1.3: The d-separation criterion for a graph G and statement $A \perp\!\!\!\perp B \,|\, C\ [G]$.

More precisely, the direction of the implication is: If every path from nodes of set A to nodes of B contains nodes of C, then the CI statement $A \perp\!\!\!\perp B \,|\, C\ [G]$ is valid for G. The direction of the implications becomes clear in the next definition. $\star$

From this implication we can generalise the definition of a Markovian measure.

Definition 1.1.8 (Markovian measure, perfectly Markovian measure)
Let P be a probability measure over node set N. Furthermore, let $\mathcal{T}(N)$ define appropriate subsets A, B and C. Then, P is defined to be a ***Markovian measure*** *with respect to a DAG G if*

$$A \perp\!\!\!\perp B \,|\, C\ [G] \Longrightarrow A \perp\!\!\!\perp B \,|\, C\ [P],\ \forall \langle A, B|C\rangle \in \mathcal{T}(N). \tag{1.1.3}$$

If, in addition, $A \perp\!\!\!\perp B \,|\, C\ [P]$ implicates $A \perp\!\!\!\perp B \,|\, C\ [G]$, we call P a ***perfectly Markovian measure****.*

Thus DAGs are appropriate to define conditional independence models, too. For a Markovian measure each valid CI statement of the graph is also valid for the distribution. Contrariwise, the existence of arcs in the graphs implies conditional dependence. Therefore, the more CI statements are valid in the graph the fewer edges exist. But a Markovian measure might contain further valid CI statements which are not valid for the graph. This follows from the direction of the implication. Therefore any graph representing a Markovian measure might contain more edges than necessary to represent the distribution. Equivalently each addition of an arc in the DAG eliminates a valid CI statement, which might still be valid for the measure. In particular, a not perfectly Markovian measure is a distribution which might not be representable by a DAG.

In 1990, Geiger and Pearl showed the existence of a Markovian and also a perfectly Markovian measure for every BN which is a DAG and thus a special graph. On the other hand, this measure defines a corresponding CI model. Let G be a DAG, then the CI model of the encoded measure is likewise defined via

$$\mathcal{M}_G := \{\langle A, B|C\rangle \in \mathcal{T}(N) : A \perp\!\!\!\perp B \,|\, C\ [G]\}.$$

Thus, every BN specified by a DAG induces a CI model. To this CI model we can apply the semi-graphoid properties.

From the graphical intuition of a Markovian measure, namely direct arcs determine the dependence among variables, we get the probability distribution of a BN to be decomposable into local distributions. This implies the Markov property defined in Section 1.1.2 as only direct influences of states constitute the distribution. The structure of the graphical network determines the combination of these local dependencies to the joint probability distribution.

1.1.4 Markov equivalence

As derived in the previous section the semi-graphoid properties can also be applied to CI statements represented in graphs.

Example 7. *Example 6, page 6, continued.* Consider again a graph G representing a CI statement $A \perp\!\!\!\perp B \,|\, C$ $[G]$ (Figure 1.3 or equivalently Figure 1.4 below). Applying the symmetry property (Property 2 in Definition 1.1.4) yields also statement $B \perp\!\!\!\perp A \,|\, C$ $[G]$ which is represented in Figure 1.5 below. If A, B and C are single nodes, then we can observe the two different DAGs (Figures 1.4 and 1.5 below) to encode the same CI model.

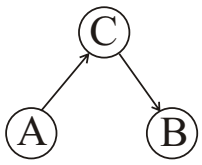

Figure 1.4: A graph G with a valid CI statement $A \perp\!\!\!\perp B \,|\, C$ $[G]$.

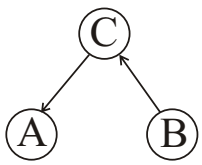

Figure 1.5: A graph G' with a valid CI statement $B \perp\!\!\!\perp A \,|\, C$ $[G']$.

Both graphs have $A \perp\!\!\!\perp B \,|\, C$ and $B \perp\!\!\!\perp A \,|\, C$ as valid CI statements. ⋆

This property can be generalised to the concept of Markov equivalence.

Definition 1.1.9 (Markov equivalent) *Two DAGs G and H are called **Markov equivalent** if and only if their classes of Markovian measures coincide, i.e. they induce the same formal independence model.*

In other words two DAGs are Markov equivalent if and only if the same CI statements are valid. Markov equivalence forms an equivalence class, which is easy to show. We introduce special structures DAGs have in order to analyse graphical criteria on Markov equivalent DAGs (see Section 1.2.1 below for more information). A thorough analysis of this topic can be found in [47].

1.2 Bayesian network structures

The BNS represents the factorisation of a distribution into its local distributions. The local distributions are defined by single arcs, and hence, the underlying DAG can be identified with the structure of the BN. Therefore, we analyse the DAG with respect to the encoded probability distribution, but also vice versa.

1.2.1 Graphical representation

We define special graphical structures a graph can have, which we later use to analyse properties of the models they encode.

Definition 1.2.1 (parent set, child, sink, underlying graph, immorality, clique) *Let $G = (N, A)$ be a DAG and $a \in N$ a node. All nodes $b \in N \setminus \{a\}$ with arcs to a $(b \to a)$ together are called the **parent set** of **child** a, or short* $\mathrm{pa}(a) := \{b \in N \setminus \{a\} : \{b, a\} \in A\}$. *For a general set of nodes a **sink** is defined to be* $\mathrm{sink}(T) = \{i \in T : \nexists j \in T \setminus \{i\} \text{ with } i \to j\}$. $\bar{G}$

is called the ***underlying*** *(undirected)* ***graph*** *of DAG G, if it is obtained from G by removing the directions of arcs. An* ***immorality*** *is an induced subgraph with distinct nodes a, b and c, if both $b, c \in \mathrm{pa}(a)$, but $\{b, c\}, \{c, b\} \notin A$. A* ***clique*** *is an induced subgraph of nodes T in which $\forall i \neq j \in T$ the edge or arc $\{i, j\}$ exist. We often identify a clique and an immorality simply with the inducing set of nodes T.*

In Section 1.1.4 we have observed two different DAGs to be equal with respect to the CI model they encode. An important result is, that two DAGs can only define the same CI model if they have the same immoralities in common, hence immoralities are a distinctive feature of Markov equivalent graphs.

Lemma 1.2.2 (Theorem 1 in [66]) *Two DAGs G and H are Markov equivalent if and only if they have the same underlying undirected graph and the same immoralities.*

Two such graphs with the same underlying undirected graph and immoralities can be transformed into each other by so-called **legal arrow reversals**. An arc $a \to b$ in G can be legally reversed to obtain H if $\mathrm{pa}(b) = \mathrm{pa}(a) \cup \{a\}$ holds in G. Two DAGs are Markov equivalent if and only if they can be obtained from each other by a sequence of such arrow reversals.

From the fact, that different DAGs may describe the same CI model, we imply the graphical representation of BNs via DAGs not to be an appropriate choice. The representation of BNs with DAGs is simply not bijective and there is a need for better graphical representatives. In literature (Verma and Pearl [66] in 1991 and Andersen, Madigan and Perlman [1] in 1997), two different concepts were introduced. Both representations are based on the analysis of common and distinctive graphical features of Markov equivalent DAGs. Such a feature is e.g. the already mentioned immorality.

1.2.1.1 Pattern graph

The first concept is based on Lemma 1.2.2 and directly translates this equivalence into a definition of a mixed graph.

Definition 1.2.3 (pattern) *The* ***pattern*** *graph of a Markov equivalence class of DAGs is a mixed graph having the same underlying undirected graph and the same immoralities as all DAGs in that class have.*

A simple construction of the pattern $\mathrm{pat}(G)$ of corresponding DAGs G follows a procedure polynomial in the number of nodes. For every node, we either have to add incident arcs, if they are part of any immorality, or add them as undirected edges, otherwise. Contrarily, if a pattern of a Markov equivalence class is given, then we can reconstruct every contained DAG by directing the undirected edges. This procedure of directing edges has to satisfy the condition, that newly directed edges should not create additional immoralities or directed cycles. This restriction is important because all immoralities must be specified already in the pattern and the resulting directed graph must be acyclic. Finding one DAG in the class of Markov equivalent graphs is polynomial in the number of nodes as well.

Example 8. Let the DAG G be given in Figure 1.6, next page. We observe the set of nodes $\{A, B, C\}$ to induce an immorality, as both arcs $B \to A$ and $A \leftarrow C$ exist and B and C are not adjacent. The arcs in immorality $\{A, B, C\}$ remain directed in $\mathrm{pat}(G)$ (see Figure 1.7, next page). In a likewise manner, we observe $\{A, D, E\}$ to induce a clique of size three, as arcs between all contained pairs of nodes exist. This is not an immorality and arcs $A \to D$, $D \to E$ and $A \to E$ are added as undirected edges to $\mathrm{pat}(G)$.

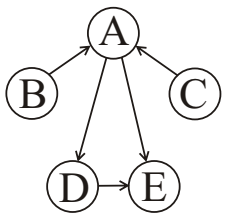

Figure 1.6: A DAG G over five nodes.

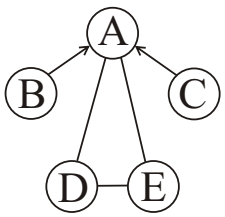

Figure 1.7: The pattern pat(G) of DAG G.

For the back direction, to reconstruct the Markov equivalent DAGs on the base of the pattern graph, we consider the first graph pat(G) given in Figure 1.8 below. The red arcs $B \to A$ and $C \to A$ are arcs within an immorality, which all DAGs G of the Markov equivalence class given by pat(G) share (compare with the second, third and fourth graph in Figure 1.8).

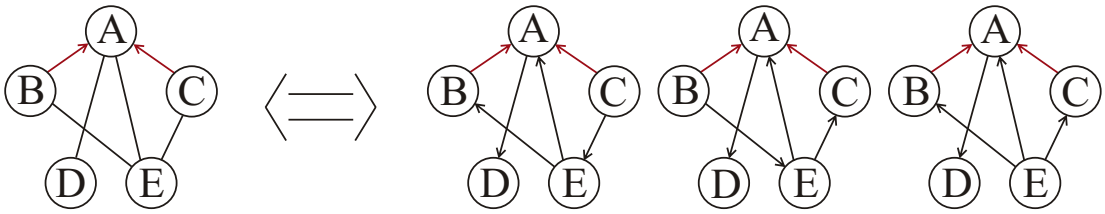

Figure 1.8: A pattern of Markov equivalent DAGs over five nodes.

To reconstruct the DAGs G induced by pat(G) we start in node A. The incident edge $A - D$ has to be directed $A \to D$ in all graphs G, as a converse direction would create additional immoralities $\{A, B, D\}$ and $\{A, C, D\}$. Analysing edges $B - E$ and $C - E$ we observe, that if both are directed $B \to E$ and $C \to E$ then an additional immorality is created. Therefore at least one of the arcs has to be outgoing in E and we separate pat(G) into graphs G_1 with both $B \leftarrow E$ and $E \to C$ (the second graph in Figure 1.8), G_2 with arcs $B \to E$ and $E \to C$ (the third graph in Figure 1.8) and G_3 with arcs $B \leftarrow E$ and $E \to C$ (the fourth graph in Figure 1.8). For the last undirected edge $A - E$ we observe, that if it is directed $A \to E$, then in each of the graphs G_1, G_2 and G_3 a directed cycle is constructed, which is $A \to E$, $E \to B$, $B \to A$ for G_1, $A \to E$, $E \to C$, $C \to A$ for G_2 and both $A \to E$, $E \to B$, $B \to A$ and $A \to E$, $E \to C$, $C \to A$ for G_3. This causes the choice of the direction of $A - E$ to be unique $E \to A$ for all the DAGs G_1, G_2 and G_3. ⋆

In the example above we can already see, that although cycles of length three, i.e. cliques $\{A, B, E\}$ and $\{A, C, E\}$ of pat(G), are not immoralities there exist nodes within these cycles (node A in both cycles) with at least two entering arcs in all DAGs induced by pat(G). This generalises to the decomposition of larger cycles into cliques of size three, which will be useful later.

1.2.1.2 Essential graph

The other graphical representation of a Markov equivalence class is based on the pattern but inherits more information about the contained DAGs. DAGs of one equivalence class may have, additionally to those of an immorality, some arcs in common. These arcs are implicitly determined by the immoralities, as a redirection of them would change the amount of immoralities and thus the Markov equivalence class itself. Therefore, they are called protected.

Definition 1.2.4 (protected edge) *Let a Markov equivalence class of DAGs G be given. An edge/arc $\{i, j\}$ with $i \neq j$ contained in any G is called* ***protected*** *if the arc $i \to j$ is contained in all DAGs G of the Markov equivalence class.*

We can apply this definition of protected edges, of course, to the arcs within an immorality, too.

Definition 1.2.5 (essential graph) *The **essential graph** of a Markov equivalence class of DAGs is the pattern graph of that class in which all protected edges of that class are directed.*

Analogously to the pattern graph of G, there exist simple procedures for a construction of the essential graph G^* from the DAGs of the Markov equivalence class and for a reconstruction of the DAGs of the class from the essential graph.

Example 9. *Example 8, page 9, continued.* We use the DAG G presented in Figure 1.9 below to construct its essential graph G^*. We use Example 8 to find pat(G) first (compare with Figure 1.10 below).

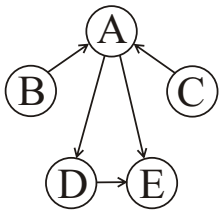

Figure 1.9: A DAG G over five nodes.

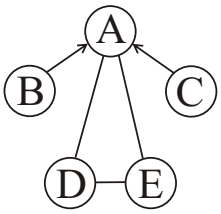

Figure 1.10: The pattern pat(G) of DAG G.

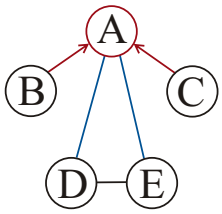

Figure 1.11: Undirected but protected edges in pat(G).

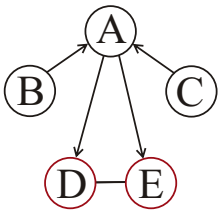

Figure 1.12: The direction of protected edges in the essential graph G^* of G.

Having computed pat(G) we iterate over all nodes which have entering arcs. We start at a node that only has entering arcs and incident undirected edges. In Figure 1.11 above, this is node A. Since the incident arcs are elements of an immorality (red coloured arcs) we need to direct all incident undirected edges as outgoing arcs (blue coloured edges), since otherwise the amount of immoralities is changed. This affects both edges $\{A, B\}$ and $\{A, E\}$. We iterate over the resulting graph (Figure 1.12 above) and search again such a node (nodes D and E). From now on entering arcs might not be part of an immorality as we have already directed some in the first step. In spite of this fact, we use the same reason, again, to direct the incident undirected edges as outgoing, if they would change the immoralities. In our example the edge $\{D, E\}$ can be directed in both ways creating or destroying no immoralities. This edge can be left undirected and Figure 1.12 represents G^*. ⋆

There exist various other methods, all based on the same rule for orienting the edges. See Section 2.2.3, page 46, for more information on this. In Figure 1.13 below the essential graph corresponding to the pattern graph given in Example 8 and therein in Figure 1.8 is constructed by detecting additional protected edges (blue coloured).

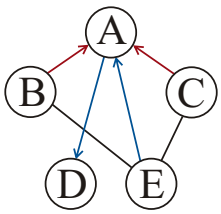

Figure 1.13: An essential graph of Markov equivalent DAGs over five nodes.

Both concepts, pattern and essential graphs, fully characterise the class of Markov equivalent DAGs due to Lemma 1.2.2.

Corollary 1.2.6 *Two DAGs G and H are Markov equivalent if and only if they have the same pattern and the same essential graph, respectively.*

1.2.2 Algebraic representation - imsets

Studený [60] introduces an algebraic understanding of BNs in terms of standard imsets which gains on one hand from an easy graphical understanding and on the other hand from a statistical context brought into algebraic formalism. This yields a geometric understanding of learning BN structures presented in [65] from which new results and insights can be obtained.

1.2.2.1 Standard imsets

Standard imsets are algebraic representations of the class of Markov equivalent DAGs. They are special integer valued functions $\mathbf{u} : \mathcal{P}(N) \to \mathbb{Z}^{2^{|N|}}$, called **imsets** (a shortcut of "**i**nteger valued **m**ulti**set**"), on the power set of N.

We use the special imset $\delta : N \to \{0,1\}^{2^{|N|}}$ with $\delta_A(S) = 1$, if and only if $S = A$, and $\delta_A(S) = 0$, if and only if $S \neq A$, which is called the **identifier of a subset** A of N.

Definition 1.2.7 (standard imset) *Let G be a DAG over a non-empty node set N. An imset $\mathbf{u} : \mathcal{P}(N) \to \mathbb{R}$ is called a **standard imset** if it has the following form:*

$$\mathbf{u}(G) = \delta_N - \delta_\emptyset + \sum_{i \in N} \left(\delta_{\mathrm{pa}(i)} - \delta_{\{i\} \cup \mathrm{pa}(i)}\right). \tag{1.2.1}$$

We use the standard imset $\mathbf{u}(G)$ as an integral vector of dimension $\mathbb{Z}^{2^{|N|}}$ indexed by subsets of N and defined by a graph G. Let two graphs G_1 and G_2 be given each consisting of only one single arc between the same nodes, say $i \to j$ in G_1 and $i \leftarrow j$ in G_2. For G_1 we observe entries $\delta_{\mathrm{pa}_{G_1}(i)}(\emptyset) = 1$, $\delta_{\mathrm{pa}_{G_1}(j)}(i) = 1$, $\delta_{\mathrm{pa}_{G_1}(i) \cup \{i\}}(\{i\}) = 1$ and $\delta_{\mathrm{pa}_{G_1}(j) \cup \{j\}}(\{i,j\}) = 1$, and for G_2 we observe $\delta_{\mathrm{pa}_{G_2}(j)}(\emptyset) = 1$, $\delta_{\mathrm{pa}_{G_2}(i)}(j) = 1$, $\delta_{\mathrm{pa}_{G_2}(j) \cup \{j\}}(\{j\}) = 1$ and $\delta_{\mathrm{pa}_{G_2}(i) \cup \{i\}}(\{i,j\}) = 1$. In both graphs we get with respect to Equation (1.2.1) above:

$$G_1 : \delta_{\mathrm{pa}_{G_1}(j)} - \delta_{\mathrm{pa}_{G_1}(j) \cup \{j\}} + \delta_{\mathrm{pa}_{G_1}(i)} - \delta_{\mathrm{pa}_{G_1}(i) \cup \{i\}} = \delta_{\mathrm{pa}_{G_1}(i)} - \delta_{\mathrm{pa}_{G_1}(j) \cup \{j\}} = \delta_\emptyset - \delta_{\{i,j\}},$$
$$G_2 : \delta_{\mathrm{pa}_{G_2}(j)} - \delta_{\mathrm{pa}_{G_2}(j) \cup \{j\}} + \delta_{\mathrm{pa}_{G_2}(i)} - \delta_{\mathrm{pa}_{G_2}(i) \cup \{i\}} = \delta_{\mathrm{pa}_{G_2}(j)} - \delta_{\mathrm{pa}_{G_2}(i) \cup \{i\}} = \delta_\emptyset - \delta_{\{i,j\}}.$$

Independent from the original direction of the arc between i and j, the standard imset $\mathbf{u}(G_1) = \mathbf{u}(G_2) = \delta_\emptyset - \delta_{\{i,j\}}$ contains an entry for the edge $\{i,j\}$ and the empty set. With this, the standard imsets of both Markov equivalent graphs are equal and they fully characterise the Markov equivalence class.

Corollary 1.2.8 (Corollary 7.1 in [60]) *Let G and H be two DAGs over N with classes (models) $\mathcal{M}_G$ and $\mathcal{M}_H$. Then $\mathcal{M}_G = \mathcal{M}_H$ if and only if $\mathbf{u}(G) = \mathbf{u}(H)$.*

Standard imsets can directly encode CI statements. Every statement $A \perp\!\!\!\perp B \,|\, C$ for pairwise disjoint sets A, B, C is associated with the imset

$$\mathbf{u}(\langle A, B|C \rangle) = \delta_{A \cup B \cup C} + \delta_C - \delta_{A \cup C} - \delta_{B \cup C}.$$

Elementary independence statements are special CI statements, likewise there exist special standard imsets called elementary imsets.

Definition 1.2.9 (elementary imsets) *Given a set of nodes N and an elementary triplet $\langle a, b|C\rangle$ with $C \subseteq N$ and $a \neq b \in N \setminus C$. The corresponding **elementary imset** is defined as*

$$\mathbf{u}(\langle a, b|C\rangle) = \delta_{\{a,b\}\cup C} + \delta_C - \delta_{\{a\}\cup C} - \delta_{\{b\}\cup C}. \tag{1.2.2}$$

The set of all elementary imsets over N is denoted with $\mathcal{E}(N)$.

It remains for us to reformulate Equation (1.2.2) above in terms of Equation (1.2.1).

Corollary 1.2.10 (compare with [60]) *Let $a \neq b \in N \setminus C$ and $C \subseteq N$ be given. An elementary imset $\mathbf{u}(\langle a, b|C\rangle)$ is also a standard imset.*

Proof. Let the remaining nodes of N be denoted with $\{i_1, \ldots, i_l\} := N \setminus (C \cup \{a, b\})$ and the elements of C with $\{c_1, \ldots c_k\}$. Then we can reformulate Equation (1.2.2) by adding zeros to obtain an equation which has the form of Equation (1.2.1).

$$\begin{aligned}
\mathbf{u}(\langle a, b|C\rangle) &= \delta_{\{a,b\}\cup C} + \delta_C - \delta_{\{a\}\cup C} - \delta_{\{b\}\cup C} \\
&= \delta_N - \delta_\emptyset + (\delta_C - \delta_{\{a\}\cup C}) + (\delta_C - \delta_{\{b\}\cup C}) + \delta_{\{a,b\}\cup C} - \delta_N + \delta_\emptyset - \delta_C \\
&= \delta_N - \delta_\emptyset + (\delta_C - \delta_{\{a\}\cup C}) + (\delta_C - \delta_{\{b\}\cup C}) + (\delta_{N\setminus\{i_1\}} - \delta_N) \\
&\quad + (\delta_{N\setminus\{i_1,i_2\}} - \delta_{N\setminus\{i_1\}}) + \ldots + (\delta_{\{a,b\}\cup C} - \delta_{\{a,b\}\cup\{i_l\}\cup C}) + (\delta_\emptyset - \delta_{c_1}) \\
&\quad + (\delta_{c_1} - \delta_{\{c_1,c_2\}}) + \ldots + (\delta_{C\setminus\{c_k\}} - \delta_C).
\end{aligned}$$

□

As elementary imsets are standard imsets we can interpret the elementary triplet $\langle a, b|C\rangle$ with respect to a graph G given in the proof above.

Example 10. Let $N := \{A, B, C, D, E\}$ be given. A DAG G over nodes N representing the (Markov equivalence class defined by the) elementary triplet $\langle A, B|\{C, D\}\rangle$ is given by Figure 1.14 below.

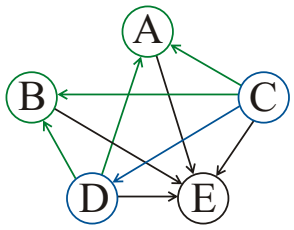

Figure 1.14: A DAG over five nodes representing the elementary imset $\mathbf{u}(\langle A, B|\{C, D\}\rangle)$.

In this graph only nodes A, B are not connected. Moreover the graph G is a DAG due to the iterative definition of parent nodes in the proof of Corollary 1.2.10. This yields the general conclusion that elementary imsets $\mathbf{u}(\langle a, b|C\rangle)$ can be represented by DAGs in which only arcs between a and b are missing. ⋆

Elementary imsets are integral vectors, because standard imsets are, by definition, integral vectors. Together with the zero imset we identify them as extreme rays $\mathcal{E}(N)$ of a pointed rational polyhedral cone $\mathcal{R}(N) := \text{cone}(\mathcal{E}(N))$. The next proposition gives the reason for this. It shows every imset of a CI statement to be expressible as a non-zero integral combination of elementary imsets. Imsets of CI statements are therefore elements of the monoid generated by the elements of $\mathcal{E}(N)$ and $\text{cone}(\mathcal{E}(N))$ can be interpreted as a pointed rational polyhedral cone.

Proposition 1.2.11 (Proposition 4.2 in [60]) *Every imset of a CI statement can be written as a non-negative integral combination of elementary imsets.*

Proof. To show this proposition, we first have to prove two formulas, similar to those of the semi-graphoid property. Let $\mathbf{u}(\langle A, B|C\rangle)$ be a non-zero imset corresponding to a CI statement. Let D be an appropriate disjoint subset, then we have

$$\begin{aligned}\mathbf{u}(\langle A, B\cup D|C\rangle) &= \delta_{A\cup B\cup D\cup C} + \delta_C - \delta_{A\cup C} - \delta_{B\cup D\cup C}\\ &= (\delta_{A\cup B\cup D\cup C} - \delta_{B\cup D\cup C} + \delta_{D\cup C} - \delta_{A\cup D\cup C}) + (\delta_{A\cup D\cup C} - \delta_{D\cup C}\\ &\quad + \delta_C - \delta_{A\cup C}) \;=\; \mathbf{u}(\langle A, B|D\cup C\rangle) + \mathbf{u}(\langle A, D|C\rangle).\end{aligned} \tag{1.2.3}$$

On the other hand we analogously get

$$\begin{aligned}\mathbf{u}(\langle A\cup D, B|C\rangle) &= \delta_{A\cup D\cup B\cup C} + \delta_C - \delta_{A\cup D\cup C} - \delta_{B\cup C}\\ &= (\delta_{A\cup D\cup B\cup C} - \delta_{A\cup D\cup C} + \delta_{D\cup C} - \delta_{B\cup D\cup C}) + (\delta_{D\cup B\cup C} - \delta_{D\cup C}\\ &\quad + \delta_C - \delta_{B\cup C}) \;=\; \mathbf{u}(\langle A, B|D\cup C\rangle) + \mathbf{u}(\langle D, B|C\rangle).\end{aligned} \tag{1.2.4}$$

With both formulas together we can decompose $\mathbf{u}(\langle A, B|C\rangle)$ into elementary imsets by iteratively considering $A\setminus\{a_i\}$ and $B\setminus\{b_j\}$. Hence, let $A := \{a_1, \ldots, a_k\}$ and $B := \{b_1, \ldots, b_k\}$ be given. We define $C_{k'l'} := C \cup \left(\bigcup_{j=1}^{k'}\{a_i\}\right) \cup \left(\bigcup_{i=1}^{l'}\{b_j\}\right)$ for $0 \le k' < k$ and $0 \le l' < l$ and overall conclude, that

$$\mathbf{u}(\langle A, B|C\rangle) = \sum_{\substack{0\le k'<k,\\ 0\le l'<l}} \mathbf{u}(\langle a_{k'+1}, b_{l'+1}|C_{k'l'}\rangle),$$

with possible repetitions of elementary imsets $\mathbf{u}(\langle a_{k'+1}, b_{l'+1}|C_{k'l'}\rangle)$. □

Standard imsets of CI statements are special integral elements of the monoid spanned by the elementary imsets. The following definition characterises the set of all elements of the monoid and therefore a superset of the standard imsets of CI statements.

Definition 1.2.12 (combinatorial imset) *Let* $\mathbf{u}$ *be an imset over* N *and* $\mathcal{E}(N)$ *the set of elementary imsets. Every* $\mathbf{u}$ *that satisfies*

$$\mathbf{u} = \sum_{\mathbf{v}\in\mathcal{E}(N)} \alpha_{\mathbf{v}}\mathbf{v}, \text{ where } \alpha_{\mathbf{v}} \in \mathbb{Z}_+,$$

is called a ***combinatorial imset****. The set of combinatorial imsets is denoted with* $\mathcal{C}(N)$*.*

Last but not least, we call every integral element $\mathbf{u}$ of $\mathcal{R}(N)$ a **structural imset**. The set of structural imsets is denoted with $\mathcal{S}(N) := \mathcal{R}(N) \cap \mathbb{Z}^t$, $t = 2^{|N|}$. All these imsets are elements of the cone for which there exists an integer $n \in \mathbb{N}$ such that $n\cdot\mathbf{u} = \sum_{\mathbf{v}\in\mathcal{E}(N)} \alpha_{\mathbf{v}}\mathbf{v}$ for non-negative integers $\alpha_{\mathbf{v}} \in \mathbb{Z}_+$.

Summing up, elementary imsets are extreme rays of a cone $\mathcal{R}(N)$, combinatorial imsets are the elements of the monoid generated by the elementary imsets, structural imsets are the integral elements of $\mathcal{R}(N)$ and standard imsets are special elements of the monoid above and they contain all elementary imsets.

1.2.2.2 Inclusion neighbours

Markov equivalence defines two models to be equal and corresponding properties of the graphical representation can be obtained. Standard imsets can be used to define another

question concerning CI models, which deals with the properties of models included in one another.

Definition 1.2.13 (inclusion neighbours, upper/lower neighbour) *Let K and L be two DAGs with corresponding CI models $\mathcal{M}_K$ and $\mathcal{M}_L$ and node set N. If $\mathcal{M}_L \subseteq \mathcal{M}_K$ and if there exists no $H \in$ DAGs such that $\mathcal{M}_L \subsetneq \mathcal{M}_H \subsetneq \mathcal{M}_K$, then we call K an **upper neighbour** of L and L an **lower neighbour** of K. Both are called **inclusion neighbours**.*

The inclusion neighbourhood of a DAG is the union of all its upper and lower neighbours. Inclusion neighbours are determined by validity of additional CI statements. If we use K, L like above, then there exists an additional conditional independence statement valid for K which is not valid for L. Moreover, any probability distribution determined by a DAG is contained in any lower neighbour of this DAG, as all of these lower neighbours contain fewer CI statements. If we use the graphical interpretation instead, then lower neighbours correspond to addition of arcs and upper neighbours to deletion of arcs. These additions and deletions of arcs translate into additions and subtractions of elementary imsets, which yields an easy algebraic characterisation of inclusion neighbourhood.

Proposition 1.2.14 (Proposition 8.3 in [60]) *Let K, L be two DAGs over N such that L is obtained from K by removal of an arc $a \to b$ of K. Then $\mathbf{u}(L) - \mathbf{u}(K) = \mathbf{u}(\langle a, b|C\rangle)$, where $C = \mathrm{pa}_K(b) \setminus \{a\} = \mathrm{pa}_L(b)$.*

Proof. We have $\mathrm{pa}_K(c) = \mathrm{pa}_L(c)$, $\forall c \in N \setminus \{b\}$, $\mathrm{pa}_L(b) = C$ and $\mathrm{pa}_K(b) = C \cup \{a\}$ according to the definitions of K, L. With this, we can analyse $\mathbf{u}(L) - \mathbf{u}(K)$.

$$\begin{aligned}\mathbf{u}(L) - \mathbf{u}(K) = {} & \delta_N - \delta_\emptyset + \sum_{i \in N} \left(\delta_{\mathrm{pa}_L(i)} - \delta_{\mathrm{pa}_L(i) \cup \{i\}}\right) \\ & - \delta_N + \delta_\emptyset - \sum_{i \in N} \left(\delta_{\mathrm{pa}_G(i)} - \delta_{\mathrm{pa}_G(i) \cup \{i\}}\right) \\ = {} & \delta_C - \delta_{C \cup \{b\}} - \delta_{C \cup \{a\}} + \delta_{C \cup \{a\} \cup \{b\}} = \mathbf{u}(\langle a, b|C\rangle).\end{aligned}$$

□

Using iterative arc removals and legal arrow reversals, two models $\mathcal{M}_K, \mathcal{M}_L$ are included in one another, i.e. $\mathcal{M}_K \subseteq \mathcal{M}_L$, if and only if the difference of their standard imsets $\mathbf{u}(L) - \mathbf{u}(K)$ is a combinatorial imset. The strict inclusion $\mathcal{M}_K \subsetneq \mathcal{M}_L$ is valid if and only if the difference is a non-zero combinatorial imset. Finally, the reverse direction of Proposition 1.2.14 is true and two DAGs, with $\mathcal{M}_K \subseteq \mathcal{M}_L$, are inclusion neighbours, if and only if $\mathbf{u}(L) - \mathbf{u}(K)$ is an elementary imset.

1.3 Learning Bayesian networks

Haddawy [26] gives a good introduction to the major approaches of learning BNs focusing on various objectives and conditions. As we cannot give a complete overview due to the complexity of that topic we limit the following analysis on the major principles presented in Haddawy [26]. For general introductions we refer to [31], [38] and [39] and to [47] for a thorough analysis containing lots of examples, too.

BNs inherit a certain probability distribution which can be explicitly given or, as mostly in real life, implicitly given by a set of observations which define a database. These observations

are possible realisations of the different states of the variables of the network. If this database is given, we can constitute the BN based on this database only. This is broadly known as **learning Bayesian networks**, which means to determine the probability distribution on the base of data. This problem can be furthermore separated into problems of known structure vs. unknown structure and complete vs. incomplete data.

If we suppose the knowledge of the structure of the BN, then the local dependencies are specified and only the qualitative information, i.e. the strength of the influence, must be calculated. This is often denoted with **parameter learning** [47]. If data is added to the existing database, we can simply update the network as only local dependencies are relevant. This update is also known as **adaption** ([38] and [39]). The so-called **prior probabilities** are simply adapted to the new data to compute a **posterior probability** using the Bayesian chain rule (1.1.1), page 4. Prior distributions can be obtained e.g. from the maximum likelihood estimate via frequencies of occurrences in the data. This is useful especially for large data sets. Other prior probabilities require a uniform distribution. However, well-chosen prior distributions can help to perform the following updating procedure more effectively. Updating strategies for complete data differ from those applied to incomplete data. For example, if some values are missing at random, then prior models can be constituted to predict the missing values. Then, this model can be updated by the created data. One of the procedures of this type is called the **EM-algorithm** (see e.g. [47]) alternating between the prediction and the updating step (an expectation and a maximisation step). Otherwise, if there is a lack of observation, then some values are systematically missing. These missing random variables are called **hidden variables**.

Suppose, we do not know the structure of the BN, but have a complete database. Sometimes the structure is given by expert knowledge determining the influence among states. Certainly, in real life situations this is often not possible. Hence, also the local dependencies have to be identified with the help of the given data. Jensen [38] calls this **batch learning**, likewise this is also called **learning Bayesian network structure**. Most approaches are based on iterative statistical tests on CI statements to be valid and then adding them to the model. But goodness-of-fit guarantees can only be given under strong assumptions on the distribution and the data. Other approaches are summarised by the so-called scoring approach all using a score function. By the use of such functions we can directly evaluate each structure with respect to the data and the implicated distribution. Moreover, approximation qualities can be given. Typically, this is a two stage procedure: Assume a structure and quantify the corresponding distribution, compare it with other possibilities and iterate (score and search techniques). We will analyse these approaches in more detail in the next sections. If, again, some data is missing, the currently best distribution can be used to complete the data, similar to the already mentioned EM-algorithm.

We do need the following assumptions on the given data and the distribution to obtain results based on the scoring approach.

Assumption 1: In the following sections, we suppose ourselves to have complete data with neither missing values, nor hidden variables. This is a simplification representing a kind of ideal case. But this assumption does not necessarily implicate to have sufficiently enough data to clearly identify the true inherited dependencies and thus the (perfectly) Markovian measure. In other words, it is theoretically possible to derive a Markovian measure from the data using only the given random variables. In practice, this assumption comes along with a trust in the data reflecting general uncertainty in results made by statistical observations.

Learning with known structure - an example To visualise the definitions of a BN and the derivation of the inherit probability distribution we analyse the standard task of learning BNs

with known structure and complete data, briefly. We can use this standard task to emphasise the problem of unknown structure.

For the introduction of learning BNs in a direct way we assume a maximum likelihood estimation of the parameters using frequencies and present an example.

Example 11. *Example 4, page 3, continued.* We have given four random variables: A red light L, a mechanical failure M, an electronic failure E, and a stopping car C. We define them to have only values in $\{0, 1\}$. $L = 1$ if and only if the light flashes up, $M, E = 1$ if and only if there is a corresponding failure and $C = 1$ if and only if the car suddenly stops. We use a randomly generated database of various joint occurrences of realisations of L, M, E and C. By using L and $\neg L$ we abbreviate events of type $L = 1$ and $L = 0$, respectively. Table 1.1 below shows a random database of these four random variables with a length $d = 20$.

L	1	1	0	1	0	1	1	0	0	1	0	1	0	0	1	1	0	0	1	0
M	0	1	0	1	0	1	0	1	0	1	0	1	1	0	0	1	0	0	1	0
E	0	1	0	1	1	1	0	0	0	0	1	1	0	1	1	0	0	0	1	0
C	0	1	0	0	0	1	0	1	0	0	0	1	0	1	1	1	1	0	1	0

Table 1.1: A randomly created database with four variables and length 20.

From this database, we calculate the corresponding contingency table (Table 1.2 below) showing the amounts of occurrences $d(y)$ of the 16 combinations of 0/1-realisations y of the four random variables together. See Section 1.4.1, page 19, or [60] for notation and more information on this. The sum of $d(y)$ taken over all y is equal to d. Contingency tables can be used to calculate the probability of the joint events as illustrated in Table 1.3, next page.

L	0	1	0	0	0	1	1	1	0	0	0	1	1	1	0	1
M	0	0	1	0	0	1	0	0	1	1	0	1	1	0	1	1
E	0	0	0	1	0	0	1	0	1	0	1	1	0	1	1	1
C	0	0	0	0	1	0	0	1	0	1	1	0	1	1	1	1
$d(y)$	4	2	1	2	1	1	0	0	0	1	1	1	1	1	0	4

Table 1.2: A contingency table for a database with four random variables.

We suppose the knowledge of the underlying relations of variables and their influence upon each other. Therefore the overall network structure is shown in Figure 1.15, next page. We can now use Bayes' theorem to calculate the conditional probabilities specified in the network.

We do this exemplarily for $P(L)$, $P(\neg L)$ and for $P(C|E \cap M)$. Since there is no influence from any other random variable upon L, we can calculate $P(L)$, $P(\neg L)$ directly. Therefore, we add up all probabilities of events keeping $L = 1$ and $M, E, C \in \{0, 1\}$ (red coloured probabilities). Analogously, we add up all probabilities with $L = 0$ to obtain $P(\neg L)$ (blue coloured probabilities). Both probabilities add up to one. For calculating $P(C|E \cap M)$ we use conditional probabilities to calculate $P(C|E \cap M)$ using $P(C \cap E \cap M)/P(E \cap M)$ instead. Again, probabilities $P(C|E \cap M)$ and $P(\neg C|E \cap M)$ add up to one. For this, we likewise use probabilities given in Table 1.3, next page. Figure 1.15, next page, shows the given structure and the corresponding local conditional probabilities for the variables. The negated counterparts such as $P(\neg C|E \cap M)$, with which the given probabilities add up to one, are skipped. ⋆

		L		$\neg L$	
		M	$\neg M$	M	$\neg M$
E	C	1/5	1/20	0	1/20
	$\neg C$	1/20	0	0	2/20
$\neg E$	C	1/20	0	1/20	1/20
	$\neg C$	1/20	2/20	1/20	1/5

Table 1.3: Probabilities of joint events for four random variables.

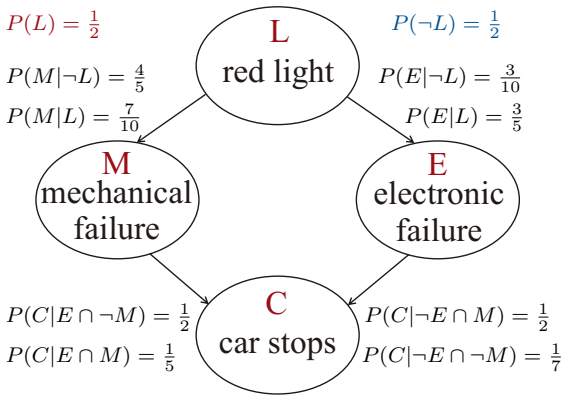

Figure 1.15: Learning a BN with four variables and known structure.

We present a second example. Although it seems to be less complex, we can use it later to indicate the hardness of computing the best structure.

Example 12. We choose a smaller set of variables containing only three random variables and a database of length three (see Table 1.4 below). The corresponding probabilities of the joint events can be seen in Table 1.5 below.

A	1	0	1
B	1	1	0
C	0	1	1

Table 1.4: A complete database with three variables and length three.

	A		$\neg A$	
	B	$\neg B$	B	$\neg B$
C	0	1/3	1/3	0
$\neg C$	1/3	0	0	0

Table 1.5: Probabilities of joint events for three random variables.

Analogously to the example above we provide a structure (Figure 1.16 below) to calculate the conditional probabilities (see Table 1.6 below). ⋆

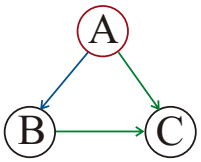

Figure 1.16: A supposed structure for three random variables.

$P(A) = 2/3$
$P(B|A) = 1/2$, $P(B|\neg A) = 1$
$P(C|B \cap A) = 0$, $P(C|B \cap \neg A) = 1$,
$P(C|\neg B \cap A) = 1$, $P(C|\neg B \cap \neg A) = 0$

Table 1.6: Conditional probabilities of three random variables.

1.4 Learning Bayesian network structures

We are interested in learning of BN structures and the hardness of it. This interest comes along with a transformation of an initially non-linear optimisation problem into a linear optimisation problem in exponential dimension. This problem can be solved to optimality using linear programming (LP) algorithms.

Assumption 2: The structure of the network is unknown. The first general assumption is a complete database. The second assumption reflects the uncertainty of relationships of random variables and implies a general lack of information on reasoning.

1.4.1 Quality criterion

A **score function** is a function measuring how good a certain BN structure given by a DAG G fits to the given database D. To define suitable score functions more notation concerning the data is necessary. We use Example 12 of the previous section as a running example to provide a better understanding of these notational conventions.

Notation: We recapitulate the different ways of representing data. One way of representation is via a given database and another is via a contingency table. A database $D : x^1, \ldots, x^d$ is an ordered sequence of length $d \geq 1$ of **realisations** of all random variables of the discrete joint sample space $X_N := \prod_{i \in N} X_i$. $\mathrm{DATA}(N, d)$ denotes the collection of all databases over N with length d. A **contingency table** is a function ascribing to each element y of the joint sample space its number of occurrences $d(y)$ in the data. Then $\mathrm{CT}(N, d)$ is the collection of these functions.

Example 13. *Example 12, page 18, continued.* We have given three nodes $N := \{A, B, C\}$. The joint sample space hence consists of the product of the corresponding three random variables. Both representations of data can be seen in Tables 1.7 and 1.8 below. ⋆

	x^1	x^2	x^3
A	1	0	1
B	1	1	0
C	0	1	1

Table 1.7: A database over the joint sample space $X_{\{A,B,C\}}$ with length three.

	y^1	y^2	y^3	y^4	y^5	y^6	y^7	y^8
A	0	1	0	0	1	1	0	1
B	0	0	1	0	1	0	1	1
C	0	0	0	1	0	1	1	1
$d(y)$	0	0	0	0	1	1	1	0

Table 1.8: A contingency table over the joint sample space $X_{\{A,B,C\}}$ with length three.

With $r(i) := |X_i| \geq 1$ we denote the number of possible states of a random variable X_i. For a fixed ordering, $y_i^1, \ldots, y_i^{r(i)}$ are the elements of X_i. We denote with y_i^k the k-th element of random variable X_i and therefore the k-th configuration of node i with $k \in \{1, \ldots, r(i)\}$. These values are given by the database and the contingency table, respectively.

Example 14. *Example 12, page 18, continued.* In our running Example 12 we have that $r(A) = 2$, $r(B) = 2$ and $r(C) = 2$. Hence, $y_A^1 = 0$, $y_A^2 = 1$ are all of the elements of X_A. Note the contingency table to consist of all elements of the joint sample space. With $k \in \{1, 2\}$, we can identify y_A^k as the k-th configuration of node A. Of course, this can likewise be done for all of the nodes. ⋆

Analogously we define values dependent on the structure of graph G. We define

$$q(i, G) := |X_{\mathrm{pa}(i)}| = \prod_{l \in \mathrm{pa}(i)} r(l) \geq 1$$

to be the number of possible parent configurations of a node $i \in N$ given in the structure G. For $\mathrm{pa}(i) = \emptyset$, we set by convention $q(i, G) = 1$. We can fix an ordering of parent configurations and denote all parent configurations of i with $z_i^1, \ldots z_i^{q(i,G)}$ (and thus all elements of $X_{\mathrm{pa}(i)}$). With $j \in \{1, \ldots, q(i, G)\}$ and z_i^j we indicate the j-th configuration among them.

Example 15. *Example 12, page 18, continued.* Although the best structure is missing we have to provide one to evaluate its score with respect to the data. Therefore, we assume the following DAG G (Figure 1.17, next page).

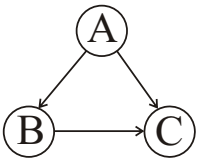

Figure 1.17: A given structure of nodes A, B and C.

Using this structure we get for node A, $\mathrm{pa}(A) = \emptyset$ and hence $q(A,G) = 1$. For node B we have, $\mathrm{pa}(B) = \{A\}$ and $q(B,G) = |X_{\mathrm{pa}(B)}| = r(A) = 2$. C has both A and B as parents and $q(C,G) = r(A) \cdot r(B) = 4$. The numbers $q(A,G)$, $q(B,G)$ and $q(C,G)$ indicate how many possible different combinations of realisations (among the states) of joint parents of a node in G can occur. These combinations are all elements of the joint space of variables $X_{\mathrm{pa}(i)}$. In our example we have e.g. for node C, $z_C^1 = (0\,0)^\intercal$ (node A has realisation zero and B, as well), $z_C^2 = (1\,0)^\intercal$, $z_C^3 = (0\,1)^\intercal$ and $z_C^4 = (1\,1)^\intercal$. Setting $j \in \{1,\ldots,4\}$, z_C^j is one of those. ⋆

Combining both the data and the structure, we compare common occurrences of the realisations in the data and the structure in the graph. We use $k(i,x)$ to identify the unique $1 \le k \le r(i)$ for which $x_i = y_i^k$, i.e. the element of X_i which corresponds to the actual realisation of node i given in the data. Analogously, $j(i,x)$ identifies the unique $1 \le j \le q(i,G)$ for which $x_{\mathrm{pa}(i)} = z_i^j$ and therefore the parent configuration which is indicated in G.

Example 16. *Example 12, page 18, continued.* For node A: $k(A,0) = 1$ and $k(A,1) = 2$. This is valid for all other nodes, as well. Then, $j(A,.) \equiv 1$ as A has no parent, $j(B,0) = 1$, $j(B,1) = 2$ and $j(C,(1\,0)^\intercal) = 2$, $j(C,(0\,1)^\intercal) = 3$ and $j(C,(1\,1)^\intercal) = 4$. The joint realisation $(0\,0)^\intercal$ for nodes A and B does not occur in the data. ⋆

With this, we define values counting how often certain realisations in the data fit to the configurations of the DAG. Given $D \in \mathrm{DATA}(N,d)$, $d \in \mathbb{N}$, and $D : x^1,\ldots,x^d$, we define:

$$\begin{aligned} d_{ij} &:= |\{1 \le l \le d : x^l_{\mathrm{pa}(i)} = z_i^j\}|, \\ d_{ijk} &:= |\{1 \le l \le d : x^l_{\{i\}\cup\mathrm{pa}(i)} = (y_i^k, z_i^j)\}| \\ &\qquad \text{for } i \in N,\ j \in \{1,\ldots,q(i,G)\},\ k \in \{1,\ldots,r(i)\}, \\ d_{[x]} &:= |\{1 \le l \le d : x^l_A = x\}| \ \text{ for } \emptyset \ne A \subseteq N,\ x \in X_A, \\ &\qquad \text{where } \sum_{k=1}^{r(i)} d_{ijk} = d_{ij} \text{ if } d_{ij} > 0. \end{aligned} \tag{1.4.1}$$

Example 17. *Example 12, page 18, continued.* Let node A be given, then $j = 1$ and $d_{A1} = d = 3$, $d_{A11} = 1$ (counting zeros in D), $d_{A12} = 2$ (counting ones). For node B: $d_{B1} = 1$, $d_{B2} = 2$ (counting zeros and ones for A, the parent node of B) and $d_{B11} = 0$ (counting jointly zero for B and for A), $d_{B12} = 1$ (counting jointly one for B and zero for A), $d_{B21} = 1$ (counting jointly zero for B and one for A) and $d_{B22} = 1$ (counting jointly one for B and for A). Note that the sum of all d_{ijk} for one node i has to be d. For node C: $d_{C1} = 0$, $d_{C2} = 1$, $d_{C3} = 1$, $d_{C4} = 1$ and $d_{C11} = 0$, $d_{C12} = 0$ (summing up to $d_{C1} = 0$), $d_{C21} = 0$, $d_{C22} = 1$, $d_{C31} = 0$, $d_{C32} = 1$, $d_{C41} = 1$ and $d_{C42} = 0$.

This can similarly be done for values $d[x]$ with $x \in X_A$ and for all non-empty sets $A \subseteq N$. ⋆

With this notation we can define the concept of score functions more formally.

Definition 1.4.1 (quality criterion)
*A **quality criterion** is a score function $Q : DAGs(N) \times DATA(N,d) \to \mathbb{R}$ assigning a real number $Q(G,D)$ to a DAG G and a database D.*

The higher the assigned value $Q(G, D)$ of $G \in \text{DAGs}(N)$ and $D \in \text{DATA}(N, d)$, the better the structure determined by G fits to the data. Quality criteria have properties similar to those of the probability distribution and therefore are implicitly indicated within the DAG G.

Definition 1.4.2 (score equivalent) *A quality criterion Q is called **score equivalent** if for two Markov equivalent graphs G and H the scores are equal, i.e.*

$$Q(G, D) = Q(H, D),\ \forall D \in DATA(N, d). \tag{1.4.2}$$

This definition/property is straightforward. For two DAGs with the same independence model, their fit to the data, and thus their score, should be equal.

For sets A with $\emptyset \neq A \subseteq N$ we call the sequence of data $D_A \in \text{DATA}(A, d) : x_A^1, \ldots, x_A^d$, $d \geq 1$, a projection of D onto A. The next property is a direct translation of the Markov property in terms of a quality criterion.

Definition 1.4.3 (decomposable) *A quality criterion is called **decomposable** if there exists a collection of functions $q_{i,B} : DATA(\{i\} \cup B, d) \to \mathbb{R}$ with $i \in N$, $B \subseteq N \setminus \{i\}$ and $d \in \mathbb{N}$ such that*

$$Q(G, D) = \sum_{i \in N} q_{i,\text{pa}(i)}(D_{\{i\} \cup \text{pa}(i)}),\ \forall G \in DAGs(N),\ \forall D \in DATA(N, d). \tag{1.4.3}$$

The Markov property of the distribution implies the overall distribution to factorise into local dependencies of nodes and their parents. Likewise, the score of the overall structure specifying how to combine the local dependencies should decompose into a sum of these local scores of nodes and their parents. A stronger property than a pure decomposability is the one of a regular criterion.

Definition 1.4.4 (regular) *A quality criterion is called **regular** if there exists a collection of functions $t_A : DATA(A, d) \to \mathbb{R}$ with $\emptyset \neq A \subseteq N$ and a constant $t_\emptyset(D_\emptyset)$ depending on the sample size and d such that*

$$\begin{aligned} Q(G, D) = &\sum_{i \in N} \left(t_{\{i\} \cup \text{pa}(i)}(D_{\{i\} \cup \text{pa}(i)}) - t_{\text{pa}(i)}(D_{\text{pa}(i)}) \right), \\ &\forall G \in DAGs(N),\ \forall D \in DATA(N, d). \end{aligned} \tag{1.4.4}$$

With this, the local scores of a decomposable quality criterion should further decompose into scores of parents and the sets of a node and their parents. A straightforward question is the relation of the three properties introduced.

Lemma 1.4.5 (Lemma 8.3 in [60]) *Assume $r(i) \geq 2$, $\forall i \in N$. A quality criterion Q for learning DAG models is regular if and only if it is decomposable and score equivalent.*

There exist several broadly used quality criteria based on different purposes to weight the similarities of the data and the BNS. For example, from a maximum likelihood interpretation (via frequencies of occurrences) there follows the so-called **Bayesian information criterion** (BIC) which is regular:

$$\text{BIC}(G, D) = \sum_{i \in N} \sum_{j=1}^{q(i,G)} \left(\frac{\ln d}{2} - r(i) \frac{\ln d}{2} + \sum_{k=1}^{r(i)} d_{ijk} \ln \frac{d_{ijk}}{d_{ij}} \right). \tag{1.4.5}$$

In the BIC, relative frequencies of joint occurrences of structures in the data and in the graph are summed up according to the maximum log-likelihood estimator.

Example 18. *Example 12, page 18, continued.* We use our running example and the therein calculated values to compute the score of the given structure with respect to the data. Note that we use the following convention: $\ln(0/.) := 0$.

$$
\begin{aligned}
\mathrm{BIC}(G,D) = {} & \left(\frac{\ln 3}{2} - r(A)\frac{\ln 3}{2} + \sum_{k=1}^{2} d_{A1k} \ln \frac{d_{A1k}}{d_{A1}}\right) + \sum_{j=1}^{2}\left(\frac{\ln 3}{2} - r(B)\frac{\ln 3}{2} + \sum_{k=1}^{2} d_{Bjk} \ln \frac{d_{Bjk}}{d_{Bj}}\right) \\
& + \sum_{j=1}^{4}\left(\frac{\ln 3}{2} - r(C)\frac{\ln 3}{2} + \sum_{k=1}^{2} d_{Cjk} \ln \frac{d_{Cjk}}{d_{Cj}}\right) \\
= {} & \left(\frac{\ln 3}{2} - \ln 3 + 1\ln\frac{1}{3} + 2\ln\frac{2}{3}\right) + \left(\frac{\ln 3}{2} - \ln 3 + 0\ln\frac{0}{1} + 1\ln\frac{1}{1}\right) + \left(\frac{\ln 3}{2} - \ln 3 + 1\ln\frac{1}{2} + 1\ln\frac{1}{2}\right) \\
& + \left(\frac{\ln 3}{2} - \ln 3 + 0\ln\frac{0}{0} + 0\ln\frac{0}{0}\right) + \left(\frac{\ln 3}{2} - \ln 3 + 0\ln\frac{0}{1} + 1\ln\frac{1}{1}\right) + \left(\frac{\ln 3}{2} - \ln 3 + 0\ln\frac{0}{1} + 1\ln\frac{1}{1}\right) \\
& + \left(\frac{\ln 3}{2} - \ln 3 + 1\ln\frac{1}{1} + 0\ln\frac{0}{1}\right) \\
= {} & \left(\frac{\ln 3}{2} - \ln 3 + 1\ln\frac{1}{3} + 2\ln\frac{2}{3}\right) + \left(\frac{\ln 3}{2} - \ln 3\right) + \left(\frac{\ln 3}{2} - \ln 3 + 1\ln\frac{1}{2} + 1\ln\frac{1}{2}\right) + \left(\frac{\ln 3}{2} - \ln 3\right) \\
& + \left(\frac{\ln 3}{2} - \ln 3\right) + \left(\frac{\ln 3}{2} - \ln 3\right) + \left(\frac{\ln 3}{2} - \ln 3\right) \qquad = -7\frac{\ln 3}{2} - 3\ln 3 = -6.5\ln 3
\end{aligned}
$$

The goodness of this result is skipped to the next sections when we compare different structures and their quality of representing the given data. ⋆

Another approach is called the **Bayesian approach** (with a corresponding **Bayesian criterion**) based on a prior distribution for each model. Other approaches are based on the encoding length of the graphical model (**minimum description length**). See [60] and the references therein for further information.

The definition of an appropriate score function brings us in the position to state the nonlinear optimisation problem corresponding to learning BN structures:

$$
\begin{aligned}
\max \quad & Q(G,D) \\
\text{s. t.} \quad & G \in \mathrm{DAGs}(N).
\end{aligned}
\tag{1.4.6}
$$

We maximise the quality criterion such that the desired graph is a DAG over nodes N. The attained maximum is the DAG best fitting to the given data. This graph may not reflect the underlying relations of random variables, i.e. the real probability distribution. However, it best represents the database and is therefore "globally" optimal with respect to the representation of that data. This is simply due to a not necessarily perfect Markovian measure and a representation via a discrete and finite set of data. Intuitively, the data cannot fully represent every distribution. Moreover to this, the data might be biased to a certain value indicating other relations.

1.4.2 Scoring based solution algorithms

As already indicated, we can distinguish two types of algorithms based on significance tests for validity of CI statements and on maximisation of a quality criterion. Studený argues in [60] that statistical based algorithms are not appropriate for learning BN structures represented by standard imsets. Additionally, he observes these methods to be only valid under strong assumptions on the data.

Algorithms of the other class are separated in search algorithms and polyhedral based primal algorithms. The first class performs a structured search through the space of all valid struc-

tures. The second class of algorithms uses a representation of the structure which enables us to compute the best structure directly using linear programming methods. All the methods we present depend on the properties of the quality criterion.

Assumption 3: Independently from the actual type, the quality criterion is assumed to be score equivalent and decomposable.

Algorithms of the first class are (among others) the Greedy equivalence search (short GES algorithm), an algorithm based on dynamic programming and a Branch and Bound based procedure also extending the dynamic programming approach. All three algorithms directly use the graphical representation via DAGs.

1.4.2.1 Greedy equivalence search

The **GES algorithm**, introduced by Meek in 1997 [45], is a broadly used local search procedure using the inclusion neighbours of DAGs (Definition 1.2.13, page 15). It consists of two phases of iterative addition and deletion of arcs, first considering all lower and then all upper neighbours of a current DAG.

Algorithm 1.4.6 (GES algorithm)
Input: *The set of DAGs G, a database D and a decomposable and score equivalent quality criterion $Q(G, D) \in \mathbb{R}$.*
Output: *The score $\overline{Q}$ and the DAG $\overline{G}$ attaining this maximum score with respect to inclusion.*

Let $G \in DAGs$ be the empty graph.

1. ***first phase - addition of edges***

   Among all lower neighbours of G, with one additional edge, choose one, say L, with the maximum positive increase in score $Q(L, D) - Q(G, D)$ and set $G := L$. Repeat until no further positive increase can be obtained.

   Let G be the resulting DAG of the first phase.

2. ***second phase - deletion of edges***

   Among all upper neighbours of G, with one edge less, choose one, say K, with the maximum positive increase in score $Q(K, D) - Q(G, D)$ and set $G := K$. Repeat until no further positive increase can be obtained.

The first phase is a simple greedy procedure computing a model containing the distribution generating the underlying DATA, which is called the **generative distribution**. This DAG may contain additional edges which are not necessary to describe the generative distribution. The second phase increases the amount of independence statements by removing arcs but keeping the generative property of the described distribution. The resulting DAG is a local maximum having no better solution among its inclusion neighbours.

Example 19. *Example 12, page 18, continued.* We again use our running example to find the best structure representing the given data. We start with the empty graph G_0 having a score of $\mathrm{BIC}(G_0, D) = -3/2 \ln 3 + 3 \ln(1/3) + 6 \ln(2/3) = 6 \ln 2 - 10.5 \ln 3 \approx -7.377$. Any addition of a single arc yields a graph G_1 with score $\mathrm{BIC}(G_1, D) = -2 \ln 3 + 2 \ln(1/3) + 4 \ln(2/3) + \ln(1/2) + \ln(1/2) = -8 \ln 3 + 2 \ln 2 \approx -7.403$, which is worse. This score is even worse than the initially supposed structure. Consequently, no addition will be performed.

As the empty graph has no edges no deletion will be performed in the second phase and the empty graph is the solution produced by the GES algorithm. By doing a more extensive search a graph with a higher score can be found.

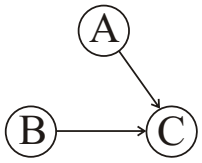

Figure 1.18: The optimal structure for our database with three nodes.

This optimal structure G^* in Figure 1.18 above has a score of $\mathrm{BIC}(G^*, D) = -3\ln 3 + 2\ln(1/3) + 4\ln(2/3) = -9\ln 3 + 4\ln 2 \approx -7.115$, which is maximal among all possible structures. Due to the symmetry within the given data all graphs with one immorality have this score and are therefore optimal. The GES cannot obtain the overall optimal solution. This fact is known as the **GES failure**. Besides, the GES solution can be arbitrarily bad compared to the optimal solution, as the amount of variables and data can be increased such that the parent/sink relations are kept fixed. Likewise, we can change the proportion of d_{ijk} to result in symmetric databases also increasing the difference of the GES and the optimal solution. Such an example with a higher divergence of both solutions can be seen in [65]. Expanding symmetrically this example increases the divergence of the scores of the solutions. ⋆

In literature (e.g. [12], [65]), general assumptions are analysed under which the GES algorithm is able to compute the optimal solution. For this purpose, the authors define the notion of **perfect data**, i.e. the given data completely encodes a probability distribution and this probability distribution fully generates the data. Thus, the fundamental relations can indeed be explained via a BN, which is approximated sufficiently by the data. Therefore, it is possible to find the underlying relation of variables by considering this database only. On the other hand, it is necessary to consider so-called **consistent** quality criteria. Note that the BIC quality criterion is consistent. With a consistent quality criterion, we have an increase in score if and only if independence statements are either added if they are necessary or deleted if they are not satisfied for this probability distribution (see also [14]). This definition coincides with a kind of monotonicity in the addition of CI statements with respect to the real distribution. With these definitions, we implicitly assume the knowledge of the inherited distribution, as we also assume the distribution to be fully represented. In real data this is typically not the case.

Apart from this GES failure, the GES is a simple and fast procedure. This algorithm is especially used for practical purposes, when good structures are needed quickly.

1.4.2.2 Dynamic programming approach

The GES failure might imply a full enumeration of the space of valid structures to be useful, but the consumption of time is far too high. The number $a_{|N|}$ of DAGs in a graph over node set N is super-exponential and can be recursively obtained (see [27]) via

$$a_{|N|} := \sum_{k=1}^{|N|} (-1)^{k-1} \binom{|N|}{k} 2^{k(|N|-k)} a_{|N|-k}.$$

Also a full enumeration of classes of Markov equivalent graphs, i.e. patterns or essential graphs, is futile. Although the exact ratio is unknown, computational results in [24] show the

number of classes to be about 27 percent of the total number of DAGs. Already for 10 nodes, there exist $4,175,098,976,430,598,143$ DAGs and $1,118,902,054,495,975,141$ classes!

However, due to the decomposability assumption on the score function, dynamic programming can be applied and reduces the search space tremendously. The dynamic programming approach was developed by Silander and Myllymäki in 2006 [55], whose work is based on earlier results by Koivisto and Sood in 2004. In contrast to the GES algorithm dynamic programming techniques are useful for "smaller" sizes of nodes, but for these a certificate of optimality can be given. Silander and Myllymäki show computations for $|N| = 29$ variables. Additional bounds on the degree of nodes speed up the procedure, but a high memory storage persists. Therefore there is a bound in the applicability not far above about 30 variables.

Given a topological ordering of nodes which provides the resulting graph to be acyclic directed. The principle of the gain from implementing dynamic programming comes from decomposable scores. Local scores $q_{i,\mathrm{pa}(i)}(D_{\{i\}\cup\mathrm{pa}(i)})$ (in Equation (1.4.3), page 21) of nodes together with their parents can be considered independent from the other local scores. In a score-maximal complete structure each local score is score-maximal, too. The sink is the score-best sink for this set of parents and the parent set is the score-best parent set with respect to this child. Any non-maximal local set in the above sense is not a part of the overall optimal network structure. These non-optimal local sets can be skipped in the remaining search reducing the overall enumeration. The remaining task is to find the best ordering of nodes, among the $|N|!$ possible onces, by performing dynamic programming for each of those.

To gain from this relation Silander and Myllymäki define every DAG G based on a fixed ordering of nodes. Therefore, let $G := (G_1, \ldots, G_n)$ be given with $G_i := \mathrm{pa}(i)$ is the parent set of node i and $|N| = n$ according to the fixed ordering. The formula ord_i denotes the element i of the given ordering. Then a necessary condition on sets G_i and the ordering is, that all parents of i precede node i with respect to the ordering and hence, $G_i \subseteq \bigcup_{j=1}^{i-1} \mathrm{ord}_j$. A fundamental observation is: G_1 is the empty set and $n \notin G_i$, $\forall i = 1. \ldots, n-1$. The nodes i are sinks for sets $G_1 \cup \ldots \cup G_i$ as the ordering preserves i to have only parents of smaller order. This we iteratively apply to the induced subgraph without i in which the assumed ordering is still valid.

The data is presented in the form of contingency tables $\mathrm{CT}(W)$ with D^W being the projection onto a set of variables $W \subseteq N$, which are rows of the $|N| \times d$ data matrix. Additionally, there are **conditional frequency tables** $CFT(v, W)$ counting how many different values of $v \in N$ occur together with data vectors corresponding to $W \setminus \{v\} \subseteq N$.

As already indicated, the quality criterion must be decomposable, which is

$$\mathrm{score} = \sum_{i=1}^{|N|} \mathrm{score}_i(G_i) = \sum_{i=1}^{|N|} \mathrm{score}(CFT(i, G_i)).$$

The local scores are $\mathrm{score}_i(G_i)$ and with respect to the conditional frequency table they are $\mathrm{score}(CFT(i, G_i))$, with nodes i and their parent sets G_i. The best choice of parents of nodes i among sets of possible parents C is then,

$$g_i^*(C) := \mathrm{argmax}_{g \subseteq C}\, \mathrm{score}_i(g). \tag{1.4.7}$$

Roughly speaking, the dynamic programming based algorithm computes the scores of all local sets. For every child, these local sets are evaluated to find its best parents to get the relation best-parents/child. Then, for all sets of possible parents, the best sink with respect to the induced subgraph is computed. This obtains the relation parents/best-sink. Having all best-parents/best-sink(child) relations, the best ordering of nodes must be computed to reconstruct the best network.

Algorithm 1.4.7 (Dynamic programming for learning BN structures)
Input: *A decomposable quality criterion* score *and a database given as a conditional frequency table* CFT.
Output: *An optimal network structure* G^* *and its score.*

1. $\forall i \in N$ *and* $\forall G_i \subseteq \mathbb{N} \setminus \{i\}$ *compute* $\text{score}_i(G_i)$ *and score*$(\text{CFT}(i, G_i))$ *of* (i, G_i), $\forall i \in N$. *($n \cdot 2^{n-1}$ different combinations)*

2. *Find the best parents* G_i *among* (i, G_i), $\forall i \in N$, *using* $g := G_i \subseteq C := N \setminus \{i\}$, *Equation (1.4.7) and*

$$\begin{aligned} \text{score}_i(g_i^*(C)) &:= \max\{\text{score}_i(C), \overline{\text{score}}(C)\} \text{ with} \\ \overline{\text{score}}(C) &:= \max_{c \in C} \text{score}_i(g_i^*(C \setminus \{c\})). \end{aligned}$$

   (result: $|N|$ *pairs* $(i, \text{pa}^*(i))$*)*

3. $\forall W \subseteq N$ *compute the best sinks among all* $s \in W$ *using*

$$\begin{aligned} \text{sink}^*(W) &:= \text{argmax}_{s \in W} \underline{\text{score}}(W, s) \text{ with} \\ \underline{\text{score}}(W, s) &:= \text{score}_s(g_s^*(W \setminus \{s\})) + \text{score}(G^*(W \setminus \{s\})). \end{aligned}$$

   The best sub-networks $G^*(W \setminus \{s\})$ *consisting of nodes* $(W \setminus \{s\})$ *are indicated in* $\text{score}(G^*(W \setminus \{s\}))$. *(result:* $2^{|N|}$ *pairs* $(\text{sink}^*(W), W)$*)*

4. *Using all best sinks, compute iteratively the best orderings of nodes via*

$$\text{ord}_i^*(N) := \text{sink}^* \left(N \setminus \bigcup_{j=i+1}^{|N|} \{\text{ord}_j^*(N)\} \right).$$

5. *With the best ordering of nodes and best set of parents the best network defined by*

$$G^*_{\text{ord}_i^*(N)} := g^*_{\text{ord}_i^*(N)} \left(N \setminus \bigcup_{j=1}^{i-1} \{\text{ord}_j^*(N)\} \right), \ \forall i \in N,$$

   can be constructed.

Example 20. *Example 12, page 18, continued.* The database of our running example with $N := \{A, B, C\}$ is highly symmetric. Therefore, the first step of Algorithm 1.4.7 above results in the scores: $\text{score}_i(\emptyset) = -3.5 \ln 3 + 2 \ln 2$, $\text{score}_i(j) = -\ln 3 - 2 \ln 2$, $\forall j \in N \setminus \{i\}$, and $\text{score}_i(N \setminus \{i\}) = -2 \ln 3$, $\forall i \in N$.

The second step consists of finding the best parents for each node in N using the scores computed in the first step. Again, due to symmetry we obtain

$$g_i^*(j) = \text{argmax}\{\text{score}_i(\emptyset), \text{score}_i(j)\} = \emptyset, \ \forall j \in N \setminus \{i\}.$$

Hence, $\overline{\text{score}}(N \setminus \{i\}) = \max\{\text{score}_i(\emptyset), \text{score}_i(\emptyset)\} = -3.5 \ln 3 + 2 \ln 2$ and with this, again $\forall i \in N$, $\text{score}_i(g_i^*(N \setminus \{i\}) = \max\{\text{score}_i(N \setminus \{i\}), -3.5 \ln 3 + 2 \ln 2\} = -2 \ln 3$. The best parents for each node are the corresponding other nodes.

In the third step the best sinks for a subset of nodes are computed. This computation includes the score of the best graph consisting of fewer nodes. These are already computed and equal due to the dynamic programming principle and the symmetry of this example. For the sake of simplicity we skip them and indicate their value with x. For sets $W \subseteq N$ of single nodes

s, we know $\text{sink}^* = \text{argmax}\,\underline{\text{score}(\{s\}, s)} = s$, $\forall s \in N$. Let $W = N \setminus \{i\}$, then

$$\begin{aligned}\underline{\text{score}(N \setminus \{i\}, j)} &= \text{score}_j(g_j^*(N \setminus \{i,j\})) + \text{score}(G^*(N \setminus \{i,j\})) \\ &= \text{score}_j(\emptyset) + x = -3.5 \ln 3 + 2 \ln 2 + x,\ \forall j \in N \setminus \{i\}.\end{aligned}$$

The nodes are symmetric and we choose

$$\text{sink}^*(\{A,B\}) = A,\ \text{sink}^*(\{A,C\}) = A,\ \text{sink}^*(\{B,C\}) = B.$$

If $W = N$, we have

$$\begin{aligned}\underline{\text{score}(N, i)} &= \text{score}_i(g_i^*(N \setminus \{i\})) + \text{score}(G^*(N \setminus \{i\})) = \text{score}_i(N \setminus \{i\}) + (\ldots) \\ &= -2 \ln 3 + x,\ \forall i \in N,\end{aligned}$$

and we again choose $\text{sink}^*(N) = A$.

The fourth step comprises to find the best ordering. Starting in $i = |N| = C$, we get $\text{ord}_C^*(N) = \text{sink}^*(N) = A$. Analogously for $i = |N| - 1 = B$ and $i = |N| - 2 = A$, $\text{ord}_B^*(N) = \text{sink}^*(N \setminus \{A\}) = B$ and $\text{ord}_A^*(N) = \text{sink}^*(N \setminus \{A,B\}) = C$, respectively.

These orderings are used to reconstruct the best network in the last step, i.e. for $i = A$, $G_C^* = g_C^*(N) = \{A,B\}$, for $i = B$, $G_B^* = g_B^*(N \setminus \{A,B\}) = \emptyset$, and $G_A^* = \emptyset$. The resulting optimal structure consists of only one immorality with sink A, due to the choice we have made during the computation and the change of the order.

This is not a full enumeration, a fact that becomes visible for larger $|N|$. From the second step of Algorithm 1.4.7 on we only consider parents which are either the empty set or the corresponding other nodes. All intermediate parent sets and hence the corresponding variables (i,j), $\forall i \in N$ and $\forall j \in N \setminus \{i\}$ can be skipped. ⋆

1.4.2.3 Branch and Bound based approach

Faced with the problem of memory usage of the dynamic programming approach, de Campos et al. [17] observe the actual required number of local scores to be tremendously smaller. Some parent sets of nodes can simply be excluded, because they are not contained in any optimal solution with respect to the scores of subsets. In later sections we use this fact, as well, and we only refer to the respective Section 5.3.1, page 98, at this point.

With these reduced local sets the authors constitute a cache of parent sets only needed. With this cache they perform a Branch and Bound algorithm. The algorithm consists of three steps. In the initialisation the scores of sets of nodes and the reduced sets of possible parents are computed. A priority queue then consists of graphs obtained by combining local sets of currently best parents, which are allowed to contain directed cycles. These graphs are sorted by score in the queue and might contain additional structural information.

In the first step of the algorithm the first graph of the queue is taken and its score is compared to the already best known score. If this score is smaller than the currently best score, then the graph is deleted and the next one is taken until a graph has a better score. If the current graph is a DAG, then its score is taken as the currently best known and the first step is iterated. If the graph is not a DAG then the second step is performed. In the second step of the algorithm a directed cycle is searched in the current graph. The following third step branches over all arcs in that cycle. New graphs are constructed with the additional structural information of certain arcs to be present or not. The scores of the new graphs are computed using already known scores and the graphs are added to the queue. The bounding step can be seen as the initial choice of graphs with scores better than the currently best one.

At any point in the computation a feasible DAG with the currently best score can be provided such that this algorithm can be used as a heuristic as well. But only if the complete procedure is finished, a certificate of optimality can be given.

If the initial parent sets cannot be reduced, then the algorithm is initialised with the same sets as in the dynamic programming approach. In spite of this, the authors can show this reduction to be the key point in the reduction of complexity in real life applications. This enables them to solve problems with up to 57 variables introducing bounds on the number of iterations. The same reduction of the search space of parent sets can be applied to the dynamic programming approach also leading to a faster algorithm with less memory needed.

1.4.2.4 Polyhedral approach via standard imsets

The second class of algorithms is based on a reformulation of the DAG in terms of algebraic representatives. The structure learning problem can then be solved with linear programming methods. In this section we present the polyhedral approach based on standard imsets. In the following chapters we also consider polyhedral approaches based on other representatives such as presented in Chapter 2, page 37, and introduced by Jaakkola et al. [37]. The advantage of any of the representatives is that any regular quality criterion becomes linear, when applied to them.

Lemma 1.4.8 (Lemma 8.7 in [60]) *Let Q be a regular quality criterion. Then there exist functions $s : DATA(N,d) \to \mathbb{R}$ and $t : D \in DATA(N,d) \mapsto \mathbf{t}(D) \in \mathbb{R}^n$ with $n = 2^{|N|}$ such that*

1. *$\forall A \subseteq N$, $|A| \leq 1$: $t_A(D) = 0$, $\forall D \in DATA(N,d)$,*
2. *$\forall A \subseteq N$, $|A| \geq 2$: the coordinate mapping $D \mapsto t_A(D)$ depends on the projection D_A and*
3. *the formula*

$$Q(G,D) = s(D) - \langle \mathbf{t}(D), \mathbf{u}(G) \rangle \tag{1.4.8}$$

holds for any $G \in DAGs(N)$ and $D \in DATA(N,d)$. The functions s, t are uniquely determined by these two requirements.

The reader is referred to [60] for a proof of this lemma.

The vector $\mathbf{t}(D)$ is called the **data vector** and, due to the lemma above, it is not only dependent on the data, but also on the quality criterion. To indicate this, we sometimes use the notion $\mathbf{t}^Q(D)$. Let DATA(N,d) be a given database, in case of the BIC the data vector for $\forall A \subseteq N$, $|A| \geq 2$, can be computed via

$$t_A^{\mathrm{BIC}}(D) := d \cdot \sum_{x \in X_A} \frac{d_{[x]}}{d} \cdot \ln \frac{\frac{d_{[x]}}{d}}{\prod_{i \in A} \frac{d_{[x_i]}}{d}} - \frac{\ln d}{2} \cdot \left(|A| - 1 + \prod_{i \in A} r(i) - \sum_{i \in A} r(i) \right). \tag{1.4.9}$$

A regular quality criterion provides a linear objective function. But for applying polyhedral techniques the corresponding representatives need to constitute a polyhedron. This means the optimal structure to be attained in a valid standard imset for any score equivalent and decomposable quality criterion. The resulting task is that we have to show the set of standard imsets to be the vertices of a polytope.

Theorem 1.4.9 (Theorem 4 in [65]) *Let $S := \{\mathbf{u}(G) : G \in DAGs(N)\}$ be the standard imsets over N. Then S is the set of vertices of a rational polytope $P(S) := \text{conv}(S) \subseteq \mathbb{R}^{2^{|N|}}$ called the **standard imset polytope**. The dimension of the polytope is $2^{|N|} - |N| - 1$.*

Proof. We refer to the appendix of [65] for a complete proof. Besides, we obtain a much simpler proof later. We refer to this in Section 2.1, page 37. Within we can show a projected set of standard imsets to be a subset of the 0/1-cube. □

Having a linear objective function (Lemma 1.4.8) and a polytope of standard imsets the problem of learning BN structures (Problem (1.4.6), page 22) turns into a linear problem to solve.

Conclusion 1.4.10 (compare with [65]) *Let $DATA(N, d)$ be a given database. Determining a network structure G best explaining the data is equal to*

$$\begin{aligned} \max \quad & s(D) - \langle \mathbf{t}(D), \mathbf{u}(G) \rangle \\ \text{s.t.} \quad & \mathbf{u}(G) \in P(S), \end{aligned} \tag{1.4.10}$$

where values $s(D)$ and $\mathbf{t}(D)$ are constants, $\forall D \in DATA(N, d)$, given by the data.

Apart from the computation of a single linear program (LP), this program comes along with some further complications. First of all, we do have given the vertices of $P(S)$ implicitly only. A canonical approach is to compute the inequality description of $P(S)$, solve Optimisation problem (1.4.10) with any LP solver and get the best BNS out of the best standard imset. But the LP to solve is of dimension exponential in the number of nodes. Moreover, the computation of the inequality description is practically infeasible due to the complexity. In fact, we do not have an explicit inequality description of $P(S)$ for more than four variables. In spite of this, a conjectured facet description of $P(S)$ exists which does at least provide a LP-relaxation of $P(S)$. But this is an implicit description and may therefore be not useful for practical purpose. See Section 3.5.3, page 85, for an overview.

For both practical and theoretical purposes, we need fundamental simplifications of the problem and its structure. One possible simplification is a reduction of the variation of the network structure as we consider only subfamilies of DAGs and the polytope of standard imsets of these subfamilies. Other ideas can gain from the decomposability of the objective function or from a graphical interpretation of the underlying DAGs.

1.5 Restricted structures - reduction of complexity

As already noticed, a reduction in complexity of Problem (1.4.10) above can be achieved by limiting the possible structures the network can have. We call this **restricted learning**. In contrast to this, we call the learning of not-further restricted structures **unrestricted learning**. In practice, restrictions are often given by an expert system providing the information of limited variations of the structure. This happens e.g. when time has to be considered and certain sequences of variables over time are forbidden or prescribed. In spite of this, the most frequent distinction of structures is one that separates between directed and undirected models with respect to the pattern or the essential graph, respectively. There exists a broad class of submodels whose pattern graphs are undirected and these are decomposable models. We will not differ any longer between DAGs and their patterns and essential graphs, respectively, we only indicate differences when necessary. In the following section we introduce several commonly investigated substructures and give an overview of special solution algorithms for the corresponding structure learning problems.

1.5.1 Chordal graphs

We again follow [60] to obtain definitions and characterisations appropriate for the algebraic and later geometric representation of learning BN structures.

Decomposable models are those DAG models G whose pattern graph $\text{pat}(G)$ is undirected. Undirected patterns $\text{pat}(G)$ correspond to DAG models G without immoralities. In a graph theoretical sense, undirected $\text{pat}(G)$ is a chordal graph, which is also denoted a triangulated graph (see below for a definition). For further information we refer to Section 2 in [42]. As the decomposable model G has no immorality, every cycle C in $\text{pat}(G)$ must contain cliques of size three. Otherwise C does not contain a node with at least two entering arcs in C and C is directed. Cliques, instead, imply a node with at least two entering arcs, but not yet an immorality. This leads to a full triangulation of each cycle C of $\text{pat}(G)$ of length at least four into cliques of size three. This is exactly the definition of a chordal graph $\text{pat}(G)$.

Definition 1.5.1 (chordal graph) *Let $G = (V, E)$ be an undirected graph. Let the set $C = \{v_1, \ldots, v_l\}$ denote a cycle in G with ordered edges $E(C) := \{\{v_1, v_2\}, \ldots, \{v_l, v_1\}\}$. Then G is called a **chordal graph** if for every cycle C in G with $l \geq 4$ there exists an edge $\tilde{e} = \{v_i, v_j\}$ in G such that $\tilde{e} \notin E(C)$ but $v_i, v_j \in C$, that is C has a **chord** in G.*

The triangulation of cycles is therefore equivalent to have diagonals in every such cycle. Hence both names can be used and are equivalent. The corresponding model is called decomposable, as the set of inclusion-maximal cliques of a chordal graph can be ordered $C_1, \ldots, C_m$, $m \geq 1$, such that

$$\forall i \in [2, m]\ \exists k \in [1, i) \text{ with } S_i := C_i \cap \left(\bigcup_{j<i} C_j \right) \subseteq C_k.$$

This property is called the **running intersection property** and the sets S_i are called **separators** of the maximal cliques. Particularly, the separators do not depend on the ordering of the maximal cliques. The ordering of maximal cliques implies the separators to be contained in each intersection of two consecutive cliques in that ordering.

Example 21. Let there be given a graph with 7 nodes visible in Figure 1.19 below.

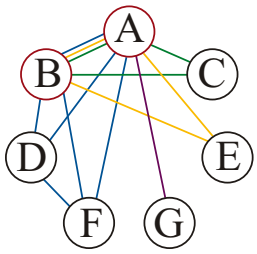

Figure 1.19: A chordal graph over 7 nodes with its maximal cliques and separators.

In this graph $C_1 := \{A, B, D, F\}$, $C_2 := \{A, B, C\}$, $C_3 := \{A, B, E\}$ and $C_4 := \{A, G\}$ are maximal cliques. The indicated ordering satisfies the running intersection property and node sets $S_1 = \{A, B\}$ and $S_2 = \{A\}$ are the respective separators. ⋆

Because of this graphical intuition of decomposable models, these models are broadly studied and algorithms for especially learning of decomposable models are investigated. Most of the algorithms are based upon statistical tests for CI or local search. For example in [18] guided statistical tests are used to reduce the arcs in the graph and therefore increase the number of CI statements. Contrarily, a greedy hill-climbing algorithm together with a consistent and locally-consistent quality criterion is introduced in [3]. There exist many variations of both

approaches with numerous corresponding literature. We analyse greedy based approaches in more detail in the next section but we skip the presentation of methods based on statistical tests due to the scope of this thesis.

However, chordal graphs inherit a useful property, when standard imsets are applied to them.

Lemma 1.5.2 (Lemma 7.2 in [60]) *Let H be an undirected chordal graph over node set N and let $\mathcal{C} := \{C_1, \ldots, C_m\}$, $m \geq 1$, be the set of all its maximal cliques satisfying the running intersection property, then*

$$\mathbf{u}(H) = \delta_N - \sum_{i=1}^{m} \delta_{C_i} + \sum_{i=2}^{m} \delta_{S_i}, \tag{1.5.1}$$

where $S_i := C_i \cap \left(\bigcup_{j<i} C_j\right)$, for $i = 2, \ldots, m$, are the separators. The set of these separators is denoted with $\mathcal{S}$. Moreover,

$$\mathbf{u}(H) = \delta_N - \sum_{C \in \mathcal{C}} \delta_C + \sum_{S \in \mathcal{S}} w(S)\delta_{S_i}, \tag{1.5.2}$$

where $w(S)$ is the multiplicity of a separator $S \in \mathcal{S}$.

The proof of Lemma 1.5.2 is based on induction on the number of maximal cliques and can be seen in [60].

Example 22. *Example 21, page 30, continued.* In Figure 1.19 of the previous example the first separator S_1 has multiplicity two and the second S_2 has multiplicity one. According to Equation (1.5.2) we have

$$\mathbf{u}(H) = \delta_N - \delta_{\{A,B,D,F\}} - \delta_{\{A,B,C\}} - \delta_{\{A,B,E\}} - \delta_{\{A,G\}} + 2 \cdot \delta_{\{A,B\}} + \delta_A.$$

⋆

The considerations in [60] lead to a characterisation of standard imsets of chordal graphs according to the relations of combinatorial and standard imsets.

Corollary 1.5.3 (Corollary 7.2 in [60]) *Let H be an undirected chordal graph over node set N. Then $\mathbf{u}(H)$ is a combinatorial imset, $\mathcal{M}_H = \mathcal{M}_{\mathbf{u}(H)}$ and $\mathbf{u}(H)$ coincides with the standard imsets for all DAGs G over N for which $M_G = \mathcal{M}_H$.*

Example 23. *Example 21, page 30, continued.* Applying the corollary above, we can write $\mathbf{u}(H)$ of the previous examples as a sum of elementary imsets, which is:

$$\begin{aligned}\mathbf{u}(H) = {} & \mathbf{u}(\langle F, G|\{A, B, C, D, E\}\rangle) + \mathbf{u}(\langle E, F|\{A, B, C, D\}\rangle) + \mathbf{u}(\langle E, G|\{A, B, C, D\}\rangle) \\ & + \mathbf{u}(\langle C, F|\{A, B, D\}\rangle) + \mathbf{u}(\langle C, E|\{A, B, D\}\rangle) + \mathbf{u}(\langle C, G|\{A, B, D\}\rangle) \\ & + \mathbf{u}(\langle D, E|\{A, B\}\rangle) + \mathbf{u}(\langle C, D|\{A, B\}\rangle) + \mathbf{u}(\langle D, G|\{A, B\}\rangle) + \mathbf{u}(\langle B, G|A\rangle).\end{aligned}$$

⋆

Chordal graphs and the GES algorithm Modified versions of the GES algorithm can be applied to any graphical model and hence decomposable models, as well. The problem which appears when using greedy-type algorithms to chordal graphs is that the solution is not

necessarily optimal. Already a certificate of optimality with respect to inclusion cannot be provided for sure. The reason is the set of chordal graphs to neither constitute a matroid nor an independence system.

In [3] the circumstances and additional properties under which a greedy based algorithm can find the inclusion-optimal solution are analysed. For this they define an extension of the property of a quality criterion to be consistent in case of chordal graphs. They define $\text{neigh}_G(a)$ to be the **neighbourhood** of a in G, i.e. all nodes in G that are adjacent to a.

Definition 1.5.4 (locally consistent) *Let G and H be two equal graphs only differing in edge $\{a, b\}$ which is in G but not in H. Let $\mathcal{M}$ be a respective set of CI statements. Then a quality criterion Q for chordal graphs is **locally consistent**, if*

a) $a \perp\!\!\!\perp b \,|\, (\text{neigh}_G(a) \cap \text{neigh}_G(b)) \in \mathcal{M} \implies Q(H) > Q(G)$ *and*

b) $a \perp\!\!\!\perp b \,|\, (\text{neigh}_G(a) \cap \text{neigh}_G(b)) \notin \mathcal{M} \implies Q(H) < Q(G)$.

So, if $\text{neigh}_G(a) \cap \text{neigh}_G(b) = \emptyset$ and a and b do not have any neighbours in common, then $a \perp\!\!\!\perp b \,|\, \emptyset \in \mathcal{M}$. This is true as a and b are directly connected in G and $Q(H) > Q(G)$. This happens for example if the edge $\{a, b\}$ is contained in a cycle of length ≥ 4 without a chord. If, in contrast, $\text{neigh}_G(a) \cap \text{neigh}_G(b) \neq \emptyset$ is valid, then statement $a \perp\!\!\!\perp b \,|\, (\text{neigh}_G(a) \cap \text{neigh}_G(b))$ is not valid for $\mathcal{M}$ and the edge $\{a, b\}$ is contained in a clique of size three. Hence, there is $Q(H) < Q(G)$.

The relation between consistency and local consistency is that a decomposable and consistent quality criterion for any DAG is locally consistent for chordal graphs. The mentioned monotonicity of a consistent score function can also be seen in the definition above.

In practice, the consistency and local consistency property for score functions of chordal graphs is only satisfied for (infinitely) large sample sizes. But if the local consistency is satisfied, a greedy-type algorithm can identify the optimal solution with respect to inclusion.

Proposition 1.5.5 (Proposition 2 in [3]) *If Q is a score function for a chordal graph that is consistent and locally consistent for a CI model $\mathcal{M}$, then local optima of Q with respect to the inclusion neighbours are inclusion optima for $\mathcal{M}$.*

We omit the proof and refer to [3].

The GES algorithm is a greedy procedure and hence these results are also valid for the GES. A consequence is, that, for e.g. the BIC, the GES can certainly find the inclusion-optimal solution but which may not be globally optimal.

As a subgraph of an undirected chordal graph is chordal as well, we can separate the class of decomposable models further into substructures. The mostly occurring structures are related to tree structures.

1.5.2 Undirected trees

Trees and branchings (directed trees) are a widely used class of considered subfamilies of decomposable models. Because of their simple graphical characterisation, a simple characterisation of the probability distribution of the model can be derived as well. Chow and Liu [15] exploit this to obtain an algorithm computing the optimal tree structure, these undirected trees are known as Chow-Liu trees. We present their approach:

A dependence tree is a DAG whose nodes have at most one parent node. With respect to their patterns, they can be considered as undirected trees. For these dependence trees, we know the corresponding joint probability distribution to factorise into so-called **second-order distributions**. These second-order distributions are conditional probabilities among two nodes: A node and its single parent node.

Definition 1.5.6 *Let D be a DAG upon node set N. Then a **dependence tree** is an ordered set of nodes $X = \{x_i \in N : i = 1 \ldots, n\}$ such that $0 \leq \mathrm{pa}(i) < i$ with $|\mathrm{pa}(i)| \leq 1$, $\forall i \in N$.*

For an unknown permutation of nodes $m_1, \ldots, m_n$ and $P(x_i|x_0) := P(x_i)$ the joint probability distribution of the dependence tree t based on n random variables can be computed as

$$P_t(x) = \prod_{i=1}^{n} P(x_{m_i}|x_{m_{\mathrm{pa}_t(i)}}).$$

These at most $n(n-1)/2$ many second-order distributions are directly specified by the underlying DAG and its arcs.

Chow and Liu propose a solution algorithm based on the closeness in approximating the real distribution P by another one P_a. This closeness is defined as

$$I(P, P_a) = \sum_{x} P(x) \log \frac{P(x)}{P_a(x)} \geq 0. \tag{1.5.3}$$

By standardisation, this value is greater or equal than zero such that a best approximation is the one equal to $P(x)$ and therefore the P_a that attains $I(P, P_a) = 0$. The problem of finding the best dependence tree is therefore to minimise the closeness in approximation given by Equation (1.5.3) above.

To find an appropriate scoring of different possible dependence trees a mutual information of dependence between variables x_i and x_j is introduced, which is

$$I(x_i, x_j) := \sum_{x_i, x_j} P(x_i, x_j) \log \frac{P(x_i, x_j)}{P(x_i)P(x_j)} \geq 0 \tag{1.5.4}$$

resulting in non-negative weights $I(x_i, x_j)$ of arcs $x_i x_j$, which can be summed up to obtain the weight of the overall tree. Let T_n be the set of all dependence trees upon n variables, then a maximal weight dependence tree t is the one with

$$\sum_{i=1}^{n} I(x_i, x_{\mathrm{pa}_t(i)}) \geq \sum_{i=1}^{n} I(x_i, x_{\mathrm{pa}_{t'}(i)}),\ \forall t' \in T_n.$$

This can be realised as a linear function to be maximised over the arc set of a DAG, such that this DAG is a dependence tree. Chow and Liu show this to be indeed the same as minimising Equation (1.5.3).

Lemma 1.5.7 (compare with [15]) *Let T_n be the set of all DAGs that are dependence trees and let D be a database. Furthermore, let P be the distribution encoded in D and P_t with $t \in T_n$ the distributions encoded in the trees t. Then the following two problems are equivalent, i.e. both yield the dependence tree with a corresponding distribution equal to the*

real distribution P. That means

$$\begin{array}{lll} \min \quad I(P, P_t) & \Longleftrightarrow & \max \quad \sum_{i=1}^{n} I(x_i, x_{\mathrm{pa}_t(i)}) \\ \text{s.t.} \quad t \in T_n & & \text{s.t.} \quad t \in T_n. \end{array}$$

In contrast to the general applicability of a greedy based algorithm, they limit the objective function, which is in this case obtained by a maximum likelihood criterion (compare with Equation (1.5.3)).

1.5.3 Polytrees

Pearl [51] generalises the concept of learning dependence trees presented by Chow and Liu to so-called **polytrees**. A polytree is a DAG having no cycle in its underlying undirected version. A node is allowed to have more than one parent with respect to Chow-Liu trees and this means immoralities can be contained and the pattern might not be undirected. More formally, Pearl assumes the probability defining the BN to factorise into conditional probabilities of nodes given their parents (not further restricted). Moreover, he supposes also the probability of the parent set to factorise into single probabilities, i.e. the parents of a node are independent.

The learning procedure can therefore be separated into first the basic algorithm presented by Chow and Liu obtaining a "best" undirected tree and second a test for immoralities and implied arcs using a test for mutual independence. More precisely, consider an already computed partly directed subgraph $b \to a - c$. If b and c are mutually independent, then there is no edge/arc connecting them and $b \to a \leftarrow c$ is an immorality. Otherwise, if they are dependent, then c is a child of a and $b \to a \to c$ is obtained, because cycles are not allowed.

In contrast to the results of Chow and Liu, this procedure can only find the optimal solution for the restricted case of perfectly Markovian measures.

1.6 Known complexity results

Several complexity results are presented in literature. Some of them are dealing with the general problem of learning BN structures or approximating them and others are dealing with limited structures only. We refer to the results corresponding to the structures of Section 1.5, page 29.

1.6.1 General DAGs

A lot of interest is devoted to the topic of finding complexity results of the general problem of learning BN structures analysing different optimisation strategies, score functions and representations of data. For example, Chickering, Heckerman and Meek show in [13] the large-sample learning problem to be $\mathcal{NP}$-hard even when the distribution is perfectly Markovian, which can be "optimally" solved with the GES algorithm. On the other hand, Chickering (in 1996) showed that learning BN structures is $\mathcal{NP}$-complete when using a certain Bayesian score. This is valid even if the number of parents is limited to a constant.

1.6.2 Chordal graphs

For several maximum likelihood based quality criteria Srebro [56] can show the learning of decomposable models to be $\mathcal{NP}$-hard for a fixed bound $k+1$ of the maximum size of the cliques. He derived this by showing the existence of a pseudo-polynomial transformation of the learning of chordal graphs into the decision that a maximum likelihood Markov network of tree-width at most k exists or not. Markov networks are generally undirected models (see [60] for a definition and properties). This problem itself is $\mathcal{NP}$-hard even for $k = 2$ and so is the learning problem. The transformation follows from a generalisation of the work by Chow and Liu.

Although approximation algorithms can be found which provide a certain quality depending on k, these are rather weak. Moreover to this, any approximation within a constant bound is $\mathcal{NP}$-hard [40].

1.6.3 Undirected trees

Due to the last section Chow and Liu provide a maximum likelihood procedure polynomial in the number of nodes computing an optimal dependence tree. Polynomiality is given, as the basic step of their procedure is a greedy algorithm applied to spanning trees in the case of a non-negative objective function and a complete undirected graph. For this result, the complexity of computing the probabilities from data and hence the objective function is considered as a constant independent of the actual optimisation procedure.

A similar result is obtained by Heckerman et al. [32] for the Bayesian scoring criterion.

1.6.4 Polytrees

Pearl's approach of learning polytrees is not a procedure obtaining the optimal solution in general. Even worse, Dasgupta [16] shows the corresponding decision problem of learning polytrees to be $\mathcal{NP}$-complete and that there exists a constant $c > 1$ such that it is $\mathcal{NP}$-hard to find a c-approximation for a polytree having at most two parents for every node.

1.6.5 Paths

In contrast to the polynomiality of identifying optimal undirected trees, Meek [46] shows the learning of paths to be $\mathcal{NP}$-hard for the maximum likelihood, the minimum description length and the Bayesian score. In this context, a path is a directed graph with one node having no parent node and all other nodes having exactly one parent. Hence its pattern is an undirected tree with degrees on nodes not exceeding two. He obtains the complexity result by explicitly constructing the polynomial size database (supposing a uniform distribution) to transform the known Hamiltonian path decision problem (see [22]) into this path problem.

2 Characteristic imsets

2.1 New representatives

In this section we introduce a new combinatorial representation of BN structures. These representatives are vectors obtained from standard imsets by an affine transformation. Therefore they inherit all properties of standard imsets. However, many theoretical results are easier to prove using the new representatives as they provide some additional properties. Of course there exist other similar representations, we refer to them in Section 2.4, page 48.

By means of standard imsets the non-linear problem of learning BN structures turns into a linear problem over $P(S)$. Typically, LP solvers can be applied to solve this problem. But two drawbacks motivate another change in the representation. First, there is neither a facet description of $P(S)$ and an explicit relaxation P of $P(S)$ with $P \cap \mathbb{Z}^{2^{|N|}} = P(S) \cap \mathbb{Z}^{2^{|N|}}$ known, nor is it possible to compute them for problems of larger sizes. Already for $|N| = 5$ this direct computation of inequalities is computationally infeasible at the moment. Second, even if we have at least the relaxation P, then standard imsets are an exponential representation of BN structures and an appropriate treatment of "practical" problem sizes of the learning problem is hard to achieve. Contrariwise, a facet description of $P(S)$ enables us to perform pure LP methods of polynomial, but in $2^{|N|}$, complexity. Hence, even if there exists any description of $P(S)$ additional information of the underlying BN structures must be incorporated to apply any linear (integer) optimisation method.

The problem of standard imsets is their highly involved and exponential representation via rather arbitrary integer vectors $\mathbf{u}(G)$. Both, properties of the underlying structures and theoretical results for their representation are hard to derive. Here we suggest: A better representation $\mathbf{c}(G)$ of the network structures G and $\mathrm{pat}(G)$, respectively, is one that uses a binary encoding with a direct graphical interpretation of G or $\mathrm{pat}(G)$. On the other hand, the properties of $\mathbf{c}(G)$ should be similar to those of $\mathbf{u}(G)$. For example, any score equivalent and decomposable quality criterion $Q(G)$ still should be linear in $\mathbf{c}(G)$. But the linearity of $Q(G)$ in terms of $\mathbf{u}(G)$ comes from the exponential encoding of $\mathbf{u}(G)$. Note that the usage of a polynomial size encoding of G via incidence vectors $\mathbf{y}$ of arcs leads to the well-known **acyclic subgraph polytope** P_{AC} (see [25]). However, the representative $\mathbf{y}$ has useful properties, most notably it is a polynomial representation of G. Unfortunately, a typical quality criterion $Q(G)$ is not linear when applied to $\mathbf{y}$. Any such encoding of edges/arcs (or nodes and edges/arcs) therefore leads to the initial non-linear Learning problem (1.4.6), page 22.

This chapter is based on our papers [33], [63] and [34].

2.1.1 More notation on graphs

In this section we introduce some additional graphical conventions. We use [60] to provide an overview of those.

Let $\mathrm{pat}(G)$ be a pattern of a DAG G. Then an induced subgraph for $\{a, b, c\}$ (or simply $\{a, b, c\}$) is called a **flag**, if $a \to b - c$ (hence $\{a, b\}$ is a directed arc and $\{b, c\}$ an undirected

edge in the mixed graph) and a, c are not adjacent. A **terminal node** within a set $T \subseteq N$ is a node $i \in T$ such that there is no $j \in T \setminus \{i\}$ with $i \to j$ in G. A terminal node is hence one possible sink of the set T. If we use the term terminal node with respect to immoralities or cliques T, then we understand a terminal node to be a node $i \in T$ with $j \to i$, $\forall i \in T$.

Let G be a mixed graph over node set N without any multiple edges. A **chain** for G is a partition of nodes into an ordered sequence of non-empty disjoint subsets $B_1, \ldots, B_n$, $n \geq 1$. This order specifies edges between nodes of B_i and B_j with $i < j$ to be directed from B_i to B_j and edges within a set B_i to be undirected. The corresponding graph is called a **chain graph** (e.g. the graph in Figure 2.1 below). Obviously, an undirected graph is a chain graph with $n = 1$ and a DAG is one with a partition into every single node. The ordering of the partition implies no directed cycle to exist.

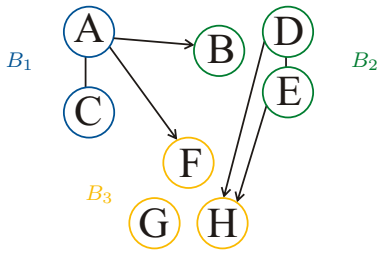

Figure 2.1: A chain graph with a possible partition of nodes.

Given a set of nodes C in a chain graph G, the set of **parents of** C, denoted by $\mathrm{pa}_G(C)$, is the set of nodes $j \in N$ such that $j \to i$ for some $i \in C$.

2.1.2 Definition of characteristic imsets

In this section we introduce the notion of a characteristic imset and prove some useful facts about it. For example, we show this imset to be a 0/1-vector for any underlying structure G.

Definition 2.1.1 *Given a DAG G over N, let $\mathbf{u}(G)$ be the standard imset for G. We introduce*

$$\begin{aligned} \mathrm{portrait}(\mathbf{u}(G)) &:= (\mathrm{portrait}_T(\mathbf{u}(G)))_{T\subseteq N, |T|>1} \in \mathbb{Z}^{2^{|N|}-|N|-1} \text{ with} \\ \mathrm{portrait}_T(\mathbf{u}(G)) &:= \sum_{X: T\subseteq X\subseteq N} \mathbf{u}_X(G) \text{ for } T \subseteq N,\ |T| > 1. \end{aligned}$$

*We call the **upper portrait** or **portrait** of $\mathbf{u}(G)$, or simply of G, $\mathrm{portrait}(\mathbf{u}(G))$. Moreover, we call*

$$\mathbf{c}(G) := \mathbf{1} - \mathrm{portrait}(\mathbf{u}(G)) \in \mathbb{Z}^{2^{|N|}-|N|-1} \tag{2.1.1}$$

*the **characteristic imset** of G.*

As their definition implies, characteristic imsets can be obtained by an affine linear transformation from standard imsets. We defer the concrete result after the next theorem. The reason is to have a direct characterisation of elements of characteristic imsets first. As we show below, their entries directly encode the immoralities of the graph.

Theorem 2.1.2 *Let G be a DAG over N. For any set $T \subseteq N$ with $|T| \geq 2$ we have $c_T(G) \in \{0, 1\}$ and $c_T(G) = 1$ if and only if there exists a node $i \in T$ with $T \setminus \{i\} \subseteq \mathrm{pa}_G(i)$. In particular, $\mathbf{c}(G) \in \{0, 1\}^{2^{|N|}-|N|-1}$.*

Proof. Consider the defining Equation (1.2.1), page 12, for the standard imset. For any $T \subseteq N$, $|T| \geq 2$, the entry $\text{portrait}_T(\mathbf{u}(G))$ can be computed as

$$\text{portrait}_T(\mathbf{u}(G)) = 1 + \sum_{i \in N : T \subseteq \text{pa}_G(i)} 1 - \sum_{i \in N : T \subseteq \text{pa}_G(i) \cup \{i\}} 1.$$

Hence, we get

$$\begin{aligned} c_T(G) = 1 - \text{portrait}_T(\mathbf{u}(G)) &= \sum_{i \in N : T \subseteq \text{pa}_G(i) \cup \{i\}} 1 - \sum_{i \in N : T \subseteq \text{pa}_G(i)} 1 \\ = \sum_{i \in N : T \subseteq \text{pa}_G(i) \cup \{i\}, i \in T} 1 &= \sum_{i \in T : T \setminus \{i\} \subseteq \text{pa}_G(i)} 1. \end{aligned}$$

For a fixed set T we assume to exist two different elements $i, j \in T$ with $T \setminus \{i\} \subseteq \text{pa}_G(i)$ and $T \setminus \{j\} \subseteq \text{pa}_G(j)$. This implies both $i \in \text{pa}_G(j)$ and $j \in \text{pa}_G(i)$. The simultaneous existence of the arcs $i \to j$ and $j \to i$ contradicts the fact of G being acyclic. Thus, for each $T \subseteq N$, there is at most one $i \in T$ with $T \setminus \{i\} \subseteq \text{pa}_G(i)$. Consequently,

$$c_T(G) = \sum_{i \in T : T \setminus \{i\} \subseteq \text{pa}_G(i)} 1 \ \in \{0, 1\},$$

and with this $\mathbf{c}(G) \in \{0, 1\}^{2^{|N|} - |N| - 1}$. □

Using this fact, we can define entries of characteristic imsets independently from standard imsets and directly based on the graph they encode. This direct definition helps to formulate the proof of the following statement in an easier way.

Lemma 2.1.3 *Let G be a DAG over N. There exists an affine linear transformation between $\mathbb{Z}^{2^{|N|}}$ and $\{0, 1\}^{2^{|N|} - |N| - 1}$ mapping $\mathbf{u}(G)$ to $\mathbf{c}(G)$.*

Proof. Definition 2.1.1 specifies the mapping between $\mathbf{u}(G)$ and $\mathbf{c}(G)$ via the equation $c_T(G) = 1 - \sum_{X : T \subseteq X \subseteq N} \mathbf{u}_X(G)$, $\forall T \subseteq N$ with $|T| \geq 2$. This mapping is affinely linear.

On the other hand, also the inverse of the portrait-map, the so-called Möbius inversion [6], is linear and

$$u_B(G) = \sum_{A : B \subseteq A \subseteq N} (-1)^{|A \setminus B|} \cdot (1 - c_A(G)), \ |B| \geq 2, \tag{2.1.2}$$

defines an entry of the standard imset based on the corresponding entry of the characteristic imset. This can be shown by substituting Equation (2.1.2) above into Equation (2.1.1). We observe,

$$c_T(G) = 1 - \sum_{X : T \subseteq X \subseteq N} \sum_{A : X \subseteq A \subseteq N} (-1)^{|A \setminus X|} + \sum_{X : T \subseteq X \subseteq N} \sum_{A : X \subseteq A \subseteq N} (-1)^{|A \setminus X|} c_A(G).$$

The sum $\sum_{A : X \subseteq A \subseteq N} (-1)^{|A \setminus X|}$ can be expressed as an alternating sum of binomial coefficients with value zero for $N \neq X$, and value one for $N = X$. Therefore, for each set T there is only one summand of the first sum remaining with value one and

$$c_T(G) = \sum_{X : T \subseteq X \subseteq N} \sum_{A : X \subseteq A \subseteq N} (-1)^{|A \setminus X|} c_A(G).$$

For the second sum we fix a set X with $T \subsetneq X$. Then there exists another set $\tilde{X}$ with $T \subseteq \tilde{X} \subsetneq X$ and $|X \setminus \tilde{X}| = 1$. For all sets A with $X \subseteq A$ there is $\tilde{X} \subseteq A$, as well, and

$c_A(G) + (-1)c_A(G) = 0$ for each choice X. Hence, the only remaining entry of the right side on the equation is the one with $T = X = A$ and the equation is true. □

2.1.3 Characteristic vs. standard imsets

This affine transformation has important consequences on the properties of characteristic imsets. All properties of standard imsets are true for characteristic imsets as well, but also vice versa, some properties of characteristic imsets are also valid for standard imsets.

We call $\mathrm{P}_{\mathbf{c}} = \mathrm{conv}\{\mathbf{c}(G) : G \in \mathrm{DAGs}(N)\}$ the **characteristic imset polytope**, which is contained in the 0/1-cube of dimension $2^{|N|} - |N| - 1$. Moreover to this, by applying the affine transformation also $P(S)$ is a polytope.

Lemma 2.1.4 *For any set N, the only integral points in the standard imset polytope $P(S)$ and in the characteristic imset polytope $\mathrm{P}_{\mathbf{c}}$, respectively, are their vertices.*

Proof. The result holds true for any 0/1-polytope and thus also for $\mathrm{P}_{\mathbf{c}}$. Moreover, the portrait map is an affinely linear map between $\mathbf{u}(G)$ and $\mathbf{c}(G)$, which is especially one-to-one. This shows, that the result holds for $P(S)$, as well. □

The affine transformation reduces the lengthy proof of Theorem 1.4.8, page 28, and Theorem 1.4.9, page 29, to a note.

Both polytopes $\mathrm{P}_{\mathbf{c}}$ and $P(S)$ are therefore isomorphic implying a bijective map between vertices and edges of the two polytopes. From this isomorphism the basic structure of the polytopes can be concluded.

First of all, characteristic imsets are unique representatives of Markov equivalence classes, too. Second, every decomposable and score equivalent quality criterion applied to characteristic imsets is linear. For this purpose, we define a revised data vector.

Definition 2.1.5 (revised data vector) *Given a score equivalent, decomposable criterion Q and a database D, let $\mathbf{t}(D)$ denote the data vector with respect to Q. We introduce the **revised data vector** $\mathbf{r}(D)$ as an element of $\mathbb{R}^{|\mathcal{P}(N)_{\geq 2}|}$ with $\mathcal{P}(N)_{\geq 2} := \{A \subseteq N : |A| \geq 2\}$ and*

$$r_A(D) = \sum_{B \subseteq A, |B| \geq 2} (-1)^{|A \setminus B|} t_B(D) \tag{2.1.3}$$

for $A \subseteq N$, $|A| \geq 2$.

With this definition, we can compute the score of a given graph G using the corresponding characteristic imset.

Lemma 2.1.6 *Every score equivalent and decomposable criterion Q has the form*

$$Q(G, D) = s_\emptyset(D) + \langle \mathbf{r}(D), \mathbf{c}(G) \rangle, \tag{2.1.4}$$

where $s_\emptyset(D)$ is the score of the empty graph over nodes N.

Proof. We substitute Equation (2.1.2), page 39, into Equation (1.4.8), page 28:

$$Q(G,D) = s(D) - \sum_{B\subseteq N,|B|\geq 2} t_B(D) \cdot \sum_{A:B\subseteq A\subseteq N} (-1)^{|A\setminus B|}(1-c_A(G)).$$

The limitation to entries with $|B| \geq 2$ is possible due to Lemma 1.4.8, page 28, and therein especially Condition 1. Now we change the order of summation and get

$$Q(G,D) = s(D) - \sum_{A\subseteq N,|A|\geq 2} (1-c_A(G)) \cdot \sum_{B\subseteq A,|B|\geq 2} (-1)^{|A\setminus B|} t_B(D)$$

in which the second sum $\sum_{B\subseteq A,|B|\geq 2}(-1)^{|A\setminus B|}t_B(D)$ can be replaced by the revised data vector $r_A(D)$. We get by Equation (2.1.3) that

$$\begin{aligned} Q(G,D) &= s(D) - \sum_{A\subseteq N,|A|\geq 2} (1-c_A(G)) \cdot r_A(D) \\ &= s(D) - \sum_{A\subseteq N,|A|\geq 2} (1-0) \cdot r_A(D) + \sum_{A\subseteq N,|A|\geq 2} c_A(G) \cdot r_A(D) \\ &= Q(G^{\emptyset},D) + \sum_{A\subseteq N,|A|\geq 2} c_A(G) \cdot r_A(D). \end{aligned}$$

For the third equation we observe the characteristic imsets of the empty graph to be identically zero. We can thus replace $s(D) - \sum_{A\subseteq N,|A|\geq 2}(1-0)\cdot r_A(D)$ by $Q(G^{\emptyset},D)$. Similar to the standard imset case, we additionally define $s_{\emptyset}(D) := Q(G^{\emptyset},D)$ and obtain the equation to prove. □

Summing up, the following two linear integer problems (IP) are equivalent, i.e. they yield the same structure G best fitting to the given data base $\mathrm{DATA}(N,d)$ with respect to the used quality criterion Q:

$$\begin{array}{l|l} \max \quad s^Q(D) - \langle \mathbf{t}^Q(D), \mathbf{u}(G)\rangle & \max \quad s^Q_{\emptyset}(D) + \langle \mathbf{r}^Q(D), \mathbf{c}(G)\rangle \\ \text{s.t.} \quad \mathbf{u}(G) \in P(S), & \text{s.t.} \quad \mathbf{c}(G) \in \mathrm{P}_{\mathbf{c}}, \end{array} \tag{2.1.5}$$

where values $s^Q(D)$ and $\mathbf{t}^Q(D)$ are constants, $\forall D \in \mathrm{DATA}(N,d)$, given by the data. $P(S)$ is the standard imset polytope.

where values $s^Q_{\emptyset}(D)$ and $\mathbf{r}^Q(D)$ are constants, $\forall D \in \mathrm{DATA}(N,d)$, given by the data. $\mathrm{P}_{\mathbf{c}}$ is the characteristic imset polytope.

Example 24. *Example 12, page 18, continued.* We continue Example 12, page 18, for applying the representation with characteristic imsets. The initial given pattern of G consists of one clique over three nodes. Let $\mathbf{c}(G)$ be the corresponding characteristic imset, i.e.

$$\mathbf{c}(G) = \begin{array}{c} \{a,b\} \quad \{a,c\} \quad \{b,c\} \quad \{a,b,c\} \\ (\quad 1 \qquad 1 \qquad 1 \qquad 1 \quad). \end{array}$$

The zero vector represents the empty graph. To calculate the revised data vector we need the data vector with respect to standard imsets. Hence, we use the BIC and Equation (1.4.9), page 28.

$$\begin{aligned} t_i^{\mathrm{BIC}}(D) &= 0, \ \forall i \in N, \\ t_T^{\mathrm{BIC}}(D) &= 3 \cdot \left(\frac{1}{3} \ln \frac{\frac{1}{3}}{\frac{1}{3}\cdot\frac{2}{3}} + \frac{1}{3} \ln \frac{\frac{1}{3}}{\frac{1}{3}\cdot\frac{2}{3}} + \frac{1}{3} \ln \frac{\frac{1}{3}}{\frac{2}{3}\cdot\frac{2}{3}} \right) - \frac{\ln 3}{2} \cdot (1+4-4) \\ &= 2\ln\frac{3}{2} + \ln\frac{3}{4} - \frac{1}{2}\ln 3 = 2.5\ln 3 - 4\ln 2, \ \forall T \subseteq N \text{ with } |T| = 2, \end{aligned}$$

$$t_N^{\mathrm{BIC}}(D) = 3 \cdot \left(\frac{1}{3} \ln \frac{\frac{1}{3}}{\frac{1}{3} \cdot \frac{2}{3} \cdot \frac{2}{3}} + \frac{1}{3} \ln \frac{\frac{1}{3}}{\frac{1}{3} \cdot \frac{2}{3} \cdot \frac{2}{3}} + \frac{1}{3} \ln \frac{\frac{1}{3}}{\frac{1}{3} \cdot \frac{2}{3} \cdot \frac{2}{3}} \right) - \frac{\ln 3}{2} \cdot (2 + 8 - 6)$$
$$= 3 \ln \frac{9}{4} - 2 \ln 3 = 4 \ln 3 - 6 \ln 2$$

The revised data vector and the data vector coincide for sets of cardinality two

$$r_T^{\mathrm{BIC}}(D) = t_T^{\mathrm{BIC}}(D) = 2.5 \ln 3 - 4 \ln 2, \ \forall T \in N \text{ with } |T| = 2, \text{ and}$$
$$r_N^{\mathrm{BIC}}(D) = -3 \cdot (2.5 \ln 3 - 4 \ln 2) + 4 \ln 3 - 6 \ln 2 = -3.5 \ln 3 + 6 \ln 2.$$

We can now recompute the scores of the graph with one edge G_1, the optimal structure G^* with one single immorality and the initial one using the score of the empty graph G_0 to replace $s_\emptyset(D)$. ⋆

2.2 Graphical interpretation

The binary encoding with characteristic imsets provides a more direct translation of graphical structures than standard imsets do. Theorem 2.1.2, page 38, directly enables us to evaluate entries with respect to structural properties.

Corollary 2.2.1 *Let G be a pattern of a DAG with nodes N and $\mathbf{c}(G)$ be the corresponding characteristic imset. If set $T \subseteq N$ is an immorality or a clique of size at least two, then $c_T(G) = 1$.*

Proof. If T is an immorality, then there exists a node $i \in T$ with $j \to i$, $\forall j \in T \setminus \{i\}$. Clearly, $T \setminus \{i\} \subseteq \mathrm{pa}_G(i)$ and $c_T(G) = 1$. Otherwise, if T is a clique, then there exists an orientation of arcs such that $j \to i$, $\forall j \in T \setminus \{i\}$, (see proof of Corollary 2.2.2 below for more details). Analogously, there is $c_T(G) = 1$. □

The implication in the corollary above can also be used the other way around. If $|T| \leq 3$ and $c_T(G) = 1$, then either T is an immorality, a clique (of size three) or an edge. We analyse this reverse implication in more detail using the class of chordal graphs. This we use later to generalise results about patterns of DAGs.

2.2.1 Characteristic imsets of chordal graphs

Given a chordal undirected graph G, the corresponding characteristic imset $\mathbf{c}(G)$ is the representation for any DAG $\overrightarrow{G}$ Markov equivalent to G. The observation that characteristic imsets are unique representatives of Markov equivalence classes makes this definition correct.

Corollary 2.2.2 *Let G be an undirected chordal graph over the node set N. Then, for $T \subseteq N$, $|T| \geq 2$, we have $c_T(G) = 1$ if and only if T is a clique in G.*

Proof. First, as G is an undirected pattern the corresponding DAGs do not contain immoralities. We can direct the edges of G in such a way that we obtain an equivalent DAG $\overrightarrow{G}$ without immoralities. Let $T \subseteq N$ be given with $\mathbf{c}_T(G) = 1$. As $c_T(\overrightarrow{G}) = c_T(G) = 1$, there exists a node $i \in T$ such that $T \setminus \{i\} \subseteq \mathrm{pa}_{\overrightarrow{G}}(i)$. Assume now, for a contradiction, that there are two nodes $j, k \in T \setminus \{i\}$ not connected by an edge in $\overrightarrow{G}$ and hence j and k are not

connected in G, either. Then, $j \to i \leftarrow k$ is an immorality in $\overrightarrow{G}$ and in G too, a contradiction. Hence, all nodes in $T \setminus \{i\}$ must be pairwise connected by an edge in G and T is a complete set of G.

Second, if T is a clique, then there exists an orientation of arcs such that there is a node $i \in T$ with $j \to i$ for all other nodes $j \in T \setminus \{i\}$. More precisely, let us assume the contrary, i.e. there is not such a node i. If T is a clique, then each subset of T is a clique. We choose subsets T_1, T_2 and T_3 of T with $|T_1| = |T_2| = |T_3| = |T| - 1$. At most one of these sets does not contain i. We easily observe $T_1 \cup T_2 = T$ for each combination of two sets. Using inductive arguments, we can assume all of them to contain nodes $k_1 \in T_1$, $k_2 \in T_2$ and $k_3 \in T_3$ having the claimed property. If all of k_1, k_2 and k_3 are distinct, then we have w.l.o.g. $j \to k_1$, $\forall j \in T_1 \setminus \{k_1\} := T \setminus \{k_1, k_2\}$ and $j \to k_2$, $\forall j \in T_2 \setminus \{k_2\} := T \setminus \{k_1, k_2\}$. Analysing T_3 we now get a contradiction. Due to the choice of T_1, T_2 and T_3 the node k_3 is also an element of either T_1 or T_2 and with this it has an arc towards k_1 or k_2, respectively, contradicting the inductive argument. Hence, all nodes k_1, k_2 and k_3 are equal and $c_T(G) = 1$. □

Applying this observation to special undirected chordal graphs, namely undirected forests, we obtain the following characterisation:

Corollary 2.2.3 *Let G be an undirected forest with node set N. Then, for $T \subseteq N$, $|T| \geq 2$, we have $\mathbf{c}_T(G) = 1$ if and only if T is an edge of G and*

$$\mathbf{c}(G) = \begin{pmatrix} \chi(G) \\ \mathbf{0} \end{pmatrix},$$

where $\chi(G)$ denotes the characteristic vector of the set of edges of G.

This is true as the only cliques of cardinality two in a forest are its edges.

2.2.2 Characteristic imsets of general DAGs and patterns

A similar result holds for any DAG G.

Corollary 2.2.4 *Let G be a DAG over N and $\bar{G}$ its underlying undirected graph. Then for any two-element subset $\{a, b\} \subseteq N$, we have $\mathbf{c}_{\{a,b\}}(G) = 1$ if and only if $a \to b$ or $b \to a$ is an arc of G, which implies*

$$\mathbf{c}(G) = \begin{pmatrix} \chi(\bar{G}) \\ * \end{pmatrix},$$

where $*$ denotes the remaining components of $\mathbf{c}(G)$.

Proof. This is an immediate consequence of Theorem 2.1.2. If $\mathbf{c}_T(G) = 1$ for $T = \{a, b\}$, then the only $i \in T$ with $T \setminus \{i\} \subseteq \mathrm{pa}_G(i)$ are either a or b. □

This implies $\mathbf{c}(G)$ to contain the characteristic vector $\chi(\bar{G})$ of the set of edges of $\bar{G}$ motivating our terminology. We show now the convertion of $\mathbf{c}(G)$ into the pattern graph $\mathrm{pat}(G)$.

Theorem 2.2.5 *Let G be a DAG over N and $a, b \in N$ are distinct nodes. Then the following holds:*

(i) $a, b \in N$ are connected in G if and only if $\mathbf{c}_{\{a,b\}}(G) = 1$, otherwise $\mathbf{c}_{\{a,b\}}(G) = 0$.

(ii) $a \to b$ *is contained in an immorality in* G *if and only if there exists a node* $i \in N \setminus \{a, b\}$ *with* $\mathbf{c}_{\{a,b,i\}}(G) = 1$ *and* $\mathbf{c}_{\{a,i\}}(G) = 0$. *The latter condition implies* $\mathbf{c}_{\{a,b\}}(G) = 1$ *and* $\mathbf{c}_{\{b,i\}}(G) = 1$.

Proof. The Condition (i) follows from Corollary 2.2.4 and Theorem 2.1.2.

For (ii) we assume $a \to b \leftarrow i$ to induce an immorality in G. Then $\mathbf{c}_{\{a,b,i\}}(G) = 1$ by Theorem 2.1.2 and the necessity of the other conditions follows from (i). Conversely, if $\mathbf{c}_{\{a,b,i\}}(G) = 1$, then either $\{a, b\} \subseteq \mathrm{pa}(i)$, $\{a, i\} \subseteq \mathrm{pa}(b)$ or $\{b, i\} \subseteq \mathrm{pa}(a)$. The first option implies $a \to i \leftarrow b$, the second $a \to b \leftarrow i$ and the third $i \to a \leftarrow b$. Now, $\mathbf{c}_{\{a,i\}}(G) = 0$ implies a and i not to be adjacent in G, which excludes two options and implies $a \to b \leftarrow i$ to be an immorality. □

Example 25. Consider the following pattern (Figure 2.2 below).

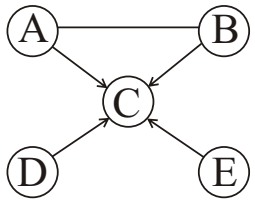

Figure 2.2: A pattern of a DAG over five nodes.

The corresponding characteristic imset is a 0/1-vector in dimension 26. With an appropriate ordering of sets we have,

$$
c = \begin{array}{cccccccccc}
 & \{A,B\} & \{A,C\} & \{A,D\} & \{A,E\} & \{B,C\} & \{B,D\} & \{B,E\} & \{C,D\} & \{CE\} & \{DE\} \\
(& 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
 & \{A,B,C\} & \{A,B,D\} & \{A,B,E\} & \{A,C,D\} & \{A,C,E\} & \{A,D,E\} & \{B,C,D\} & \{B,C,E\} & \{B,D,E\} & \{C,D,E\} \\
| & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\
 & \{A,B,C,D\} & \{A,B,C,E\} & \{A,B,D,E\} & \{A,C,D,E\} & \{B,C,D,E\} & & N & & & \\
| & 1 & 1 & 0 & 1 & 1 & | & 1 &). & &
\end{array}
$$

We easily can see the first 10 entries to coincide with the characteristic vector of the pattern. Moreover, we can reconstruct the pattern by first including all edges specified in the characteristic vector. In the second step we observe the edge $\{A, C\}$ to be directed $A \to C$ as $c_{\{A,C,D\}} = 1$, $c_{\{A,C\}} = 1$, $c_{\{C,D\}} = 1$ but $c_{\{A,D\}} = 0$. By the same reason, all of the edges $\{B, C\}$, $\{C, D\}$ and $\{C, E\}$ can be directed to arcs $B \to C$, $D \to C$ and $E \to C$. Summing up, $\{A, C, D\}$, $\{A, C, E\}$, $\{B, C, D\}$, $\{B, C, E\}$ and $\{C, D, E\}$ are immoralities. ⋆

All these facts together imply a pattern to be fully characterised by entries for sets of size two and three. Note this fact to be wrong for essential graphs. In fact, it is false if and only if there are more additional protected edges than those within an immorality.

Corollary 2.2.6 *Let* G *be a DAG over* N. *The characteristic imset* $\mathbf{c}(G)$ *is determined uniquely by its values for sets of cardinality two and three.*

Proof. By Theorem 2.2.5 these values determine both the edges and immoralities in G. In particular, they define the pattern $\mathrm{pat}(G)$. As explained in Section 1.2.1.1, page 9, the edges and immoralities uniquely describe the BNS and, therefore, the respective standard and characteristic imsets. □

The entries for sets of cardinality two and three determine the pattern and the pattern determines the DAG. The DAG, on the other hand, determines the corresponding characteristic imset, which defines the entries for sets of cardinality at least four, as well. More specifically, based upon the following lemma, the components $\mathbf{c}_S(G)$ for $|S| \geq 4$ can be derived iteratively from the components for $|S| \leq 3$. Another consequence of the lemma below is that there is no linear function obtaining the entries for $|S| \geq 4$ from those of $|S| \leq 3$.

Example 26. *Example 25, page 44, continued.* We use the previous Example 25 to visualise this fact. We call the given characteristic imset $\mathbf{c}_1$ and the following one $\mathbf{c}_2$ visualised in Figure 2.3 below.

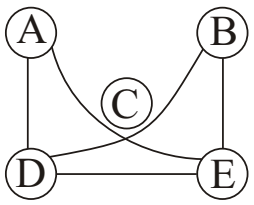

Figure 2.3: A second pattern of a DAG over five nodes.

Note, we fix the ordering of sets throughout the rest of this and the remaining chapters.

$$\mathbf{c}_2 = (0\,0\,1\,1\,0\,1\,1\,0\,0\,1\,0\,0\,0\,0\,0\,1\,0\,0\,1\,0\,0\,0\,0\,0\,0)$$

We show the mapping of the first entries for $|S| \leq 3$ of both vectors, $\mathbf{c}_1|_{\leq 3}$ and $\mathbf{c}_2|_{\leq 3}$, not to coincide with the sum of the mappings of the entries for $|S| \geq 4$, $\mathbf{c}_1|_{\geq 4}$ and $\mathbf{c}_2|_{\geq 4}$. More formally, let $\pi_{\geq 4}$ be the projection of the first entries for sets $|S| \leq 3$ onto the last ones, i.e.

$$\pi_{\geq 4} : \{0,1\}^{\binom{|N|}{2}+\binom{|N|}{3}} \to \{0,1\}^{\sum_{k=4}^{|N|}\binom{|N|}{k}} \text{ with } \pi_{\geq 4}(\mathbf{c}|_{\leq 3}) = \mathbf{c}|_{\geq 4}.$$

Then $\pi_{\geq 4}(\mathbf{c}_1|_{\leq 3}) + \pi_{\geq 4}(\mathbf{c}_2|_{\leq 3}) \neq \pi_{\geq 4}(\mathbf{c}_1|_{\leq 3} + \mathbf{c}_2|_{\leq 3})$, because $\mathbf{c}_1|_{\leq 3} + \mathbf{c}_2|_{\leq 3}$ indicates a complete graph (all edges in the joint graph of Figures 2.2 and 2.3 above exist). A complete graph has the **1**-vector as its characteristic imset, whereas $\pi_{\geq 4}(\mathbf{c}_1|_{\leq 3}) + \pi_{\geq 4}(\mathbf{c}_2|_{\leq 3})$ does not contain entries of value one only. ⋆

However, these entries for sets of cardinality at least four can be derived iteratively from the smaller ones, i.e. entries for sets of cardinality k imply those for cardinality $k+1$. The following Helly-type result generalises this fact.

Lemma 2.2.7 *Let there be given a DAG G over N and sets $S \subseteq N$, $|S| \geq 4$. Then the following conditions are equivalent.*

(a) $\mathbf{c}_S(G) = 1$,

(b) there exist $|S| - 1$ *subsets* T *of* S *with* $|T| = |S| - 1$ *and* $\mathbf{c}_T(G) = 1$,

(c) there exist three subsets T *of* S *with* $|T| = |S| - 1$ *and* $\mathbf{c}_T(G) = 1$.

Proof. The implication "$(a) \Rightarrow (b)$" simply follows from Theorem 2.1.2, page 38. The implication "$(b) \Rightarrow (c)$" follows immediately. To show "$(c) \Rightarrow (a)$" we first fix a terminal node i within S. Now, (c) implies at least two sets $T \subseteq S$, $|T| = |S| - 1$, to exist which contain i. Let $\tilde{T}$ be one of them. Since $\mathbf{c}_{\tilde{T}}(G) = 1$ by Theorem 2.1.2 there exists a node $k \in \tilde{T}$ with $j \to k$ for every $j \in \tilde{T} \setminus \{k\}$. If $i \neq k$, then $i \to k$, which contradicts i to be terminal in S and $i = k$ (compare with Figure 2.4, next page).

The same holds for the other subset and hence also the arcs from the nodes not contained in sets T are directed to i. Since these two sets T cover S we have $j \to i$ for every $j \in S \setminus \{i\}$ and Theorem 2.1.2 implies $\mathbf{c}_S(G) = 1$. □

The reasoning used in this proof is similar to the arguments used in the proof of Corollary 2.2.2.

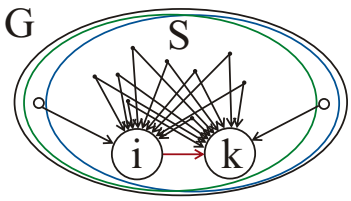

Figure 2.4: A set of nodes S covered by two subsets (blue and green) containing node i, which is not terminal.

2.2.3 Reconstruction of the essential graph from a characteristic imset

Theorem 2.2.5 allows us to reconstruct the essential graph of G. Indeed, the Conditions (i) and (ii) directly characterise the pattern graph $\mathrm{pat}(G)$. In general, there could be additional arcs in the essential graph. Fortunately, there is a graphical algorithm transforming $\mathrm{pat}(G)$ into the corresponding essential graph G^*, which requires polynomially, in the number of nodes, many steps. More specifically, Theorem 3 in [44] shows the repeated exhaustive application of the **orientation rules** from Figure 2.5 below to give the essential graph G^*, provided $\mathrm{pat}(G)$ is the pattern of a DAG G.

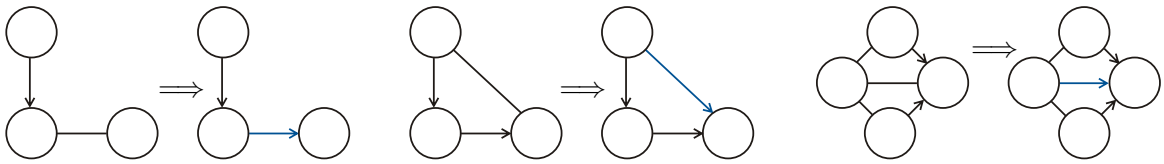

Figure 2.5: Orientation rules 1, 2 and 3 for obtaining essential graphs and general DAGs.

But there is also the possibility of determining the arcs in G^* directly on the basis of $\mathbf{c}(G)$. This fact is based on the following generalisation of Theorem 2.1.2. For this purpose, we need the definition of a characteristic imset of a more general graph than a DAG. We therefore consider chain graphs and define the following. The characteristic imset $\mathbf{c}(H)$ for a chain graph H being Markov equivalent to a DAG G is defined as $\mathbf{c}(G)$. As we take any Markov equivalent graph, $\mathbf{c}(G)$ is equal for all of those, and hence it does not depend on the choice of the DAG itself.

Theorem 2.2.8 *Let H be a chain graph without flags which is equivalent to a DAG G over N. For sets $T \subseteq N$, $|T| \geq 2$, the entry $\mathbf{c}_T(H)$ is equal to one if and only if*

$$\exists \text{ a clique } C \text{ with } \emptyset \neq C \subseteq T \text{ in } H \text{ and } T \setminus C \subseteq \mathrm{pa}_H(C). \tag{2.2.1}$$

We refer to [34] or [63] for a detailed proof of this fact.

The above result is also applicable for the other direction. Applied to the essential graph G^* in place of H, it gives the direct formula for $\mathbf{c}(G)$ on the basis of G^*. This leads to a procedure for testing whether a given vector $\mathbf{c} \in \mathbb{Z}^{2^{|N|}-|N|-1}$ is a characteristic imset for a DAG G over N. Clearly, a necessary condition (see Theorems 2.1.2 and 2.2.5) is: If $\mathbf{c}$ has the value one for a three-element set, then at most one of its two-element subsets can have the value zero (compare also with Lemma 2.2.7).

If this condition holds, we determine the conjectured pattern of G using the Conditions (i) and (ii) in Theorem 2.2.5. If there is no disagreement about directing edges in the pattern, then, by consecutive application of the orientation rules, the conjectured essential graph for

G is obtained. We can use the graphical characterisation of an essential graph in Theorem 4.1 of [1] (see Theorem 3.3.2, page 65) to check whether it is indeed the essential graph. In this case, this condition basically means to check whether the resulting graph is a chain graph and whether its undirected components induce chordal graphs. If the respective components are chordal graphs, we can determine the characteristic imset for the conjectured essential graph by Theorem 2.2.8. It remains to check whether it coincides with the given vector $\mathbf{c}$.

2.3 Characteristic imsets of inclusional neighbours

The inclusion of BN structures based on characteristic imsets is another interesting aspect to consider. As standard imsets are able to recognise inclusion neighbours (see Proposition 1.2.14, page 15), there must be the possibility to identify inclusion neighbours also on the basis of characteristic imsets. Indeed, this is the case due to Lemma 2.1.3, page 39. The first step is the characterisation of a minimal non-trivial inclusion.

Corollary 2.3.1 *Let K, L be two DAGs over node set N. The structure induced by K strictly contains the structure induced by L while there is no other DAG between them if and only if there exists an elementary imset $\mathbf{u}_{\langle a,b|C\rangle}$ such that $\mathbf{c}(K) - \mathbf{c}(L) = \text{portrait}(\mathbf{u}(\langle a, b|C\rangle))$.*

Proof. Two statistical models are in inclusion relation if the difference of the respective standard imsets $\mathbf{u}(L) - \mathbf{u}(K)$ is an elementary imset $\mathbf{u}(\langle a, b|C\rangle)$ (Remark 8.10 and Corollary 8.4 in [60]). This fact can be applied to the portrait of the models, as well, and

$$\begin{aligned}\text{portrait}(\mathbf{u}(\langle a, b|C\rangle)) &= \text{portrait}(\mathbf{u}(L)) - \text{portrait}(\mathbf{u}(K)) \\ &= (\mathbf{1} - \text{portrait}(\mathbf{u}(K))) - (\mathbf{1} - \text{portrait}(\mathbf{u}(L))) = \mathbf{c}(K) - \mathbf{c}(L)\end{aligned}$$

follows, because the portrait map from Definition 2.1.1, page 38, is an invertible linear transformation. □

The inclusion relation described above corresponds graphically to the following situation. There exists a DAG K' Markov equivalent to K and a DAG L' Markov equivalent to L such that L' is obtained from K' by removal of one arc (Lemma 8.5 in [60] or [12]).

As the portrait of an elementary imset is a non-negative vector, we get the following consequence of Corollary 2.3.1. If the BNS defined by K includes the BNS defined by L, then $\mathbf{c}(K) - \mathbf{c}(L) \geq 0$. This is a simple necessary condition for inclusion, but not a sufficient one. Example 27 below gives the counterexample.

Example 27. If we consider $\mathbf{c}(K) = (0\,1\,1\,1)$ and $\mathbf{c}(L) = (0\,1\,1\,0)$, then $\mathbf{c}(K) \geq \mathbf{c}(L)$ follows directly, but the structures defined by K and L are not in the inclusion relation, which can be seen in Figure 2.6 below. ⋆

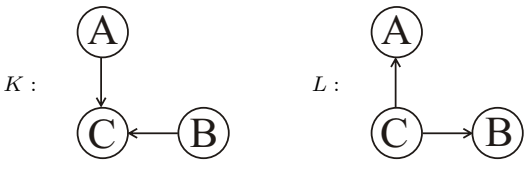

Figure 2.6: An example of two characteristic imsets with $\mathbf{c}(K) \geq \mathbf{c}(L)$, which are not in inclusion relation.

Corollary 2.3.2 *Let K, L be two DAGs over N. Then the structure induced by K contains the structure induced by L if and only if*

$$\mathbf{c}(K) - \mathbf{c}(L) = \sum_{\mathbf{u} \in \mathcal{E}(N)} \alpha_{\mathbf{u}} \cdot \text{portrait}(\mathbf{u}),$$

where $\mathbf{u}$ are elementary imsets and $\alpha_{\mathbf{u}}$ are non-negative integer coefficients.

Proof. Since there is a finite number of DAGs over N, the inclusion premise implies a sequence of DAGs $K = G_1, \ldots, G_n = L$, $n \geq 1$ to exist such that, for $i = 1, \ldots, n-1$, G_i and G_{i+1} are in the relation mentioned in Corollary 2.3.1.

Conversely, if $\mathbf{c}(K) - \mathbf{c}(L)$ is a combination of portraits of elementary imsets, then Corollary 2.3.1 implies the inclusion. This fact holds because $\mathbf{c}(K) = \mathbf{c}(L) + \sum_{\mathbf{u} \in \mathcal{E}(N)} \alpha_{\mathbf{u}} \cdot \text{portrait}(\mathbf{u})$ can be separated into smaller sums of characteristic imsets and portraits of elementary imsets adding only non-negative values. If one single sum of a portrait and a characteristic imset is a binary vector it can be further added with portraits (etc.). These intermediate binary vectors do not have to be characteristic imsets themselves. It is only important to show the addition of a portrait of an elementary imset to result in the addition of an edge and K and L to be in inclusion relation.

For showing this, we observe $\text{portrait}_T(\mathbf{u}(\langle a, b|C\rangle)) = 1$ if and only if $T \subseteq \{a, b\} \cup C$ and $a, b \in T$. This equivalence is the same as $c_T(\mathbf{u}(\langle a, b|C\rangle)) = 0$ if and only if $T \subseteq \{a, b\} \cup C$ and $a, b \in T$. Consider two graphs G_1, G_2 and $\mathbf{c}(G_1) + \mathbf{1} - \mathbf{c}(G_2)$, which is the addition of a portrait and a characteristic imset. The resulting vector is binary if and only if $\mathbf{c}(G_2) \geq \mathbf{c}(G_1)$. With respect to $\mathbf{c}(\mathbf{u}(\langle a, b|C\rangle))$, this means $c_T(L') := c_T(L) + 1 - c_T(\mathbf{u}(\langle a, b|C\rangle)) = 1$ while $c_T(L) = 0$ if and only if $c_T(\mathbf{u}(\langle a, b|C\rangle)) = 0$, i.e. $T \subseteq \{a, b\} \cup C$ and $a, b \in T$. The new characteristic imset $\mathbf{c}(L')$ contains additionally the edge $\{a, b\}$ and eventually new immoralities/cliques containing this edge.

As we suppose the sum of these portraits to obtain the difference of $\mathbf{c}(K)$ and $\mathbf{c}(L)$, there must be appropriate elementary imsets. Each intermediate binary vector $\mathbf{c}(L')$ has one edge more and the models induced by K and L are in inclusion relation. □

2.4 Related work

In literature, we have come across two papers in which similar binary representations are introduced with two different objectives. Niepert [48], presented in Section 6.5.1, page 129, introduces a binary representative of CI statements to test the CI implication problem computationally:

Definition 2.4.1 *Let $s = \langle A, B|C\rangle$ be a CI statement over node set N, then vector $\mathbf{v}$ with*

$$v_T(s) = 1 \iff C \subseteq T \text{ and } A, B \nsubseteq T \tag{2.4.1}$$

is called the vector representative of s.

We analyse and relate this reference to our method in more detail in Section 6.5.1. On the other hand Jaakkola et al. [37] introduce binary representatives of BN structures to learn BN structures (see Section 5.4, page 102). An advantage of both representatives is that they are not limited to the presented applications.

2.4.1 Representation due to Jaakkola et al. [37]

The paper by Jaakkola et al. [37] is devoted to the topic of learning BN structures computationally. They analogously use special 0/1-vectors $\eta(G)$ to encode DAGs G over N. This vector $\eta(G)$ is of dimension $|N| \cdot 2^{(|N|-1)}$, which is larger than the dimension of a corresponding characteristic imset. Specifically, the components of $\eta(G)$ are indexed by pairs $(i|B)$, where $i \in N$ and $B \subseteq N \setminus \{i\}$. Given a DAG G over N, the respective vector $\eta(G)$ is defined as follows:

$$\eta_{(i,B)}(G) = \begin{cases} 1 & \text{if } B = \mathrm{pa}_G(i), i \in N, \\ 0 & \text{otherwise.} \end{cases}$$

Deriving the representation in more detail, we start by defining $Pa(i) \subseteq \mathcal{P}(N \setminus \{i\})$ to be the set of all possible parent sets of a node i. One element of this is a set $s_i \in Pa(i)$ which is then a realisation of a parent set of i in a graph G. For all nodes i, we use $\eta_i \in \{0,1\}^{2^{(|N|-1)}}$ to indicate the parent sets, i.e. $\eta_{(i,s_i)}(G) = 1$ if and only if s_i is the actual parent set of i in graph G. The complete binary vector $\eta := (\eta_1, \ldots, \eta_{|N|})$ is then a vector of dimension $|N| \cdot 2^{(|N|-1)}$. Moreover, this vector is highly sparse with exactly $|N|$ non-zero entries, since there is exactly one parent set for each node.

Applying a decomposable quality criterion to this representation yields a linear objective, as every single score $Q_i(s_i)$ directly evaluates entries $\eta_{(i,s_i)}$, for all $s_i \in Pa(i)$. The complete linear objective function is the scalar product of η and the given quality criterion, which is $\eta \cdot Q = \sum_{i=1}^{|N|} \sum_{s_i \in Pa(i)} \eta_{(i,s_i)} Q_i(s_i)$. The corresponding structure learning task can be formulated as a linear integer optimisation problem.

$$\begin{aligned} \max \quad & \sum_{i=1}^{|N|} \sum_{s_i \in Pa(i)} \eta_{(i,s_i)} Q_i(s_i) \\ \text{s.t.} \quad & \eta \in P_\eta \end{aligned} \tag{2.4.2}$$

As $\eta(G)$ is a binary representation, the set of vectors η does constitute the vertices of a polytope P_η for sure. In spite of this, Jaakkola et al. do not solve the linear program directly using LP solvers. We defer the description of P_η and their detailed computational methods for learning BN structures presented in [37] to Chapters 3, page 51, and 5, page 95.

2.4.2 Relation of Jaakkola et al. [37] and characteristic imsets

The representative $\eta(G)$ of Jaakkola et al. is in one-to-one correspondence to a DAG G, while the characteristic imset $\mathbf{c}(G)$ corresponds to the Markov equivalence class of graphs containing G. There is no affine (linear!) mapping transforming $\mathbf{c}(G)$ to $\eta(G)$. Contrarily, $\mathbf{c}(G)$ can be considered as a linear function of $\eta(G)$. This can be shown, by observing the standard imset $\mathbf{u}(G)$ to be an affine function of $\eta(G)$:

$$u_T(G) = \delta_N(T) - \delta_\emptyset(T) + \sum_{(i|B):B=T} \eta_{(i,B)}(G) - \sum_{(i|B):\{i\}\cup B=T} \eta_{(i,B)}(G), \ \forall T \subseteq N.$$

Moreover, we already know $\mathbf{c}(G)$ to be an affine function of $\mathbf{u}(G)$, and the composition above provides an affine transformation of $\eta(G)$ to $\mathbf{c}(G)$. However, for the empty graph $G_\emptyset$, both $\eta(G_\emptyset)$ and $\mathbf{c}(G_\emptyset)$ are the zero vector. As the zero vector is transformed to the zero vector, the affine transformation must be linear.

3 Polytopes of (restricted) Bayesian network structures

3.1 The characteristic imset polytope

We consider the following polytope

$$\mathrm{P}_{\mathbf{c}} := \mathrm{conv}\{\mathbf{c}(G) : G \in \mathrm{DAGs}\} \tag{3.1.1}$$

as the characteristic imset polytope and the optimisation problem of maximising $\mathbf{w}^\intercal\mathbf{c}$ over $\mathbf{c} \in \mathrm{P}_{\mathbf{c}}$, where $\mathbf{w}$ is a linear objective function.

If we would have an inequality description P of $\mathrm{P}_{\mathbf{c}}$, which is a *facet description*, then only integral vertices occur and pure LP-methods can be applied. In this case every optimal LP-solution with respect to P is integral and a characteristic imset of a DAG G. However, if P does not coincide with a facet description, then we distinguish two cases. First, P contains exactly all $\mathbf{c}(G)$ as its integral points and we call it a *LP-relaxation*. We can apply IP-methods (e.g. branch and bound) in order to get the best BNS contained in P coinciding with the integer optimal solution. These IP-methods are computationally costly and not all vertices of P are integral. Second, P can contain additional integral points which do not coincide with a characteristic imset $\mathbf{c}(G)$ of any DAG G. The integer optimal solution $x^{\mathrm{opt}} \in P$ does not need to be a characteristic imset and is thus not valid for $\mathrm{P}_{\mathbf{c}}$. In spite of this x^{opt} may provide structural information leading to the best BNS. We call P in this case to provide a *valid inequality description* of $\mathrm{P}_{\mathbf{c}}$.

Throughout the complete chapter we derive several descriptions P of $\mathrm{P}_{\mathbf{c}}$ partly providing a LP-relaxation and partly providing a valid inequality description. To provide also facet descriptions we consider restricted structures of all DAGs and the corresponding restricted polytopes of characteristic imsets which inherit a fundamental simplification. These restricted structures are related to those presented in Sections 1.5, page 29, and in Chapter 4.

3.1.1 Decomposition of the characteristic imset polytope

The dynamic programming approach makes use of the decomposition property of the score function. Once a topological ordering of nodes is known, the best DAG respecting this ordering can be obtained by finding the best choice of parents for each node independently. Therein, we already know the potential parents of node i, namely $1, \ldots, i-1$. This decomposition and simplification of the optimisation problem can be seen in $\mathrm{P}_{\mathbf{c}}$, as well.

Let $\mathrm{P}_{\mathbf{c},\mathrm{sink}}(j)$ be the polytope of characteristic imsets of all DAGs G^j_{sink} which have only arcs entering node j. The vertices of $\mathrm{P}_{\mathbf{c},\mathrm{sink}}(j)$ are characteristic imsets of the form $\mathbf{c}(G^j_{\mathrm{sink}})$. According to Theorem 2.1.2, page 38, each, especially binary, vector $\mathbf{c}(G) \in \mathrm{P}_{\mathbf{c}}$ can be decomposed into a sum of unique vectors $\mathbf{c}(G^j_{\mathrm{sink}})$, for all terminal nodes j.

Theorem 3.1.1 *Let π be an ordering of nodes in N and*

$$\begin{aligned} \mathrm{P}_{\mathbf{c}} &:= \{\mathbf{c}(G) : G \in \mathit{DAGs}\}, \\ \mathrm{P}_{\mathbf{c}}^{\pi} &:= \{\mathbf{c}(G) : G \in \mathit{DAGs\ compatible\ with\ ordering}\ \pi\} \end{aligned}$$

and for each fixed $j \in N$ let

$$\begin{aligned} \mathrm{P}_{\mathbf{c},\mathrm{sink}}(j) &:= \{\mathbf{c}(G) : G \in \mathit{DAGs\ containing\ only\ arcs}\ i \to j\}, \\ \mathrm{P}_{\mathbf{c},\mathrm{sink}}^{\pi}(j) &:= \{\mathbf{c}(G) : G \in \mathit{DAGs\ containing\ only\ arcs}\ i \to j, \pi(i) < \pi(j)\} \end{aligned}$$

be given. Then, we have

$$\begin{aligned} \mathrm{P}_{\mathbf{c}} &\subseteq \mathrm{P}_{\mathbf{c},\mathrm{sink}}(1) + \mathrm{P}_{\mathbf{c},\mathrm{sink}}(2) + \ldots + \mathrm{P}_{\mathbf{c},\mathrm{sink}}(|N|), \\ \mathrm{P}_{\mathbf{c}}^{\pi} &= \mathrm{P}_{\mathbf{c},\mathrm{sink}}^{\pi}(1) + \mathrm{P}_{\mathbf{c},\mathrm{sink}}^{\pi}(2) + \ldots + \mathrm{P}_{\mathbf{c},\mathrm{sink}}^{\pi}(|N|). \end{aligned}$$

Proof. The inclusion relation holds since each DAG G can be decomposed into $|N|$ subgraphs on node set N in which we collect only those arcs entering node j, $j = 1, \ldots, |N|$. By Theorem 2.1.2 the characteristic imsets of these $|N|$ subgraphs G^j_{sink}, $j = 1, \ldots, |N|$, exactly add up to $\mathbf{c}(G)$. The inclusion relation can be strict since the parent sets of two distinct nodes $i, j \in N$ cannot be chosen independently as otherwise $i \to j$ and $i \leftarrow j$, a directed cycle, could be contained in G.

If we impose an ordering of nodes $\pi \in S_N$ in the permutation group S_N of nodes N, say $1 < 2 < \ldots < |N|$, and consider only graphs G^j_{sink} compatible with this ordering, then directed cycles cannot occur. We call G to be compatible with the ordering π if for any arc $i \to j$ in G we have $\pi(i) < \pi(j)$. According to this definition the potential parents of node i are only $\pi^{-1}(1), \ldots, \pi^{-1}(\pi(i) - 1)$.

The equality relation holds, since due to the given ordering π of the nodes, we can choose for each node $i \in N$ its parent set $\mathrm{pa}(i)$ independently without creating a cycle. □

If we additionally assume the score function $\mathbf{w}$ to be decomposable, then we can, for given ordering π, compute the best fitting model of $\mathrm{P}_{\mathbf{c}}^{\pi}$ by considering the smaller problems for $j = 1, \ldots, |N|$:

$$\begin{aligned} \max \quad & \mathbf{w}^{\intercal}\mathbf{c} \\ \text{s.t.} \quad & \mathbf{c} \in \mathrm{P}_{\mathbf{c},\mathrm{sink}}^{\pi}(j) \end{aligned}$$

and construct the overall optimal solution from of the optimal solutions for each subproblem.

Finding an overall best fitting model comprises to optimise each of the $|N|$ smaller problems and to construct $\mathbf{c}^{\mathrm{opt}}$ by adding one, possibly not unique, vertex of each of the maxima of the $|N|$ subproblems.

As there exists an ordering of nodes for every DAG, we can reduce the size of each smaller problem. If we define an ordering π, then each graph G^j_{sink} of the polytope $\mathrm{P}_{\mathbf{c},\mathrm{sink}}^{\pi}(j)$ consists of edges among at most $\pi(j) - 1$ nodes, which is a significant reduction in size.

This decomposition of the problem/polytope is an analogon of the property used for the dynamic programming approach (see Section 1.4.2.2, page 24). Both decompositions imply to find the best parents for each single node independently. It remains to find the best ordering of nodes, which is the central problem also for the dynamic programming approach.

3.1.2 Geometric neighbours in the characteristic imset polytope

Both Optimisation problems (2.1.5), page 41, and (1.4.10), pages 29 and 41, use an exponential representation of BN structures. To solve either of them an inequality description of the polytopes $\mathrm{P}_{\mathbf{c}}$ and $P(S)$ is needed. One possible way is to compute the inequalities of $\mathrm{P}_{\mathbf{c}}$ and $P(S)$ directly on the base of the characteristic and standard imsets, respectively, but this task is computationally infeasible, already for $|N| \geq 5$. The first idea presented in [65] is to compute the **skeleton** of the standard imset polytope, which is a graph consisting of the vertices and edges of a polytope. This skeleton can be used to perform a Simplex-based algorithm on the edges of the polytope. Up to now, there does not exist a complete description of the edges of $\mathrm{P}_{\mathbf{c}}$ and $P(S)$, respectively, and no explicit inequality description of $P(S)$. Note that due to isomorphic polytopes, any characterisation of the inequalities or edges of the one polytope can be used to derive a corresponding characterisation of the other. However, some families of the edges of $\mathrm{P}_{\mathbf{c}}$ and $P(S)$ can be derived.

We use the notion of geometric edges to characterise the edges of $\mathrm{P}_{\mathbf{c}}$ first. Two graphs G and H over N are called geometric neighbours if and only if $\mathbf{c}(G)$ and $\mathbf{c}(H)$ lie on the same edge in $\mathrm{P}_{\mathbf{c}}$. A fundamental result from [65] is, that every pair of inclusion neighbours are also geometric neighbours. The GES algorithm only adds and deletes edges in graphs and by doing this it makes use of the inclusion neighbours only but not of the geometric neighbours.

Definition 3.1.2 (geometric neighbours) *Let $\mathbf{c}(G)$ and $\mathbf{c}(H)$ be two characteristic imsets of $\mathrm{P}_{\mathbf{c}}$. The graphs G and H are called **geometric neighbours** if and only if there does not exist any $\lambda_K \geq 0$ such that*

$$\mathbf{c}(G) - \mathbf{c}(H) = \sum_{\substack{K \in DAGs, \\ K \neq G}} \lambda_K (\mathbf{c}(K) - \mathbf{c}(H)) \tag{3.1.2}$$

for $\mathbf{c}(K) \in \mathrm{P}_{\mathbf{c}}$.

Some geometric neighbours already follow from the decomposition property, Theorem 3.1.1, of $\mathrm{P}_{\mathbf{c}}$.

We use Definition (3.1.2) of geometric neighbours and evaluate Theorem 2.1.2, page 38, with respect to the empty graph.

Remark 3.1.3 *The graph G is not a geometric neighbour of the empty graph H if G consists of at least two nodes i, j with at least one entering arc.*

The empty graph H has the zero vector as its characteristic imset, $\mathbf{c}(H) = \mathbf{0}$. If graph G has at least two nodes i, j with at least one entering arc, then G^i_{sink} and G^j_{sink} are not empty and not G itself and we observe $\mathbf{c}(G) - \mathbf{c}(H) = \lambda_i(\mathbf{c}(G^i_{\text{sink}}) - \mathbf{0}) + \lambda_j(\mathbf{c}(G^j_{\text{sink}}) - \mathbf{0})$. This G satisfies Equation (3.1.2) with $\lambda_i = 1$, $\lambda_j = 1$ and G is not a geometric neighbour of the empty graph H.

Remark 3.1.3 provides an example of families of geometric neighbours but it does not provide a complete characterisation of all geometric edges. See [62] for more geometric neighbours in terms of the standard imset polytope. Moreover, this family of geometric neighbours can also be applied for the restricted polytopes in Section 3.4, page 77, of course if and only if both neighbours belong to this restricted structure in consideration.

For using families of edges for the optimisation problem, only heuristics can be applied. These heuristics are primal methods similar to the Simplex method applied to the edges of $\mathrm{P}_{\mathbf{c}}$. An

example of such a method is the GES algorithm using the family of inclusion neighbours. An open question is hence to completely characterise all families of geometric neighbours in $\mathrm{P}_{\mathbf{c}}$ in terms of pairs of graphs. The advantage of the usage of geometric neighbours is given within the graphs. Characterisations of neighbours in terms of graphs can gain from effective graph algorithms.

However, other descriptions of $\mathrm{P}_{\mathbf{c}}$ must be derived in order to apply polyhedral methods to optimality. So we have to go back to derive a general inequality description.

3.2 LP-relaxation of the characteristic imset polytope

3.2.1 Extended formulation

A valid LP-relaxation of $\mathrm{P}_{\mathbf{c}}$ contains all characteristic imsets $\mathbf{c}(G)$ describing valid patterns $\mathrm{pat}(G)$ of DAGs G. On the other hand, all 0/1-vectors not describing characteristic imsets are not contained in an appropriate LP-relaxation of the reformulated $\mathrm{P}_{\mathbf{c}}$. We derive an inequality description for the LP-relaxation of $\mathrm{P}_{\mathbf{c}}$ based on the analysis of properties of $\mathbf{c}(G)$ of DAGs G. For this purpose we introduce an extended formulation $\mathrm{P}_{\mathrm{ext}}$ consisting of $|N|(|N|-1)$-dimensional incidence vectors $\mathbf{y}(G)$ and characteristic imsets $\mathbf{c}(G)$ of all DAGs G:

$$\mathrm{P}_{\mathrm{ext}} := \mathrm{conv}\{(\mathbf{y}(G), \mathbf{c}(G)) : G \in \mathrm{DAGs}\}. \tag{3.2.1}$$

The projection of $\mathrm{P}_{\mathrm{ext}}$ onto entries corresponding to $\mathbf{c}(G)$ equals the characteristic imset polytope $\mathrm{P}_{\mathbf{c}}$ as a valid characteristic imset is contained in each vertex of $\mathrm{P}_{\mathrm{ext}}$ which is a 0/1-vector, i.e.

$$\mathrm{P}_{\mathrm{ext}}\,|_{\mathbf{c}} \cap \mathbb{Z}^{2^{|N|}-|N|-1} = \mathrm{P}_{\mathbf{c}} \cap \mathbb{Z}^{2^{|N|}-|N|-1}.$$

By convention we use $\mathbf{y} \in \mathrm{P}_{\mathrm{ext}}$ to abbreviate $(\mathbf{y}, \mathbf{x}) \in \mathrm{P}_{\mathrm{ext}}$ for a not necessarily specified $\mathbf{x}$.

Both a LP-relaxation of $\mathrm{P}_{\mathbf{c}}$ and of $\mathrm{P}_{\mathrm{ext}}$ yield an inequality description only containing valid characteristic imsets as integral points. The difference are the incidence vectors $\mathbf{y}(G)$, for all DAGs G, which are additionally contained in $\mathrm{P}_{\mathrm{ext}}$ and any LP-relaxation of it. In this section we derive a LP-relaxation of $\mathrm{P}_{\mathrm{ext}}$ and in the subsequent Section 3.3, page 64, we derive an inequality description of $\mathrm{P}_{\mathbf{c}}$.

Let $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ be a LP-relaxation of $\mathrm{P}_{\mathrm{ext}}$. A valid binary vector $(\mathbf{y}, \mathbf{x}) \in \mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ consists of an incidence vector $\mathbf{y}$ describing a DAG $G(\mathbf{y})$ and a vector $\mathbf{x}$ describing the pattern graph $G(\mathbf{x}|_{\leq 3})$ of $G(\mathbf{y})$ based on the entries of $\mathbf{x}|_{\leq 3}$ for sets of size ≤ 3. If and only if the binary variables $\mathbf{y}(G)$ in $\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}$ are describing a DAG G and if the binary variables $\mathbf{x}$ in $\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}$ are defined to describe $\mathrm{pat}(G)$ appropriately on the base of $\mathbf{y}(G)$, then $\mathbf{x}$ is a valid characteristic imset $\mathbf{c}(G)$ of a DAG G. This equivalence provides $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ to be a LP-relaxation of $\mathrm{P}_{\mathrm{ext}}$. We present inequalities providing $\mathbf{y}(G)$ to describe a DAG G and inequalities corresponding to properties of a vector $\mathbf{x}$ to be a characteristic imset $\mathbf{c}(G)$ of a DAG G given by $\mathbf{y}$.

In the next sections we introduce the inequality description of $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ and repeat properties from Chapter 2, page 37, necessary to describe a characteristic imset on the base of a DAG given by $\mathbf{y}$. For the sake of simplicity we give an outline of the proof that $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ induces indeed a valid LP-relaxation of $\mathrm{P}_{\mathrm{ext}}$ in the following section.

3.2.2 Outline for the proof of a LP-relaxation

We obtain an appropriate definition of inequalities in $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ by analysing the properties which provide $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ to be a LP-relaxation. For this purpose we present an outline of the basic steps

of a proof to show the following theorem and obtain the properties needed for the inequalities:

Theorem 3.2.1 *Let* $\mathrm{P}_{\mathbf{c}}$ *be the characteristic imset polytope over node set* N*, let* P_{ext} *be an extended polytope in dimension* $2^{|N|}-|N|-1+|N|(|N|-1)$*. Then,* $\mathrm{P}^{\text{rel}}_{\text{ext}}$ *is a valid LP-relaxation of* P_{ext} *and* $\mathrm{P}^{\text{rel}}_{\text{ext}}|_{\mathbf{c}} \cap \mathbb{Z}^{2^{|N|}-|N|-1} = \mathrm{P}_{\mathbf{c}} \cap \mathbb{Z}^{2^{|N|}-|N|-1}$.

Note that if $\mathrm{P}^{\text{rel}}_{\text{ext}}$ provides a LP-relaxation of P_{ext}, the first part of the theorem above, then the second part of it is a direct consequence of $\mathrm{P}_{\text{ext}}|_{\mathbf{c}} \cap \mathbb{Z}^{2^{|N|}-|N|-1} = \mathrm{P}_{\mathbf{c}} \cap \mathbb{Z}^{2^{|N|}-|N|-1}$.

3.2.2.1 Iterative definition of variables in the extended formulation

Let $\mathbf{y}$ be a 0/1-vector contained in a LP-relaxation $\mathrm{P}^{\text{rel}}_{\text{ext}}$ of the extended formulation and let $G(\mathbf{y})$ be the directed graph which we obtain by interpreting $\mathbf{y}$ as an incidence vectors of arcs. Given $\mathbf{y}$, $\mathbf{x}(\mathbf{y})$ denotes the definition of a vector $\mathbf{x}$ of $(\mathbf{y},\mathbf{x}) \in \mathrm{P}^{\text{rel}}_{\text{ext}}$ on the base of $\mathbf{y}$.

We conclude: If and only if the incidence vector $G(\mathbf{y})$ describes a DAG, then there exists a characteristic imset $\mathbf{c}(G(\mathbf{y}))$ such that a correct interpretation of $\mathbf{x}(\mathbf{y})$ yields $\mathbf{x}(\mathbf{y}) = \mathbf{c}(G(\mathbf{y}))$ and $(\mathbf{y},\mathbf{x}(\mathbf{y}))$ is a valid integral point in $\mathrm{P}^{\text{rel}}_{\text{ext}}$ and in P_{ext}.

To satisfy this equivalence, first, $G(\mathbf{y})$ must describe a DAG and, second, $\mathbf{x}$ must correspond to a correct interpretation of $\mathbf{y}$. Provided the definition of $\mathbf{x}$ is appropriate, then a valid $\mathbf{y}$ defines the integral point $(\mathbf{y},\mathbf{x})$ in $\mathrm{P}^{\text{rel}}_{\text{ext}}$ and in P_{ext}, uniquely.

Properties of characteristic imsets Let us recall some properties of characteristic imsets we need to define $\mathbf{x}$ on the base of a DAG G. We limit ourselves on sets $T \subseteq N$ with $|T| \geq 2$.

A fundamental property of characteristic imsets specifies the iterative definition of $\mathbf{c}(G)|_{\geq 4}$ on the base of $\mathbf{c}(G)|_{\leq 3}$. In particular in Lemma 2.2.7, page 45, we observe,

(a) $\mathbf{c}_T(G) = 1$,

(b) there exist $|T|-1$ subsets T_i of T with $|T_i| = |T|-1$ and $\mathbf{c}_{T_i}(G) = 1$,

(c) there exist three subsets T_i of T with $|T_i| = |T|-1$ and $\mathbf{c}_{T_i}(G) = 1$,

for a DAG G, a characteristic imset $\mathbf{c}(G)$ and sets $T \subseteq N$, $|T| \geq 4$. Due to monotonicity it is sufficient to provide the validity of implications "$(a) \Longrightarrow (b)$" and "$(c) \Longrightarrow (a)$" to satisfy all equivalences.

This property has the following consequences: First $\mathbf{c}(G)$ of a DAG G is uniquely determined by the entries for sets T with $|T| \leq 3$. Second, the entries for sets T with $|T| \geq 4$ are uniquely determined on the base of entries with $|T| \leq 3$. Given the $\binom{|N|}{2} + \binom{|N|}{3}$ entries of $\mathbf{c}(G)|_{\leq 3}$, the entries of $\mathbf{c}(G)|_{\geq 4}$ can be obtained uniquely.

Theorem 2.2.5, page 43, implies a direct graphical translation of $\mathbf{c}(G)|_{\leq 3}$. This graphical translation coincides with $\mathrm{pat}(G)$. The edges of $\mathrm{pat}(G)$ are determined by $\mathbf{c}(G)|_{=2}$ and the immoralities and cliques of size three are determined by $\mathbf{c}(G)|_{=3}$.

Because of this iterative definition of characteristic imsets our strategy to derive inequalities of $\mathrm{P}^{\text{rel}}_{\text{ext}}$ is to define $\mathbf{x}(G)|_{\leq 3}$ for a DAG G and to derive inequalities providing the Implications "$(a) \Longrightarrow (b)$" and "$(c) \Longrightarrow (a)$" of Lemma 2.2.7, page 45, above, which defines $\mathbf{x}(G)|_{\geq 4}$. If and only if G is a DAG, then $\mathbf{x}(G)$ is a valid characteristic imset, i.e. $\mathbf{x}(G) = \mathbf{c}(G)$.

Note, that the choice of $\mathbf{y}(G)$ of $(\mathbf{y}(G),\mathbf{x}(G)) \in \mathrm{P}^{\text{rel}}_{\text{ext}}$ is not unique for a vector $\mathbf{x}$, because the pattern $G(\mathbf{x}|_{\leq 3})$ is a representation of the Markov equivalence class of all DAGs, in which G is contained. In fact, there exist valid binary vectors $(\mathbf{y}(G),\mathbf{x})$ in $\mathrm{P}^{\text{rel}}_{\mathbf{c}}$, in which $\mathbf{x}$ is fixed and $\mathbf{y}(G)$ varies, for all DAGs G of the same Markov equivalence class.

3.2.2.2 Summary of necessary properties to provide a LP-relaxation of the extended formulation

To complete the outline of the proof of Theorem 3.2.1 we have to show the inequality description of $\mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}}$, introduced in the next section, to encode the properties above. We repeat these necessary properties under which Theorem 3.2.1 is valid and which a LP-relaxation of $\mathrm{P}_{\mathrm{ext}}$ should provide:

Lemma 3.2.2 *The binary vector* $(\mathbf{y}, \mathbf{x})$ *is an element of* $\mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}}$ *if and only if*

1. $\mathbf{y}$ *is the incidence vector of the arcs of a DAG* G*,*
2. $\mathbf{x}|_{=2}$ *defines the underlying undirected graph* $\bar{G}$ *of* G*,*
3. $\mathbf{x}|_{=3}$ *defines the immoralities and cliques of size three of* G *and*
4. $\mathbf{x}$ *satisfies Lemma 2.2.7, page 45.*

3.2.3 Derivation of inequalities

In this section we derive inequalities describing $\mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}}$ with respect to properties of $\mathbf{c}(G) \in \mathrm{P}_{\mathbf{c}}$. Each subsequent sets of inequalities represent equivalently a property of a $\mathbf{c}(G)$ and a valid incidence vector $\mathbf{y}(G)$ of a DAG G. If and only if the inequalities are satisfied, then $\mathbf{x}$ of a 0/1-vector $(\mathbf{y}(G), \mathbf{x}) \in \mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}}$ is contained in $\mathrm{P}_{\mathrm{ext}}$ and in $\mathrm{P}_{\mathbf{c}}$.

We use Lemma 3.2.2 and the specified Arguments: 1 - the *incidence vectors of directed acyclic graphs*, 2 - the definition of the *underlying undirected graph*, 3 - the definition of *immoralities and cliques* and 4 - the correct *characteristic imset extension*.

1 - Incidence vectors of directed acyclic graphs For the well-known acyclic subgraph polytope $P_{\mathrm{AC}} \subseteq \mathbb{R}^{|N|(|N|-1)}$ there exists an inequality description of a valid LP-relaxation P_{C} [25]. Let $\mathbf{y} \in \{0,1\}^{|N|(|N|-1)}$ be the incidence vector of a set of arcs A of a directed graph $D = (N, A)$. Then $y_a = 1$ if and only if $a \in A$. As arcs in A are ordered pairs of nodes we use the notation $y_{i,j} = 1$ to indicate arc $i \leftarrow j$ in D. Then

$$P_{\mathrm{C}} := \{\mathbf{y} \in [0,1]^{|N|(|N|-1)} : \\ \mathbf{y}(C) \leq |C| - 1, \ \forall \text{ directed cycles } C \subseteq N\}. \tag{3.2.2}$$

In Inequalities (3.2.2) the notion $y(C)$ for a cycle $C := \{\{i_1, i_2\}, \{i_2, i_3\}, \ldots, \{i_l, i_1\}\}$ of length $l \geq 2$ and ordered arcs is used to replace the sum $\sum_{\{i,j\} \in C} y_{i,j}$. We use these inequalities directly to describe a DAG G in $\mathbf{y}(G) \in \mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}}$ and add Inequalities (3.2.2) to the description of $\mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}}$. All binary vectors $\mathbf{y} \in \mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}}$ satisfy Inequalities (3.2.2) and coincide with valid DAGs over the same node set N. In particular, we have $\mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}}|_{\mathbf{y}} \cap \mathbb{Z}^{|N|(|N|-1)} = P_{\mathrm{C}} \cap \mathbb{Z}^{|N|(|N|-1)}$.

We call the set of Inequalities (3.2.2) the **cycle-inequalities**.

2 - Underlying undirected graph The underlying undirected graph $\bar{G}$ of a DAG G contains an edge $\{i, j\}$ if and only if either $i \rightarrow j$ or $i \leftarrow j$ is contained in G. If G is a DAG, then arcs $i \rightarrow j$ and $i \leftarrow j$ do not occur simultaneously. We identify $\mathbf{x}|_{=2}$ contained in $\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}$ with the characteristic vector of the edge set of $\bar{G}$. We derive the following equations for binary vectors $\mathbf{y}$ and $\mathbf{x}$:

$$y_{i,j} + y_{j,i} = x_{\{i,j\}}, \ \forall i \neq j \in N. \tag{3.2.3}$$

These equations define $x_{\{i,j\}} = 1$, for a binary vector $\mathbf{x}$, if and only if either arc $i \to j$ or arc $i \leftarrow j$ is in G, which is the definition of the characteristic vector $\mathbf{x}|_{=2}$ of $\bar{G}$. We add Equations (3.2.3) to the description of $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ and conclude, that a binary vector $\mathbf{x}$ satisfies these equations if and only if $\mathbf{x}|_{=2}$ describes the edges of the underlying undirected graph $\bar{G}$ of G induced by binary vectors $\mathbf{y}$. If $\mathbf{x}$ is assumed to be binary, then these equations especially enforce not both arcs $i \to j$ and $i \leftarrow j$ to exist simultaneously, which would be a directed cycle of length two, and hence we can restrict $|C| \geq 3$ in the cycle-inequalities above.

3 - Immoralities and cliques G is a DAG if and only if each cycle in G contains a node with at least two parents within the cycle. Moreover the immoralities of G together with the edges of the underlying undirected graph $\bar{G}$ define $\mathrm{pat}(G)$. Therefore any vector $\mathbf{x}$ of $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ with $\mathbf{x}|_{\leq 3}$ representing $\mathrm{pat}(G)$ must contain entries for immoralities. As we use characteristic imsets for a pattern representation we also suppose this vector $\mathbf{x}|_{\leq 3}$ to specify the cliques of G. Let $T \subseteq N$, $|T| = 3$, be fixed. The following inequalities describe the fact, that if T induces an immorality or clique in G, then $x_T = 1$. Note, that the following inequalities do not provide the back direction of the implication. In order to prove every vector $\mathbf{x}|_{\leq 3}$ to describe exactly the immoralities and cliques of size three of G a second set of inequalities must be derived. However,

$$y_{i,j} + y_{i,k} \leq 1 + x_{\{i,j,k\}},\ \forall i \neq j \neq k \in N, \tag{3.2.4}$$

implies a vector $\mathbf{x}|_{\leq 3}$ to contain all the immoralities and cliques specified in $\mathbf{y}$. Let i be a terminal node, then there exist other nodes j, k and arcs $y_{i,j} = 1$ and $y_{i,k} = 1$. Therefore, $x_{\{i,j,k\}} = 1$ is the only choice to satisfy the corresponding inequality. Last but not least, if i is a terminal node, then it is contained in an immorality or clique defined by $\{i, j, k\}$, which provides the implication: If T induces an immorality or a clique in G, then $x_T = 1$. Moreover, these inequalities are valid for $\mathrm{P}_{\mathrm{ext}}$ as if both $y_{i,j} = 1$ and $y_{i,k} = 1$, then arcs $i \leftarrow j$ and $i \leftarrow k$ are present in a DAG G and $\{i, j, k\}$ induced either an immorality or a clique. With this, $c_{\{i,j,k\}}(G) = 1$ and $\mathbf{x}(G) = \mathbf{c}(G)$ satisfies the inequalities above.

The following four sets of Inequalities (3.2.5) below and (3.2.6), next page, together provide the other direction of the implication above: If $x_T = 1$, then T induces an immorality or clique in G.

Provided $G(\mathbf{y})$ is a DAG, a clique of size three is defined by the existence of three contained edges. As the edges are appropriately defined by Equations (3.2.3) the existence of all edges $x_{\{i,j\}} = 1$, $x_{\{i,k\}} = 1$ and $x_{\{j,k\}} = 1$ is sufficient to enforce $x_{\{i,j,k\}} = 1$. Second if $\{i, j, k\}$ is an immorality or a clique and $x_{\{i,j,k\}} = 1$, then at least two edges have to exist in $\bar{G}$ and $x_{\{i,j\}} = 1$, $x_{\{i,k\}} = 1$, $x_{\{j,k\}} = 1$ for at least two of those.

$$\begin{aligned} x_{\{i,j\}} + x_{\{i,k\}} + x_{\{j,k\}} &\leq 2 + x_{\{i,j,k\}}, \\ 2x_{\{i,j,k\}} &\leq x_{\{i,j\}} + x_{\{i,k\}} + x_{\{j,k\}}, \\ &\forall i \neq j \neq k \in N. \end{aligned} \tag{3.2.5}$$

The second set of inequalities is valid for $\mathrm{P}_{\mathrm{ext}}$, because if $c_{\{i,j,k\}} = 1$, then $\bar{G}$ contains at least two edges of $\{i, j\}$, $\{i, k\}$, $\{j, k\}$ and $c_{\{i,j\}} + c_{\{i,k\}} + c_{\{j,k\}} \geq 2$. Both sets of inequalities provide a necessary condition on the implication: If $x_T = 1$, then T induces an immorality or clique in G. The necessary condition is determined by the fact, that if T is an immorality or a clique of size three, then sufficiently enough edges and arcs must be present.

The last sets of inequalities provide the direction of the arcs induced by $x_T = 1$ which provides the equivalence: T is an immorality or a clique of size three if and only if there exists a terminal node in T. This equivalence represents the correct definition of a characteristic imset.

However, if $\{i, j, k\}$ induces an immorality, then exactly two edges of $\{i, j\}$, $\{i, k\}$ and $\{j, k\}$

must be contained in $\bar{G}$ and for these two edges, say $\{i,j\}$, $\{i,k\}$, the node i has both j and k as its parents. On the other hand, if i is not terminal but both edges $\{i,j\}$, $\{i,k\}$ exist and $x_{\{i,j,k\}} = 1$, then $\{i,j,k\}$ induces a clique and also edge $\{j,k\}$ has to exist.

$$\begin{aligned} x_{\{i,j\}} + x_{\{i,k\}} + x_{\{i,j,k\}} &\leq 2 + x_{\{j,k\}} + y_{i,j}, \\ x_{\{i,j\}} + x_{\{i,k\}} + x_{\{i,j,k\}} &\leq 2 + x_{\{j,k\}} + y_{i,k}, \\ &\forall i \neq j \neq k \in N,\ \forall i \in \{i,j,k\}. \end{aligned} \tag{3.2.6}$$

Inequalities (3.2.6) provide the bijection between general terminal nodes in G and cliques or immoralities specified in $\mathbf{x}$. Both inequalities are valid for $\mathrm{P}_{\mathrm{ext}}$. If $c_{\{i,j\}} = 1, c_{\{i,k\}} = 1$ and $c_{\{i,j,k\}} = 1$, then $\{i,j,k\}$ is an immorality or a clique. In both cases $\{i,j,k\}$ contains a terminal node. If this terminal node is i, then both $y_{i,j} = 1$ and $y_{i,k} = 1$ and the inequalities are satisfied. If the terminal node is not i, then one of the arcs $j \to k$ or $j \leftarrow k$ exists and $c_{\{j,k\}} = 1$ which also satisfies the inequalities above.

Let the binary vector $(\mathbf{y}, \mathbf{x}(\mathbf{y})) \in \mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ be given, which is valid for the inequalities above. We conclude: If T is an immorality or a clique, then with Inequalities (3.2.4), $c_T = x_T = 1$ for the characteristic imset $\mathbf{c}(G)$ of the DAG G defined by $G(\mathbf{y})$. Contrarily, provided $x_T = 1$ with $T = \{i,j,k\}$, then two edges, say $\{i,j\}$ and $\{i,k\}$ with $x_{\{i,j\}} = 1$ and $x_{\{i,k\}} = 1$, exist due to the second Inequalities of (3.2.5). Inequalities (3.2.6) are only satisfied if $x_{\{j,k\}} = 1$ or if both $y_{i,j} = 1$, $y_{i,k} = 1$. In the first case T is a clique, which we assume to be acyclic with respect to the cycle-inequalities and $c_T = 1$, as well. In the second case node i is terminal and Inequalities (3.2.4) apply yielding T to be an immorality or clique and $c_T = 1$.

Up to now any binary vector $(\mathbf{y}, \mathbf{x}|_{\leq 3})$ contained in $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ coincides exactly with a vector $(\mathbf{y}(G), \mathbf{c}(G)|_{\leq 3})$ for a DAG G. All the introduced inequalities together provide on the one hand an encoding of a DAG via $\mathbf{y}(G)$ and on the other hand a correct identification of $\mathbf{c}(G)|_{=2}$ and $\mathbf{c}(G)|_{=3}$. The next inequalities specify the iterative identification of $\mathbf{c}(G)|_{\geq 4}$, which are especially necessary to provide the linearity of the objective function.

4 - Characteristic imset extension Lemma 2.2.7, page 45, can be used to obtain conditions on 0/1-vectors $\mathbf{x}$ to have well-defined entries $\mathbf{x}|_{\geq 4}$ on the base of $\mathbf{x}|_{=3}$, in terms of characteristic imsets.

We follow Lemma 2.2.7 and derive the corresponding inequalities. The following figures visualise two occurring cases. Figure 3.1 below corresponds to Direction "$(c) \Longrightarrow (a)$" applied to a clique T of size four. Figure 3.2 below corresponds to Direction "$(a) \Longrightarrow (b)$" applied to a set T of size four with similar properties than an immorality.

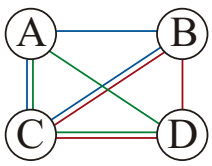

Figure 3.1: Three cliques of size three imply the existence of the covering clique $\{A,B,C,D\}$ of size four.

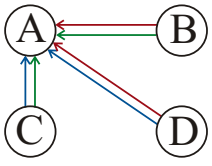

Figure 3.2: An extended immorality $\{A,B,C,D\}$ of size four implies the existence of three contained immoralities.

We consider implications "$(c) \Longrightarrow (a)$" and "$(a) \Longrightarrow (b)$" and derive the following two sets of inequalities.

$$\sum_{i \in T} x_{T \setminus \{i\}} \leq 2 + (|T| - 2) x_T, \tag{3.2.7}$$

$$(|T|-1)x_T \leq \sum_{i\in T} x_{T\setminus\{i\}},$$
$$\forall T \subseteq N,\ |T| \geq 4$$

Both sets of inequalities are a direct translation of Directions "$(c) \Longrightarrow (a)$" and "$(a) \Longrightarrow (b)$" of Lemma 2.2.7. If $\mathbf{c}(G)$ is a characteristic imset of a DAG G and there exist at least three sets $T \setminus \{i\}$ with $c_{T\setminus\{i\}}(G) = 1$, then the left hand side of the first inequalities is ≥ 3 but $\leq |T|$. Direction "$(c) \Longrightarrow (a)$" implies $c_T(G) = 1$ and the right hand side of the inequalities is equal to $|T|$. On the other hand, if there are at most two subsets with $c_{T\setminus\{i\}}(G) = 1$, then the inequalities are satisfied with $2 \leq 2$. Likewise the second set of inequalities is valid. If $c_T(G) = 1$, then there are, with "$(a) \Longrightarrow (b)$", at least $|T| - 1$ subsets with $c_{T\setminus\{i\}}(G) = 1$. The left hand side of the second inequalities is equal to $|T| - 1$ which is at most the right hand side. If $c_T(G) = 0$, the inequalities are satisfied due to non-negativity of the entries $c_{T\setminus\{i\}}(G)$.

The remaining Direction "$(b) \Longrightarrow (c)$" follows immediately. Therefore any $\mathbf{c}(G) \in \mathrm{P}_{\mathrm{ext}}$ satisfies Inequalities (3.2.7).

Since Inequalities (3.2.7) are a direct translation of Directions "$(c) \Longrightarrow (a)$" and "$(a) \Longrightarrow (b)$" of Lemma 2.2.7 we have: Any 0/1-vector $\mathbf{x}$ contained in $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ and feasible for Inequalities (3.2.7) also satisfies Lemma 2.2.7 and therefore has a valid and unique *characteristic imset extension* of $\mathbf{x}|_{\geq 4}$ on the base of $\mathbf{x}|_{=3}$.

Together with Inequalities (3.2.5) we introduce

$$\sum_{i\in T} x_{T\setminus\{i\}} \leq 2 + (|T|-2)x_T, \tag{3.2.8}$$
$$(|T|-1)x_T \leq \sum_{i\in T} x_{T\setminus\{i\}},$$
$$\forall T \subseteq N,\ |T| \geq 3,$$

and call them altogether the **cim-inequalities**. Note, that the first Inequalities of (3.2.5) are theoretically redundant with respect to Inequalities (3.2.4) but we consider them explicitly as Inequalities (3.2.8) can be formulated symmetrically for sets $T \subseteq N$, $|T| \geq 3$, this way.

3.2.4 Complete description

Let $t := 2^{|N|} - |N| - 1$ and let $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ contain Inequalities (3.2.2) (3.2.3), (3.2.6) and (3.2.8) and Equations (3.2.4). We have

$$\begin{aligned}
\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}} = \{\mathbf{y} \in [0,1]^{|N|(|N|-1)}, \quad & & \mathbf{x} \in [0,1]^t : & & (3.2.9)\\
\mathbf{y}(C) & \leq & |C| - 1, & \forall \text{ directed cycles } C \subseteq N,\\
& & & |C| \geq 3\\
y_{i,j} + y_{j,i} & = & x_{i,j}, & \forall i \neq j \in N\\
y_{i,j} + y_{i,k} & \leq & 1 + x_{\{i,j,k\}}, & \forall i \neq j \neq k \in N\\
x_{\{i,j\}} + x_{\{i,k\}} + x_{\{i,j,k\}} & \leq & 2 + x_{\{j,k\}} + y_{i,j}, & \forall i \neq j \neq k \in N,\\
x_{\{i,j\}} + x_{\{i,k\}} + x_{\{i,j,k\}} & \leq & 2 + x_{\{j,k\}} + y_{i,k}, & \forall i \in \{i,j,k\}
\end{aligned}$$

$$\begin{aligned} \sum_{i\in T} x_{T\setminus\{i\}} &\leq 2+(|T|-2)x_T, \quad \forall T\subseteq N,\ |T|\geq 3 \\ (|T|-1)x_T &\leq \sum_{i\in T} x_{T\setminus\{i\}}, \quad \forall T\subseteq N,\ |T|\geq 3\ \}. \end{aligned}$$

This description of $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ fulfils Theorem 3.2.1.

Theorem 3.2.3 $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ *as defined above is a valid LP-relaxation of* $\mathrm{P}_{\mathrm{ext}}$.

Proof. To proof this theorem, we have to show $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ to contain exactly all valid characteristic imsets as its integral points and, with this, we have to show that binary vectors $(\mathbf{y},\mathbf{x})$ in $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ coincide exactly with $(\mathbf{y}(G),\mathbf{c}(G))$ in $\mathrm{P}_{\mathrm{ext}}$, for all DAGs G. We apply Lemma 3.2.2 and conclude $\mathbf{x}$ to be a characteristic imset $\mathbf{c}(G)$ if and only if the corresponding $\mathbf{y}$ defines a DAG G (Argument 1), if edges in $G(x|_{=2})$ coincide with $\bar{G}$ of G (Argument 2), if entries of $x|_{=3}$ coincide with the immoralities and cliques of G (Argument 3) and if $x|_{\geq 4}$ is chosen according Lemma 2.2.7 (Argument 4).

The cycle-Inequalities (3.2.2) provide $\mathbf{y}$ with $(\mathbf{y},\mathbf{x})\in\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ to describe all DAGs of the same node set N. Equations (3.2.3) provide the equivalence between the edges specified in $G(\mathbf{x}|_{=2})$ and in $\bar{G}$ of the DAGs G in $\mathbf{y}$. Inequalities (3.2.4), (3.2.6) and the cim-Inequalities (3.2.8) for $|T|=3$ together provide the equivalence between immoralities and cliques of size three in G and in $G(\mathbf{x}|_{=3})$. Last but not least the cim-Inequalities (3.2.8) for $|T|\geq 4$ are a direct translation of the conditions of Lemma 2.2.7 and provide $\mathbf{x}|_{\geq 4}$ and the correctness and uniqueness of its choice.

Let $\mathbf{c}(G)$ be a characteristic imset of a DAG G, then a fixed $\mathbf{c}(G)$ defines vectors $\mathbf{y}(G)$ for each G of the same Markov equivalence class. We conclude $\mathbf{c}(G)$ to coincide with $\mathbf{x}$ for all the corresponding $\mathbf{y}(G)$ and hence $(\mathbf{y}(G),\mathbf{c}(G))\in\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$. Contrariwise, let $\mathbf{x}$ be a binary vector in $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$, then $\mathbf{x}|_{\leq 3}$ describes for all DAGs G in $\mathbf{y}(G)$ a mixed graph $G(\mathbf{x}|_{\leq 3})$ having the same underlying undirected graph as G and the same immoralities. Then $G(\mathbf{x}|_{\leq 3})$ defines the pattern of this graph G. Moreover, for $\mathbf{x}|_{\geq 4}$ Lemma 2.2.7, page 45, is valid and $\mathbf{x}$ coincides with $\mathbf{c}(G)$, which implies $(\mathbf{y}(G),\mathbf{x})\in\mathrm{P}_{\mathrm{ext}}$. □

To solve the structure learning task we use $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ given by System (3.2.9) together with the integrality condition and apply a score equivalent and decomposable score function $\mathbf{w}$ to variables $\mathbf{x}$ of $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ only:

$$\begin{aligned} \max\ & \mathbf{w}^{\intercal}\mathbf{x} \\ \text{s.t.}\ & (\mathbf{y},\mathbf{x})\in\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}, \\ & \mathbf{y}\in\mathbb{Z}^{|N|(|N|-1)}, \\ & \mathbf{x}\in\mathbb{Z}^{2^{|N|}-|N|-1}. \end{aligned} \tag{3.2.10}$$

3.2.5 Additional Inequalities and alternative formulations

There exist possible modifications of the Description (3.2.9) of $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$. Although this modifications lead to other or at least additional inequalities these modifications may provide on the one hand the intergality of variables $\mathbf{x}$ if variables $\mathbf{y}$ are integral and on the other hand alternative descriptions of DAGs defined by $\mathbf{y}$. Both modifications provide properties which are from a practical perspective useful to improve the computation times of the structure learning task.

Inequalities due to a relaxed integrality condition We introduce the following Inequalities:

$$\sum_{j\in T\setminus\{i\}} y_{i,j} \leq |T| - 2 + x_T, \quad \forall T \subseteq N,\ |T| \geq 4,\ \forall i \in T \tag{3.2.11}$$

$$\sum_{j\in T\setminus\{i\}} x_{T\setminus\{j\}} + x_T \leq |T| - 1 + x_{T\setminus\{i\}} + y_{i,j}, \quad \forall T \subseteq N,\ |T| \geq 4,\ \forall i \in T,\ \forall j \in T\setminus\{i\} \tag{3.2.12}$$

$$x_T \leq x_{T\setminus\{i\}} + x_{T\setminus\{j\}}, \quad \forall T \subseteq N,\ |T| \geq 3,\ \forall i \neq j \in T. \tag{3.2.13}$$

Inequalities (3.2.11) and (3.2.12) above generalise Inequalities (3.2.4) and Inequalities (3.2.6) for sets T with $|T| \geq 4$ because of which they are valid for $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ and $\mathrm{P}_{\mathrm{ext}}$. Inequalities (3.2.13) provide the negation of Lemma 2.2.7, page 45.

If $|T| \geq 4$ and $\mathbf{c}(G)$ is a characteristic imset, then the negation of Direction "$(a) \Longrightarrow (c)$" of Lemma 2.2.7 indicates: $c_T = 0$, if there exist at most two subsets $T_i \subseteq T$ with $|T_i| = |T| - 1$ and $c_{T_i} = 1$. In particular, as $|T| \geq 4$, there exist two nodes i, j with subsets $T\setminus\{i\}$, $T\setminus\{j\}$ and $c_{T\setminus\{i\}} + c_{T\setminus\{i\}} = 0$. If $|T| = 3$, then at least two edges in T must exist to provide a terminal node in T, hence $c_T = 0$ if $c_{\{i,j\}} + c_{\{i,k\}} = 0$ for two edges $\{i,j\}$ and $\{i,k\}$ in T is valid. On the other hand, let $c_T = 1$ for $|T| \geq 3$, then with Inequalities (3.2.8) at least $|T| - 1$ subsets T_i exist with $|T_i| = |T| - 1$ and $c_{T_i} = 1$, which implies $c_{T\setminus\{i\}} + c_{T\setminus\{i\}} \geq 1$ for all $i \neq j \in T$. This shows Inequalities (3.2.13) to be valid for $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$.

Moreover, we can conclude all of the inequalities above to be valid for the integral vectors in $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$. The advantage of the addition of these inequalities is visible by the implied integrality of $\mathbf{x}$. Let us assume $\mathbf{y}$ to be a binary vector with $(\mathbf{y}, \mathbf{x}) \in \mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ added by Inequalities (3.2.11) (3.2.12) and (3.2.13), then $\mathbf{x}$ is integral, too.

We define $\mathrm{P}^{\mathrm{rel}\,\prime}_{\mathrm{ext}}$ to coincide with $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ together with Inequalities (3.2.11), (3.2.12) and (3.2.13).

Theorem 3.2.4 *Let $\mathrm{P}^{\mathrm{rel}\,\prime}_{\mathrm{ext}}$ be defined as above and $(\mathbf{y}, \mathbf{x}) \in \mathrm{P}^{\mathrm{rel}\,\prime}_{\mathrm{ext}}$. First, the integral points coincide, i.e. $\mathrm{P}^{\mathrm{rel}\,\prime}_{\mathrm{ext}} \cap \mathbb{Z}^t$ equals $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}} \cap \mathbb{Z}^t$, $t = |N|(|N|-1) + 2^{|N|} - |N| - 1$. Second, if vectors $\mathbf{y} \in \{0,1\}^{|N|(|N|-1)}$, then $\mathbf{x} \in \{0,1\}^{2^{|N|}-|N|-1}$.*

Proof. For the description of $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ we can already observe the integrality of variables $x_{\{i,j\}}$, $\forall i \neq j \in N$, provided $\mathbf{y}$, with $(\mathbf{y}, \mathbf{x}) \in \mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$, is integral.

Likewise Inequalities (3.2.4) and Inequalities (3.2.6) provide the integrality of $\mathbf{x}|_{=3}$. If, in Inequalities (3.2.4), $y_{i,j} = 1$ and $y_{i,k} = 1$, then $x_{\{i,j,k\}} = 1$, too. Contrarily, if $y_{i,j} = 0$ and we assume $x_{\{i,j,k\}} \neq 0$, then we distinguish between $x_{\{i,j\}} = 1$, $x_{\{i,k\}} = 1$ and $x_{\{i,j\}} = 0$ or $x_{\{i,k\}} = 0$, for all $i \in \{i,j,k\}$, in the Inequalities (3.2.6). In the first case, we have $x_{\{i,j\}} + x_{\{i,k\}} + x_{\{i,j,k\}} > 2$ which implies $x_{\{j,k\}} = 1$ or both $y_{i,j} = 1$, $y_{i,k} = 1$. If $x_{\{j,k\}} = 1$ we can apply the first of the cim-Inequalities (3.2.8) for $|T| = 3$ and obtain $x_{\{i,j,k\}} = 1$. If both $y_{i,j} = 1$ and $y_{i,k} = 1$, then a contradiction with our assumption is yield. In the second case we have $x_{\{i,j\}} = 0$ or $x_{\{i,k\}} = 0$, for all nodes i, and $x_{\{l,l_1\}} = 0$, $x_{\{l,l_2\}} = 0$ follows for a node l with $\{l, l_1, l_2\} = \{i,j,k\}$. We use Inequalities (3.2.13) for $|T| = 3$ and get $x_{\{i,j,k\}} \neq 0 = x_{\{l,l_1\}} + x_{\{l,l_2\}} = 0$, a contradiction.

The reformulated versions of Inequalities (3.2.4) and (3.2.6), i.e. (3.2.11) and (3.2.12), together with (3.2.13) imply by induction: If all entries x_S are integral for $|S| < |T|$, $|S| \geq 2$ and $|T| \geq 3$, then x_T is integral.

The case $|T| = 3$ is clear due to above. Let $|T| = n+1$ and $|S| \leq n$. By induction hypothesis x_S is integral if the corresponding $\mathbf{y}$ of $(\mathbf{y}, \mathbf{x}) \in \mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}\,\prime}$ are integral. First of all, if T induces a clique, then for all subsets $S \subseteq T$ with $|S| = |T| - 1$ there is $x_S = 1$ as well as $x_T = 1$, due to the second Inequalities of (3.2.8). Generally, if T contains a terminal node, then Inequalities (3.2.11) imply $x_T = 1$ since $\mathbf{y}$ is integral. Therefore, let both, a subset $S \subseteq T$, $|S| = |T| - 1$, with $x_S = 0$ and $y_{i,j} = 0$ for all $i \in T$ and $j \in T \setminus \{i\}$ be given, such that $x_T = 1$ is not necessarily be implied by Inequalities (3.2.11), for any choice of the terminal node $i \in T$. For deriving a contradiction we assume $x_T \neq 0$.

First we apply Inequalities (3.2.12) and distinguish between $\sum_{j \in T\setminus\{i\}} x_{T\setminus\{j\}} = |T| - 1$, for a node $i \in T$, and $\sum_{j \in T\setminus\{i\}} x_{T\setminus\{j\}} < |T| - 1$, $\forall i \in T$. The first case yields $x_{T\setminus\{i\}} = 1$, and T is a clique, then x_T is integral, or $y_{i,j} = 1$, $\forall j \in T \setminus \{i\}$ and $i \in T$, and T is an immorality, then x_T is integral. Therefore, $\sum_{j \in T\setminus\{i\}} x_{T\setminus\{j\}} < |T| - 1$, $\forall i \in T$, and moreover let T induce no immorality. We conclude at most two subsets S_1, S_2, $|S_1| = |S_2| = |T| - 1$, to exist with $x_{S_1} = 1$ and $x_{S_2} = 1$. Applying Inequalities (3.2.13) yields $x_T = x_{T\setminus\{i\}} + x_{T\setminus\{j\}} = 0$ for two other sets, a contradiction. □

For a repetition, we define

$$
\begin{aligned}
\mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}\,\prime} := \{\mathbf{y} \in [0,1]^{|N|(|N|-1)}, \quad & \mathbf{x} \in [0,1]^t : && (3.2.14)\\
\mathbf{y}(C) \quad & \leq \quad |C| - 1, && \forall \text{ directed cycles } C \subseteq N,\ |C| \geq 3\\
y_{i,j} + y_{j,i} \quad & = \quad x_{i,j}, && \forall i \in N,\ \forall j \in N \setminus \{i\}\\
\sum_{j \in T\setminus\{i\}} y_{i,j} \quad & \leq \quad |T| - 2 + x_T, && \forall T \subseteq N,\ |T| \geq 3,\ \forall i \in T\\
\sum_{j \in T\setminus\{i\}} x_{T\setminus\{j\}} + x_T \quad & \leq \quad |T| - 1 + x_{T\setminus\{i\}} + y_{i,j}, && \forall T \subseteq N,\ |T| \geq 3,\\
& && \forall i \in T,\ \forall j \in T \setminus \{i\}\\
\sum_{i \in T} x_{T\setminus\{i\}} \quad & \leq \quad 2 + (|T| - 2)x_T, && \forall T \subseteq N,\ |T| \geq 3\\
(|T| - 1)x_T \quad & \leq \quad \sum_{i \in T} x_{T\setminus\{i\}}, && \forall T \subseteq N,\ |T| \geq 3\\
x_T \quad & \leq \quad x_{T\setminus\{i\}} + x_{T\setminus\{j\}}, && \forall T \subseteq N,\ |T| \geq 3,\\
& && \forall i \neq j \in T\ \}
\end{aligned}
$$

and consider the optimisation problem

$$
\begin{aligned}
\max \quad & \mathbf{w}^\intercal \mathbf{x} \\
\text{s.t.} \quad & (\mathbf{y}, \mathbf{x}) \in \mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}\,\prime}, \\
& \mathbf{y} \in \mathbb{Z}^{|N|(|N|-1)}.
\end{aligned} \qquad (3.2.15)
$$

The relaxed integrality condition can be exploited when solving the Optimisation problem (3.2.15) with branch and bound techniques. Given a fractional LP-solution only the $\mathbf{y}$ variables which are only polynomial many need to be considered for branching. This way, an integer solution $(\mathbf{y}, \mathbf{x})$ may be obtained after fewer branching steps resulting in better computation times.

Alternative directed cycle formulations The cycle-Inequalities (3.2.2) provide the acyclicity of every ordered subset of arcs of a DAG G. An alternative definition of a DAG yields: G is a DAG if and only if every cycle C of $\bar{G}$ contains an immorality (compare with Figure 3.3, next page) or a clique (compare with Figure 3.4, next page) in G. Both structures provide a node

to exist in C which has at least two parents within C. With respect to $\text{pat}(G)$ we conclude these immoralities and cliques to be specified in $\text{pat}(G)$ and with this in the characteristic imset $\mathbf{c}(G)$.

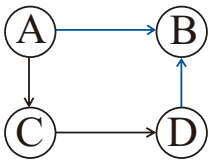

Figure 3.3: An acyclic directed cycle with one immorality.

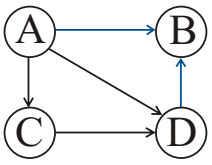

Figure 3.4: A acyclic directed cycle with two cliques.

Provided arcs are well defined in $\mathbf{y}$ and cliques of G are acyclic directed in $\mathbf{y}$ we can define an alternative version of the cycle-inequalities.

$$\begin{aligned} y_{i,j} + y_{j,k} + y_{k,i} &\leq 2,\ \forall i \neq j \neq k \in N, \\ y_{j,i} + y_{i,k} + y_{k,j} &\leq 2,\ \forall i \neq j \neq k \in N, \\ \sum_{\substack{A \subseteq T, \\ |A|=2}} x_A &\leq |T| - 1 + \sum_{\substack{B \subseteq N, \\ |B|=3}} x_B,\ \forall T \subseteq N,\ |T| \geq 4 \end{aligned} \tag{3.2.16}$$

The first two sets of Inequalities of (3.2.16) are defining no directed cycle of length three to exist. Roughly speaking, the third inequalities imply: If a set T induces a cycle in $\bar{G}$, as it contains at least $|T|$ edges, then an immorality or a clique of size three has to be contained in G^T, which is indicated with $x_B = 1$, $|B| = 3$. Together with $x_{\{i,j\}} \in [0,1]$ for all edges, $y_{i,j} + y_{j,i} = x_{\{i,j\}}$ and the inequalities above, all cycles are acyclic directed in a DAG G specified by $\mathbf{y}$.

More precisely, let T be a minimal cycle in $\text{pat}(G)$ containing $|T| \geq 4$ edges/arcs specified by $\mathbf{c}(G)$. For all such cycles T, a valid characteristic imset $\mathbf{c}(G)$ of a DAG G contains an immorality since T is a minimal cycle and there exists a set $B \subseteq T$ with $|B| = 3$ and $c_B(G) = 1$ and the inequality is satisfied. Let T be a cycle with additional diagonals contained within, i.e. T is no minimal cycle in $\text{pat}(G)$. In fact, as G is a DAG, there exists a triangulation of T into cliques of size three (compare with Figure 3.5 below) and eventually minimal cycles $S \subseteq T$ with at least four edges. Cycles S are already considered, therefore let T contain a triangulation into cliques of size three. Each of the possible triangulations contain $|T|+|T|-3$ edges/arcs with $|T|-2$ cliques of size three. This satisfies $|T|+|T|-3 \leq |T|-1+|T|-2$ coinciding with the fulfilment of Inequalities (3.2.16). If $\text{pat}(G)$ can be triangulated in more than one way, then only the additional diagonals (say d many) and the additional cliques ($d+1$ many) have to be added independently fulfilling the inequalities.

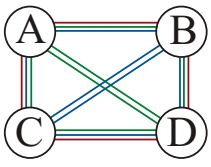

Figure 3.5: An undirected cycle of nodes $T = \{A, B, C, D\}$ with two contained triangulations (blue and green).

Contrarily, let $(\mathbf{y}, \mathbf{x})$ be given such that $\mathbf{x}$ is defined appropriately on the base of $\mathbf{y}(D)$ and let D, specified by $\mathbf{y}$, contain a directed cycle T of (minimal) length ≥ 4, which is equivalent to have no immorality or clique inside T. More precisely, with respect to the pattern graph G^T is undirected and not chordal, a contradiction. The set T then does not contain a set $B \subseteq T$, $|B| = 3$, with $x_B = 1$. The Inequalities (3.2.16) for set T corresponds to the cycle-inequalities not satisfied for cycles T and therefore also not satisfied for $\mathbf{x}$.

We call this set of inequalities the **DAG-inequalities**.

The advantage of the DAG- compared to the cycle-inequalities lies in their amount. The cycle inequalities are defined for every directed cycle, for which the order of the contained edges must be considered. For the DAG-inequalities this order is not important. In spite of this the cycle-inequalities, only using the $\mathbf{y}$ variables, are much stronger as they are directly influencing $\mathbf{y}$.

3.3 Valid inequality descriptions for the characteristic imset polytope

The previous section is devoted to the derivation of a LP-relaxation of an extension of the characteristic imset polytope $\mathrm{P}_{\mathrm{ext}}$. The use of an extension is particularly important as the definition of the inequalities becomes easier if they are only dependent on the definition of the incidence vectors of a DAG. In spite of this, the previous section does not provide a LP-relaxation of $\mathrm{P}_{\mathbf{c}}$ itself. Hence an alternative approach is to define valid inequalities directly for $\mathrm{P}_{\mathbf{c}}$ which provide a LP-relaxation $\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}$ of $\mathrm{P}_{\mathbf{c}}$. The derivation of inequalities only using characteristic imsets is more complex as various additional properties have to be ensured. Most prominent, the acyclicity of the described directed graph G must be ensured only on the base of its characteristic imset $\mathbf{c}(G)$, which, on the other hand, only encodes the characteristic vector of the underlying undirected graph of G. However, up to know we have not succeeded to derive a provable LP-relaxation, but we present a conjectured list of inequalities.

We derive an inequality description for $\mathrm{P}_{\mathbf{c}}$ based on the analysis of properties of characteristic imsets to encode patterns of DAGs. This description $\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}}$ consists of inequalities each representing a certain property of a valid characteristic imset. If the inequalities are added to the system successively, then each addition of a set of inequalities results in a polytope containing less integral points not equal to a characteristic imset. If all inequalities are added, then the polytope $\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}}$ is obtained.

We conjecture this iterative addition of inequalities to $\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}}$ and properties of characteristic imsets to $\mathbf{x} \in \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}}$ to yield a LP-relaxation $\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}$ of $\mathrm{P}_{\mathbf{c}}$ using no extended formulation. For the sake of simplicity, we give an outline of the properties needed to be encoded by the inequalities. This outline is based on a complete characterisation of a binary vector $\mathbf{x} \in \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}}$ to be a characteristic imset. To obtain this characterisation we use properties of characteristic imsets from Section 3.2, page 54.

3.3.1 Conjectured LP-relaxation of the characteristic imset polytope

We present a possible outline of the basic steps of a proof to show the following conjecture:

Conjecture 3.3.1 *Let* $\mathrm{P}_{\mathbf{c}}$ *be the characteristic imset polytope over node set* N*, then* $\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}}$ *is a LP-relaxation* $\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}$ *of* $\mathrm{P}_{\mathbf{c}}$ *and* $\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}} \cap \mathbb{Z}^t = \mathrm{P}_{\mathbf{c}}^{\mathrm{rel}} \cap \mathbb{Z}^t = \mathrm{P}_{\mathbf{c}} \cap \mathbb{Z}^t$*,* $t = 2^{|N|} - |N| - 1$*.*

3.3.1.1 Patterns, essential graphs and their characteristic imsets

A proof of Conjecture 3.3.1 which directly shows all inequalities to be necessary and sufficient for a LP-relaxation of $\mathrm{P}_{\mathbf{c}}$ may use the graphical properties of characteristic imsets.

Properties of patterns and essential graphs Moreover to the properties of a characteristic imset we need properties of their graphical translation, the pattern graph. As analysed in Section 1.2.1.2, page 10, a DAG G contains arcs which are common over all DAGs of the same Markov equivalence class of G. Some of these arcs are already directed in the pattern $\text{pat}(G)$ of G, namely those in an immorality. The remaining protected arcs of the Markov equivalence class can be derived by an exhaustive application of the orientation rules in Section 2.2.3, page 46, or in Figure 3.6 below.

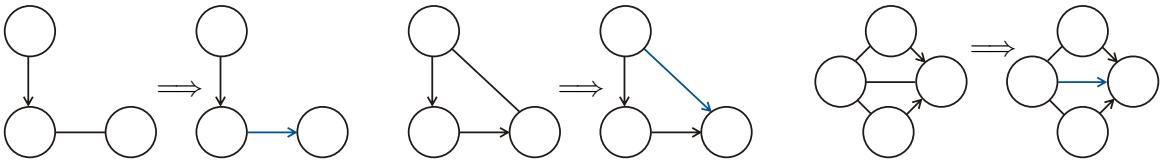

Figure 3.6: Orientation rules 1, 2 and 3.

The obtained graph is the essential graph G^* of (the Markov equivalence class of) G. The directed arcs in an essential graph G^* are all protected but also the undirected edges of G^* have a property useful to characterise G^*. The essential graph G^* of Markov equivalent DAGs G is a chain graph (see Section 2.1.1, page 37, for a definition) whose undirected components are chordal graphs. These properties coincide with Proposition 4.1 in [1]. A reconstruction of a DAG G from G^* comprises to direct the undirected edges of the chordal components of G^*. If and only if this direction is possible, then G^* is the essential graph of a DAG G. In [1] the representation of G^* in terms of a chain graph is derived by observing no flag to exist in G^*. In particular, if the orientation rules are applied to a general mixed graph, then the resulting graph does not contain a flag.

Direct characterisation of an essential graph Andersson et al. [1] use the presented properties of the essential graph G^* to define conditions on mixed graphs which are essential graphs.

Theorem 3.3.2 (Theorem 4.1 in [1]) *A graph G' is equal to the essential graph G^* of a DAG G if and only if G' satisfies the following four conditions:*

(i) G' is a chain graph,

(ii) Every undirected component of G' is a chordal graph,

(iii) G' does not contain a flag and

(iv) Every arc in G' is strongly protected.

We interpret the four conditions in the following way: Conditions (iii) and (iv) are valid for a mixed graph G' if the orientation rules are applied and, in particular, are valid. Therefore G' is obtainable by an iterative application of the orientation rules in every order without creating a contradiction. If Condition (i) is not satisfied, then G' contains a directed cycle. The mixed graph G' is not a chain graph if there are at least two components B_i and B_j with $i < j$ and nodes $v \in B_i$ and $w \in B_j$ such that $v \leftarrow w$ is an arc in G'. If this contradiction of the order of B_i and B_j exists, then there is a sequence of nodes and arcs $v \rightarrow v_{i+1} \rightarrow \ldots \rightarrow v_{j-1} \rightarrow w \rightarrow v$ of components $B_i, B_{i+1}, \ldots, B_{j-1}, B_j$ with $i < i+1 < \ldots < j-1 < j$ inducing a directed cycle in $\bar{G}$. If there is no connected sequence or if the cycle is not directed, then the order of B_i and B_j can be adapted to obtain a chain graph. Condition (ii) can be tested using the definition of chordal graphs via cycles of length ≥ 4 and their decomposability into cliques of size three.

By definition, chordal graphs are patterns (or essential graphs) without immoralities. The graph G' is chordal if there exists an immorality or clique for all cycles of length ≥ 4. This condition holds also for the undirected components of G' and Condition (ii) is satisfied.

3.3.1.2 Proving Conjecture 3.3.1

We observe a characteristic imset to satisfy Lemma 2.2.7, page 45, and with this to have a valid and unique *characteristic imset extension*.

We use the mentioned graphical translation in Theorem 2.2.5, page 43, in the following way: Let $\mathbf{x}|_{\leq 3}$ be a binary vector. We interpret $\mathbf{x}|_{\leq 3}$ with the graphical translation of characteristic imsets. This interpretation yields a mixed graph $G(\mathbf{x}|_{\leq 3})$. This mixed graph is a pattern $\text{pat}(G)$ of a DAG G if and only if $\mathbf{x}|_{\leq 3}$ is a valid representation of a Markov equivalence class of DAGs. We call this graphical interpretation the *pattern interpretation*.

However, this graphical translation between $\mathbf{x}|_{\leq 3}$ and $G(\mathbf{x}|_{\leq 3})$, a possible pattern graph, must be bijective. The vector $\mathbf{x}|_{=2}$ is interpreted as the characteristic vector of an undirected graph (compare with Corollary 2.2.4, page 43). With this graphical translation the underlying undirected graph of $G(\mathbf{x}|_{\leq 3})$ is in a bijective relation to $G(\mathbf{x}|_{=2})$. The entries $\mathbf{x}|_{=3}$ determine the immoralities and cliques of size three. Theorem 2.2.5 provides one direction, namely if $\mathbf{x}|_{=3}$ induces an immorality, then $G(\mathbf{x}|_{\leq 3})$ contains the corresponding directed arcs. Cliques of size three are provided by the existence of all contained edges. On the other hand, if $G(\mathbf{x}|_{\leq 3})$ contains an immorality or a clique of size three, which is not specified in $\mathbf{x}|_{=3}$, then $\mathbf{x}|_{\leq 3} \neq \mathbf{c}(G)|_{\leq 3}$, $\forall G \in$ DAGs. Directed arcs in $G(\mathbf{x}|_{\leq 3})$ only occur within immoralities. This implies that $\mathbf{x}|_{\leq 3}$ may contain fewer immoralities than in $G(\mathbf{x}|_{\leq 3})$, which happens if the corresponding arcs of these immoralities are also contained in other immoralities in $G(\mathbf{x}|_{\leq 3})$ and iteratively also in $\mathbf{x}|_{\leq 3}$. We identify this case of fewer immoralities and cliques with the *inconsistency of immoralities* encoded in $\mathbf{x}|_{\leq 3}$. Moreover, inconsistent immoralities can also occur if contained arcs are directed in $G(\mathbf{x}|_{\leq 3})$ in both ways, which is a contradiction.

Of course, besides the inconsistent choice of immoralities and a possible contradiction of the orientation rules, $G(\mathbf{x}|_{\leq 3})$ is not a pattern of a DAG if the directed arcs in $G(\mathbf{x}|_{\leq 3})$ already imply directed cycles. The orientation rules do only direct undirected edges but not arcs, hence arcs in $G(\mathbf{x}|_{\leq 3})$ are fixed. We call $G(\mathbf{x}|_{\leq 3})$ in this case to contain *directed cycles*.

The mixed graph $G(\mathbf{x}|_{\leq 3})$ is not a valid pattern if and only if there exists no DAG G with $\bar{G} = G(\mathbf{x}|_{=2})$ having the same immoralities. In terms of the essential graph, a characteristic imset and with respect to the last paragraph, this equivalence becomes: $\mathbf{x} \neq \mathbf{c}(G)$ if and only if $G(\mathbf{x}|_{\leq 3})$ cannot be directed to the valid essential graph G^* of G. Due to Theorem 3.3.2 there exists a direct characterisation of the latter condition, i.e. a condition on G^* to be an essential graph. Furthermore, the iterative application of orientation rules in Figure 3.6 can be reformulated to conditions on $G(\mathbf{x}|_{\leq 3})$ in terms of: If these orientation rules are satisfied and yield no contradiction in every possible combination, then an iterative application is possible. We call this the *reconstruction of a DAG* from a binary vector $\mathbf{x}$.

A graphical based proof of Conjecture 3.3.1 should use the five presented properties as basic argumentative steps. The key idea is to specify the properties needed to interpret a binary vector $\mathbf{x}$ as a characteristic imset and with this to reconstruct a corresponding DAG. If $\mathbf{x}$ does not satisfy one of the properties, then $\mathbf{x} \neq \mathbf{c}(G)$, $\forall G \in$ DAGs. Let a binary $\mathbf{x}$ be given, we conclude:

1. $\mathbf{x} \neq \mathbf{c}(G)$, $\forall G \in$ DAGs, if it does not satisfy the *characteristic imset extension*.
2. $\mathbf{x} \neq \mathbf{c}(G)$, $\forall G \in$ DAGs, if does not have a *pattern interpretation* $G(\mathbf{x}|_{\leq 3})$.

3. $\mathbf{x} \neq \mathbf{c}(G)$, $\forall G \in$ DAGs, if $G(\mathbf{x}|_{\leq 3})$ has *inconsistent immoralities*.

4. $\mathbf{x} \neq \mathbf{c}(G)$, $\forall G \in$ DAGs, if $G(\mathbf{x}|_{\leq 3})$ contains *directed cycles*.

5. $\mathbf{x} \neq \mathbf{c}(G)$, $\forall G \in$ DAGs, if $G(\mathbf{x}|_{\leq 3})$ cannot be *reconstructed to a DAG*.

Argument 1 is equivalent to $\mathbf{x}$ to satisfy Lemma 2.2.7, page 45. Argument 2 is equivalent to $\mathbf{x}$ to satisfy Theorem 2.2.5, page 43. Argument 3 is equivalent to directed arcs in $G(\mathbf{x}|_{\leq 3})$ to induce immoralities in $\mathbf{x}|_{=3}$. The most crucial parts in the proof concern Arguments 4 and 5. Argument 4 is equivalent to $G(\mathbf{x}|_{\leq 3})$ to contain directed cycles of arcs included in immoralities. Argument 5 is equivalent to Theorem 3.3.2. Besides, Arguments 4 and 5 are related in the sense that also during the reconstruction of the DAG, the mixed graph must not have directed cycles.

The argumentation presented coincides with a complete case study of conditions on $\mathbf{x} \neq \mathbf{c}(G)$, $\forall G \in$ DAGs.

3.3.2 Derivation of inequalities

In this section we derive the inequalities with respect to the properties of $\mathbf{c} \in \mathrm{P}_{\mathbf{c}}$. Each subsequent set of inequalities represents equivalently a property of $\mathbf{c}$. We conjecture: If and only if the inequalities are satisfied, then the 0/1-vector $\mathbf{x}$ is contained in $\mathrm{P}_{\mathbf{c}}$ and $\mathbf{x} = \mathbf{c}(G)$ for a DAG G. For each of the inequalities separately we need to show them to coincide with Arguments 1 to 5 above.

1 - Characteristic imset extension We use Inequalities (3.2.7), page 58, from Section 3.2, page 54, and obtain with $t = 2^{|N|} - |N| - 1$:

$$\begin{aligned}\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(1)}} := \{x \in [0,1]^t : & \\ \sum_{i \in T} x_{T \setminus \{i\}} \quad &\leq \quad 2 + (|T| - 2)x_T, \qquad \forall T \subseteq N,\ |T| \geq 4 \\ (|T| - 1)x_T \quad &\leq \quad \sum_{i \in T} x_{T \setminus \{i\}}, \qquad \forall T \subseteq N,\ |T| \geq 4\ \}.\end{aligned}$$

Lemma 3.3.3 *For the characteristic imset polytope, there is* $\mathrm{P}_{\mathbf{c}} \subseteq \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(1)}}$.

Corollary 3.3.4 *Any binary* $\mathbf{x} \in \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(1)}}$ *has the property of a valid characteristic imset extension.*

Both Corollary 3.3.4 and Lemma 3.3.3 above follow from Section 3.2.3, page 56.

2 - Pattern interpretation The graphical interpretation $G(\mathbf{x}|_{\leq 3})$ of a binary vector $\mathbf{x}$ is given by Theorem 2.2.5. Therein, $G(\mathbf{x}|_{\leq 3})$ contains an edge $a - b$ if and only if $x_{\{a,b\}} = 1$. This provides a bijection between $G(\mathbf{x}|_{=2})$ and $\mathbf{x}|_{=2}$. On the other hand, $a \to b \leftarrow c$ induces an immorality in $G(\mathbf{x}|_{\leq 3})$ if and only if there exists a node $c \neq a, b$ with $x_{\{a,b,c\}} = 1$ and $x_{\{a,b\}} = 1$, $x_{\{b,c\}} = 1$ but $x_{\{a,c\}} = 0$. Moreover to this, the set $\{a, b, c\}$ is a clique in $G(\mathbf{x}|_{\leq 3})$ if and only if $x_{\{a,b\}} = 1$, $x_{\{b,c\}} = 1$ and $x_{\{a,c\}} = 1$. Inequalities encoding cliques and immoralities are similar to the previous Inequalities (3.2.7), page 58. In fact, setting $|T| \geq 3$ in Inequalities

(3.2.7) is enough to provide the *pattern interpretation*. We use the cim-Inequalities (3.2.8), page 59, and obtain with $t = 2^{|N|} - |N| - 1$:

$$\begin{aligned}\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(2)}} := \{x \in [0,1]^t :\\ \sum_{i \in T} x_{T\setminus\{i\}} &\leq 2 + (|T|-2)x_T, &\forall T \subseteq N,\ |T| \geq 3\\ (|T|-1)x_T &\leq \sum_{i \in T} x_{T\setminus\{i\}}, &\forall T \subseteq N,\ |T| \geq 3\ \}.\end{aligned}$$

Lemma 3.3.5 *For the characteristic imset polytope, there is* $\mathrm{P}_{\mathbf{c}} \subseteq \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(2)}}$.

Lemma 3.3.5 follows from Section 3.2 as well.

Corollary 3.3.6 *Any binary* $\mathbf{x} \in \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(2)}}$ *has the property of a pattern interpretation.*

Proof. Let a binary vector $\tilde{\mathbf{x}} \in \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(2)}}$ be given. Let its graphical interpretation $G(\tilde{\mathbf{x}}|_{\leq 3})$ contain a clique T but $\tilde{x}_T = 0$. As T is a clique all edges are present and $\tilde{x}_{T\setminus\{i\}} = 1$ for all edges $T \setminus \{i\}$ of T. Then the first of the cim-inequalities is not fulfilled with $3 \leq 2$. On the other hand, let the graphical interpretation $G(\tilde{\mathbf{x}}|_{\leq 3})$ not contain an immorality T but $\tilde{x}_T = 1$ (the case of $\tilde{x}_T = 0$ but $G(\tilde{\mathbf{x}}|_{\leq 3})$ contains the immorality T is considered as *inconsistent immoralities*; see below). We observe w.l.o.g. that $\tilde{x}_{T\setminus\{i\}} = 0$ for at least two subsets $T \setminus \{i\} \subseteq T$ and $2 \leq 1$ in the second of the cim-inequalities. Either way, the binary vector $\tilde{\mathbf{x}}$ is not contained in $\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(2)}}$, a contradiction. □

3 - Inconsistency of immoralities First of all, as already shown, the first set of the cim-inequalities provides the bijection between cliques in $\mathbf{x}|_{=3}$ and in $G(\mathbf{x}|_{\leq 3})$. Likewise the cim-inequalities provide the interpretation of immoralities. The remaining direction comprises to find immoralities specified in $G(\mathbf{x}|_{\leq 3})$ but not in $\mathbf{x}|_{=3}$ or to find immoralities in $G(\mathbf{x}|_{\leq 3})$ contradicting each other. As already analysed these additional or wrong immoralities of $G(\mathbf{x}|_{\leq 3})$ are induced by arcs of $G(\mathbf{x}|_{\leq 3})$ contained in other immoralities which are, with an iterative argumentation, contained in $\mathbf{x}|_{=3}$. We interpret this implication with an extension of Lemma 2.2.7.

Let T, $|T| \geq 4$, be a set and $i \in T$ with an induced subgraph $G^{T\setminus\{i\}}$ which is not complete, then we have

$$\begin{aligned}(a)' :\ c_T(G) = 1 \iff (d) :\ &\text{there exist two subsets } T_j \subseteq T,\ j = 1,2, \text{ with}\\ &|T_j| = |T| - 1,\ i \in T_j,\ G^{T_j\setminus\{i\}} \text{ not complete and } \mathbf{c}_{T_j}(G) = 1.\end{aligned} \tag{3.3.1}$$

If sets T_j have cardinality three, then $\mathbf{c}_{T_j}(G) = 1$ implies them to be immoralities with terminal node i. Together with $c_T(G) = 1$ and $G^{T\setminus\{i\}}$, the set T has the same property, i.e. $T \setminus \{i\} \subseteq \mathrm{pa}(i)$. This property is particularly visible if two sets $T_j \subseteq T$, $j = 1,2$, exist with $\mathbf{c}_{T_j}(G) = 1$, $G^{T_j\setminus\{i\}}$ empty, $T_1 \cup T_2 = T$ and $i \in T_1, T_2$ (e.g. compare with Figure 3.7, page 69). In other words, the union of two immoralities has special properties like each of the both.

We call sets T with $|T| \geq 4$, their terminal node $i \in T$ and $G^{T\setminus\{i\}}$ not complete an **extended immorality**.

Corollary 3.3.7 *Let G be a DAG and $\mathbf{c}(G)$ be the corresponding characteristic imset. If there are two immoralities (or extended immoralities) T_1, T_2 with the same terminal node in G, then $c_{T_1}(G), c_{T_2}(G) = 1$ and $c_{T_1 \cup T_2}(G) = 1$.*

This coincides with Direction "$(d) \Longrightarrow (a)'$" in Equivalence (3.3.1), and specifies the property of *(in)consistent immoralities*. Note that Implication "$(a)' \Longrightarrow (d)$" follows from "$(a) \Longrightarrow (b)$" and is already incorporated in the cim-conditions. Orientation rule 3 is a special extended immorality. In the respective induced subgraph one additional arc is protected and one immorality is contained both with the same terminal node.

If T_1 and T_2 are extended immoralities with respect to Orientation rule 3, then undirected edges of $G(\mathbf{x}|_{\leq 3})$ are contained. This we consider later again with respect to the property of a *reconstruction of a DAG* when undirected edges are directed according to the orientation rules.

We use the *characteristic imset extension* and conclude the occurrence of minimal cases to combine (extended) immoralities corresponding to sets T with $|T| = 4, 5, 6, 7$. All remaining higher dimensional cases can be derived inductively with the *characteristic imset extension*. In the proceeding analysis we often distinguish between these immoralities to be edge-disjoint or not. Figure 3.7 below shows the minimal example of such an addition of two non-edge-disjoint immoralities of size three (coloured blue and green) to one of size four. In Figure 3.8 below we can see the corresponding minimal example of an extended immorality of size five, which is implied by two edge-disjoint immoralities of size three (coloured blue and green).

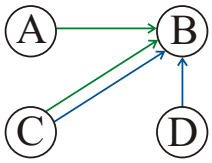

Figure 3.7: Two non-edge-disjoint immoralities imply an extended immorality with four nodes.

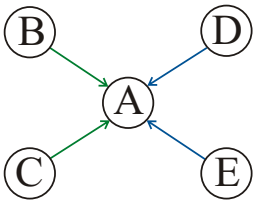

Figure 3.8: Two edge-disjoint immoralities imply an extended immorality with five nodes.

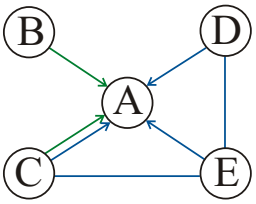

Figure 3.9: An immorality and a non-edge-disjoint extended immorality imply an extended immorality with five nodes.

Figures 3.9, 3.10, 3.11, 3.12, this page, show the minimal combinations of immoralities and extended immoralities which are implied by Orientation rule 3.

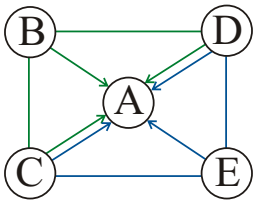

Figure 3.10: Two non-edge-disjoint extended immoralities imply an extended immorality with five nodes.

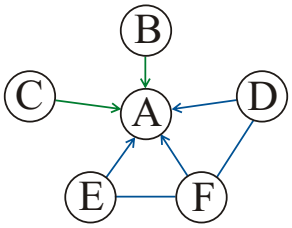

Figure 3.11: An immorality and an edge-disjoint extended immorality imply an extended immorality with six nodes.

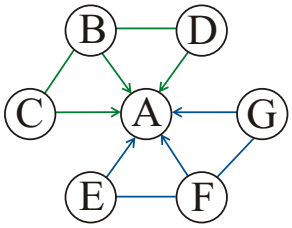

Figure 3.12: Two edge-disjoint extended immoralities imply an extended immorality with seven nodes.

Any inequality forcing this "addition of immoralities" has to incorporate terminal nodes. If they differ, then Equivalence (3.3.1) cannot be applied. Moreover a contradiction is yield if both terminal nodes are contained in the respective other (extended) immorality, because

a cycle of length two is created (compare with e.g. Figures 3.13 below and 3.14 below). In the other case, setting the entry of the superset to one is then sufficient to imply all other contained (extended) immoralities to be existing, as well. Furthermore it is sufficient to consider sets of cardinality at most 7, because two edge-disjoint extended immoralities of Orientation rule 3 are sufficient for covering (compare with Figure 3.12).

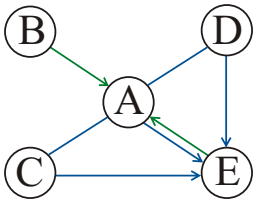

Figure 3.13: An immorality and a non-edge-disjoint extended immorality imply a directed cycle with nodes A, E.

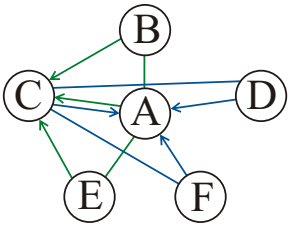

Figure 3.14: Two non-edge-disjoint extended immoralities imply a directed cycle with nodes A, C.

For the following Inequalities (3.3.2) we distinguish between the size of T and of subsets T_1, T_2. If $|T| = 4$, then $|T_1| = |T_2| = 3$. If $|T| = 5$, then $\{|T_1|, |T_2|\} \in \{\{|3,3\}, \{3,4\}, \{4,4\}\}$. If $|T| = 6$, then $\{|T_1|, |T_2|\} \in \{\{3,4\}, \{4,4\}\}$ and if $|T| = 7$, then $|T_1| = |T_2| = 4$. Let $t := 2^{|N|} - |N| - 1$ and

$$
\begin{aligned}
\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(3)}} := \{x \in [0,1]^t : & \\
\sum_{i \in T} x_{T \setminus \{i\}} \quad &\leq \quad 2 + (|T| - 2) x_T, && \forall T \subseteq N,\ |T| \geq 3 \\
(|T| - 1) x_T \quad &\leq \quad \sum_{i \in T} x_{T \setminus \{i\}}, && \forall T \subseteq N,\ |T| \geq 3 \\
& && \qquad (3.3.2) \\
x_{T_1} + x_{T_2} \quad &\leq \quad 1 + x_T && \forall T \subseteq N,\ |T| = 4,5,6,7, \\
& \quad + x_{T_1 \setminus \{k_1\}} + x_{T_2 \setminus \{k_2\}}, && \forall T_1, T_2 \subseteq T,\ T_1 \cup T_2 = T, \\
& && \forall k_1, k_2 \in T_1 \cap T_2 \ \}.
\end{aligned}
$$

Lemma 3.3.8 *For the characteristic imset polytope, there is* $\mathrm{P}_{\mathbf{c}} \subseteq \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(3)}}$.

Proof. Let G be a DAG and $\mathbf{c}(G)$ be the corresponding characteristic imset. First of all, any characteristic imset with entries $c_{T_1}(G) = 0$ or $c_{T_2}(G) = 0$ clearly is contained in $\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(3)}}$. Second, consider only the case $c_{T_1}(G) = 1$ and $c_{T_2}(G) = 1$. Due to the *characteristic imset extension* this means nodes in both sets $k_1 \in T_1$ and $k_2 \in T_2$ with $T_1 \setminus \{k_1\} \subseteq \mathrm{pa}_G(k_1)$ and $T_2 \setminus \{k_2\} \subseteq \mathrm{pa}_G(k_2)$ to be existent, respectively. If T_1 or T_2 is a clique, then $c_{T_1 \setminus \{j\}}(G) = 1$, $\forall j \in T_1$, and $c_{T_2 \setminus \{j\}}(G) = 1$, $\forall j \in T_2$, also implying the inequality to be valid. Analogously, if $k_1 \in T_1 \setminus T_2$ or $k_2 \in T_2 \setminus T_1$, then for all $k \in T_1 \cap T_2$ there exist arcs $k \to k_1$, $k \to k_2$, arcs $j \to k_1$ for $j \in T_1 \setminus \{k, k_1\}$ and also arcs $j \to k_2$ for $j \in T_2 \setminus \{k, k_2\}$. This likewise implies the inequalities to be valid due to $c_{\{j,k_1\}}(G) = 1$ and $c_{\{j,k_2\}}(G) = 1$. Contrarily, let $k_1 \in T_1 \cap T_2$ and $k_2 \in T_2 \setminus T_2$, then $c_{T_2 \setminus \{k\}}(G) = 1$, $\forall k \in T_1 \cap T_2$. If both $k_1 \neq k_2 \in T_1 \cap T_2$, then T_1 and T_2 induce a cycle with nodes k_1, k_2 which is not possible for $\mathbf{c}(G)$. But, if $k_1 = k_2$ both terminal nodes are equal and $c_{T_1 \cup T_2}(G) = 1$.

We can conclude, every characteristic imset of a DAG G to satisfy Inequalities (3.3.2) and hence to be contained in $\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(3)}}$. □

Conjecture 3.3.9 *Any binary* $\mathbf{x} \in \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(3)}}$ *has no inconsistent immoralities.*

To show this Corollary we have to show no inconsistency to be contained in $\mathbf{x} \in \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(3)}}$ and the above case study to be complete. This is not yet sufficiently be proven.

4 - Directed cycles A graphical translation $G(\mathbf{x}|_{\leq 3})$ of a binary vector $\mathbf{x}$ is clearly not $\mathrm{pat}(G)$ of a DAG G, if directed cycles are contained. We use the corresponding property: The induced graph of $\mathrm{pat}(G)$ containing all arcs is a DAG G' itself. On the other hand, a directed graph G' is a DAG if and only if for all cycles C in G' there is a node i in C having at least two parents within that cycle. From this equivalence we derive inequalities equivalently encoding this condition. We distinguish between a DAG G' that has

1. no directed cycle C of length ≥ 4.

   For this consider the DAG-inequalities (3.2.16), page 63.

2. no directed cycle C of length three.

   Theoretically, patterns of cycles of length three are cliques of size three and the edges are undirected. However, there are cases when the existence of special immoralities induce directed arcs within C. A directed cycle C of length three occurs in $G(\mathbf{x}|_{\leq 3})$ only if all directed arcs in C are fixed. This happens if the arcs of C are contained in immoralities (compare with Figure 3.15 below). We can observe two of the therein immoralities to be enough to enforce the direction of C.

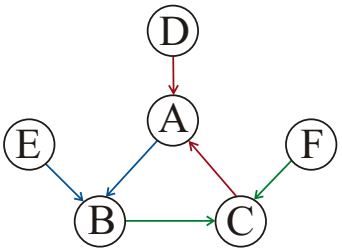

Figure 3.15: A directed cycle of length three implied by immoralities.

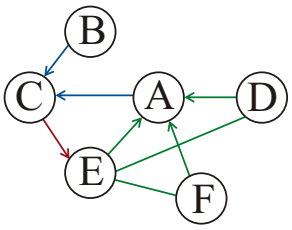

Figure 3.16: A directed cycle of length three implied by an immorality and an extended immorality.

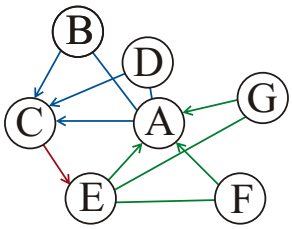

Figure 3.17: A directed cycle of length three implied by extended immoralities.

Similar cases can occur with a combination of immoralities and extended immoralities implied by Orientation rule 3 (Figures 3.16 and 3.17 above). However, as this observation results in the application of orientation rules we comment on it again in the next section. The only way to enforce one of the arcs to be conversely directed is to enforce the existence of additional immoralities or cliques and the particular arc to be contained in it (coloured red in the next Figure 3.18).

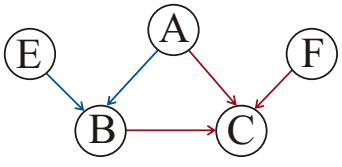

Figure 3.18: Additional immoralities to change the direction of a directed cycle of length three.

The corresponding inequalities enforce these additional immoralities to exist only in case if directed cycles of length three are implied by (extended) immoralities. This happens only if at least two (extended) immoralities T_1 and T_2 (with terminal nodes k_1 and k_2) and one cycle T_3 of length three are existing according to Figures 3.15, 3.16 and 3.17.

3. no directed cycle of length two.

   Directed cycles of length two containing arcs of $G(\mathbf{x}|_{\leq 3})$ are implied by *inconsistent immoralities* and with this are already analysed.

From the analysis of cycles of length three inequalities can be derived. Similar to Inequalities (3.3.2) we consider the cardinalities of T and of T_1, T_2 in Inequalities (3.3.3) separately. If $|T| = 5$, then $|T_1| = |T_2| = 3$. If $|T| = 6$, then $|T_1| = 3$ and $|T_2| = 4$ and if $|T| = 7$, then $|T_1| = |T_2| = 4$. Let $t := 2^{|N|} - |N| - 1$ and

$$
\begin{aligned}
\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(4)}} := \{x \in [0,1]^t :\ & \\
\sum_{i \in T} x_{T\setminus\{i\}} &\leq 2 + (|T|-2)x_T, && \forall T \subseteq N,\ |T| \geq 3 \\
(|T|-1)x_T &\leq \sum_{i \in T} x_{T\setminus\{i\}}, && \forall T \subseteq N,\ |T| \geq 3 \\
x_{T_1} + x_{T_2} &\leq 1 + x_T && \forall T \subseteq N,\ |T| = 4,5,6,7,\ \forall T_1, T_2 \subseteq T, \\
&\quad + x_{T_1\setminus\{k_1\}} + x_{T_2\setminus\{k_2\}}, && T_1 \cup T_2 = T,\ \forall k_1, k_2 \in T_1 \cap T_2 \\
\sum_{\substack{A \subseteq T \\ |A|=2}} x_A &\leq \sum_{\substack{B \subseteq T \\ |B|=3}} x_B + |T| - 1, && \forall T \subseteq N,\ |T| \geq 4 \\
x_{T_1} + x_{T_2} + x_{T_3} &\leq 2 + x_{T_1\setminus\{k_1\}} && \forall T \subseteq N,\ |T| = 5,6,7,\ \forall T_1, T_2 \subseteq T,\ |T_1 \cap T_2| = 1, \\
&\quad + x_{T_2\setminus\{k_2\}} && \forall k_1 \in T_1,\ \forall k_2 \in T_2,\ k_1 \neq k_2,\ k_1 \in T_1 \cap T_2 \text{ or} \\
&\quad + x_{(T_3\setminus T_2 \cup T_2\setminus T_1)}, && k_2 \in T_1 \cap T_2,\ \forall T_3 \subseteq T,\ T_3 \neq T_1, T_2,\ |T_3| = 3, \\
& && |T_1 \cap T_2 \cap T_3| = 1,\ k_1, k_2 \in T_3\ \}.
\end{aligned}
\tag{3.3.3}
$$

Lemma 3.3.10 *For the characteristic imset polytope, there is* $\mathrm{P}_{\mathbf{c}} \subseteq \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(4)}}$.

Proof. Let $\mathbf{c}(G)$ be a characteristic imset of a DAG G. First of all the DAG-inequalities are valid due to Section 3.2.5, page 60.

Second, the Inequalities (3.3.3) are not trivially satisfied only in case $c_{T_1}(G) = 1$, $c_{T_2}(G) = 1$, $c_{T_3}(G) = 1$ on the left hand side of the inequalities and $c_{T_1\setminus\{k_1\}}(G) = 0$ and $c_{T_2\setminus\{k_2\}}(G) = 0$ on the right hand side. For a set T with five nodes and for well chosen sets T_1 and T_2 with terminal nodes k_1 and k_2, we have an induced subgraph with immoralities T_1 of nodes say a, b, k_1 and T_2 with nodes k_1, c, k_2. Note, since $c_{T_1\setminus\{k_1\}}(G) = 0$ and $c_{T_2\setminus\{k_2\}}(G) = 0$, both nodes k_1 and k_2 must be terminal. There are two possibilities of T_3, first it consists of nodes k_1, k_2 and a, or it consists of nodes k_1, k_2 and b. By construction and with $c_{T_3}(G) = 1$, T_3 is a clique in both cases. Together with the respective notation, Figure 3.19, next page, shows both possible induced subgraphs for $|T| = 5$.

As $\mathbf{c}(G)$ is a valid characteristic imset, both induced subgraphs are DAGs. This implies the first subgraph to contain arc $a \to k_2$ and the additional immorality or clique $\{a, k_2, c\}$. The second subgraph contains arc $b \to k_2$ and the additional immorality or clique $\{b, k_2, c\}$. For each of the two cases we have $c_{T_3\setminus T_2 \cup T_2\setminus T_1}(G) = 1$, because $T_3 \setminus T_2 \cup T_2 \setminus T_1 = \{a\} \cup \{k_2, c\}$, for the first choice of T_3, and $\{b\} \cup \{k_2, c\}$, for the second choice.

The same fact can be derived for cardinalities $|T| = 5, 6$ with an appropriate adjustment of sets $T_3 \setminus T_2 \cup T_2 \setminus T_1$, which are of cardinality four in this case. □

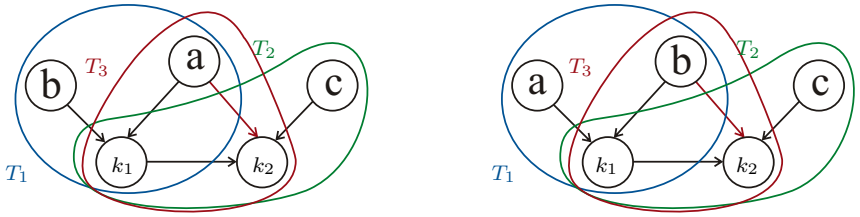

Figure 3.19: Two possible induced subgraphs satisfying Inequalities (3.3.3).

Conjecture 3.3.11 *Any binary* $\mathbf{x} \in \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(4)}}$ *contains no directed cycle.*

The Corollary above is true for directed arcs in $G(\mathbf{x}|_{\leq 3})$. But with respect to a later application we also have to show this in case of contained undirected edges. This is not yet sufficiently be proven.

5 - Reconstruction of a DAG A reconstruction of a directed graph D from a mixed graph $G(\mathbf{x}|_{\leq 3})$ of a binary vector $\mathbf{x}$ may not be a DAG for several reasons. First, the application of the orientation rules is not possible and so the derivation of the potential essential graph $G^*(\mathbf{x}|_{\leq 3})$. In particular, the orientation rules do not contradict each other. This can happen if additional immoralities are created which are not specified in $G(\mathbf{x}|_{\leq 3})$ or, second, the resulting directed graph D can contain directed cycles. If both, no additional immoralities are created and if the graph is acyclic, then Theorem 3.3.2 provides conditions to test whether $G(\mathbf{x}|_{\leq 3})$ can be directed acyclic.

We proceed with an analysis of the different possibilities to iteratively apply the orientation rules (short OR1, OR2, OR3) and the corresponding contradictions.

- **OR1 vs. OR1:** Two undirected edges $\{a, b\}$ and $\{b, c\}$ are directed $a \to b$ and $b \leftarrow c$ in $G(\mathbf{x}|_{\leq 3})$ according to Orientation rule 1, a contradiction. Iteratively, we can conclude the existence of other nodes d, e with directed arcs $d \to a$ and $c \leftarrow e$ implying the initial direction. An iterative reconstruction of the arcs leads to two immoralities T_1 and T_2 (compare with Figure 3.20, page 74,) which are connected in $G(\mathbf{x}|_{\leq 3})$ with an undirected path P (with a, b, c contained in P). For the sake of simplicity let P be minimal with respect to T_1 and T_2. Concluding, we can observe additional immoralities to be implied according to Orientation rule 1 if there exists an undirected path P in $G(\mathbf{x}|_{\leq 3})$ between two terminal nodes $k_1 \neq k_2$ of two immoralities T_1 and T_2 (compare with Figure 3.21, page 74, and Figure 1 in [67]). In this case, both T_1 and T_2 imply the incident edges, say $\{k_i, l_i\}$, $i = 1, 2$, within P to be directed $k_1 \to l_1$ and $l_2 \leftarrow k_2$. This leads into node b on P having two entering arcs.

  The only way to have a valid pattern in this case is to imply the additional immorality induced by a, b, c on P or to change the initial immoralities T_1, T_2 such that one of the incident edges $\{k_i, l_i\}$, $i = 1, 2$, is not allowed to be directed $k_i \leftarrow l_i$, $i = 1, 2$. These edges are allowed to be directed if there are immoralities and cliques containing l_i and k_i for $i = 1$ or $i = 2$ (compare with Figures 3.23 and 3.24, page 74), or if one of the immoralities T_1 and T_2 is a clique (compare with Figure 3.22, page 74).

  The existence of a path P is crucial for the creation of the additional immoralities. In order to be only tight in case P exists, the corresponding inequalities have to incorporate this. The most direct way to do so is to use the DAG-inequalities. For technical reasons, we assume P in the inequalities to have no other nodes in common with T_1 and T_2 than their terminal nodes.

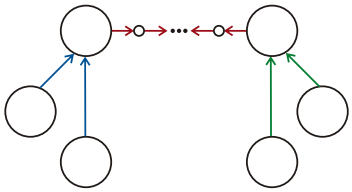

Figure 3.20: An additional immorality on a path between two immoralities.

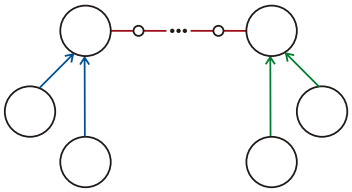

Figure 3.21: An undirected path between two terminal nodes of immoralities.

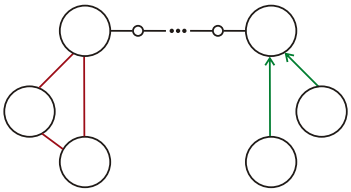

Figure 3.22: An undirected path between a clique and an immorality.

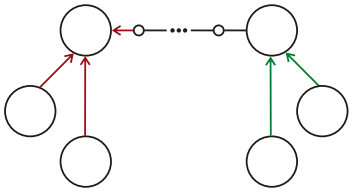

Figure 3.23: Additional immoralities with the same terminal node on a path between two immoralities.

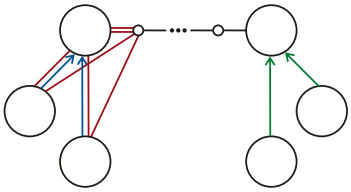

Figure 3.24: Additional cliques with the same terminal node on a path between two immoralities, Orientation rule 3.

- **OR1 vs. OR2:** Two cases can occur. Either a directed cycle T is implied by two immoralities sharing a node (see Inequalities (3.3.3)) or a directed cycle is implied by two disjoint immoralities (compare with Figure 3.25 below).

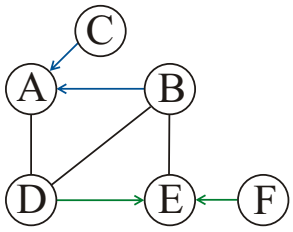

Figure 3.25: Directed cycles implied by two disjoint immoralities.

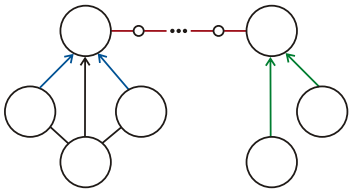

Figure 3.26: An undirected path between two terminal nodes of (extended) immoralities.

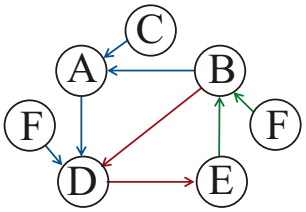

Figure 3.27: Directed cycles implied by three immoralities.

- **OR1 vs. OR3:** Independent from the additional protected edge in Orientation rule 3 an immorality T_1 is contained in the induced subgraph (coloured blue in Figure 3.26 above). Orientation rule 1 and 3 lead to a contradiction if there exists another immorality T_2 and an undirected path between the terminal node of T_1 and T_2, which we consider in case: "OR1 vs. OR1" (compare with Figure 3.26).
- **OR2 vs. OR2:** A contradiction is yield if the protected edges of two cliques of size three imply a directed cycle. But the initial direction is implied by immoralities, leading to case: "OR1 vs. OR1" or "OR1 vs. OR2" (compare with Figure 3.27 above).
- **OR2 vs. OR3:** The protected edges in a clique of size three are implied by the protected edge of Orientation rule 3. This we consider in Inequalities (3.3.3).
- **OR3 vs. OR3:** The protected edges of two extended immoralities with respect to Orientation rule 3 contradict each other. This is a special case already considered in Inequalities (3.3.2).

The case study above results in two additional sets of inequalities. We use the abbreviation $T(k_1, k_2) := T \setminus (T_1 \cup T_2) \cup \{k_1, k_2\}$, $T(k_1) := T \setminus (T_1 \cup T_2) \cup \{k_1\}$ and $T(k_2) := T \setminus (T_1 \cup T_2) \cup \{k_2\}$. Let $t := 2^{|N|} - |N| - 1$ and

$$
\begin{aligned}
\overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(5)}} := \{x \in [0,1]^t :\ & \\
\sum_{i \in T} x_{T\setminus\{i\}} &\leq 2 + (|T|-2)x_T, && \forall T \subseteq N,\ |T| \geq 3 \\
(|T|-1)x_T &\leq \sum_{i \in T} x_{T\setminus\{i\}}, && \forall T \subseteq N,\ |T| \geq 3 \\
x_{T_1} + x_{T_2} &\leq 1 + x_T && \forall T \subseteq N,\ |T| = 4,5,6,7,\ \forall T_1, T_2 \subseteq T, \\
&\quad + x_{T_1\setminus\{k_1\}} + x_{T_2\setminus\{k_2\}}, && T_1 \cup T_2 = T,\ \forall k_1, k_2 \in T_1 \cap T_2 \\
\sum_{\substack{A \subseteq T \\ |A|=2}} x_A &\leq \sum_{\substack{B \subseteq T \\ |B|=3}} x_B + |T| - 1, && \forall T \subseteq N,\ |T| \geq 4 \\
x_{T_1} + x_{T_2} + x_{T_3} &\leq 2 + x_{T_1\setminus\{k_1\}} && \forall T \subseteq N,\ |T| = 5,6,7\ \forall T_1, T_2 \subseteq T,\ |T_1 \cap T_2| = 1, \\
&\quad + x_{T_2\setminus\{k_2\}} && \forall k_1 \in T_1,\ \forall k_2 \in T_2,\ k_1 \neq k_2,\ k_1 \in T_1 \cap T_2 \text{ or} \\
&\quad + x_{(T_3\setminus T_2 \cup T_2\setminus T_1)}, && k_2 \in T_1 \cap T_2,\ \forall T_3 \subseteq T,\ T_3 \neq T_1, T_2,\ |T_3| = 3, \\
& && |T_1 \cap T_2 \cap T_3| = 1,\ k_1, k_2 \in T_3
\end{aligned}
\tag{3.3.4}
$$

$$
\begin{aligned}
x_{T_1} + x_{T_2} &\leq |T(k_1,k_2)| && \forall T \subseteq N,\ |T| \geq 5,\ \forall T_1, T_2 \subseteq T,\ |T_1| = |T_2| = 3, \\
+ \sum_{\substack{A \subseteq T(k_1,k_2) \\ |A|=2}} x_A &\quad + \sum_{\substack{B \subseteq T(k_1,k_2) \\ |B|=3}} x_B && |T_1 \cap T_2| < 2,\ \forall k_1 \in T_1,\ \forall k_2 \in T_2,\ k_1 \neq k_2, \\
& \quad + x_{T_1\setminus\{k_1\}} + x_{T_2\setminus\{k_2\}} && \forall i \in T_1 \setminus \{k_1\},\ \forall j \in T_2 \setminus \{k_2\} \\
& \quad + \sum_{l \in T(k_2)} x_{\{l,i,k_1\}} && \\
& \quad + \sum_{l \in T(k_1)} x_{\{l,j,k_2\}}, &&
\end{aligned}
\tag{3.3.5}
$$

$$
\begin{aligned}
x_{T_1} + x_{T_2} &\leq 3 + x_{T_1 \cup T_3} + x_{T_2 \cup T_3} && \forall T \subseteq N,\ |T| = 6,\ \forall T_1, T_2 \subseteq T, \\
+ x_{T_3} + x_{T_4} &\quad + x_{T_1\setminus\{k_1\}} + x_{T_2\setminus\{k_2\}}, && T_1 \cup T_2 = T,\ |T_1| = |T_2| = 3,\ \forall T_3, T_4 \subseteq T, \\
& && |T_3 \cap T_4| = 2,\ |T_1 \cap T_3| = 2,\ |T_2 \cap T_4| = 2, \\
& && \forall k_1 \in T_1 \cap T_3,\ k_2 \in T_2 \cap T_4\ \}.
\end{aligned}
$$

Lemma 3.3.12 *For the characteristic imset polytope, there is* $\mathrm{P}_{\mathbf{c}} \subseteq \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(5)}}$.

Proof. We consider Inequalities (3.3.4) first. Let T_1 or T_2 be no clique and no immorality in $\mathrm{pat}(G)$, then $c_{T_1}(G) = 0$ or $c_{T_2}(G) = 0$, which implies $c_{T_1}(G) + c_{T_2}(G) \leq 1$ and

$$1 + \sum\nolimits_{\substack{A \subseteq T(k_1,k_2) \\ |A|=2}} c_A \leq 1 + |T(k_1,k_2)| - 1 + \sum\nolimits_{\substack{B \subseteq T(k_1,k_2) \\ |B|=3}} c_B$$

to be satisfied, for all $k_1 \in T_1$, $k_2 \in T_2$, due to the DAG-inequality.

Let T_1 or T_2 be cliques with $c_{T_1}(G) = c_{T_2}(G) = 1$. Then T_1 or T_2 is complete with $c_{T_1\setminus\{k_1\}}(G) = 1$ or $c_{T_2\setminus\{k_2\}}(G) = 1$, for all k_1, k_2. Together with the DAG-inequalities,

$$
\begin{aligned}
c_{T_1}(G) + c_{T_2}(G) + \sum_{\substack{A \subseteq T(k_1,k_2) \\ |A|=2}} c_A(G) &\leq 2 + |T(k_1,k_2)| - 1 + \sum_{\substack{B \subseteq T(k_1,k_2) \\ |B|=3}} c_B(G) \\
&= |T(k_1,k_2)| + \sum_{\substack{B \subseteq T(k_1,k_2) \\ |B|=3}} c_B(G) + 1 \\
&\leq |T(k_1,k_2)| + \sum_{\substack{B \subseteq T(k_1,k_2) \\ |B|=3}} c_B(G) + c_{T_1\setminus\{k_1\}}(G) + c_{T_2\setminus\{k_2\}}(G)
\end{aligned}
$$

follows. The same follows if T_1, T_2 are immoralities and k_1 or k_2 is not terminal.

Let T_1 and T_2 be immoralities with terminal nodes k_1, k_2, then $c_{T_1}(G) = c_{T_2}(G) = 1$ and $c_{T_1\setminus\{k_1\}}(G) = c_{T_2\setminus\{k_2\}}(G) = 0$. If components T_1 and T_2 are not connected in $\mathrm{pat}(G)$, then

the DAG-inequalities imply

$$\sum\nolimits_{\substack{A\subseteq T(k_1,k_2)\\|A|=2}} c_A < |T(k_1,k_2)| - 1 + \sum\nolimits_{\substack{B\subseteq T(k_1,k_2)\\|B|=3}} c_B$$

and therefore also

$$\begin{aligned} 1+1+\sum\nolimits_{\substack{A\subseteq T(k_1,k_2)\\|A|=2}} c_A &\leq 2+|T(k_1,k_2)|-2+\sum\nolimits_{\substack{B\subseteq T(k_1,k_2)\\|B|=3}} c_B \\ &= 1+|T(k_1,k_2)|+\sum\nolimits_{\substack{B\subseteq T(k_1,k_2)\\|B|=3}} c_B. \end{aligned}$$

We conclude T_1 and T_2 to be immoralities which are connected with a k_1, k_2-path P in $\mathrm{pat}(G)$ and k_1 and k_2 are their terminal nodes.

Let l_1 be the last node on P before node k_1 and l_2 be the last node before k_2. If there exist arcs $l_1 \to k_1$ or $l_2 \to k_2$ in G, then there exist immoralities or cliques $\{l_1, k_1, i\}$ or $\{l_2, k_2, j\}$, for all $i \in T_1 \setminus \{k_1\}$ and $j \in T_2 \setminus \{k_2\}$. This implies for each i or j

$$\sum\nolimits_{l\in T(k_2)} c_{\{l,i,k_1\}}(G) + \sum\nolimits_{l\in T(k_1)} c_{\{l,j,k_2\}}(G) = c_{\{l_1,i,k_1\}}(G) + c_{\{l_2,j,k_2\}}(G) \in \{1,2\}$$

implying Inequalities (3.3.4) to be valid

If both $k_1 \to l_1$ and $k_2 \to l_2$ are contained in G, then there exists an immorality in every path P with nodes $V(P) \subseteq T(k_1, k_2)$. This causes $c_{\bar{B}}(G)$ to be one for at least one $\bar{B} \subseteq T(k_1, k_2)$ with $|\bar{B}| = 3$ and this set is independent from any cycle in P. We get,

$$\begin{aligned} c_{T_1}(G) + c_{T_2}(G) + \sum_{\substack{A\subseteq T(k_1,k_2)\\|A|=2}} c_A(G) &\leq 2+|T(k_1,k_2)|-1+\sum_{\substack{B\subseteq T(k_1,k_2)\\|B|=3}} c_B(G) \\ &= |T(k_1,k_2)| + \sum_{\substack{B\subseteq T(k_1,k_2)\\|B|=3, B\neq\bar{B}}} c_B(G) + c_{\bar{B}}(G). \end{aligned}$$

This proves the Inequalities (3.3.4) to be valid.

Second, consider Inequalities (3.3.5). We observe them to be not directly fulfilled for entries $c_{T_1}(G) = 1$, $c_{T_2}(G) = 1$, $c_{T_3}(G) = 1$ and $c_{T_4}(G) = 1$. Moreover, we assume $c_{T_1\setminus\{k_1\}}(G) = 0$ and $c_{T_2\setminus\{k_2\}}(G) = 0$ and T_1, T_2 are immoralities with terminal nodes k_1 and k_2. The only possibility for T_3 and T_4 with $c_{T_3}(G) = 1$, $c_{T_4}(G) = 1$ is the choice given in Figure 3.25 with $T_1 = \{a, b, c\}$, $T_2 = \{d, e, f\}$. We distinguish between $T_3 = \{a, b, d\}$, $T_4 = \{b, d, e\}$ and $T_3 = \{a, d, e\}$, $T_4 = \{a, b, e\}$.

Let T_3 or T_4 be not a clique, then both are no cliques. In this case $T_3 = \{a, b, d\}$, $T_4 = \{b, d, e\}$, no directed cycle is constructed and both $c_{T_1\cup T_3}(G) = 1$ and $c_{T_2\cup T_4}(G) = 1$ satisfying Inequalities (3.3.5). If both T_3 or T_4 are cliques (for both choices of T_3 and T_4), then $c_{T_1\cup T_3}(G) = 1$ or $c_{T_2\cup T_4}(G) = 1$, as $\mathbf{c}(G)$ implies no directed cycle. □

Conjecture 3.3.13 *Any binary* $\mathbf{x} \in \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(5)}}$ *has a valid reconstruction of a DAG.*

The proof of this conjecture depends on the completeness of the cases considered to obtain contradictions of the orientation rules. We do not have a sufficient proof for that.

3.3.3 Complete description

Conclusion 3.3.14 *Let* $\mathrm{P}_{\mathbf{c}}$ *be the characteristic imset polytope over node set* N*, then the relation* $\mathrm{P}_{\mathbf{c}} \subseteq \overline{\mathrm{P}_{\mathbf{c}}^{\mathrm{rel}}} := \mathrm{P}_{\mathbf{c}}^{\mathrm{rel}(5)}$ *is valid.*

Proof. $\overline{\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}}$ is a valid inequality description of $\mathrm{P}_{\mathbf{c}}$ if and only if each of the inequalities are valid for $\mathrm{P}_{\mathbf{c}}$. This fact follows from the iterative definition of $\overline{\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}}$ and Lemma 3.3.12. □

As $\overline{\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}}$ is not proven to be a facet description of $\mathrm{P}_{\mathbf{c}}$ we focus ourselves on the LP-relaxation $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ of Section 3.2.4, page 59.

3.4 Polytopes of restricted characteristic imsets

The LP-relaxation $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ of Section 3.2.4, page 59, is complicated and not very tight in comparison to a facet description of $\mathrm{P}_{\mathrm{ext}}$. Moreover, the decomposition into smaller polytopes according to Section 3.1.1, page 51, does not help to solve optimisation problems of larger sizes. Using the restriction of the underlying structure instead can give further simplifications, better descriptions and even additional characterisations of geometric neighbours. In the end this leads to good running times of learning restricted BN structures computationally. We start with the most simple structure.

3.4.1 Undirected forests

Let $\mathrm{DAGs}(K)$ denote all DAGs on N whose underlying undirected graph is a subgraph of K and let $\mathrm{DAGs}_{\mathrm{UF}}(K)$ denote all DAGs in $\mathrm{DAGs}(K)$ with edge set E whose pattern is an undirected forest. Now consider the polytope

$$\mathrm{P}_{\mathrm{UF}}(K) := \mathrm{conv}\{\mathbf{c}(G) : G \in \mathrm{DAGs}_{\mathrm{UF}}(K)\}.$$

Learning an undirected forest is the same as maximising a linear function over $\mathrm{P}_{\mathrm{UF}}(K)$. By Corollary 2.2.3, page 43, $\mathrm{P}_{\mathrm{UF}}(K)$ is equivalent to

$$\hat{\mathrm{P}}_{\mathrm{UF}}(K) := \mathrm{conv}\{\chi(\bar{G}) : G \in \mathrm{DAGs}_{\mathrm{UF}}(K)\} \subseteq \{0,1\}^{\binom{|N|}{2}}.$$

The set $\{\chi(\bar{G}) : G \in \mathrm{DAGs}_{\mathrm{UF}}(K)\}$ constitutes a well-known matroid for any K. This is the so-called **graphic matroid** or **cycle matroid**. Many properties are known about $\hat{\mathrm{P}}_{\mathrm{UF}}(K)$ and, equivalently, about $\mathrm{P}_{\mathrm{UF}}(K)$ and linear optimisation over it (see e.g. [54]).

Corollary 3.4.1 *Let $\mathbf{c}(F_1), \mathbf{c}(F_2) \in \mathrm{P}_{UF}(K)$. Then the graphs F_1 and F_2 are geometric neighbours with respect to $DAGs_{UF}(K)$ if and only if*

- *$F_1 \subseteq F_2$ and $|F_2 \setminus F_1| = 1$, or*
- *$F_2 \subseteq F_1$ and $|F_1 \setminus F_2| = 1$, or*
- *$|F_1| = |F_2|$, F_1 and F_2 differ in one pair of edges, i.e. $|F_1 \triangle F_2| = 2$, and their union contains a cycle.*

While the first two cases just mean that F_1 and F_2 are inclusion neighbours, the third condition corresponds to swaps of pairs of edges. The matroid $\{\chi(\bar{G}) : G \in \mathrm{DAGs}_{\mathrm{UF}}(K)\}$ also provides a facet description of $\hat{\mathrm{P}}_{\mathrm{UF}}(K)$ and thus also of $\mathrm{P}_{\mathrm{UF}}(K)$. We denote with $|k(G)|$ the number of components in a general graph G and $x_C := \sum_{e \in C} x_e$. The description

$$\begin{array}{lll} x_T & = 0, & \forall T \subseteq N, |T| < 2, |T| > 2, \\ 0 \leq x_e & \leq 1, & \forall e \in E, \\ x_C & \leq |N| - |k(C)|, & \forall \text{ inclusion-minimal cycles } C \text{ in } K, \end{array}$$

provides a complete facet description of all points in $\mathrm{P}_{\mathrm{UF}}(K)$. This inequality description is even TDI (see e.g. [54]).

But also restricted learning of undirected spanning trees is possible.

3.4.2 Undirected spanning trees

Let K be a connected graph on node set N and let $\mathrm{DAGs}_{\mathrm{UST}}(K)$ denote all DAGs in $\mathrm{DAGs}(K)$ with edge set E whose pattern is an undirected spanning tree of a connected graph K. We consider the following polytope

$$\mathrm{P}_{\mathrm{UST}}(K) := \mathrm{conv}\{\mathbf{c}(G) : G \in \mathrm{DAGs}_{\mathrm{UST}}(K)\}.$$

Learning an undirected spanning tree is the same as maximising a linear function over $\mathrm{P}_{\mathrm{UST}}(K)$. By Corollary 2.2.3 $\mathrm{P}_{\mathrm{UST}}(K)$ is equivalent to

$$\hat{\mathrm{P}}_{\mathrm{UST}}(K) := \mathrm{conv}\{\chi(\bar{G}) : G \in \mathrm{DAGs}_{\mathrm{UST}}(K)\} \subseteq \{0,1\}^{\binom{|N|}{2}}.$$

While $\{\chi(\bar{G}) : G \in \mathrm{DAGs}_{\mathrm{UF}}(K)\}$ forms a matroid for any K, the set of undirected spanning trees $\{\chi(\bar{G}) : G \in \mathrm{DAGs}_{\mathrm{UST}}(K)\}$ is the set of bases of this matroid. In fact, $\hat{\mathrm{P}}_{\mathrm{UST}}(K)$ is the so-called base polytope of the matroid $\{\chi(\bar{G}) : G \in \mathrm{DAGs}_{\mathrm{UF}}(K)\}$ and it is a face of $\hat{\mathrm{P}}_{\mathrm{UF}}(K)$. This gives us also a facet description of $\mathrm{P}_{\mathrm{UST}}(K)$. We define $x_E := \sum_{e\in E} x_e$. The complete facet description of all points in $\mathrm{P}_{\mathrm{UST}}(K)$ is

$$\begin{array}{lll} x_T & = 0, & \forall T \subseteq N, |T| < 2, |T| > 2, \\ 0 \leq x_e & \leq 1, & \forall e \in E, \\ x_C & \leq |N| - |k(C)|, & \forall \text{ inclusion-minimal cycles } C \text{ in } K, \\ x_E & = |N| - 1, & \end{array}$$

which again is a TDI description. Moreover, as $\mathrm{P}_{\mathrm{UST}}(K)$ is a face of $\mathrm{P}_{\mathrm{UF}}(K)$ defined by the valid inequality $x_E \leq |N| - 1$, it inherits the edges from $\mathrm{P}_{\mathrm{UF}}(K)$ that keep x_E constant.

Corollary 3.4.2 *Let $\mathbf{c}(F_1), \mathbf{c}(F_2) \in \mathrm{P}_{UST}(K)$ be given. Then the graphs F_1 and F_2 are geometric neighbours with respect to $DAGs_{UST}(K)$ if and only if F_1 and F_2 differ in one pair of edges, that is $|F_1 \triangle F_2| = 2$.*

This simply follows from Corollary 3.4.1 applied to the equal size of spanning trees.

3.4.3 Undirected spanning trees and forests with degree bounds

A line-tree and a line-forest are trees and forests, respectively, without any node exceeding a degree bound of two (compare with Section 1.6.5, page 35). Let $\mathrm{DAGs}_{\mathrm{USLT}^2}(K)$ denote all DAGs in $\mathrm{DAGs}(K)$ with edge set E whose pattern is an undirected spanning line-tree of a connected graph K on N. Let $\mathrm{DAGs}_{\mathrm{ULF}^2}(K)$ denote all DAGs in $\mathrm{DAGs}(K)$ with edge set E whose pattern is an undirected line-forest of a graph K on N. Then, by Corollary 2.2.3 and the previous sections, the corresponding polytopes are

$$\begin{aligned} \mathrm{P}_{\mathrm{ULF}}{}^2(K) &:= \mathrm{P}_{\mathrm{UF}}(K) \cap \{d(i) \leq 2,\ \forall i \in N\} \text{ and} \\ \mathrm{P}_{\mathrm{USLT}}{}^2(K) &:= \mathrm{P}_{\mathrm{UST}}(K) \cap \{d(i) \leq 2,\ \forall i \in N\}, \end{aligned}$$

for $d(i)$ denoting the degree of a node i of K. This means that we have to add inequalities

$$\sum_{e \in \delta(i)} x_e \leq 2, \ \forall i \in N, \tag{3.4.1}$$

to the inequality system of $\mathrm{P}_{\mathrm{UF}}(K)$ and $\mathrm{P}_{\mathrm{UST}}(K)$, respectively. We define $\delta(i)$ to be the set of all incident edges of node i.

This can easily be generalised for any degree bound $d(i) \leq k \leq |N| - 1$, $k \in \mathbb{N}$, imposed on the nodes in the forests and spanning trees, respectively. So we do have the following LP-relaxations:

$$\begin{array}{lll} x_T & = 0, & \forall T \subseteq N, |T| < 2, |T| > 2 \\ 0 \leq x_e & \leq 1, & \forall e \in E, \\ x_C & \leq |N| - |k(C)|, & \forall \text{ inclusion-minimal cycles } C \text{ in } K, \\ \sum\limits_{e \in \delta(i)} x_e & \leq k, & \forall i \in N, \end{array}$$

for the polytope of undirected degree bounded forests ${\mathrm{P}_{\mathrm{ULF}}}^k(K)$ and

$$\begin{array}{lll} x_T & = 0, & \forall T \subseteq N, |T| < 2, \\ 0 \leq x_e & \leq 1, & \forall e \in E, \\ x_C & \leq |N| - |k(C)|, & \forall \text{ inclusion-minimal cycles } C \text{ in } K, \\ x_E & = |N| - 1, & \\ \sum\limits_{e \in \delta(i)} x_e & \leq k, & \forall i \in N, \end{array}$$

for the polytope of undirected degree bounded spanning trees ${\mathrm{P}_{\mathrm{USLT}}}^k(K)$.

To get a somehow analogous and short description of the skeleton as in the forest case we have to show the polytope of degree bounded forests to be combinatorial, which cannot be concluded easily. This definition means for us to show for any two non-adjacent and distinct vertices $\mathbf{x}_1$ and $\mathbf{x}_2$ two other distinct vertices $\mathbf{x}_3$ and $\mathbf{x}_4$ to exist, for which $\mathbf{x}_1 + \mathbf{x}_2 = \mathbf{x}_3 + \mathbf{x}_4$. This fact holds in the matroid case of the previous mentioned subfamily, but does not generally hold for independence systems (see [30]) of degree bounded forests.

The motivation for the search of geometric neighbours, also for more complicated structures, comes from the graphical translation. If there would be a characterisation of two characteristic imsets to be adjacent only on the basis of the graphs they encode, then this implies an easy and probably polynomial characterisation of at least one geometric edge. Edge-based algorithms mentioned in the beginning of this chapter may lead to fast algorithms or heuristics solving the structure learning task efficiently.

3.4.4 Chordal graphs

Let $\mathrm{DAGs}_{\mathrm{CG}}(K)$ denote all DAGs in $\mathrm{DAGs}(K)$ whose pattern is a chordal subgraph of a graph K on N.

$$\mathrm{P}_{\mathrm{CH}} := \mathrm{conv}\{\mathbf{c}(G) : G \in \mathrm{DAGs}_{\mathrm{CG}}(K)\}$$

This polytope cannot be reduced to a polynomial size description, as also higher dimensional entries of $\mathbf{c}(G)$ may be non-zero (compare with Corollary 2.2.2, page 42). But the set of characteristic imsets describing chordal graphs is a subset of those describing any pattern of a DAG and with this $\mathrm{P}_{\mathrm{CH}} \subseteq \mathrm{P}_{\mathbf{c}}$. Therefore all inequalities valid for $\mathrm{P}_{\mathbf{c}}$ are also valid for P_{CH}.

Chordal graphs are graphical models without immoralities, hence some of the inequalities of the description of $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$, $\mathrm{P}^{\mathrm{rel}\,\prime}_{\mathrm{ext}}$ and $\overline{\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}}$ are trivially satisfied and therefore redundant. The remaining conditions to be satisfied are induced by a *characteristic imset extension* applied to chordal graphs and the decomposition of cycles of length ≥ 4 into cliques of size three. These two properties give an "if and only if"-condition on binary vectors $\mathbf{x}$ to be characteristic imsets $\mathbf{c}(G)$ of chordal graphs G. For the first condition we use a modification of the cim-Inequalities (3.2.8), page 59, and for the second condition we use the DAG-Inequalities (3.2.16), page 63. A consequence of the use of only these inequalities is, that binary variables $\mathbf{y}$ of the extended formulation $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ do not have to be used anymore and only variables $\mathbf{x}$ remain representing characteristic imsets. Moreover $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}|_{\mathbf{c}}$, $\mathrm{P}^{\mathrm{rel}\,\prime}_{\mathrm{ext}}|_{\mathbf{c}}$ and $\overline{\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}}$ coincide if the cim- and the DAG-inequalities are contained in the inequality descriptions only.

3.4.4.1 LP-relaxation of the polytope of characteristic imsets of chordal graphs

The following polytope with $t := 2^{|N|} - |N| - 1$ contains all $\mathbf{c} \in \mathrm{P}_{\mathrm{CH}}$.

$$
\begin{aligned}
\mathrm{P}^{\mathrm{rel}\,(1)}_{\mathrm{CH}} := \{\mathbf{x} \in [0,1]^t :\ & \\
\sum_{i \in T} x_{T\setminus\{i\}} \quad &\leq \quad 2 + (|T|-2)x_T, \qquad \forall T \subseteq N,\ |T| \geq 3, && (3.4.2)\\
(|T|-1)x_T \quad &\leq \quad \sum_{i \in T} x_{T\setminus\{i\}}, \qquad \forall T \subseteq N,\ |T| \geq 3, && (3.4.3)\\
\sum_{\substack{A \subseteq T,\\ |A|=2}} x_{\{i,j\}} \quad &\leq \quad |T| - 1 + \sum_{\substack{B \subseteq T\\ |B|=3}} x_B, \qquad \forall T \subseteq N,\ |T| \geq 4 \ \ \} && (3.4.4)
\end{aligned}
$$

However, the description of $\mathrm{P}^{\mathrm{rel}\,(1)}_{\mathrm{CH}}$ implies the existence of immoralities due to Inequalities (3.4.3) and therefore 0/1-vectors $\mathbf{x} \in \mathrm{P}^{\mathrm{rel}\,(1)}_{\mathrm{CH}}$ with $\mathbf{x} \notin \mathrm{P}_{\mathrm{CH}}$.

Chordal graphs are much easier to describe than patterns of "arbitrary" DAGs. Their additional structural information can easily be incorporated into an inequality description of the LP-relaxation of P_{CH} leading to stronger inequalities.

We now analyse Inequalities (3.4.3) in more detail. With Corollary 2.2.2, page 42, and Lemma 2.2.7, page 45, we conclude, that for sets $T \subseteq N$, $|T| \geq 3$, the entry $c_T(G)$ is one if and only if there exist $|T|$ subsets $T_i \subseteq T$, $|T_i| = |T| - 1$, with $c_{T_i}(G) = 1$. The forward direction leads to inequalities

$$|T| x_T \leq \sum_{i \in T} x_{T\setminus\{i\}},\ \forall T \subseteq N,\ |T| \geq 3.$$

Therefore, $c_T(G) = 1$ if and only if T is a clique and this is equivalent to the graphical translation $\mathrm{pat}(G)$ of the characteristic imset to be undirected. These inequalities can be further simplified to the Inequalities (3.4.5) below. Let $t := 2^{|N|} - |N| - 1$ given,

$$
\begin{aligned}
\mathrm{P}^{\mathrm{rel}}_{\mathrm{CH}} := \{\mathbf{x} \in [0,1]^t :\ & \\
\sum_{i \in T} x_{T\setminus\{i\}} \quad &\leq \quad 2 + (|T|-2)x_T, \qquad \forall T \subseteq N,\ |T| \geq 3,\\
x_T \quad &\leq \quad x_{T\setminus\{i\}}, \qquad \forall T \subseteq N,\ |T| \geq 3, && (3.4.5)\\
\sum_{\substack{A \subseteq T,\\ |A|=2}} x_{\{i,j\}} \quad &\leq \quad |T| - 1 + \sum_{\substack{B \subseteq T\\ |B|=3}} x_B, \qquad \forall T \subseteq N,\ |T| \geq 4\ \}.
\end{aligned}
$$

Theorem 3.4.3 *Let P_{CH} be the characteristic imset polytope over the node set N, then $\mathrm{P}^{\mathrm{rel}}_{CH} \cap \mathbb{Z}^t = \mathrm{P}_{CH} \cap \mathbb{Z}^t$, $t := 2^{|N|} - |N| - 1$.*

Proof. This fact follows from Theorem 3.2.1, page 55, and the DAG-Inequalities (3.2.16) applied to characteristic imsets of chordal graphs. By the help of the cim-Inequalities (3.4.2) and Inequalities (3.4.5) binary vectors $\mathbf{x} \in \mathrm{P}^{\mathrm{rel}}_{\mathrm{CH}}$ describe graphical translations $G(\mathbf{x}|_{\leq 3})$ which are undirected graphs and which satisfy Lemma 2.2.7. Undirected graphs $G(\mathbf{x}|_{\leq 3})$ are chordal if and only if every cycle of length ≥ 4 has a diagonal and hence decomposes into cliques of size three. This property is provided due to the DAG-Inequalities (3.4.4). □

Learning the best chordal graph $G(\mathbf{x}|_{\leq 3})$ is equal to the optimisation problem:

$$\begin{aligned} \max \quad & \mathbf{w}^\intercal \mathbf{x} \\ \text{s.t.} \quad & \mathbf{x} \in \mathrm{P}^{\mathrm{rel}}_{\mathrm{CH}}, \\ & \mathbf{x} \in \{0,1\}^{2^{|N|}-|N|-1}. \end{aligned} \tag{3.4.6}$$

But even more inequalities valid for P_{CH} and $\mathrm{P}^{\mathrm{rel}}_{\mathrm{CH}}$, respectively, can be derived.

3.4.4.2 Additional inequalities and alternative formulations

We derive some additionally valid inequalities and observe some of them to have the property to define a learning problem of chordal graphs in which some of the integrality conditions can be skipped.

Inequalities due to a relaxed integrality condition Using again Lemma 2.2.7 and Inequalities (3.4.5) we can further conclude for $\mathbf{c}(G) \in \mathrm{P}_{\mathrm{CH}}$: If $c_T(G) = 1$, then also $c_{\{i,j\}}(G) = 1$, for all edges $\{i,j\}$. This implication obtains Inequalities (3.4.7) below, i.e. $x_T \leq x_{\{i,j\}}$, $\forall T \subseteq N$, $|T| \geq 3$, $\forall i \neq j \in T$ (compare with Figure 3.28 below). The back direction, i.e. Inequalities (3.4.2), ensures also higher subsets of T to induce cliques.

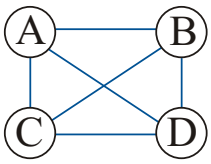

Figure 3.28: A clique induced by $\{A, B, C, D\}$ implies all (blue) edges to exist.

Using these Inequalities (3.4.7) instead of Inequalities (3.4.5) yields a LP-relaxation of P_{CH} in which only the integrality of variables $\mathbf{x}|_{=2}$ is important.

Let $t := 2^{|N|} - |N| - 1$, then

$$\begin{aligned} \mathrm{P}^{\mathrm{rel}\,\prime}_{\mathrm{CH}} := \{x \in [0,1]^f : & \\ \sum_{i \in T} x_{T\setminus\{i\}} \quad &\leq \quad 2 + (|T|-2)x_T, \qquad \forall T \subseteq N,\ |T| \geq 3, \\ x_T \quad &\leq \quad x_{\{i,j\}}, \qquad \forall T \subseteq N,\ |T| \geq 3,\ \forall i \neq j \in T, \\ \sum_{\substack{A \subseteq T,\\ |A|=2}} x_{\{i,j\}} \quad &\leq \quad |T| - 1 + \sum_{\substack{B \subseteq T\\ |B|=3}} x_B, \qquad \forall T \subseteq N,\ |T| \geq 4\ \} \end{aligned} \tag{3.4.7}$$

is a description in which the integrality on $\mathbf{x}$ can be relaxed.

Theorem 3.4.4 *Let $\mathrm{P}^{\mathrm{rel}\,\prime}_{CH}$ be defined as above and $\mathbf{x} \in \mathrm{P}^{\mathrm{rel}\,\prime}_{CH}$. First, the integral points coincide, i.e. $\mathrm{P}^{\mathrm{rel}\,\prime}_{CH} \cap \mathbb{Z}^t$ equals $\mathrm{P}^{\mathrm{rel}}_{CH} \cap \mathbb{Z}^t$, $t = 2^{|N|} - |N| - 1$. Second, if entries $x_{\{i,j\}} \in \{0,1\}$, $\forall i \neq j \in N$, then $x_T \in \{0,1\}$, $\forall T \subseteq N$, $|T| \geq 3$.*

Proof. We suppose $\mathbf{x}|_{=2}$ to be integral and $\mathbf{x} \in \mathrm{P}_{\mathrm{CH}}^{\mathrm{rel}\,\prime}$. Analogous to the proof of Theorem 3.2.4 we deduce first $\mathbf{x}|_{=3}$ to be integral once $\mathbf{x}|_{\leq 2}$ is integral and use induction to show $\mathbf{x}|_{\geq |T|}$ with $|T| \geq 4$ to be integral provided $\mathbf{x}|_{<|T|}$ is integral.

Let $T \subseteq N$, $|T| = 3$, be given and let T induce a clique. All edges $\{i,j\} \subseteq T$ exist in the graphical translation $G(\mathbf{x}|_{\leq 3})$ and $x_{\{i,j\}} = 1$. To fulfil the first of the cim-Inequalities (3.4.2) $x_T = 1$ is the only choice. Contrarily, let T induce no clique, then at least one edge in T does not exist, say $\{i,j\}$ with $x_{\{i,j\}} = 0$. We apply Inequalities (3.4.7) for this edge $\{i,j\}$ and hence also $x_T = 0$.

For the next step we assume $|T| = n+1$ and $|S| \leq n$ such that entries x_S are supposed to be integral and x_T is analysed. Similar to above we apply the first of the cim-Inequalities (3.4.2) and observe $|T \setminus \{i\}| \leq n$, $\forall i \in T$, and conclude x_T to be integral if T induces a clique. For T to imply no clique, we can directly apply Inequalities (3.4.7) and observe $x_T = 0$. □

To compute the best chordal graph $G(\mathbf{x}|_{\leq 3})$ we consider the optimisation problem

$$\begin{aligned} \max \quad & \mathbf{w}^\intercal \mathbf{x} \\ \text{s.t.} \quad & \mathbf{x} \in \mathrm{P}_{\mathrm{CH}}^{\mathrm{rel}\,\prime}, \\ & \mathbf{x}|_{=2} \in \{0,1\}^{|N|(|N|-1)/2}. \end{aligned} \tag{3.4.8}$$

Additional valid inequalities For an explanation of the approach to derive Inequalities (3.4.9), page 83, consider the DAG-inequalities. If $x_T = 0$, therefore T induces no clique, then these inequalities coincide with the previous DAG-inequalities. But if $x_T = 1$, then T induces a clique and all subsets are cliques. As G^T is a complete graph we have, $\sum_{\{i,j\} \subseteq T} x_{\{i,j\}} = \binom{|T|}{2}$ and $\sum_{\substack{B \subseteq T \\ |B|=3}} x_B = \binom{|T|}{3}$. Both are monotonous increasing functions with $\binom{|T|}{2} \leq \binom{|T|}{3}$, $\forall T \subseteq N$ with $|T| \geq 4$. For these functions we can conclude $\binom{|T|}{2} + 1 = |T| - 1 + \binom{|T|}{3}$ in case of $|T| = 4$ and $\binom{|T|}{2} + 1 < |T| - 1 + \binom{|T|}{3}$ for $|T| > 4$. Figure 3.29 below shows a clique with nodes T of size four with all contained cliques of size three.

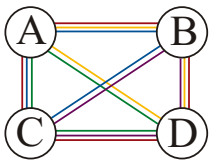

Figure 3.29: If $T = \{A, B, C, D\}$ induces a clique, then all cliques of size three exist.

Inequalities (3.4.10), page 83, can be derived simply by observing the following fact: If a graph G^T has two cliques within, say $T \setminus \{i\}$ and $T \setminus \{j\}$, then the only missing edge in G^T is edge $\{i,j\}$. The existence of this edge implies T to be a clique (compare with Figure 3.30 below). The term $x_{T \setminus \{i,j\}}$ is added to ensure satisfiability also in case the edge $\{i,j\}$ is not present.

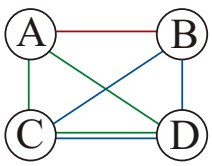

Figure 3.30: $T = \{A, B, C, D\}$ induces a clique if there exist two cliques of size $|T| - 1$ (blue and green) and the missing edge (red).

The remaining two Inequalities are rather technical and a graphical intuition is hard to

present. However, both are valid by distinction of T to be a clique or not and doing some case study on this.

We use Inequalities (3.4.2) and split the sum on the left hand side yielding

$$\sum_{i \in T \setminus \{j\}} x_{T \setminus \{i\}} + x_{T \setminus \{j\}} \leq 2 + (|T| - 2) x_T.$$

This we add with $x_{T \setminus \{j\}} \leq 1$ to obtain

$$\sum_{i \in T \setminus \{j\}} x_{T \setminus \{i\}} + 2 x_{T \setminus \{j\}} \leq 3 + (|T| - 2) x_T.$$

Inequalities (3.4.11), this page, are a stronger version of the inequalities above, but they are still valid. If T is a clique all subsets are cliques and

$$3 + (|T| - 2) x_T \leq 2 x_T + \sum_{i \in T \setminus \{j\}} x_{T \setminus \{i,j\}}, \quad (3 + |T| - 2 \leq 2 + |T| - 1).$$

If T is not a clique, then at most two subsets are cliques. First, let $T \setminus \{j\}$ be not a clique, as well. Then $x_{T \setminus \{j\}} = 0$ and $\sum_{i \in T \setminus \{j\}} x_{T \setminus \{i\}} \leq 2$. As for all $j \in T$, $T \setminus \{i, j\} \subseteq T \setminus \{i\}$ for one such i there are at least as many subsets $T \setminus \{i, j\}$ with $x_{T \setminus \{i,j\}} = 1$ as there are $x_{T \setminus \{i\}} = 1$ (if $x_{T \setminus \{i_1\}} = 1$ and $x_{T \setminus \{i_2\}} = 1$, then $x_{T \setminus \{i_1,j\}} = 1$ and $x_{T \setminus \{i_2,j\}} = 1$ for $i_1, i_2 \in T \setminus \{j\}$) and the inequalities are fulfilled. Second, let $T \setminus \{j\}$ be a clique, then for at most one $i \in T \setminus \{j\}$ there is $x_{T \setminus \{i\}} = 1$ and

$$\sum_{i \in T \setminus \{j\}} x_{T \setminus \{i\}} + 2 x_{T \setminus \{j\}} \leq 3 \leq |T| - 1.$$

On the other hand, $T \setminus \{i, j\} \subseteq T \setminus \{j\}$, for all $i \in T \setminus \{j\}$, implies $x_{T \setminus \{i,j\}} = 1$, $\forall i \in T \setminus \{j\}$. The corresponding sum is equal to $|T| - 1$ and the inequalities are valid.

To derive Inequalities (3.4.12), this page, we again distinguish between T to be a clique or not. If it is a clique, all subsets are cliques as well and

$$\sum_{i \in T \setminus T_j} x_{T \setminus \{i\}} = |T| - |T_j| = x_T + \sum_{i \in T \setminus (T_j \cup \{k\})} x_{T_j \cup \{i\}} = 1 + |T| - |T_j| - 1.$$

Contrarily, let T be not a clique. Again, there exist at most two subsets which are cliques and $\sum_{i \in T \setminus T_j} x_{T \setminus \{i\}} \leq 2$. Let those be $T \setminus \{l\}$ and $T \setminus \{m\}$ $(l, m \notin T_j)$. For inequalities with $k = l$ or $k = m$ there is $T_j \cup \{i\} \subseteq T \setminus \{l\}$ or $T_j \cup \{i\} \subseteq T \setminus \{m\}$, respectively, $\forall i \in T \setminus (T_j \cup \{k\})$. As $|T \setminus (T_j \cup \{k\})| \geq 2$ the inequalities are satisfied. For inequalities with $k \neq l$ and $k \neq m$ there is $l, m \in T \setminus (T_j \cup \{k\})$ and therefore $T_j \cup \{l\} \subseteq T \setminus \{m\}$, $T_j \cup \{m\} \subseteq T \setminus \{l\}$, which also implies the inequalities to be valid.

Below, there is the list of these derivable additional valid inequalities.

$$\sum_{\{i,j\} \subseteq T} x_{\{i,j\}} + x_T \leq |T| - 1 + \sum_{\substack{B \subseteq T \\ |B| = 3}} x_B, \qquad \forall T \subseteq N,\ |T| \geq 4, \tag{3.4.9}$$

$$x_{T \setminus \{i\}} + x_{T \setminus \{j\}} + x_{\{i,j\}} \leq 1 + x_T + x_{T \setminus \{i,j\}}, \qquad \forall T \subseteq N,\ |T| \geq 4,\ \forall \{i,j\} \subseteq T, \tag{3.4.10}$$

$$\sum_{i \in T \setminus \{j\}} x_{T \setminus \{i\}} + 2 x_{T \setminus \{j\}} \leq 2 x_T + \sum_{i \in T \setminus \{j\}} x_{T \setminus \{i,j\}}, \qquad \forall T \subseteq N,\ |T| \geq 4,\ \forall j \in T, \tag{3.4.11}$$

$$\sum_{i \in T \setminus T_j} x_{T \setminus \{i\}} \leq x_T + \sum_{i \in T \setminus (T_j \cup \{k\})} x_{T_j \cup \{i\}}, \qquad \forall T \subseteq N,\ |T| \geq 4,\ \forall T_j \subseteq T, \tag{3.4.12}$$

$$1 \leq |T_j| \leq |T| - 3, \ \forall k \in T \setminus T_j$$

3.4.4.3 Geometric neighbours of chordal graphs

In contrast to the set of forests and degree bounded forests, respectively, the set of chordal graphs itself does not even constitute an independence system. This causes the geometric neighbours to be hard to derive. Moreover, they cannot be concluded easily from the neighbours of forests (even graphically).

3.4.5 Bounds on degrees of nodes and cardinalities of cliques

Both bounds on the degree of nodes and bounds on the maximum size of cliques can easily be incorporated into the description of polytopes of restricted characteristic imsets. This can be done in a direct way by adding inequalities of type (3.4.1), page 79, for bounds on degrees of nodes, or by limiting the maximum cardinality of subsets T of characteristic imsets. We use the second formulation technique as this can be done independently from the inequality description. The corresponding variables are simply set to zero and can be skipped from the complete description. See Section 5.3.1.2, page 101, for the necessary modifications of the problem.

3.5 The standard imset polytope

3.5.1 Decomposition of the standard imset polytope

We have already analysed the property of the characteristic imset polytope to decompose into smaller polytopes (see Section 3.1.1, page 51). Furthermore, we have related this to the dynamic programming approach (see Section 1.4.2.2, page 24). The purpose of this derivation is an analogous problem description in terms of standard imsets and a characterisation of geometric neighbours. As the characteristic imset polytope and the standard imset polytope are isomorphic the decomposition into smaller polytopes is also visible in the standard imset polytope. As this is a straightforward result we skip the derivation here and state only the results.

For this purpose we derive a formula similar to the characterisation of edges of the standard imset polytope $P(S)$ (see Section 1.4.2.4, page 28).

Lemma 3.5.1 *Let* $\mathbf{u}(G)$ *be a standard imset of a DAG* G*, then*

$$\mathbf{u}(G) - \mathbf{u}(\emptyset) = \sum_{i \in N} (\mathbf{u}(G^i_{\text{sink}}) - \mathbf{u}(\emptyset)), \tag{3.5.1}$$

where $\mathbf{u}(G^i_{\text{sink}})$ *is the standard imset of the graph consisting of all arcs in* G *entering node* i*.*

Proof. Using the formula for standard imsets, we get for the left hand side

$$\begin{aligned}\mathbf{u}(G) - \mathbf{u}(\emptyset) &= \delta_N - \delta_\emptyset + \sum_{i \in N} (\delta_{\text{pa}(i)} - \delta_{\text{pa}(i) \cup \{i\}}) - \delta_N + \delta_\emptyset - |N| \cdot \delta_\emptyset + \sum_{i \in N} \delta_{\{i\}} \\ &= \sum_{i \in N} (\delta_{\text{pa}(i)} - \delta_{\text{pa}(i) \cup \{i\}} - \delta_\emptyset + \delta_{\{i\}}).\end{aligned}$$

On the right hand side, we analogously get

$$\begin{aligned}\sum_{i\in N}(\mathbf{u}(G^i_{\text{sink}})-\mathbf{u}(\emptyset)) &= \sum_{i\in N}(-\delta_\emptyset - \sum_{j\in N\setminus\{i\}}\delta_{\{j\}}+\delta_{\text{pa}(i)}-\delta_{\text{pa}(i)\cup\{i\}}+\sum_{i\in N}\delta_{\{i\}})\\ &= \sum_{i\in N}(-\delta_\emptyset+\delta_{\text{pa}(i)}-\delta_{\text{pa}(i)\cup\{i\}}+\delta_{\{i\}}),\end{aligned}$$

showing equality of both sides. □

Note, that if the pattern of G corresponds to an undirected forest, each node of G has at most one arc entering. Therefore $\mathbf{u}(G^i_{\text{sink}})$, $\forall i \in N$, simplifies to $\mathbf{u}(e)$ for each arc e of G.

We now interpret this equation in a rather general way. Let $P_{\text{sink}}(i)$ be the polytope of standard imsets of all DAGs having only arcs entering node i. Then its vertices are standard imsets of the form $\mathbf{u}(G^i_{\text{sink}})$.

Corollary 3.5.2 *Let* $\text{P}_{\text{sink}}(i)$ *be the standard imset polytope of all DAGs having only arcs entering node* i*. Then the standard imset polytope* $P(S)$ *of all DAGs is contained in the Minkowski sum of polytopes:*

$$\text{P}_{\text{sink}}(1)+\text{P}_{\text{sink}}(2)+\ldots+\text{P}_{\text{sink}}(|N|)-(|N|-1)\mathbf{u}(\emptyset).$$

Proof. The Minkowski sum of polytopes is defined to be the convex hull of the sum of single vertices for each smaller polytope. This yields again a polytope. By Equation (3.5.1) each vertex in $P(S)$ can be written as the sum of vertices $\mathbf{u}(G^i_{\text{sink}})$ of $\text{P}_{\text{sink}}(i)$, $\forall i \in N$, and $\mathbf{u}(\emptyset)$. □

The reverse direction is false. Only if we impose an ordering π of nodes, we can decompose the problem of finding the best standard imset by solving each smaller problem:

$$\begin{aligned}\max\ & \mathbf{c}^\intercal\mathbf{u}\\ \text{s.t.}\ & \mathbf{u}\in \text{P}^\pi_{\text{sink}}(i),\end{aligned}$$

for nodes i.

Finding the overall best fitting model comprises to optimise each of the $|N|$ smaller problems and to construct $\mathbf{u}^{\text{opt}}$ by adding the vertices attaining the maxima and by subtracting with $(|N|-1)\cdot\mathbf{u}(\emptyset)$.

In spite of this separation into smaller problems, Equation (3.5.1) leads to another interpretation.

3.5.2 Geometric neighbours in the standard imset polytope

Using directly Equation (3.5.1) or indirectly the isomorphic transformation between the standard imset polytope and the characteristic imset polytope, geometric neighbours can be derived on the base of the decomposition of the standard imset polytope and Remark 3.1.3, page 53.

3.5.3 Conjectured facet description of the standard imset polytopes

In spite of having a explicit inequality description of $P(S)$, there is a list of inequalities presented in [64] providing a LP-relaxation of $P(S)$ and which are furthermore conjectured to provide a facet description.

The authors Studený and Vomlel of [64] divide the list of facets by means of standard imsets as integral points of the cone $\mathcal{E}(N)$ (type (A) and (B)) and by means of additional non-valid points (type (C)).

3.5.3.1 Equality constraints

The following two types of inequalities define $P(S)$ to be contained in a linear subspace of dimension $2^{|N|} - |N| - 1$.

$$\sum_{S \subseteq N} u_S = 0, \tag{A.1}$$

$$\sum_{S: i \in S \subseteq N} u_S = 0, \quad \forall i \in N \tag{A.2}$$

With Equations (A.1) and (A.2) above each of the entries corresponding to subsets S with cardinality $|S| \leq 1$ are direct consequences of the remaining entries. The amount of sets S is exactly $|N| + 1$, which causes the dimension of the polytope.

3.5.3.2 Non-specific inequality constraints

The second class of inequalities defines all elements, and thus $P(S)$ itself, to be elements of the cone $\mathcal{R}(N)$ generated by elementary imsets $\mathcal{E}(N)$. Studený and Vomlel use the outer description of $\mathcal{E}(N)$, which can be described via supermodular functions defined in Lemma 6.2.6, page 118.

$$\langle m, \mathbf{u} \rangle \geq 0, \qquad \text{for every supermodular function } m : \mathcal{P}(N) \to \mathbb{R}^{2^{|N|}-|N|-1}.$$

This outer description plays a central role in Chapter 6, page 115, such that we skip the corresponding definitions to the respective sections within Chapter 6. But note, that this is an infinite list of inequalities. An equivalent formulation derived in Chapter 6 can be used to get a finite list of inequalities instead:

$$\langle m, \mathbf{u} \rangle > 0, \quad \forall m \in \mathcal{K}_l^{\diamond}(N). \tag{B}$$

As m is not explicitly given, this fact coincides with the computation of the faces of the cone $\mathcal{R}(N)$, which is computationally infeasible for problems of larger sizes (compare with Chapter 6).

3.5.3.3 Specific inequality constraints

Up to now, every point in $\mathcal{R}(N)$ is valid. But $P(S)$ is bounded and there must exist points in $\mathcal{R}(N)$ not being elements of the polytope. The last class of Inequalities bounds $P(S)$ in $\mathcal{R}(N)$ and is defined by so-called "ascending classes of sets" which are in this case defined to be closed under supersets.

Definition 3.5.3 (closed under supersets) *A class $\mathcal{A} \subseteq \mathcal{P}(N)$ is called **closed under supersets** if*

$$\forall S \in \mathcal{A} \text{ with } S \subseteq T \subseteq N \text{ then } T \in \mathcal{A}.$$

To obtain non-redundant inequalities only those classes containing sets with at least one element are considered in [64].

$$\sum_{S\in\mathcal{A}} u_S \leq 1, \qquad \text{for any system } \emptyset \neq \mathcal{A} \subseteq \mathcal{P}(N)_{\geq 1} \text{ closed under supersets} \tag{C}$$

Still, some of the constraints of type (C) are redundant with respect to Equations (A.1), (A.2) and Inequalities (B). Therefore Studený and Vomlel introduce a "standard form" of the Inequalities (C) above,

$$\sum_{S\in\mathcal{B}} k_S \cdot u_S \leq 1, \qquad \text{for } \mathcal{B} \subseteq \mathcal{P}(N)_{\geq 2} \text{ and } k_S \in \mathbb{Z} \text{ for } S \in \mathcal{B}.$$

3.5.3.4 LP-relaxation of the standard imset polytope

Elementary imsets are standard imsets and we moreover know that the origin of the cone $\mathcal{R}(N)$ is a standard imset, as well. The origin coincides with the **0**-imset defined by the complete graph. This implies the Inequalities (B) to be necessary and sufficient for defining the "lower" part of the polytope. Equivalently, we know the geometric neighbours of the **0**-imset (compare with the blue part in Figure 3.31 below). The problem is the "upper" part of the polytope comprising the geometric neighbours of the empty graph and edges/facets defined by all other standard imsets (the red part in Figure 3.31 below). These facets should be defined by Inequalities (C).

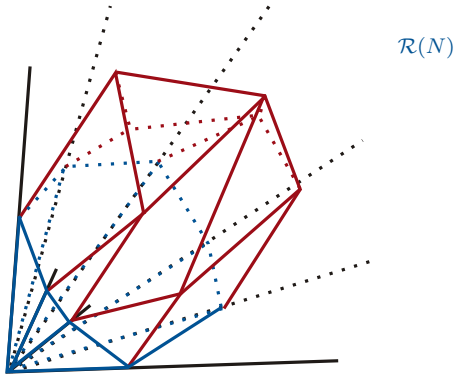

Figure 3.31: The standard imset polytope is entirely contained in the cone $\mathcal{R}(N)$ of elementary imsets.

However, all of the presented inequalities are valid for $P(S)$ but it remains to show this for Inequalities (C) only.

Lemma 3.5.4 (Lemma 1 in [64]) *If* **u** *is a standard imset over* N *then the Condition (C) is valid.*

This fact gives the validity of the inequality description given by (A.1), (A.2), (B) and (C). Recently, in Corollary 23 of [61], the following conjecture of [64] has been proven:

Conjecture 3.5.5 (compare with [64]) *Inequalities (A.1), (A.2), (B) and (C) together provide a LP-relaxation of* $P(S)$*.*

But it is still open to show these inequalities to constitute a facet description of $P(S)$. The hope of the authors in [64] for proving this is based on confirmed computations made for $|N| \leq 4$.

Conjecture 3.5.6 (compare with [64]) *Inequalities (A.1), (A.2), (B), and (C) together provide a facet description of $P(S)$*

The major drawback of the Inequalities (A.1), (A.2), (B) and (C) is, that those of type (B) are not explicitly given. In spite of this, for the optimisation procedure the generation of inequalities at the time they are needed, e.g. in a row generation procedure (compare with Section 5.2.2, page 97), may be possible and the structure can be exploited to compute also the non-specific Inequalities (B).

However, this drawback causes the need for other methods to derive inequalities. These methods should rather gain from a graphical intuition which characteristic imsets can give. With this graphical intuition we provide an explicit LP-relaxation of $\mathrm{P}_{\mathbf{c}}$ in Section 3.2, page 54.

3.6 Related work

We use the representation by Jaakkola et al. [37] of DAGs to present their inequalities describing the corresponding polytope P_η (see Section 2.4.1, page 49).

Jaakkola et al. analogously use an exponential representation of directed acyclic graphs. Likewise to the description via standard and characteristic imsets, respectively, the cycle-inequalities from the acyclic subgraph polytope cannot be applied directly.

Let $C \subseteq N$ be a subset which they call a **cluster of nodes**. Let $I_C(s_i)$ be one if $C \cap s_i = \emptyset$ and zero otherwise. The authors use I_C to indicate a parent set to lie outside a cluster or to be the empty set. An equivalent definition of a DAG is that the set of parents of every cluster is either outside the cluster or the empty set. This causes at least one of the variables $\eta_{(i,s_i)}$ to be one for all $s_i \in Pa(i)$ and the inequalities

$$\sum_{i \in C} \sum_{s_i \in Pa(i)} \eta_{(i,s_i)} I_C(s_i) \geq 1 \tag{c1}$$

describe this fact. Together with inequalities defining $\eta_{(i,s_i)}$ appropriately a complete LP-relaxation of P_η called $\mathcal{P}_{cluster}$ can be obtained.

$$\begin{aligned} \mathcal{P}_{cluster} := \{\eta \in [0,1]^{|N|\cdot 2^{|N|}} : & \\ \eta_{(i,s_i)} &\geq 0, && \forall i \in N,\ \forall s_i \in Pa(i), \\ \sum_{s_i \in Pa(i)} \eta_{(i,s_i)} &= 1, && \forall i \in N, \\ \sum_{i \in C} \sum_{s_i \in Pa(i)} \eta_{(i,s_i)} I_C(s_i) &\geq 1, && \forall i \in N,\ \forall C \subseteq N\ \}. \end{aligned} \tag{3.6.1}$$

This is not a facet description. The authors of [37] observe this description to be only a facet description for the restricted class of spanning trees, which is a straightforward result.

4 Complexity of learning (restricted) Bayesian network structures

4.1 Derivation of complexity results of learning Bayesian network structures

The search for efficient algorithms for learning BN structures comes along with the study of complexity results of some restricted and unrestricted learning tasks. In Section 1.6.1, page 34, we present common substructures together with the corresponding learning tasks and the obtained complexity results. Most of the known complexity results are based on the hardness of the reconstruction of the data, that are necessary to represent a certain structure, restricted or not. Any obtained results are strongly dependent on the assumptions made on the way of representing the data and their structure (etc.). To make our results transparent we start the analysis by clarifying again the necessary assumptions.

Assumptions

A reduction in complexity can be achieved by limiting the possible structures the BN can have. These structures are called restricted. In the following, we restrict our attention to learning *decomposable models*, i.e. learning the best DAG among all DAGs whose pattern are undirected. To be more general, we assume to have given an undirected graph K over N with an edge set $E(K)$, which is not necessarily the complete graph. We wish to learn a DAG G maximising a certain quality criterion and whose pattern is a subgraph of K of a certain restricted type. In particular, we are interested in learning undirected forests and spanning trees with and without degree bounds and in learning undirected chordal graphs, as well.

We only make minimal assumptions on the database D and on the quality criterion Q to be optimised. We assume to have given a *complete* database D over node set N (see Section 1.3, page 15). Additionally, we require the quality criterion to be regular, i.e. *score equivalent* and *decomposable* (see Section 1.4.2, page 22). The representation of the quality criterion Q is in form of a comparison oracle for $Q(\cdot, D)$ such that we do not need to consider D as an explicit part of the input. Given two graphs G_1 and G_2, the comparison oracle for $Q(\cdot, D)$ can decide whether $Q(G_1, D) < Q(G_2, D)$, $Q(G_1, D) > Q(G_2, D)$ and $Q(G_1, D) = Q(G_2, D)$.

4.2 Complexity of learning restricted Bayesian network structures with characteristic imsets

We proceed in an analogous way as in Section 3.4, page 77, and derive the corresponding complexity results for our characteristic imset representation presented in Chapter 2, page 37. The problem of learning the DAG best representing a database therefore becomes the problem of maximising a certain linear functional over the set of possible structures, which is Problem (2.1.5), page 41. Some of the results already coincide with those of Section 1.6,

page 34, but are much easier to derive with characteristic imsets. On the other hand, we strengthen several complexity results.

This chapter is based on our joint papers [33] and [62].

4.2.1 Undirected forests and spanning trees

By Corollary 2.2.3, page 43, we know that every DAG whose essential graph is an undirected forest G has $\binom{\chi(G)}{\mathbf{0}}$ as its characteristic imset. The score of G is given by a linear objective

$$Q(G) := \mathbf{w}^\intercal \mathbf{c}(G)$$

with $\mathbf{w} : E(G) \to \mathbb{R}$. Given a not necessary complete undirected graph K, the problem of learning the best undirected forest is therefore equivalent to finding a maximum weight subgraph G of K that is a forest. The same holds for spanning trees, except that K must be connected. These are two well-known combinatorial problems that can be solved in polynomial time via greedy-type algorithms (see Section 3.4.1, page 77, or e.g. [54]). The input to the two algorithms are the graph K and a linear function $\mathbf{w} : E(K) \to \mathbb{R}$ whose values are provided by the given database D. We conclude the following statement to be true.

Lemma 4.2.1 *Given an undirected graph K and a comparison oracle for $Q(\cdot, D)$, the problems of finding a maximum score subgraph of K that is*

a) a forest,

b) a spanning tree,

can be solved in time polynomial in $|N|$ and in the encoding length of K and the comparison oracle for $Q(\cdot, D)$.

Chow and Liu [15] provide a polynomial procedure in the number of nodes for maximising the maximum log-likelihood criterion which computes an optimal dependence tree (Section 1.6.3, page 35). The core of their algorithm is the greedy algorithm and they apply it to a non-negative objective function only. For their equal result, the complexity of computing the probabilities from data is omitted due to the comparison oracle for $Q(\cdot, D)$ and hence the complexity of computing the objective function as well.

Our result combines this previous result by only supposing a decomposable and score equivalent quality criterion, whereas Chow and Liu use a maximum likelihood criterion. A nice property of undirected forests is that their characteristic vectors define a matroid. From this fact a greedy-type algorithm maximising a linear score function $\mathbf{w}$ over $\hat{\mathrm{P}}_{\mathrm{UF}}(K)$ follows. This greedy algorithm basically coincides with the algorithm used in [15].

Algorithm 4.2.2 (greedy algorithm)
Input: *A set of nodes N, an undirected graph K and a linear function $\mathbf{w} : E(K) \to \mathbb{R}$.*
Output: *A forest $F \subseteq K$ of maximum score $Q(F) = \mathbf{w}^\intercal \mathbf{c}(F)$.*

1. *Sort $E(K) := \{s_1, s_2, \ldots, s_{|E(K)|}\}$ according to $w_{s_1} \geq w_{s_2} \geq \ldots \geq w_{s_{|E(K)|}}$.*
2. *Set $F := \emptyset$.*
3. *For $i = 1$ to $|E(K)|$ do*

   If $F \cup \{s_i\}$ is a forest, set $F := F \cup \{s_i\}$.

4. Return F.

Note this greedy algorithm to run in time polynomial in $|N|$ and in the number of calls to a comparison oracle for the score function $Q(\cdot, D)$. If the comparison oracle is queried on DAGs T_1 and T_2, then it decides whether $\mathbf{w}^\intercal\mathbf{c}(T_1) < \mathbf{w}^\intercal\mathbf{c}(T_2)$, $\mathbf{w}^\intercal\mathbf{c}(T_1) = \mathbf{w}^\intercal\mathbf{c}(T_2)$, or $\mathbf{w}^\intercal\mathbf{c}(T_1) > \mathbf{w}^\intercal\mathbf{c}(T_2)$. This greedy-type algorithm can also be used to prove similar properties of the GES algorithm (compare with Section 1.4.2.1, page 23).

Corollary 4.2.3 *The GES algorithm always finds an undirected forest of maximum score in time polynomial in $|N|$ and in the number of calls to a comparison oracle for $Q(\cdot, D)$, as well.*

Proof. To see this, simply observe that the first phase of the GES algorithm adds edges greedily and obtains an optimal solution. As the solution is optimal due to the result above, it then never removes an edge in the second phase. □

The set of spanning trees is the base system of the matroid of forests and, thereby, there exists a greedy-type algorithm maximising a score function $\mathbf{w}(\cdot)$ over $\hat{\mathrm{P}}_{\mathrm{UST}}(K)$. Chow and Liu suppose the score function to be non-negative and with this every forest of maximum score to be a spanning tree. As we only suppose a decomposable and score-equivalent score function we only need to reformulate Algorithm (4.2.2), page 90, in order to obtain spanning trees.

Algorithm 4.2.4 (greedy swapping algorithm)
Input: *A set of nodes N, an undirected connected graph K and a linear score function $Q(\cdot, D) = \mathbf{w}^\intercal\mathbf{c}(\cdot)$ given by $\mathbf{w} : E(K) \to \mathbb{R}$.*
Output: *A spanning tree $T \subseteq K$ of maximum score $Q(T)$.*

1. *Choose any spanning tree T of K.*
2. *While there are edges $e \in K \setminus T$, $f \in T$ such that $T' := T \setminus \{f\} \cup \{e\}$ is a spanning tree of K with $Q(T') > Q(T)$*

   choose T' among the spanning trees that maximises $Q(T')$ and set $T := T'$.
3. *Return T.*

Note this greedy algorithm to run in time polynomial in $|N|$ and in the number of calls to a comparison oracle for the score function $Q(\cdot, D)$, as well.

This enables us to interpret the classic procedure for learning dependence trees ([15] and Section 1.5.2, page 32): It can be interpreted as learning an undirected forest by maximisation of a so-called maximised log-likelihood criterion which is a special non-negative decomposable and score equivalent quality criterion of the edges over a complete graph K.

4.2.2 Undirected forests and spanning trees with degree bounds

Problems of learning undirected forests and of learning undirected spanning trees, respectively, are solvable in polynomial time and the incidence vectors of edges suffice for representation (see Section 3.4.3, page 78). In contrast to this, learning an undirected forest/spanning tree with a given degree bound $d(i) \leq k$, $\forall i \in N$, is $\mathcal{NP}$-hard for $|N| - 1 > k \geq 2$. For $k = 1$ this problem is equal to the well-known problem of finding a maximum weight matching in K, which is in the general case polynomially solvable in $|N|$ (see Chapter 30 in [54]).

Meek [46] has shown the following result for the case of $k = 2$ (see Section 1.6.5, page 35). We generalise this to any k with $|N| - 1 > k \geq 2$.

Theorem 4.2.5 *Given an undirected graph K and a comparison oracle for $Q(\cdot, D)$, the problems of finding a maximum score subgraph of K that is*

(a) a forest,

(b) a spanning tree,

and fulfils the degree bounds $d(i) \leq k$, $\forall i \in N$, is $\mathcal{NP}$-hard in the encoding length of K and the comparison oracle for $Q(\cdot, D)$.

Proof. We deduce part (b) from the following feasibility problem. In Section 3.2.1 of [22] is shown that the following decision problem:

Bounded degree spanning tree
Instance: An undirected graph K and a constant $2 \leq k < |N| - 1$.
Question: "Is there a spanning tree for K in which no node has degree exceeding k?"

is $\mathcal{NP}$-complete by reduction onto the Hamiltonian path problem.

Part (a) now follows by considering the subfamily of problems in which the linear objective takes only strictly positive values. Then every optimal forest with the bounded degree is a spanning tree. The problem of finding a maximum-weight forest with a given degree bound is now equivalent to finding a maximum-weight spanning tree with a given degree bound. As the feasibility problem for the latter is already $\mathcal{NP}$-complete for a special objective function, part (a) follows for general objective functions. □

As the set of bounded forests is only an independence system but not a matroid the greedy algorithm 4.2.2 results in a locally optimal solution only and so the GES, too. However, this locally optimal solution provides an approximation quality. The minimum obtainable approximation factor, given by Theorem 13.19 in [41], is

$$\min_{F \subseteq K} \frac{r_U(F)}{r_M(F)} \geq \frac{1}{|N|},$$

based on the minimum cardinality of a base of a set F, $r_U(F)$ given by an independent system U, and the maximum cardinality, $r_M(F)$ given by a matroid M. Of course, this greedy algorithm runs in time polynomial in $|N|$ and in the number of calls to a comparison oracle for the score function $Q(\cdot, D)$, that, when queried on DAGs T_1 and T_2, decides whether $\mathbf{w}^\intercal \mathbf{c}(T_1) < \mathbf{w}^\intercal \mathbf{c}(T_2)$, $\mathbf{w}^\intercal \mathbf{c}(T_1) = \mathbf{w}^\intercal \mathbf{c}(T_2)$, or $\mathbf{w}^\intercal \mathbf{c}(T_1) > \mathbf{w}^\intercal \mathbf{c}(T_2)$ (see Section 3.4.2, page 78).

Corollary 4.2.6 *The GES algorithm always finds an undirected degree bounded forest of at least $1/|N|$ of the maximum score in time polynomial in $|N|$ and in the number of calls to a comparison oracle for $Q(\cdot, D)$.*

Again, we define the Greedy swapping algorithm 4.2.4 in the degree bounded spanning tree case. Likewise, we observe that this greedy algorithm runs in time polynomial in $|N|$ and in the number of calls to a comparison oracle for the score function $Q(\cdot, D)$.

4.2.3 Chordal graphs

In this section, we show learning of general decomposable models to be $\mathcal{NP}$-hard.

Theorem 4.2.7 *Given an undirected graph K and a complete database D, the problem of finding a maximum score chordal subgraph of K is $\mathcal{NP}$-hard in the encoding length of K and D.*

Proof. We show the following $\mathcal{NP}$-hard problem to be polynomially transformable to learning undirected chordal graphs.

Clique of given size
Instance: An undirected graph K and a constant $2 \leq k \leq |N| - 1$.
Question: "Is there a clique in K of size at least k?"

We define now a suitable learning problem that would solve this problem. By Corollary 2.2.2, page 42, we know for any chordal graph G the entry $\mathbf{c}_T(G)$ to be one if and only if $T \subseteq N$, $|T| \geq 2$, is a clique, otherwise this entry is zero. Hence, the score of G is determined by the values of the objective for the cliques T in G. In particular, we can define the values for the cliques in such a way, that, when transforming the learning problem to the problem of maximising a linear functional $\mathbf{w}^\intercal \mathbf{x}$ over the characteristic imset polytope, the entries $\mathbf{w}_T$ are zero if $|T| < k$ and are strictly positive if $|T| \geq k$. The definition of $\mathbf{w}$ implies the maximum score among all chordal subgraphs of K to be strictly positive if and only if there exists a chordal graph containing a clique T of size $|T| \geq k$. On the other hand, such a chordal graph exists if and only if K has a clique of this size, which is the task mentioned above.

The learning problem for a general class of decomposable and score equivalent quality criteria is at least as hard as for one score function among them. Showing the hardness result for at least one particular function yields the complete class to be at least as hard as the special instance. This special instance does not imply all instances for all possible score functions to be equally hard. For other specific decomposable and score equivalent score functions the learning problem still may be polynomial time solvable. A straight forward question is to find such score functions leading to polynomial time solvable problems. □

4.2.4 Chordal graphs with a bounded size of cliques

Let us consider a variation of the previous task by the introduction of an upper bound k on the size of cliques introduced by Srebro [56] (see Section 1.6.2, page 35). If $k \leq 2$, we get the problems of learning undirected forests/matchings, which we already know to be solvable in polynomial time (compare with Sections 4.2.1 and 4.2.2). Srebro has shown the problem of learning decomposable models to be $\mathcal{NP}$-hard for a fixed bound on the size of the cliques and a maximised log-likelihood criterion. This implies also the unbounded case to be $\mathcal{NP}$-hard as we can choose the bound on the cliques to be $|N|$.

The reason for the learning of chordal graphs to be an $\mathcal{NP}$-hard problem is based on the construction of a specific decomposable and score equivalent objective function. The same reason applies in the case of chordal graphs with a bound on the maximum clique size. The special score function which Srebro [56] used to show his hardness result implies the complete class of decomposable and score equivalent quality criteria to result in $\mathcal{NP}$-hard problems. In spite of this, we believe this hardness result to be also valid and provable for our approach.

5 Computations of learning (restricted) Bayesian network structures

5.1 Learning (restricted) Bayesian network structures with polyhedral approaches

Chapters 2, page 37 and 3, page 51, introduce a new combinatorial representative of Bayesian network structures and the derivation of theoretical results. Moreover, the hardness results presented in Chapter 4, page 89, are based on a theoretical analysis of the restricted learning tasks. However, using state-of-the-art software it may be possible to turn this theoretic approach into a practical one and solve Problem (2.1.5), page 41, which is

$$\begin{aligned} \max \quad & s^Q_\emptyset(D) + \langle \mathbf{r}^Q(D), \mathbf{c}(G) \rangle \\ \text{s.t.} \quad & \mathbf{c}(G) \in \mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}, \end{aligned} \tag{5.1.1}$$

for moderate problem sizes and different restricted structures. Values $s^Q_\emptyset(D)$ and $\mathbf{r}^Q(D)$, $\forall D \in \mathrm{DATA}(N, d)$, are given by the data. Value $s^Q_\emptyset(D)$ is a constant and is skipped in the computation. $\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}$ is a LP-relaxation of the characteristic imset polytope of possibly restricted structures given by a set of inequalities (see Chapter 3, page 51).

Although Problem (5.1.1) contains an amount of both variables and inequalities exponential in $|N|$, additional structural information can easily be incorporated into the inequality description. These additional structural information yield learning problems of restricted structures. As shown in Section 3.4, page 77, the inequality descriptions of restricted structures may lead to more compact formulations reducing the overall computational effort. Enumerative approaches as presented in Section 1.4.2, page 22, cannot exploit any additional structural restriction in the same way and their memory storage remains high. Moreover, the search still must be performed for all structures and only once a valid DAG is found it can be excluded from consideration due to these additional structural restrictions.

In spite of any benefits from restricted structures, the exponential representation, a description of $\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}$ and the computation of $\mathbf{r}^Q(D)$ still requires an extensive analysis for a possible reduction. We introduce two approaches for obtaining reduced descriptions of $\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}$ and $\mathbf{r}^Q(D)$ in Section 5.3.1, page 98. The first approach makes use of the objective function and of the underlying graphical structures. Applying this approach to characteristic imsets leads to fewer variables needed to describe $\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}$, which also helps to reduce the amount of inequalities of the presented LP-relaxations in Chapter 3. The second approach introduces fundamental simplifications of the structures and hence of the polytope with respect to restricted learning. Such a simplification of the structure is a bound on the number of possible parents of each node.

Additionally, methods for affecting the actual performance of solving the IP (5.1.1) are introduced in Section 5.3.2, page 102. These methods are similar to branch-and-cut approaches.

We close the analysis of the description of Problem (5.1.1) with a presentation of related work. Jaakkola et al. provide a similar approach for learning unrestricted BN structures, which is

based on a column generation technique and a branch and bound method which seems to be appropriate for various real life instances. We present a sketch of their algorithm.

For a repetition of fundamentals of branch and bound algorithms combined with column or row generation, we present an overview of the respective concepts in Section 5.2, this page. We introduce both the branch-and-cut based approach that we use and the column generation method that Jaakkola et al. is based on in this section.

5.2 Fundamentals of branch and bound with column and row generation

Branch and bound approaches for integer programming can be combined with techniques for solving large linear programs. Such techniques are row and column generation which are dual approaches. We introduce both and their possible combinations with branch and bound. For this purpose we follow the overviews given in [4] and [20].

5.2.1 Column generation

Let the following linear program be given:

$$\begin{aligned} \min \quad & \mathbf{c}^\intercal \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b}, \\ & \mathbf{x} \in \mathbb{R}_+^{|J|}. \end{aligned} \tag{5.2.1}$$

where J represents variables with x_j, $j \in J$, of Problem (5.2.1) above. Linear programming techniques like the Simplex-method cannot be applied to solve this LP as we assume the dimension $|J|$, and hence the number of columns of $\mathbf{A}$, to be too large. Column generation is a technique starting with a smaller version of the problem above restricted to a subset $\tilde{J} \subseteq J$ of the columns, i.e. variables x_j, $j \in \tilde{J}$, and solving this:

$$\begin{aligned} \min \quad & \tilde{\mathbf{c}}^\intercal \mathbf{x} \\ \text{s.t.} \quad & \tilde{\mathbf{A}}\mathbf{x} \leq \mathbf{b}, \\ & \mathbf{x} \in \mathbb{R}_+^{|\tilde{J}|}. \end{aligned} \tag{5.2.2}$$

As the columns of Problem (5.2.1) are restricted to those corresponding to $\tilde{J}$ the new objective function is $\tilde{\mathbf{c}}$ and the new coefficient matrix is $\tilde{\mathbf{A}}$. Any feasible solution $\mathbf{x}^{\tilde{J}}$ of Problem (5.2.2) above is also a feasible solution $\mathbf{x}$ of Problem (5.2.1) if the remaining variables x_j, $j \in J \setminus \tilde{J}$, are set to zero. Let λ be the dual solution of the primal optimal solution $\mathbf{x}^{\tilde{J}}$ of Problem (5.2.2) and let $\mathbf{x}$ be the corresponding feasible solution to (5.2.1). The dual solution λ can be used to compute the reduced costs of variables x_j, $j \in J$, of $\mathbf{x}$. If there is a variable $x_{j'}$, $j' \in J$, with a negative reduced cost then the corresponding column is added to the restricted problem with $\tilde{J} = \tilde{J} \cup \{j'\}$ as it decreases the costs obtained by Problem (5.2.2). This computation of reduced costs v is itself a program to solve:

$$v := \min_{y \in Y} \{c_y - \lambda A_{\cdot y}\}, \tag{5.2.3}$$

where $A_{\cdot y}$ denotes the y-th column of A. The set Y of objects y corresponds to a selection of columns of A and the respective entries of c. In practical application columns Y of A typically represent objects y like paths in graphs or subgraphs or assignments. An implicit treatment of columns Y in terms of the mentioned objects is often simpler in practice than

dealing with the corresponding variables in Problem (5.2.1) explicitly. If $v < 0$, then the primal variable x_j and column j of $\mathbf{c}$ and $\mathbf{A}$, obtained from the solution $j := y^*$ attaining the minimum of Problem (5.2.3) are added to the Problem (5.2.2). If, contrarily, $v \geq 0$ then no additional variable $x_j \neq 0$, $j \in J \setminus \tilde{J}$, added to $\mathbf{x}^{\tilde{J}}$ reduces the current objective value of $\mathbf{x}^{\tilde{J}}$. The feasible solution $\mathbf{x}^{\tilde{J}}$ with $x_j = 0$, $\forall j \in J \setminus \tilde{J}$, is therefore optimal for Problem (5.2.1) as well.

The primal Problem (5.2.1) is called the **master problem**, the restricted Problem (5.2.2) is called the **restricted master problem** and the Problem (5.2.3), to compute the reduced costs and the next column to enter, is called the **pricing problem**.

Branch-and-price and price-and-branch The above column generation solves linear problems to optimality. Several methods generalise this concept to linear integer programming. A common approach is the combination of branch and bound and column generation. Two possibilities occur in how to combine both steps.

The first approach uses the column generation to solve the LP in each node of the branch and bound tree. The branch and bound itself is not affected by the column generation and as both methods are obtaining optimal solutions independently so does the combination of both. This way of combining both methods is called **branch-and-price**.

A second way to combine branch and bound with column generation is to relax the given IP and to solve the obtained LP using column generation. The derived restricted master program together with the integer conditions is solved by using branch and bound. The restricted master program contains the best choice of non-zero variables for the linear program but this choice does not need to be the best choice of variables for the integer case. The lack of a certificate for optimality causes this type of combining column generation and branch and bound to yield only a heuristic. To distinguish the sequence of applying both techniques, this method is called **price-and-branch**.

5.2.2 Row generation

A dual approach to column generation leads to the addition of inequalities instead of variables (columns). We assume Problem (5.2.1) to contain too many variables to be solved directly. The dual problem to (5.2.1) contains an inequality for each column of the primal problem and hence $|J|$ many. As the primal problem contains too many columns the dual problem contains too many inequalities. The inclusion of variables by means of column generation is equal to an inclusion of inequalities to the dual problem.

If an Optimisation problem (5.2.1) contains, on the other hand, too many inequalities then an iterative addition of inequalities starting with a subset of inequalities is appropriate. The optimisation problem is defined for a restricted subset of its inequalities only and an optimal solution is computed. This solution is feasible for the overall optimisation problem if it satisfies also the remaining inequalities. If there is an inequality among the remaining ones which is not fulfilled for the current solution this inequality is added to the restricted system. An optimal solution to the new system is computed and the iterative addition and optimisation is performed until the optimal solution is feasible for all inequalities. The inequalities to add cut off the current optimal solution, from which the type of procedure of the row generation comes from: A **cutting plane procedure**.

Likewise to the column generation, an iterative addition of implicit inequalities performs often better than the explicit definition in an inequality system to solve. This benefit comes first from the storage of the inequalities. For an iterative procedure like row generation such

inequalities are often created at the moment they are tested to be fulfilled. Second, the current optimal solution has similar properties like the columns of the coefficient matrix in the column generation, due to duality of the approaches. For example, graphical procedures on these optimal solutions/objects can give quick certificates on their validity or invalidity.

Branch-and-cut A combination of branch and bound and row generation is called **branch-and-cut**. In each solution of the LP in the branching step additional and preferably facet defining inequalities are added to cut off the current non-integer solution. In practice a set of additional inequalities is used as a pool of inequalities to be added in the cutting step. The generation of facet defining inequalities is a topic of research on its own. As we use row generation not in terms of facet defining inequalities we refer the interested reader to the corresponding literature, e.g. Chapter 23 in [53], on cutting planes, rounding cuts, Gomory cuts, etc. for more information.

5.3 Towards a compact description of the characteristic imset polytope

We do not have a facet description or a LP-relaxation of $\mathrm{P}_{\mathbf{c}}$ yet and thus an application of pure linear programming may lead into fractional solutions of Problem (5.1.1), page 95. Instead, we use a possibly weak LP-relaxation $\mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}}$ of an extension $\mathrm{P}_{\mathrm{ext}}$ of $\mathrm{P}_{\mathbf{c}}$ with $\mathrm{P}_{\mathrm{ext}}\,|_{\mathbf{c}} \cap \mathbb{Z}^t = \mathrm{P}_{\mathbf{c}} \cap \mathbb{Z}^t$, $t = 2^{|N|} - |N| - 1$, and simply derivable inequalities and try to solve Problem (5.1.1) with it, i.e. Problem (3.2.10), page 60, with $(\mathbf{0}, \mathbf{w}) = (\mathbf{0}, \mathbf{r}^Q(D))$. Of course an extensive analysis of the underlying IP can help to improve the description of $\mathrm{P}_{\mathrm{ext}}^{\mathrm{rel}}$ by a reduction of the number of variables and inequalities.

5.3.1 Methods for improving the description

All of the inequality descriptions in Chapter 3, page 51, are of exponential size, as they are defined over nearly all subsets of N. Even for moderate problem sizes this fact causes an IP solver to store and to handle this amount of inequalities. Even doing this in a sparse representation means to store/handle more than $2^{|N|}$ rows simultaneously! Even worse, the same problem occurs with the variables of the problems.

Anyway, for handling problems of size $|N| = 29$, like for dynamic programming in Section 1.4.2.2, page 24, general simplifications of Problems (5.1.1) and (3.2.10), respectively, must be imposed.

5.3.1.1 Pruning

An idea for improving the description is **pruning**. De Campos et al. [17] and Jaakkola et al. [37] introduce a method for reducing the amount of variables based on an exploitation of the decomposable scores into local scores of parent sets. If parent sets are dominated by other parent sets with respect to their scores they are called pruned and the corresponding variables of the parent sets are set to zero. The corresponding variables can be removed from the Description (3.6.1), page 88, of Problem 2.4.2, page 49:

$$\begin{aligned} \max \quad & \sum_{i=1}^{|N|} \sum_{s_i \in Pa(i)} \eta_{(i,s_i)} Q_i(s_i) \\ \text{s. t.} \quad & \eta \in \mathcal{P}_{cluster}. \end{aligned} \tag{5.3.1}$$

We first introduce the concept of pruning with respect to the representation of Jaakkola et al. and de Campos et al. (compare with Sections 2.4.1, page 49, 3.6, page 88, and 1.4.2, page 22). Then, we modify the concept of pruned parent sets in terms of an application for Problem (5.1.1) and characteristic imsets.

Lemma 5.3.1 (compare with [37] and Lemma 1 in [17]) *Let Q be a score equivalent and decomposable quality criterion and G be a best DAG over N with respect to Q. Moreover, let $i \in N$ be a node and $Pa(i)$ the set of possible parent sets of i. If for $T \in Pa(i)$ there exists a set $X \subsetneq T$ with $Q_i(X) \geq Q_i(T)$, then we can set $\eta_{(i,T)} = 0$ in an optimal DAG G.*

Proof. Let $\tilde{G}$ be a DAG with $\eta_{(i,T)} = 1$. Then, a graph G equal to $\tilde{G}$ but with $\eta_{(i,T)} = 0$ is a DAG, as well, as it has fewer arcs than DAG $\tilde{G}$. Moreover, G has a better objective value than $\tilde{G}$ as $Q_i(X) \geq Q_i(T)$, for at least the local score of node i. The choice of the parents for the remaining nodes in G and $\tilde{G}$ is independent from i and X due to the decomposable score function Q. □

Lemma 5.3.2 *Let Q be a score equivalent and decomposable quality criterion and G be a best DAG over N with respect to Q. If for all nodes $i \in T \subseteq N$ of a given subset T there exists a set $X \subsetneq T \setminus \{i\}$ with $Q_i(X) \geq Q_i(T \setminus \{i\})$, then we can set $c_T(G) = 0$ in an optimal DAG G.*

Proof. Let $\tilde{G}$ be a DAG with $c_T = 1$. Then, a graph G equal to $\tilde{G}$ but with $c_T = 0$ is a DAG, as well, as it has fewer arcs than DAG $\tilde{G}$. Moreover, G has a better objective value than $\tilde{G}$ as $Q(G, D) \geq Q(\tilde{G}, D)$: According to Equation (2.1.4), page 40, the corresponding score is

$$\begin{aligned} Q(\tilde{G}, D) = s_\emptyset(D) + \langle \mathbf{r}^Q(D), \mathbf{c}(\tilde{G}) \rangle &= s_\emptyset(D) + \sum_{A \subseteq N, |A| \geq 2} c_A(\tilde{G}) \cdot r_A^Q(D) \\ &= s_\emptyset(D) + \sum_{A \subseteq T, |A| \geq 2} c_A(\tilde{G}) \cdot r_A^Q(D) + \sum_{A \subseteq N, |A| \geq 2, A \not\subseteq T} c_A(\tilde{G}) \cdot r_A^Q(D) \end{aligned}$$

Evaluating local scores $Q_i(X)$ and $Q_i(T)$ yields

$$\begin{aligned} s_\emptyset(D) + \sum_{A \subseteq X, |A| \geq 2} c_{A \cup \{i\}} \cdot r_{A \cup \{i\}}^Q(D) &= Q_i(X) \\ \geq Q_i(T \setminus \{i\}) = s_\emptyset(D) + \sum_{\substack{A \subseteq T, |A| \geq 2, \\ i \in A}} c_A \cdot r_A^Q(D), \ \forall i \in T. \end{aligned}$$

Note that there might exist different sets X for different terminal nodes i. It is only important to have at least one better parent set for one i. We have

$$\begin{aligned} Q(\tilde{G}, D) = s_\emptyset(D) &+ \sum_{\substack{A \subseteq T, |A| \geq 2, \\ i \in A}} c_A(\tilde{G}) \cdot r_A^Q(D) + \sum_{\substack{A \subseteq T, |A| \geq 2, \\ i \notin A}} c_A(\tilde{G}) \cdot r_A^Q(D) \\ &+ \sum_{\substack{A \subseteq N, |A| \geq 2, \\ A \not\subseteq T}} c_A(\tilde{G}) \cdot r_A^Q(D) \\ = Q_i(T \setminus \{i\}) &+ \sum_{\substack{A \subseteq T, |A| \geq 2, \\ i \notin A}} c_A(\tilde{G}) \cdot r_A^Q(D) + \sum_{\substack{A \subseteq N, |A| \geq 2, \\ A \not\subseteq T}} c_A(\tilde{G}) \cdot r_A^Q(D), \ \forall i \in T \\ \leq Q_i(X) &+ \sum_{\substack{A \subseteq T, |A| \geq 2, \\ i \notin A}} c_A(\tilde{G}) \cdot r_A^Q(D) + \sum_{\substack{A \subseteq N, |A| \geq 2, \\ A \not\subseteq T}} c_A(\tilde{G}) \cdot r_A^Q(D), \ \forall i \in T \end{aligned}$$

$$= s_\emptyset(D) + \sum_{A\subseteq X,|A|\geq 2} c_{A\cup\{i\}}(G)\cdot r^Q_{A\cup\{i\}}(D) + \sum_{\substack{A\subseteq T,|A|\geq 2,\\ i\notin A}} c_A(\tilde{G})\cdot r^Q_A(D)$$
$$+ \sum_{\substack{A\subseteq N,|A|\geq 2,\\ A\not\subseteq T}} c_A(\tilde{G})\cdot r^Q_A(D) \quad =: Q(G,D),\ \forall i\in T.$$

Entries $c_{A\cup\{i\}}$ with $A\subseteq X$ are independent from c_A with $A\not\subseteq T$. We conclude G to be a valid DAG having at least the same score as $\tilde{G}$. □

This result shows pruning to be applicable also for characteristic imsets and Problem (5.1.1). In particular, pruning is an operation done when computing the object function $\mathbf{r}^Q(D)$. But for the respective entries of $\mathbf{r}^Q(D)$ all subsets, also the pruned ones, need to be considered. With Lemma 2.1.6, page 40, and its proof we conclude

$$Q(G,D) = s_\emptyset(D) + \sum_{A\subseteq N,|A|\geq 2} c_A(G)\cdot r^Q_A(D).$$

If G is a graph having only arcs from parents X to a node i, then its score is the local score of node i and X. Therefore,

$$Q(G,D) = Q_i(X) = s_\emptyset(D) + \sum_{A\subseteq X,|A|\geq 2} 1\cdot r^Q_{A\cup\{i\}}(D).$$

We use scores of subgraphs G' with $Q(G',D) = Q_i(X')$ and $X' = X\setminus\{j\}$, for a node $j\in X$, to compute the coefficient of the objective function $r^Q_{X\cup\{i\}}(D)$ corresponding to $c_{X\cup\{i\}} = 1$:

$$r^Q_{X\cup\{i\}}(D) = Q(G,D) - s_\emptyset(D) - \sum_{A\subsetneq X,|A|\geq 2} r^Q_{A\cup\{i\}}(D)$$
$$= Q(G,D) - Q(G',D) \ = \ Q_i(X) - Q_i(X').$$

Even if there exist sets $X'\cup\{i\}\subseteq X\cup\{i\}$ which are pruned, then the respective objective coefficients $r^Q_{X'\cup\{i\}}(D)$ must be considered in the above formula. A negative effect is that pruning does not limit the overall computation of $\mathbf{r}^Q(D)$ but only the needed memory usage of the remaining non-pruned sets.

We summarise the pruning algorithm to compute the set of non-pruned subsets $\mathcal{T}$ and the corresponding variables $\mathbf{c}_T$, $T\in\mathcal{T}\subseteq\mathcal{P}(N)_{\geq 2}$, with the following basic steps:

1. set $\mathcal{T} = \mathcal{P}(N)_{\geq 2}$
2. for all sets $T\in\mathcal{T}$ do:
   a) set prune=false
   b) for all $i\in T$ do:
      i. compute $Q_i(T\setminus\{i\})$
      ii. for all $X\subsetneq T\setminus\{i\}$ with $|X|\geq 2$ do:
         A. compute $Q_i(X)$
         B. if $Q_i(X)\geq Q_i(T\setminus\{i\})$, set prune=true and continue at step b) with next $i\in T$
      iii. if prune=false, return T as non-pruned and continue at step 1. with next $T\in\mathcal{T}$

c) return T as pruned and set $\mathcal{T} := \mathcal{T} \setminus T$

3. return c_T, $\forall T \in \mathcal{T}$, as variables for Problem (5.1.1)

This procedure can be sped up using several strategies of searching the sets T and X. For example it is useful to order sets T and X by inclusion, because the computed scores of non-pruned subsets can then be used to compute the current score without unnecessary recalculation. Moreover, sets of cardinality two are never pruned and are a good choice for sets X to prune sets T.

The hope for an application of such techniques comes from observations made in practice. Often sparse networks containing less dependencies are occurring. Moreover they have higher scores as most of the score functions are designed to punish the use of many arcs/edges in a BN. Furthermore a reduction of the amount of parents does not change the acyclicity of the graph and therefore always yields a valid solution.

5.3.1.2 Additional bounds on the number of parents

In contrast to most (integer) LP applications the computation of the objective function and their coefficients is itself already a time-consuming task in Problem (5.1.1). To get the objective function coefficients $r_T^Q(D)$ in a reasonable amount of time, initial bounds on the amount of possible parents of nodes need to be imposed. A second reason for imposing bounds is that pruning is not applicable for every restricted structure.

Example 28. Consider a chordal graph given in Figure 5.1 below. Let sets $T := \{A, B, C\}$ and $\tilde{X} := \{A, D, E\}$ be given.

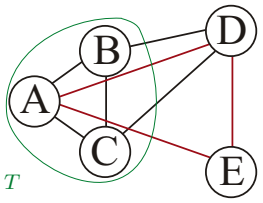

Figure 5.1: A chordal graph G in which pruning is not applicable.

Let set $T \cup \{D\}$ be pruned, because for nodes D and A the set $\{B, C\}$ is a better parent set, for node C sets $\{A, B\}$ or $\{B, D\}$ are better parent sets and $\{A, C\}$ or $\{C, D\}$ for node B. The variable $c_{T\cup\{D\}}$ is then zero. Provided all edges in T are existing the only possibility for $c_{T\cup\{D\}} = 0$ is the edge $\{A, D\}$ to be non-existing and $c_{\{A,D\}} = 0$. Therefore $c_{\tilde{X}} = 0$, as well, as $\tilde{X}$ is no clique anymore. Node E is not allowed to have parents $\{A, D\}$ which may not be valid with respect to the objective function and the optimal solution. Further choices of parents are not independent and so pruning is not possible. ⋆

A possibility to reduce the complexity of the problem, especially for chordal graphs, is to introduce bounds on the number of parents. This technique cannot be used in case of chordal graphs only, but also for other restricted and unrestricted structures. Bounding the number of parents is equivalent to bounding the maximal size of the sets T corresponding to entries c_T.

Corollary 5.3.3 *Let G be a DAG over N and $k \in \mathbb{N}$ with $1 \leq k \leq |N| - 1$, then $|\operatorname{pa}(i)| \leq k$, $\forall i \in N$, if and only if $c_T(G) = 0$, for all $T \subseteq N$ with $|T| > k + 1$, in G.*

Proof. This follows directly from the fact, that in a set of nodes and parents there exists a direction of arcs such that there is a node $i \in T$ with $j \to i$, for all $j \in T \setminus \{i\}$. Bounding the amount of parents of i is now equivalent to bounding the cardinality of T in $c_T(G)$. □

Another possibility to incorporate bounds on the number of parents is to add inequalities to the description of (5.1.1). In Section 3.4.5, page 84, we argue this to be an inappropriate way as the number of inequalities is increased. Even worse the imposed simplification of the problem does not lead to a real reduction of the description. Therefore, we always introduce bounds on the number of parents by considering $c_T = 0$ for the remaining sets T.

5.3.2 Row and column generation

Row generation can be applied to tighten the formulation of an problem. We use this method in a similar way as the branch-and-cut approach presented in Section 5.2.2, page 97. We start to solve a reduced description of Problem (5.1.1), with e.g. branch and bound, to get an optimal integer solution. This integer solution is tested to satisfy also the remaining inequalities. Only those additional inequalities which the current solution not fulfils are needed and added to the description of (5.1.1). A lot of inequalities are not necessary to define the integer solution of Problem (5.1.1). In a LP only dimensional many inequalities (i.e. facets) are necessary to define the optimal solution. We could obtain a tremendous reduction of the description of Problem (5.1.1) if this description contains only the needed inequalities.

The difference between our row generation based procedure and the branch-and-cut approach is that we start in a reduced problem description and check each computed integer solution against the remaining set of inequalities. Only those inequalities among the remaining ones are added which are cutting off the current integer solution of the reduced problem. In each step of the row generation a complete IP solution algorithm, e.g. branch-and-bound, is performed. To provide correctness of the obtained integer solutions we have to check all inequalities. As the variables of Problem (2.1.5), page 41, correspond to DAGs of (restricted) BNs we can rather check the correctness by validation of the directed graph induced by the current integer solution than by satisfiability of the remaining inequality. The advantage is that we can exploit quick graphical procedures and we do not need to store the remaining inequalities explicitly.

Unfortunately, the number of variables necessary to describe Problem (5.1.1) are exponential in $|N|$, too. Of course, column generation techniques can be applied, but a reason for not considering column generation on Problem (5.1.1) for these computations is pruning. With pruning the number of variables for typical real life instances is reduced tremendously.

5.4 Related work

Based on the inequality description derived in [37], Jaakkola et al. are computing optimal BN structures of several presented datasets. For this purpose they present an algorithm similar to the column generation approach (see Section 5.2.1, page 96).

5.4.1 Learning Bayesian network structures with Jaakkola et al. [37]

The authors of [37] are faced with the same problem from which also other descriptions using a linear decomposable quality criterion suffer. The Inequality description (3.6.1), page 88, contains a number of inequalities and variables exponential in $|N|$. To deal with this,

Jaakkola et al. first compute the dual of the corresponding relaxed problem of maximising a quality criterion over the Polytope (3.6.1). Then this dual problem is reformulated to obtain a shorter but non-linear description:

$$\begin{aligned} \min \quad & \sum_{i \in N} \max_{s_i \in Pa(i)} \left\{ Q_i(s_i) + \sum_{i \in C : C \subseteq N} \lambda_C I_C(s_i) \right\} - \sum_{C \subseteq N} \lambda_C \\ \text{s.t.} \quad & \lambda_C \geq 0, \quad \forall C \subseteq N. \end{aligned} \tag{5.4.1}$$

The advantage of this dual formulation is that is has only non-negativity constraints. Additionally, although the objective function is non-linear it can be solved directly by analysing its structure. In terms of column generation this is the master problem. However, Problem (5.4.1) above contains a prohibitively large amount of variables. The corresponding restricted master problem is defined to contain a set of cluster variables λ_C with $C \in \mathcal{C}$ only. The set $\mathcal{C}$ is successively increased in a procedure based on a pricing problem. In each step of the solution of the restricted master problem a DAG corresponding to the primal variables is obtained. If its objective value is equal to the dual objective value, then it is the optimal structure of the primal Problem (5.3.1), page 98, due to duality. If the objective values are not equal, then other cluster variables are added to the restricted master problem according to their gain obtained in the pricing problem. If none of the obtained DAGs has been optimal, then the dual objective of the restricted master problem is used as an upper bound for a branch and bound procedure.

The advantage of the column generation based method is to provide good solutions to the structure learning problem quickly. Moreover, their branch and bound procedure always terminates with an optimal solution, but its performance strongly depends on the size of the problem.

However, this approach and its implementation has drawbacks. First and most prominent, the pricing problem is not solved to optimality. Clusters are only added up to a certain amount and clusters to add are only added in a greedy way. Second, for the implementation of their approach the authors do not use state-of-the-art software to solve linear and integer programs occurring during the column generation and the branch and bound method. This lack of use of contemporary software leaves room for practical improvements.

We summarise the major steps of the procedure presented in [37].

Algorithm 5.4.1 (Column generation based procedure)
Column generation:

1. ***Initialisation:*** *set $\lambda_C = 0$, $\forall C \subseteq N$, i.e. consider the empty set of clusters $\mathcal{C} = \emptyset$ in the restricted master problem:*

$$\begin{aligned} \min \quad & \sum_{i \in N} \max_{s_i \in Pa(i)} \left\{ Q_i(s_i) + \sum_{i \in C : C \in \mathcal{C}} \lambda_C I_C(s_i) \right\} - \sum_{C \in \mathcal{C}} \lambda_C \\ \text{s.t.} \quad & \lambda_C \geq 0, \quad \forall C \in \mathcal{C}. \end{aligned} \tag{5.4.2}$$

2. *Update λ_C for all clusters $C \in \mathcal{C}$ by solving the restricted master problem for $\mathcal{C}$. Variables λ_C for all clusters $C \in \mathcal{C}$ are updated by analysing the gain in objective. This gain results in a piecewise linear function, which is solved by coordinate descent.*

3. *Decode a DAG from the optimal solution λ_C^* of the restricted Master problem (5.4.2) above. If its score is equal to the dual score, then it is an optimal structure. If not, then proceed.*

The variables λ^*_C are encoding clusters which, on the other hand, encode parent set configurations of DAGs. Supposing an order of nodes, a DAG can be reconstructed from the clusters. The parent set configuration is then obtained by maximising the parent sets consistent with this ordering.

4. *Choose a new cluster $\bar{C}$ to add, and set $\mathcal{C} := \mathcal{C} \cup \{\bar{C}\}$. This coincides with the pricing problem.*
   Jaakkola et al. do not solve the pricing problem explicitly, since this comprises the consideration of all cluster constraints of the primal problem (as a new cluster has to be searched for). Instead the authors add clusters in two ways corresponding to two sets of inequalities. First, if there are cycle inequalities (obtained from the LP-relaxation of the acyclic subgraph polytope P_C, see Section 3.2.3, page 56) which are not satisfied, then take the most violated one and add the corresponding cluster to $\mathcal{C}$. Second, if there are no violated cycle inequalities, then this does not necessarily mean satisfiability of the solution. These cycle inequalities are not sufficient for the integral solutions of the Problem (3.6.1), page 88. In this case, clusters $\mathcal{C}$ are added with nodes in a greedy way with respect to the gain in objective.

Branch and bound:
The original primal problem (Problem (2.4.2), page 49, with the Inequality description (3.6.1), page 88) is solved using a bound obtained from the restricted Master problem (5.4.2), page 103. The authors use the respective dual objective values as a certificate for optimality of an integer solution.

5.4.2 Computational results of Jaakkola et al. [37]

This highly involved algorithm uses two fundamental steps iteratively, a column generation approach and afterwards a branch and bound method, for which the authors do not use any state-of-the-art software. However, their computational results gain from an initial pruning step reducing the number of variables in Problem (5.3.1). The size of the problems they solve and the time needed justify hope for an application of state-of-the-art software to go even beyond the presented problem sizes. We refer to their results in Section 5.6.4, page 113.

5.5 Computational experiments

5.5.1 The databases

The examples we analyse are taken from three sources: the UCI-Machine learning repository[1], "randomly" generated datasets and used by Jaakkola et al. [37]. Information presented on the databases is taken from the respective source. We introduce the used abbreviations of the databases in the respective captions.

In general, the database or, more precisely, its background is not necessary for the procedure, as structural properties of relations among variables and their interpretation are not investigated and taken advantage of. The only requirement we need is caused by our technical assumptions: We only use integral databases without missing values or hidden variables. We do not know whether there are possible hidden values, of course. But, due to our assumptions we suppose the completeness of observed variables and consider "hidden variables" as a representation technique.

[1] http://archive.ics.uci.edu/ml/.

Solar flare database - "flare" This database was donated to the UCI-Machine learning repository by Gary Bradshaw[2] in March 1989.

Each data vector is an instance representing 13 features for one active region in the sun. The database comprises 1066 instances of solar flares which are error corrected. Measured features are

A - a code for the so-called "modified Zurich class": values ranging from 1(="A") to 7(="H"),

B - a code for the largest spot size: 1(="X"), 2(="R"), 3(="S"), 4(="A"), 5(="H") and 6(="K") from smaller to larger sizes,

C - a code for the spot distribution: 1(="X"), 2(="O"), 3(="I") and 4(="C"),

D - a code for the activity of the region: 1 for a "reduced" activity and 2 for an "unchanged" activity,

E - a code for the evolution: 1 for a "decay" and 2 for a "growth",

F - a code for the flare activity (of "M1") of the previous 24 hours: 1 for smaller than one "M1", 2 for one "M1" and 3 for more activity than one "M1",

G - a code for "historically complex": 1 for "yes" and 2 for "no",

H - a code whether this region has become "historically complex" on this pass: 1 for "yes" and 2 for "no",

I - a code for the size of the area under investigation: 1 for a "small" area and 2 for a "large" area,

J - a code of the area of the largest observed spots: 1 for "≤ 5" and 2 for "> 5",

K - the total number of common ("C-class") flares production by this region over the last 24 hours: 0 to 8,

L - the total number of moderate ("M-class") flares production by this region over the last 24 hours: 0 to 8,

M - the total number of severe ("X-class") flares production by this region over the last 24 hours: 0 to 8.

The task is to predict the last 3 variables, which are variables classifying the region under investigation, on the base of the other features. Therefore a computed structure of the network provides the underlying relationship of the first 10 variables upon the last three variables.

We compute several structures with and without additional degree bounds for either a DAG description or a chordal graph description (see Section 5.5.4, page 109). Figure 5.2, next page, provides a structure corresponding to the best pattern with no imposed degree bounds.

[2] bradshaw@clipr.colorado.EDU.

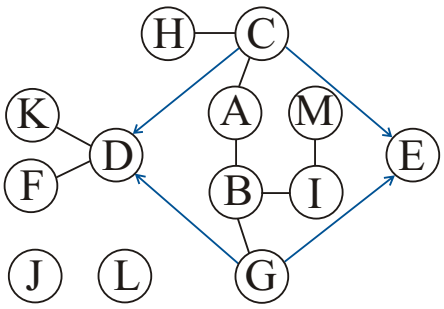

Figure 5.2: The best unrestricted Bayesian network structure representing the "flare" database.

From this, we can see, that feature J, the area of the largest observed spots, does not have any influence upon all other variables at all. Moreover, there is no relation within the data implying class variable L or the other way around.

Zoo database - "zoo" This database is also taken from the UCI-Machine learning repository. The creator is Richard Forsyth who added this database to the repository in May 1990.

A data vector represents a set of features of one animal together with its type. There are binary features for presence of "hair" (A) or "feathers" (B), the production of "eggs" (C) or "milk" (D), "airborne" (E), "aquatic" (F), "predator" (G), "toothed" (H), "backbone" (I), "breathes" (J), "venomous" (K) and "fins" (L). Moreover, there is an integer feature for the number of "legs" (M) with values 0,2,4,5,6 and 8. Other binary features are "tail" (N), "domestic" (O) and "catsize" (P). The type variable (Q) is integer from 1 to 7 with 41 of the animals contained in the first class, 10 in the second, 5 in the third, 13 in the fourth, 4 in the fifth, 8 in the sixth and 10 in the seventh type.

Altogether the database consists of the attributes of 101 animals with no missing values.

Analogous to the "flare" database we want to predict the type of the animal, which is represented by the last variable, based on its observed features. The following structure (Figure 5.3 below) is the chordal graph structure with an imposed bound on the maximum size of the cliques best representing the data.

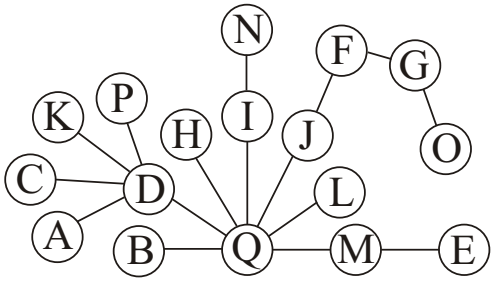

Figure 5.3: The best restricted chordal graph with bounds on clique size representing the "zoo" database.

This structure is a forest, hence the imposed maximal clique size is not attained in the optimal restricted solution. Additionally, all features are contained in the computed structure.

Randomly generated database - "random" Originally the class of randomly created binary databases has been used for testing the program for computing the objective function and for

creating the inequalities. For a good randomisation procedure creating real random values, there should be no relation among variables implied in the data. Indeed in most of the cases the empty structure with no relation turns out to be the best one, which becomes obvious by the objective function and performing a simple preprocessing step. The only set, among the initial ones, surviving this "preprocessing" with a non-trivial structure is one with 20 variables and a database of length 35, all variables within are binaries.

MicroRNA database - "mirna" This database is taken from Jaakkola et al. [37] and has been originally published in [43]. In [43] MicroRNA is investigated as a special type of RNA helping to analyse and classify human cancer.

SPECT heart database - "SPECT_tr" The database was added by Lukasz A. Kurgan and Krzysztof J. Cios (University of Colorado at Denver) to the UCI-Machine learning repository in January 2001. As specified in the repository the database describes the diagnosis of special cardiac tomography images.

Each data vector coincides with one patient who is to be classified as "normal" or "abnormal" and there are 267 patients recorded. The data vector itself summarises one image originally comprising 44 continuous features. These continuous variables have already been discretised to obtain 22 binary variables. Together there are 23 binary variables and the 267 patient records are divided into 80 instances for training and 187 for testing. We choose the training database and call it "SPECT_tr".

The following Figure 5.4 shows the best BN structure representing the "SPECT_tr" database.

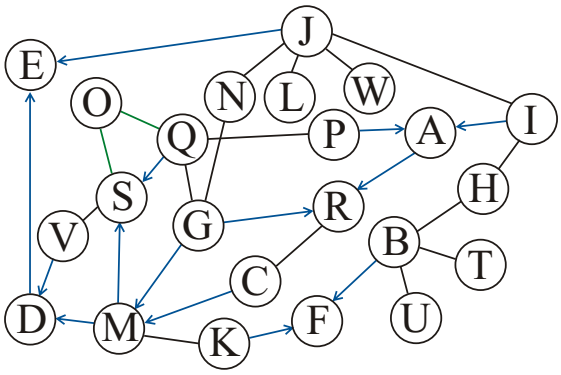

Figure 5.4: The best unrestricted Bayesian network structure representing the "SPECT_tr" database.

Phenotypic database - "phen" This database is taken from Jaakkola et al. [37] and the references therein. It is a binary database with 25 variables describing phenotypic data for C. elegans, which is a worm used as a model system to derive properties of molecular and cellular interactions.

The optimal structure for the unrestricted DAG case is particularly interesting (see Figure 5.5, next page). Initially bounded to sets of size at most five due to pruning, the maximal attained size is four (e.g. sets $\{f, h, i, p\}$, $\{f, i, k, p\}$ and $\{f, i, p, q\}$). Moreover the structure is complicated compared to the previous ones. Cliques and immoralities occur far more often than the undirected edges. Intuition would suggest this problem to be hard to solve. We will refer to this in the discussion of the computational results in Section 5.6, page 111.

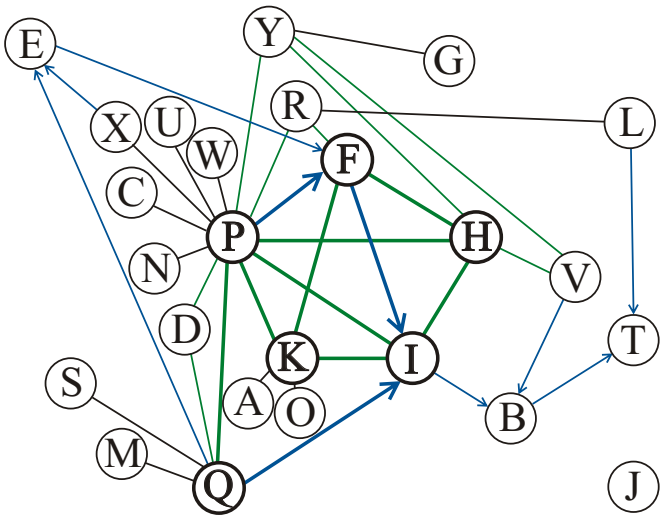

Figure 5.5: The best unrestricted Bayesian network structure representing the "phen" database.

5.5.2 Setting up the experiments

For all of the given databases the objective function is computed. For general DAGs we have first tried to reduce the dimension of the problem by pruning the variables. If the dimension is too large additional bounds on node degrees are imposed. In case of chordal graphs pruning is not allowed (compare with Section 5.3.1.2, page 101), only bounding is applied for a reduction of the dimension.

The computation times for the objective function are not reported as these are common through all scoring and searching approaches. We use the BIC as the objective function with a combination of Formulas (1.4.5), page 21, and (1.4.9), page 28. In particular, it is easier to apply pruning within the score of graphs than with respect to standard or characteristic imsets. So we use the scores of graphs to prune the variables needed to represent Problem (5.1.1), page 95, (see Section 5.3.1.1, page 98). Recall that, even when having a non-prunable subset, all scores of subsets must be considered for computing the respective score requiring a lot of computational effort.

For a computation of the inequalities the actual sets representing the variables of the characteristic imsets are needed. The reason is that subset relations such as inclusions and exclusions of these subsets are needed to derive the inequalities. The list of non-pruned sets can be derived while computing the objective. This list can of course be very large and must be represented and stored efficiently. Moreover, we have to choose a representation enabling us to search subsets fast so that the computation of the inequalities can be done quickly.

For the learning of unrestricted DAGs and DAGs with bounds on the number of parents we use the Inequality description (3.2.14), page 62, in its alternative cycle formulation using the DAG-inequalities (3.2.16), page 63. Although Description (3.2.14) can be used to relax the integrality condition, it consists of too many inequalities. However, for defining a LP-relaxation of P_{ext} not all of Inequalities (3.2.11), (3.2.12) and (3.2.13), page 61, are necessary. We add them according to specified parameters. For learning chordal graphs with and without bounds on the clique sizes we use the LP-relaxation $P_{CH}^{rel\ (1)}$ of Section 3.4.4.1, page 80.

The aim of the computational experiments is to visualise the strength of the rather theoretic approach and the gain which can already be achieved by simply solving the examples with state-of-the-art IP solvers. For example, instead of having to code a row generation procedure cutting off invalid integer solutions, we use "lazy constraints" which is a tool provided by the

optimisation solver CPLEX 12.1 [36]. We have to add these inequalities to the description but they are only used for verification or falsification of an integer solution, not during the computation itself. The actual amount of inequalities in the description is hence kept small.

We use a C/C++ program for computing the objective function. Another C/C++ program then creates the inequalities and uses the Callable Library of CPLEX 12.1 [36] for solving the Optimisation problem (5.1.1). With this interface we can define the corresponding linear integer Program (5.1.1) directly within CPLEX and solve it within the program. The time is measured with respect to the needed CPU time. The machine used consists of 4 AMD-Opteron 6174 12-core CPUs with 2.2 GHz and a total of 128GB RAM.

5.5.3 Parameter tuning

Additionally to "lazy constraints", we perform row generation on the DAG-inequalities (3.2.16). The program uses parameters specifying up to which bound on the cardinality of sets T of nodes the inequalities are *all* added to the system or are added as "lazy constraints". The cardinality of T comes from the definition of characteristic imsets in Chapter 2, page 37, and the derivation of the inequalities in Chapter 3, page 51. An integer solution is checked to contain directed cycles or not. If it is not a DAG, then the respective inequality cutting off this solution is added to the system. An integer solution may contain more than one directed cycle. Adding more than one cutting inequality may lead to fewer optimisation steps in the row generation. A second parameter indicated with "-cl" in the tables of Section 5.5.4.1, page 110, specifies how many directed cycles should be searched for and hence how many DAG-inequalities should be added to the system in one single step of the row generation. Independently from this row generation providing the integer optimal solution to be a DAG, we use a third parameter indicated with "-ip" to influence the size of the description. This parameter influences the number of Inequalities (3.2.12), page 61, added to the system. The corresponding parameters are reported in each of the tables in Section 5.5.4.1.

5.5.4 Comparisons

The results are listed in four tables each for learning unrestricted patterns of DAGs, for learning restricted patterns of DAGs with a bound on the degree of the nodes, for learning chordal graphs and for learning chordal graphs with a bound on the clique size. In each of the tables the original number of variables of the problem ("dimension 1") and the resulting number of variables ("dimension 2"), after pruning or bounding, respectively, is reported. Together with these dimensions we report on the maximum cardinality of a set T ("maximum $|T|$") with respect to the encoding with characteristic imsets. Moreover the duration of the computation is listed. We distinguish two ways of measuring the time: The first time reported ("time 1") just measures the duration of the optimisation step including the verification of optimality and the row generation. The second time ("time 2") additionally includes the setup of the problem. Both do not include the computation of the objective function, which we consider as given. The parameters used for the row generation are listed below each of the tables.

We do not compare our results with other methods for several reasons. First, most of the methods (as presented in Section 1.4.2, page 22) are heuristic methods not providing a certificate for optimality which is often enough the time consuming step. The only procedures obtaining provable optimal solutions are the dynamic programming approach and the method by Jaakkola et al. [37] presented in Section 5.4, page 102. Second, we only present preliminary results. As argued in Sections 1.4.2.2, page 24, and 5.1, page 95, enumerative procedures cannot exploit additional structural information as the polyhedral approach can do. However,

we use these tests only as a presentation of the power of both polyhedral approaches together with state-of-the-art software.

5.5.4.1 Learning unrestricted patterns of DAGs

Database	$\|N\|$	dimension 1	dimension 2	maximum $\|T\|$	time 1	time 2
flare[1]	13	8,178	110	4	0.74s	0.79s
flare[2]	13	8,178	110	4	0.44s	0.48s
flare[3]	13	8,178	110	4	**0.06s**	**0.19s**
SPECT_tr[4]	23	8,388,584	349	3	17m 17s	18m 15s
SPECT_tr[5]	23	8,388,584	349	3	**6m 13s**	**15m 21s**
SPECT_tr[6]	23	8,388,584	349	3	25m 12s	36m 05s
phen[7]	25	33,554,406	890	5	2h 48m 53s	2h 49m 14s
phen[8]	25	33,554,406	890	5	7m 28s	7m 53s
phen[9]	25	33,554,406	890	5	**4m 58s**	**5m 35s**

1- Called with parameters: 4 -cl 1 -ip 4.

2- Called with parameters: 5 -cl 1 -ip 3.

3- Called with parameters: 6 -cl 1 -ip 5.

4- Called with parameters: 8 -cl 4 -ip 3.

5- Called with parameters: 9 -cl 1 -ip 4.

6- Called with parameters: 10 -cl 1 -ip 3.

7- Called with parameters: 4 -cl 6 -ip 4.

8- Called with parameters: 5 -cl 1 -ip 3.

9- Called with parameters: 5 -cl 1 -ip 5.

Table 5.1: Timing results for learning unrestricted patterns of DAGs.

The next tables show the respective computational results for the restricted learning.

5.5.4.2 Learning patterns of DAGs with degree bounds

Imposing bounds on the node degrees is only necessary for the random generated database and the "mirna" database. Note that an additional pruning step can cause the nodes to have an even smaller degree.

Database	$\|N\|$	dimension 1	degree bound	dimension 2	maximum $\|T\|$	time 1	time 2
Random[1]	20	60,439	5	218	4	3.17s	3.77s
Random[2]	20	60,439	5	218	4	4.49s	5.39s
Random[3]	20	60,439	5	218	4	**1.67s**	**3.11s**
mirna[4]	22	35,420	10	295	3	126h 26m 03s	126h 26m 25s
mirna[5]	22	35,420	10	295	3	**4h 48m 16s**	**4h 55m 13s**

1- Called with parameters: 4 -cl 1 -ip 3.

2- Called with parameters: 4 -cl 1 -ip 4.

3- Called with parameters: 5 -cl 1 -ip 4.

4- Called with parameters: 8 -cl 2 -ip 3.

5- Called with parameters: 11 -cl 1 -ip 3.

Table 5.2: Timing results for learning restricted, bounds on the degree of all nodes, patterns of DAGs.

5.5.4.3 Learning chordal graphs

As already analysed pruning is not a valid problem reduction for chordal graphs. In this case there is no reduction of the problem at all and dimension 1 coincides with dimension 2.

Database	$\|N\|$	dimension 1	time 1	time 2
flare[1]	13	8,178	11.48s	12.67s

1- Called with parameters: 6 -cl 1.

Table 5.3: Timing results for learning chordal graphs.

5.5.4.4 Learning chordal graphs with bounds on clique sizes

For the other databases we have to impose bounds on the maximum size of cliques ("maximum $|C|$") in order to get results in a reasonable amount of time.

Database	$\|N\|$	dimension 1	maximum $\|C\|$	dimension 2	time 1	time 2
flare[1]	13	8,178	5	2,366	0.74s	1.29s
zoo[2]	17	131,054	4	3,196	16h 44m 16.20s	16h 44m 24.40s
Random[3]	20	60,439	4	6,175	07.22s	19.30s
SPECT_tr[4]	23	8,388,584	4	10,879	15h 4m 22.6s	15h 17m 16.6s
phen[5]	25	33,554,406	4	15,250	74.44s	06m 42.1s

1- Called with parameters: 6 -cl 1.

2- Called with parameters: 7 -cl 3.

3- Called with parameters: 4 -cl 1.

4- Called with parameters: 8 -cl 5.

5- Called with parameters: 6 -cl 3.

Table 5.4: Timing results for learning chordal graphs with bounds on clique size.

5.6 Computational results of learning with characteristic imsets

The results presented in Section 5.5.4, page 109, are based on preliminary tests. Both restricted and unrestricted learning tasks for various real life databases could be solved showing the power of the combinatorial approach with characteristic imsets using state-of-the-art software.

5.6.1 Results from computing the objective function and from pruning

First of all, the computations show pruning to be a powerful tool for an effective reduction of the dimension of the problem. For example, consider the "SPECT_tr" database. Although the problem is formulated with an initial dimension of about 8 million, it can be reduced to effectively 350 required variables. Moreover, the complexity of the solution reduces to each node having at most two parents. Table 5.1, previous page, shows, that a bound of three to four parents per node is a valid assumption from which general supposed bounds on the degrees of nodes can be obtained and later used for additional limitations of problems in higher dimensions. The additional reductions together with pruning techniques might lead to fast computations of objective functions and inequalities. Furthermore, especially in real world application, it is often of more interest to have a certain type of solution, e.g. a simple one with fewer parents. The complete problem description, of course, could comprise these additional specifications. But this would also increase the number of inequalities and might lead to higher computational requirements. Instead, additional bounds are considered as a kind of preprocessing within the computation of the objective function.

Although pruning is a justification for an extensive preprocessing within the objective computation, it takes time. Master and slave techniques, parallelisation and efficient storing of information might help to compensate this problem. In particular, for concrete problems,

as a subject of future research, efficient coding techniques lead to computable solutions far beyond the presented dimensions in Table 5.1 (and also in Table 5.2, page 110). In spite of this, there is definitely a limit for an effective computation of the objective function, which can only be extended with the evolution of hardware.

Therefore, the objective functions of many examples, especially those implying a more complicated structure, could not be computed. Even worse, although larger databases imply better solutions without any structural assumptions, the timing results become worse.

For chordal graphs only bounding the clique sizes can reduce the dimension of the problem and thus the costs of computing the objective function and later the inequalities and the solution. Without additional pruning the dimension of the problem remains high as visible in Tables 5.3 and 5.4, page 111. This is not necessarily a problem for the computation of the objective function but it is definitely a problem during the calculation of the inequalities needed to describe the optimisation problem. This fact, again, makes it even more important to consider only concrete examples to gain from the above mentioned additional programming and preprocessing techniques. In spite of this, if pruning is not done, the total computation time of the objective function reduces tremendously.

5.6.2 Results from optimisation

There are two main influences on the complexity of computing the inequalities and the solutions, which might in some cases have an opposite effect. For learning DAGs an extensive pruning reduces the dimension of the problem tremendously, resulting either in less variables or in less inequalities necessary. This is definitely an effect leading into a faster optimisation step, as well. Contrarily, also better inequalities would reduce the necessary computation time, e.g. introducing facets would improve the performance of the branch and bound approach within CPLEX (or with other optimisation software). But additional inequalities also increase the size of the description and, even worse, have to be constructed, and this additional construction takes time. Therefore, also the contrary effect is observed, additional and even stronger inequalities yield into computationally infeasible optimisation steps.

The "phen" database is an example of this trade-off between intuition (more variables yield into a bigger and more complex problem description) and complexity of a more accurate (i.e. a facet) description. This database has a large number of initial variables and also the pruning step yields large remaining subsets. Although there are databases, like the "zoo" database, with less variables, the corresponding computations could not be finished for the unrestricted case in a reasonable amount of time. For the "phen" database a larger description seems to yield a better description enabling CPLEX to solve it much faster.

But, a carefully balanced trade-off must be found anyway. For example, a concrete database can give additional information on optimal structures. Analysing the corresponding objective function and incorporating expert knowledge may provide guidance on, which inequalities to add and also might indicate which inequalities to skip (similar to row generation presented in Section 5.3.2, page 102).

The same effect occurs in case of chordal graphs. The only, although important difference is that additional inequalities can be derived more easily (e.g. those presented in Section 3.2.4, page 59) and the concrete description is smaller. A side-effect of the larger description of the general case is, that once being able to compute the best DAG this computation can be done much faster than for the corresponding chordal graph.

Again, the "phen" database is a good example. CPLEX can compute the optimal bounded chordal graph much faster for this database than for other databases ("zoo" or "SPECT_tr"). Although we impose a bound of four on the clique size, there are about 33 millions variables.

This causes the amount of inequalities needed to grow tremendously. However, it is possible for us to compute the optimal bounded chordal graph for the "zoo" database. For this set the larger description is much better, i.e. stronger, than the description of the unrestricted problem.

5.6.3 Observations from parameter tuning

In Table 5.1 we can see the trade-off between the inclusion of more inequalities and doing row generation for small databases like "flare". Adding more inequalities (according to parameters "4 -cl 1 -ip 4", "5 -cl 1 -ip 3" and "6 -cl 1 -ip 5" in Table 5.1) results in less necessary optimisation steps (from 7 iterations to only one) in the row generation and a low computation time for the pure optimisation ("time 1" for "flare"). On the other hand, the inclusion of more inequalities implies the construction of more inequalities and hence the overall process takes longer ("time 2" for "flare").

The same happens for "SPECT_tr" and "mirna", respectively. In spite of its large number of variables and larger original dimension, an addition of DAG-inequalities up to parameter value 9 and 11 (the size of the sets; compare with Section 5.5.3, page 109), respectively, is effective in this case, as well.

In the case of chordal graph fewer inequalities are necessary to specify the problem. For small databases it is even possible to include all inequalities. With such an amount of inequalities CPLEX cannot gain from the exact problem definition and the computation times grow large, as to be seen in Table 5.3.

For chordal graphs it makes sense to include more DAG-inequalities for small data sets. We can see this in Table 5.4 for the "flare" database, more or less for the "zoo" database (here the included number of inequalities simply becomes too large to be manageable for CPLEX), the randomly generated database, the "SPECT_tr" database and the "phen" database. Moreover to this, an increase in the number of DAGs added per row generation step (parameter indicated with "-cl") is effective as long as enough cycles occur.

5.6.4 Comparison to Jaakkola et al. [37]

Jaakkola et al. present in [37] computational results of learning unrestricted BN structures with problems larger than ours. Moreover to this, problems of comparable size and especially the problem constructed from the "phen" and the "mirna" database could be solved faster (approximated). This is quite surprising as we have already presented their method to be highly involved (Section 5.4, page 102), complicated and they do not use optimisation software for solving LPs occurring in their approach. Besides this, they are faced with the same problems of representation as in our approach due to an exponential number of inequalities and variables.

For example, simple optimal structures do not always occur. As we can see in the "SPECT_tr" and "mirna" database large cycles can appear. This effect can be seen in Tables 5.1 and 5.2. The parameters specified for the STECT_tr database imply **all** DAG-inequalities up to bound 10 to be included. We have tried smaller bounds as well and therefore a search for contained directed cycles independent of the CPLEX computation. The inclusion of all inequalities for the largest occurring cycles of length 10 has turned out to be much faster. For the column generation technique of Jaakkola et al. this would mean to include clusters of cardinality 10 (and 12 for "mirna").

Another possible reason for diverging results is their objective function. The main results of Jaakkola et al. are presented together with the use of a Bayesian score (see Section 1.4.1,

page 19). The use of an alternative scoring function has influence on both the pruning and the computation itself. As we have intended to present the power of the use of well-known optimisation software together with the polyhedral approach we have limited ourselves and the computations on the common BIC quality criterion.

Furthermore, we use CPLEX as a state-of-the-art software providing quick and efficient computations especially when dealing with a binary problem. Contrarily, Jaakkola et al. do not use any sate-of-the-art software.

5.6.5 Future work

Each step of the effective solution algorithm, the computation of the objective function, the inequalities and the optimisation step, indicates a potential power of the polyhedral approach via characteristic imsets.

But this power can only be exploited if additional information about concrete examples can be incorporated. Having these additional information, more involved preprocessing techniques yield problem descriptions which are easier to solve. Moreover, additional programming techniques (such as the storing of variables and construction of inequalities) can be exploited and customised for the database under investigation.

6 The conditional independence implication problem

6.1 From graphical implication to computer testing

BNs can be represented by a set of CI statements satisfying the Semi-graphoid properties 1.1.4, page 5 (and also Lemma 1.1.5, page 5). These semi-graphoid properties define the implication between valid CI statements. This implication of statements generalises to the generation of statements, i.e. the question whether certain CI statements are already implied by other CI statements. This question is known as the **CI implication problem** or the **CI inference problem**.

Pearl [51] considers graphical models as "inference engines" enabling him to determine implicated CI statements. A list of statements defines a network corresponding to a certain graph. From this graph additional statements can be derived. He conjectured the existence of a set of semi-graphoid-type graphical properties with which CI inference can be fully characterised. But Studený disproves this and he moreover shows no finite system of these properties to exist ([57], [58]). But the complexity of the algebraic descriptions of CI statements is not yet completely clarified and, because of this, computer based solutions are a straightforward alternative for testing CI inference.

These computer tests are based on an algebraic representation instead of a graphical one. With this description any discrete CI structure can be represented, which is a basic problem of graphical models as "inference engines". Proposition 1.2.11, page 14, characterises any CI statement to be a non-negative integral combination of elementary imsets, i.e. combinatorial imsets. A valid generalisation is to consider structural imsets instead. The use of structural imsets yields a sufficient condition on the representation of every CI statement. In terms of this algebraic representation the CI problem is known as the **independence implication problem**.

This chapter is based on [8] and the references therein.

6.2 Concepts concerning CI inference

We follow [60] and the definitions and results therein. Based on a presentation of the contained results, we outline an algorithm for testing independence implication introduced in [9] (see Section 6.3.1, page 122). Furthermore, we propose an alternative way for computer testing of the independence implication problem (see Section 6.3.2, page 123).

6.2.1 Conditional independence of sets of statements

Let L be a set of CI statements, called an **input list of CI statements**, and t a CI statement outside L. We say L **probabilistically implies** t if for every discrete probability distribution P the following is true: Whenever all statements in L are valid with respect to P, then t is valid with respect to P, too.

This probabilistic implication is a formal generalisation of CI implication, which is only dependent on the probability distribution and not on a graphical representation. The Markov property enables us to generalise from CI statements in graphs to CI statements valid for probability distributions. Every implication in the graph is also valid for the probability distribution, but not the other way around. A classic example of valid probabilistic CI implications are the semi-graphoid properties.

6.2.2 Structural imsets

For the reduction of repetition, we recall only the definition of a structural imset (compare with Section 1.2.2.1, page 12). An imset $\mathbf{u}$ over N is called structural if there exists an $n \in \mathbb{N}$ such that the multiple $n \cdot \mathbf{u}$ is a combinatorial imset, i.e.

$$n \cdot \mathbf{u} = \sum_{\mathbf{w} \in \mathcal{E}(N)} k_{\mathbf{w}} \mathbf{w} \quad \text{for some } n \in \mathbb{N},\ k_{\mathbf{w}} \in \mathbb{Z}_+. \tag{6.2.1}$$

The set of structural imsets is denoted with $\mathcal{S}(N)$. Due to Lemma 6.3 in [60], there exists a smallest multiple $n_* \in \mathbb{N}$, depending only on $|N|$, with

$$\mathbf{u} \in \mathcal{S}(N) \Leftrightarrow n_* \cdot \mathbf{u} \in \mathcal{C}(N)$$

for every imset $\mathbf{u}$ over N. Studený shows $n_* = 1$ for $|N| \leq 4$ and it is possible to show $n_* = 2$ for $|N| = 5$ by using Hilbert bases. However, it is still an open question to determine n_* for every $|N|$ (Theme 11 in [60]).

6.2.3 Independence implication

With these algebraic representatives of CI statements the CI inference problem translates into conditions on structural imsets.

Definition 6.2.1 (independence implication) *Let $\mathbf{u}, \mathbf{v}$ be structural imsets over N. We say $\mathbf{u}$ **independence implies** $\mathbf{v}$ and write $\mathbf{u} \rightharpoonup \mathbf{v}$ if $\mathcal{M}_{\mathbf{v}} \subseteq \mathcal{M}_{\mathbf{u}}$.*

This definition can also be derived by considering the inclusion of imsets. If CI statements are implied by one another, then their respective models are, as well. The same is true for the standard imsets.

The question remains: Which condition on structural imsets, and therefore on CI statements, implies $\mathbf{u} \rightharpoonup \mathbf{v}$. This question coincides with the search for a direct link between structural imsets and the CI statements they encode. The following definition is crucial for this relation:

Definition 6.2.2 (representable) *Let $\langle A, B|C\rangle$ be a CI statement and $\mathbf{u}$ be a structural imset. Then, $\langle A, B|C\rangle$ is called **representable** by $\mathbf{u}$ if there exists a coefficient $k \in \mathbb{N}$ with $k \cdot \mathbf{u} - \mathbf{u}(\langle A, B|C\rangle) \in \mathcal{S}(N)$ or if equivalently there exists a coefficient $l \in \mathbb{N}$ with $l \cdot \mathbf{u} - \mathbf{u}(\langle A, B|C\rangle) \in \mathcal{C}(N)$.*

With this definition of representation the rather theoretical characterisation of independence implication turns into a practical one. We can use this condition then to verify $\mathbf{u} \rightharpoonup \mathbf{v}$ computationally.

Lemma 6.2.3 (Lemma 6.1 in [60]) *Let* $\mathbf{u}, \mathbf{v}$ *be structural imsets over* N*. Then* $\mathbf{u} \rightharpoonup \mathbf{v}$ *if and only if one of the following conditions is true*

(i) $\exists l \in \mathbb{N}$ *s.t.* $l \cdot \mathbf{v} - \mathbf{u}$ *is a structural imset,*

(ii) if $\mathbf{v}$ *is a combinatorial imset, then* $\exists k \in \mathbb{N}$ *s.t.* $k \cdot \mathbf{u} - \mathbf{v}$ *is a combinatorial imset.*

Proof. We start this proof by showing the first equivalence. Let $\mathbf{u} \rightharpoonup \mathbf{v}$ be given and both $\mathbf{u}$ and $\mathbf{v}$ be structural imsets. From the definition of structural imsets we can conclude $n \in \mathbb{N}$ and $k_{\mathbf{w}} \in \mathbb{Z}_+$ to exist as such that Equation (6.2.1) holds. For any coefficient $k_{\mathbf{w}} > 0$ we define $\mathbf{w} = \mathbf{u}(\langle a, b|K\rangle)$. We can conclude imsets $\mathbf{w}$ to be representable by $\mathbf{v}$ (Definition 6.2.2). Modifying Equation (6.2.1) with respect to $n \cdot \mathbf{v} - \mathbf{w}$ yields the required condition. Then, we have $\langle a, b|K\rangle \in \mathcal{M}_{\mathbf{v}}$ and with $\mathbf{u} \rightharpoonup \mathbf{v}$ also $\langle a, b|K\rangle \in \mathcal{M}_{\mathbf{u}}$. We again use the definition of representability and conclude the existence of a coefficient $l_{\mathbf{w}} \in \mathbb{Z}_+$ with $l_{\mathbf{w}} \cdot \mathbf{u} - \mathbf{w} \in \mathcal{S}(N)$. For this coefficient and $k_{\mathbf{w}}$ we define $l := \sum_{\mathbf{w} \in \mathcal{E}(N), k_{\mathbf{w}} > 0} k_{\mathbf{w}} \cdot l_{\mathbf{w}}$. This l satisfies the first implication. To verify this, compute

$$\begin{aligned} l \cdot \mathbf{u} - \mathbf{v} &= l \cdot \mathbf{u} - n \cdot \mathbf{v} + (n-1)\mathbf{v} \\ &= \sum_{\substack{\mathbf{w} \in \mathcal{E}(N) \\ k_{\mathbf{w}} > 0}} k_{\mathbf{w}} \cdot l_{\mathbf{w}} \cdot \mathbf{u} - \sum_{\substack{\mathbf{w} \in \mathcal{E}(N) \\ k_{\mathbf{w}} > 0}} k_{\mathbf{w}} \cdot \mathbf{w} + (n-1)\mathbf{v} \\ &= \sum_{\substack{\mathbf{w} \in \mathcal{E}(N) \\ k_{\mathbf{w}} > 0}} k_{\mathbf{w}} \cdot (l_{\mathbf{w}} \cdot \mathbf{u} - \mathbf{w}) + (n-1)\mathbf{v}. \end{aligned} \tag{6.2.2}$$

As both summands in (6.2.2) above are structural imsets, $l \cdot \mathbf{u} - \mathbf{v}$ is structural, as well. To show the other direction of Statement (i) we assume the existence of a coefficient $l \in \mathbb{N}$ such that $l \cdot \mathbf{u} - \mathbf{v} \in \mathcal{S}(N)$. We choose a CI statement $\langle A, B|C\rangle \in \mathcal{M}_{\mathbf{v}}$. If $\mathbf{u} \rightharpoonup \mathbf{v}$, then any CI statement has to be contained in $\mathcal{M}_{\mathbf{u}}$, too, or, in other words, the imset $\mathbf{u}(\langle A, B|C\rangle)$ must be also implied by $\mathbf{u}$. However, we know $\mathbf{u}(\langle A, B|C\rangle)$ to be implied by $\mathbf{v}$. Then there exists a $n \in \mathbb{N}$ with $n \cdot \mathbf{v} - \mathbf{u}(\langle A, B|C\rangle) \in \mathcal{S}(N)$. We can now add two structural imsets

$$n \cdot (l \cdot \mathbf{u} - \mathbf{v}) + n \cdot \mathbf{v} - \mathbf{u}(\langle A, B|C\rangle) = (n \cdot l) \cdot \mathbf{u} - \mathbf{u}(\langle A, B|C\rangle)$$

and obtain a structural imset. Indeed by definition of representability, $\mathbf{u}(\langle A, B|C\rangle)$ is also implied by $\mathbf{u}$.

To show the second equivalence, we show Statement (ii) to be equivalent to the previous one. If the second equivalence is valid, then the first one is valid as well, as every combinatorial imset is also a structural one. Otherwise, we assume $\mathbf{v}$ to be a combinatorial imset and suppose the validity of the first equivalence. Then we obtain the structural imset $l \cdot \mathbf{u} - \mathbf{v}$. Let $n \in \mathbb{N}$ be given such that $n \cdot (l \cdot \mathbf{u} - \mathbf{v})$ is a combinatorial imset. Setting $k := n \cdot l$ leads to

$$k \cdot \mathbf{u} - \mathbf{v} = n \cdot l \cdot \mathbf{u} - \mathbf{v} = n(l \cdot \mathbf{u} - \mathbf{v}) + (n-1)\mathbf{v},$$

a sum of combinatorial imsets and therefore a combinatorial imset. This fact shows the second equivalence. □

This Lemma yields a so-called direct characterisation of independence implication.

Definition 6.2.4 (direct characterisation) *Let* $\mathbf{u}, \mathbf{v}$ *be structural imsets over* N*. The following equivalence is called the* ***direct characterisation*** *of the independence implication problem.*

$$\mathbf{u} \rightharpoonup \mathbf{v} \iff \exists k \in \mathbb{N} \text{ s.t. } k \cdot \mathbf{u} - \mathbf{v} \in \mathcal{S}(N). \tag{6.2.3}$$

Given $\mathbf{u} \in \mathcal{S}(N)$, the corresponding CI structure $\mathcal{M}_{\mathbf{u}}$ consists of all CI statements $A \perp\!\!\!\perp B \,|\, C$ with $\mathbf{u} \rightharpoonup \mathbf{u}(\langle A, B|C\rangle)$. Equivalently, the CI structure induced by $\mathbf{u}$ contains the CI structure induced by $\mathbf{v}$. The importance of structural imsets is the correspondence of CI structures induced by a discrete probability distribution P over N and $\mathcal{M}_{\mathbf{u}}$. This fact is the difference between algebraic and graphical representatives. Graphical representatives are dependent on the Markovian measure providing only one direction of the implication between the encoded model and the graphical structure. Contrarily, algebraic representations are directly encoding the distribution and the model.

Theorem 6.2.5 (Theorem 5.2 in [60]) *Let P be a probability measure over N with finite multiinformation (see Section 2.3.3 in [60] for a definition). There exists a structural imset $\mathbf{u}$ over N such that P is perfectly Markovian with respect to $\mathbf{u}$, that is, $\mathcal{M}_P = \mathcal{M}_{\mathbf{u}}$.*

An equivalent dual characterisation of independence implication can be derived from the following definitions: A real function $m : \mathcal{P}(N) \to \mathbb{R}^{2^{|N|}}$ is called **supermodular** if and only if

$$m(C \cup D) + m(C \cap D) \geq m(C) + m(D) \text{ for every } C, D \subseteq N.$$

The set of all supermodular functions over N is called $\mathcal{K}(N)$. The scalar product of m and an imset $\mathbf{u}$ over N is then

$$\langle m, \mathbf{u}\rangle := \sum_{S \subseteq N} m(S) u_S.$$

With these supermodular functions, being in fact nothing but imsets again, a dual definition of independence implication is possible.

Lemma 6.2.6 (Lemma 6.2 in [60]) *Let $\mathbf{u}, \mathbf{v}$ be structural imsets over N. Then $\mathbf{u} \rightharpoonup \mathbf{v}$ if and only if*

$$\forall m \in \mathcal{K}_l^{\diamond}(N) \text{ with } \langle m, \mathbf{v}\rangle > 0 \implies \langle m, \mathbf{u}\rangle > 0. \tag{6.2.4}$$

This condition is equivalent to

$$\text{for every } m \in \mathcal{K}(N) \text{ with } \langle m, \mathbf{v}\rangle > 0 \implies \langle m, \mathbf{u}\rangle > 0. \tag{6.2.5}$$

Moreover, Condition (6.2.4) above is also equivalent to the requirement:

$$l_* \cdot \mathbf{u} - \mathbf{v} \in \mathcal{S}(N), \text{ for every } l_* \in \mathbb{N} \text{ with } l_* \geq \langle r, \mathbf{v}\rangle \text{ and for every } r \in \mathcal{K}_l^{\diamond}(N). \tag{6.2.6}$$

The proof of this Lemma can be found in [60] on pages 118-119.

The set $\mathcal{K}_l^{\diamond}(N)$, known as the ***l*-skeleton** over N, denotes the collection of all non-zero normalised imsets contained in the extreme rays of $\mathcal{K}_l(N)$. The set $\mathcal{K}_l(N)$, on the other hand, is the class of all l-standardised supermodular functions. Note that the set of supermodular functions $\mathcal{K}(N)$ itself is a rational polyhedral cone. A ***l*-standardised** supermodular function m attains the value zero for all entries $m(S)$ with $|S| \leq 1$. In fact, the explicit l-standardisation is not necessary in the above lemma, any standardisation would suffice to derive the result. To get a definition analogous to the direct characterisation, we define the following skeletal characterisation which is the negated version of Condition (6.2.5).

Definition 6.2.7 (skeletal characterisation) *Let $\mathbf{u}, \mathbf{v}$ be structural imsets and let $\mathcal{K}(N)$ be the set of supermodular functions over N with $\langle m, \mathbf{u}\rangle, \langle m, \mathbf{v}\rangle \geq 0$, $\forall m \in \mathcal{K}(N)$. We call the following equivalence the **skeletal characterisation**:*

$$\mathbf{u} \rightharpoonup \mathbf{v} \iff \left(\langle m, \mathbf{u}\rangle = 0 \implies \langle m, \mathbf{v}\rangle = 0,\ \forall m \in \mathcal{K}(N)\right). \tag{6.2.7}$$

This skeletal notion of independence implication has been the original definition. Geometrically, it says "**u** implies **v** if every face of $\mathcal{R}(N)$ which contains **u** also contains **v**" [60], page 115.

The complete CI implication problem uses a certain set L of statements, which is transformable into a combinatorial imset $\mathbf{u}(L) := \sum_{s\in L} \mathbf{u}(s)$ via the sum of the single imsets. With this, we can derive a sufficient algebraic characterisation of the CI implication problem based on an observation in [60]. Note that this is only a sufficient condition but not a necessary one, in particular, solving the implication problem solves the CI implication problem only approximately.

Proposition 6.2.8 (following from [60] and [9]) *Let L be an input list of CI statements and t another CI statement over N. If $\mathbf{u}(L) \rightharpoonup \mathbf{u}(t)$ then L probabilistically implies t.*

Proof. To show this, we again use a special supermodular function called the multiinformation m_p (see Theorem 1.4.9, page 29) for any discrete probability distribution. This function has special properties. It is supermodular and the following implication is valid,

$$\forall s := A \perp\!\!\!\perp B \,|\, C \text{ over } N \implies \langle m_p, u(s)\rangle = 0 \iff (A \perp\!\!\!\perp B \,|\, C\,[P])\,, \tag{6.2.8}$$

which implies s to be valid for P.

We assume $\mathbf{u}(L) \rightharpoonup \mathbf{u}(t)$ and use the Skeletal characterisation (6.2.7). Now let P be a distribution. For all statements s of the input list L, we have: If s is valid for P, then, with Condition (6.2.8) above, the scalar product $\langle m_p, \mathbf{u}(s)\rangle$ is zero. As the scalar product is linear and every term itself is zero, we can define $\langle m_P, \mathbf{u}(L)\rangle := \sum_{s\in L} \langle m_P, \mathbf{u}(s)\rangle$ and assign $\langle m_P, \mathbf{u}(L)\rangle = 0$. Then, $\langle m_P, \mathbf{u}(t)\rangle = 0$ follows via the skeletal characterisation, as well. Condition (6.2.8) can now be applied yielding t to be valid for P. □

6.2.4 Algebraic criteria vs. graphical criteria

CI implication problems appear under different objectives in the field of learning BNS. But also the algebraic approach via structural imsets can be applied, which yields into questions of membership of vectors to $\mathcal{S}(N)$ or $\mathcal{C}(N)$, respectively. For example, the question, whether a certain CI statement is contained in the complete list of valid CI statements $\mathcal{I}_G$ of a DAG G, can be answered directly. This is graphically derivable by the d-separation criterion [23] or equivalently by the moralisation criterion (Lauritzen, Dawid, Larsen and Leimer, 1990) but also algebraically by the use of imsets.

As a service to the reader we visualise the difference of the graphical and algebraic approaches by recalling the graphical criteria. For this purpose, it is necessary to define certain notation. However the special notation for the graphical criteria is uncoupled from the rest of this chapter and the respective Example 29, page 120, does only contribute to a deeper understanding of alternative and common methods for testing the CI implication problem.

Notation: Let the set of **ancestors** of $A \subseteq N$ be all nodes participating in directed paths to nodes in A. Furthermore, let the **moral graph** of a graph G be a special underlying graph of G with additional edges $a - b$ whenever there exists a node c with $a, b \in \mathrm{pa}_G(c)$. Let $w : c_1, \ldots, c_n$, $n \geq 1$ be a route in a DAG which is understood to be a sequence of arcs whose underlying undirected edges constitute a path. Every node c_i, $i = 1, \ldots, n$, in w having arcs $c_{i-1} \to c_i \leftarrow c_{i+1}$ in w is called a **collider** with respect to w. A route w is then called **active** with respect to a set $C \subseteq N$ if every collider with respect to w belongs to the ancestors of C

and if every other node of w does not belong to C. If w is not active, we call it **blocked** by C.

Definition 6.2.9 (moralisation criterion, d-separation criterion) *Let G over N be a graph and $\langle A, B|C\rangle$ be a CI statement with $A, B, C \subseteq N$.*

- *If C separates between A, B in the moral graph of the graph induced by the ancestors of $A \cup B \cup C$, then $\langle A, B|C\rangle$ is represented in G according to the **moralisation criterion**.*
- *If every route from A to B in G is blocked by C, then $\langle A, B|C\rangle$ is represented in G according to the **d-separation criterion**.*

Lauritzen et al. (1990) showed the equivalence of both criteria. However, for visualisation we give an example for both methods taken from [60].

Example 29. Let the following graph in Figure 6.1 below with $N = \{a, b, c, d, e, f, g\}$ and the additional CI statement $\langle a, f|\{c, d\}\rangle$ be given. We set $A := \{a\}$, $B := \{f\}$ and $C := \{c, d\}$. The question is whether $\langle a, f|\{c, d\}\rangle$ is implied by the graph or not.

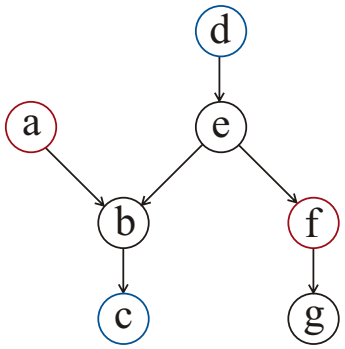

Figure 6.1: A graph over 6 nodes to demonstrate the moralisation and the d-separation criterion.

1. The ancestors of $A \cup B \cup C = \{a, f, c, d\}$ are nodes a, b, d, e. The induced subgraph together with sets A, B, C therefore does not comprise node g and edges/arcs to it. As node b has two parents a, e the moral graph has the additional edge $a - e$. The following Figure 6.2 illustrates the underlying graph of the resulting moral graph. In this graph, not all paths from a to f go through c or d (consider path $a - e - f$), and therefore $\langle a, f|\{c, d\}\rangle$ is not implied.

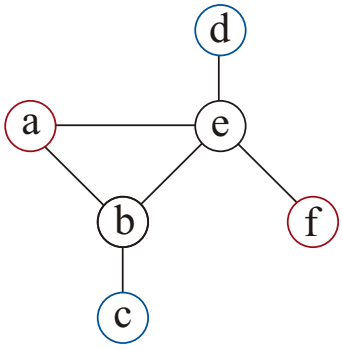

Figure 6.2: A moral graph over 6 nodes.

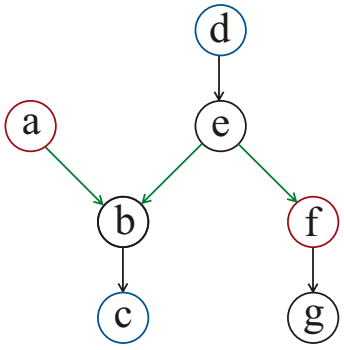

Figure 6.3: A graph over 6 nodes with an active route.

2. Consider every route from a to f, which is in this case only one, namely $a \to b \leftarrow e \to f$. We have to check whether this route is active or blocked by $\{c, d\}$. The collider in this route is b and the ancestors of $\{c, d\}$ are $\{a, b, e, d\}$. Therefore b is an element of the set of ancestors. Moreover, every other node a, e, f of the route is outside $\{c, d\}$ implying this route to be active. Statement $\langle a, f|\{c, d\}\rangle$ is not implied due to the d-separation criterion. Figure 6.3 above shows the original graph with the active route from a to f.

$\star$

Algebraically, we use structural imsets and the following fact based on results in [60] to directly derive implications on the base of graphical representations.

Proposition 6.2.10 (following from [60] and [9]) *Let G and H be DAGs over N. Then $\mathcal{I}_H \subseteq \mathcal{I}_G$ if and only if $\mathbf{u}(G) - \mathbf{u}(H) \in \mathcal{C}(N)$ and $\mathbf{u}(G) - \mathbf{u}(H) \in \mathcal{S}(N)$, respectively.*

Proof. The equivalence $\mathcal{I}_H \subseteq \mathcal{I}_G \Leftrightarrow \mathbf{u}(G) - \mathbf{u}(H) \in \mathcal{C}(N)$ is proven in Lemma 8.6 in [60]. Clearly, $\mathbf{u}(G) - \mathbf{u}(H) \in \mathcal{C}(N)$ implies $\mathbf{u}(G) - \mathbf{u}(H) \in \mathcal{S}(N)$. Conversely, if the difference $\mathbf{u}(G) - \mathbf{u}(H) \in \mathcal{S}(N)$, then, by Equivalence (6.2.3), page 117, $\mathbf{u}(G) \rightharpoonup \mathbf{u}(H)$ and the inclusion $\mathcal{M}_{\mathbf{u}(H)} \subseteq \mathcal{M}_{\mathbf{u}(G)}$ can be shown directly, because all CI statements valid for H are already implied by G. This, however, means $\mathcal{I}_H \subseteq \mathcal{I}_G$. □

Proposition 6.2.10 has the following consequence: It allows us to replace the d-separation criterion by testing whether an imset is combinatorial.

Corollary 6.2.11 (following from [60] and [9]) *Given a DAG G over N, the class $\mathcal{I}_G$ coincides with the collection of CI statements $A \perp\!\!\!\perp B \,|\, C$ such that $\mathbf{u}(G) - \mathbf{u}(\langle A, B|C\rangle)$ is a combinatorial imset and a structural imset, respectively.*

Proof. The proof is simply based on the transformation of any CI statement $A \perp\!\!\!\perp B \,|\, C$ into a graph and the application of Proposition 6.2.10. □

The problem of the graphical criteria is the inability of representing every set of CI statements graphically. Likewise, the semi-graphoid properties do not suffice. Only with the algebraic approach every implication problem can be solved. Consider the following example illustrating this problem.

Example 30. Let the following CI statements be given:

$$\langle a, b|c\rangle, \langle a, c|d\rangle, \langle a, d|b\rangle, \langle b, c|\{a, d\}\rangle.$$

These statements cannot be represented within a single graph of four nodes. An application of any graphical criterion is therefore impossible. It remains to test the semi-graphoid properties and the algebraic approach. The problem of the semi-graphoid properties is the sufficiency. If an implication is not given, then it is still not clear, whether this CI statement is valid or not. However, consider two statements $\langle a, c|b\rangle$ and $\langle b, c|a\rangle$.

To proceed with the semi-graphoid properties, we follow the different implications to construct all derivable implications. The first CI statement is among them but the second CI statement not. Only the algebraic approach can make clear, whether the second statement is implicated or not. For this purpose, consider the skeletal characterisation and a supermodular function

$$m = 2 \cdot \delta_{\{a,b,c,d\}} + \delta_{\{a,b,c\}} + \delta_{\{a,b,d\}} + \delta_{\{a,c,d\}} + 2 \cdot \delta_{\{b,c,d\}} + \delta_{\{b,c\}} + \delta_{\{b,d\}} + \delta_{\{c,d\}}.$$

For $\mathbf{u}(L) = \mathbf{u}(\langle a, b|c\rangle) + \mathbf{u}(\langle a, c|d\rangle) + \mathbf{u}(\langle a, d|b\rangle) + \mathbf{u}(\langle b, c|\{a, d\}\rangle)$ and $\mathbf{u}(t) = \mathbf{u}(\langle b, c|a\rangle)$ we have $\langle m, \mathbf{u}(L)\rangle = 0$, whereas $\langle m, \mathbf{u}(t)\rangle = 1$ proving $\langle b, c|a\rangle$ to be not implicated. $\star$

6.3 Methods for solving the independence implication problem

The algebraic characterisations, namely the direct and the skeletal ones, are clearly important when determining the implication of CI statements. Due to their exponential representation in terms of algebraic representatives the performance of the computational realisation is worth investigating. But still, the exponential representation complicates an application for higher $|N|$. Therefore we analyse other methods, as well. Some of these methods are based on linear programming, like the one we introduce in Section 6.3.2, page 123, or the one introduced by Niepert in [48] (see Section 6.5.1, page 129, for a more detailed analysis). Niepert uses a special representation of the structures of BNs, which we investigate in the corresponding Section 6.5.1, page 129, in more detail.

In this section, we describe, on the one hand, a basic method for testing CI inference algebraically (the racing algorithm or just "Racer") already presented in [9] and, on the other hand, a new method obtained by an application of linear programming (LP method). The task is equal for both methods but the LP method performs significantly better. An input list L of elementary CI statements and another elementary CI statement t over N are given and the aim is to decide whether $\mathbf{u}(L) \rightharpoonup \mathbf{u}(t)$. In addition to the computational results, we show a geometric understanding of the independence implication problem and therefore also of the original CI implication problem based on the LP method in Section 6.6, page 132.

6.3.1 "Racer" algorithm

The idea of [9] is to combine two algorithms for testing the independence implication $\mathbf{u} \rightharpoonup \mathbf{v}$. One of them, called the **verification algorithm**, is based on Condition (6.2.3), page 117, and is suitable to confirm the implication, provided it holds. However, the direct characterisation may take a long time to prove the implication not to hold, because all possible decompositions into elementary imsets have to be tested. The other algorithm, called the **falsification algorithm** based on (6.2.7), page 118, is designed to disprove the implication if it does not hold very quickly. However, it is not able to confirm $\mathbf{u} \rightharpoonup \mathbf{v}$ provided it holds. The authors of [9] call the combination of the verification and the falsification algorithm the racing algorithm.

The overall racing procedure starts with two threads, the verification one and the falsification one. Once one thread finds its proof, it stops the other one and returns its outcome.

6.3.1.1 Verification - decomposition into elementary imsets

Consider a combinatorial imset $\mathbf{u} \in \mathcal{C}(N)$, an elementary imset $\mathbf{v} \in \mathcal{E}(N)$ and the task to decide whether $\mathbf{u} \rightharpoonup \mathbf{v}$. By Condition (6.2.3), page 117, this means to test if $k \cdot \mathbf{u} - \mathbf{v}$ is a structural imset for some $k \in \mathbb{N}$. This characterisation is equivalent to

$$\exists\, l \in \mathbb{N} \text{ s.t. } l \cdot \mathbf{u} - \mathbf{v} \in \mathcal{C}(N). \tag{6.3.1}$$

This fact directly follows from Lemma 6.2.3, page 117. Indeed, as $\mathcal{C}(N) \subseteq \mathcal{S}(N)$, we have that (6.3.1) $\Longrightarrow$ (6.2.3). Now, (6.2.3) implies $n \cdot (k \cdot \mathbf{u} - \mathbf{v})$ to be a combinatorial imset for some $k, n \in \mathbb{N}$ (see Section 6.2.2, page 116, for the respective definitions). As $(n-1) \cdot \mathbf{v} \in \mathcal{C}(N)$, $\mathbf{v}$ is chosen to be an elementary imset and $n \cdot (k \cdot \mathbf{u} - \mathbf{v}) = (n \cdot k) \cdot \mathbf{u} - \mathbf{v} - (n-1) \cdot \mathbf{v} \in \mathcal{C}(N)$, we can derive $(n \cdot k) \cdot \mathbf{u} - \mathbf{v} \in \mathcal{C}(N)$. The factor $n \cdot k$ has an "upper bound" $\bar{k}$ with the following property: All vectors $\bar{k} \cdot \mathbf{u} - \mathbf{v}$ which are not combinatorial for $n \cdot k \leq \bar{k}$ are not combinatorial for any choice of $n \cdot k$. The concrete bound $\bar{k}$ is in general unknown but can be implicitly used to bound the computation. There exists the conjecture to compute $\bar{k}$ based on a mixed integer non-linear program.

The Characterisation (6.3.1) together with the implicit bound on the multipliers allows us to transform the test for independence implication to the task of deciding, whether a given candidate imset $\mathbf{y} = l \cdot \mathbf{u} - \mathbf{v}$ is combinatorial or not. A combinatorial imset $\mathbf{y}$ may have many decompositions $\mathbf{y} = \sum_{\mathbf{w} \in \mathcal{E}} \lambda_{\mathbf{w}} \mathbf{w}$, $k_{\mathbf{w}} \in \mathbb{Z}_+$ into elementary imsets but if there is no decomposition, all of of them have to be checked leading to the interpretation as a verification algorithm.

The decomposition is not limited to elementary imsets. Actually Hilbert bases can be used to find an integral decomposition of the integer valued vectors under consideration too. In case of $|N| = 3, 4$ the elementary imsets fully generate the integral points of the cone, i.e. the cone is normal, but this is not the case for $|N| \geq 5$ ([35], [10]). Already the computation of the Hilbert basis is $\mathcal{NP}$-hard and the concrete Hilbert bases for cones with $|N| \geq 6$ are unknown, which indicates that this approach is not applicable for higher dimensional examples.

6.3.1.2 Falsification - random generation of supermodular functions

The falsification is based on the Skeletal characterisation (6.2.7), page 118. Any supermodular function $m : \mathcal{P}(N) \to \mathbb{R}$ satisfying $\langle m, \mathbf{u} \rangle = 0$ and $\langle m, \mathbf{v} \rangle > 0$ suffices to disprove the implication $\mathbf{u} \rightharpoonup \mathbf{v}$. To verify the implication all, theoretically infinitely many, supermodular functions have to be tested, which is not possible. This hardness of verification leads to a preferable falsification algorithm.

Actually, we can limit the computation to a test of only l-standardised integer-valued supermodular functions. These imsets have a special form which allows us to generate them randomly. The idea is

- to randomly generate a collection of subsets of N,
- to assign them randomly selected positive integer values and zero values to the remaining sets, and
- to modify the resulting function to make it a supermodular imset.

The details of this procedure can be found in Section 3.3 of [9].

6.3.2 Linear programming based algorithm

The basic idea presented in [8] is to re-formulate the definition of independence implication in terms of the pointed rational polyhedral cone $\mathcal{R}(N) := \text{cone}(\mathcal{E}(N))$ spanned by elementary imsets. More specifically, given $\mathbf{u}, \mathbf{v} \in \mathcal{S}(N)$, the Condition (6.2.3) can be expressed in the following way:

$$\mathbf{u} \rightharpoonup \mathbf{v} \iff \exists k \in [0, \infty) \text{ with } k \cdot \mathbf{u} - \mathbf{v} \in \mathcal{R}(N). \tag{6.3.2}$$

Indeed, since $\mathcal{S}(N) \subseteq \mathcal{R}(N)$ the Implication (6.2.3) $\implies$ (6.3.2) above is evident. Conversely, provided (6.3.2) above holds with k, it holds with any $k' \geq k$ because $\mathcal{R}(N)$ is a cone and $(k' - k) \cdot \mathbf{u} \in \mathcal{R}(N)$. Therefore, there exists $k' \in \mathbb{N}$ with $k' \cdot \mathbf{u} - \mathbf{v} \in \mathcal{R}(N)$. As $k' \cdot \mathbf{u} - \mathbf{v}$ is an imset and therefore integer valued, it belongs to $\mathcal{S}(N)$, and Condition (6.2.3) holds. The point is that testing whether Condition 6.3.2 is satsfied can be viewed as the feasibility test of a LP.

Lemma 6.3.1 *Given* $\mathbf{u}, \mathbf{v} \in \mathcal{S}(N)$*, we have* $\mathbf{u} \rightharpoonup \mathbf{v}$ *if and only if the following system has a solution*

$$\begin{aligned} -\mathbf{v} &= \sum_{\mathbf{w} \in \mathcal{E}(N)} \lambda_{\mathbf{w}} \mathbf{w} - k \cdot \mathbf{u} \\ \lambda_{\mathbf{w}} &\geq 0, \ \forall \mathbf{w} \in \mathcal{E}(N) \\ k &\geq 0. \end{aligned} \tag{6.3.3}$$

Proof. The cone $\mathcal{R}(N)$ consists of conic combinations of representatives of its extreme rays. Therefore (6.3.2) can be re-written as the requirement, that $k \geq 0$ and $\lambda_{\mathbf{w}} \geq 0$ exist with $k \cdot \mathbf{u} - \mathbf{v} = \sum_{\mathbf{w} \in \mathcal{E}} \lambda_{\mathbf{w}} \mathbf{w}$. □

Note that $k \geq 0$ can be skipped, as due to the conic structure only non-negative values of k would result in a feasible problem.

The feasibility of problems of type (6.3.3) is well analysed and can e.g. be checked applying phase I of the Simplex method. In spite of the not yet proven or disproven polynomial complexity of the Simplex method, there exist algorithms (e.g. Karmarkar's algorithms, Chapter 15 in [53], which is an interior point method) providing polynomiality in $2^{|N|}$ of the overall testing procedure. In contrast to the previous presented methods, this approach is not limited to a particular number of variables (e.g $|N| \leq 5$) and problems of small dimension are solvable quickly in practice, even for $|N| = 10$.

6.4 Computational experiments

In this section we describe the results of our computational experiments. One experiment compares the "Racer" and the LP method for $|N| = 5$. The LP method easily outperforms the racing algorithm and the aim of higher dimensional experiments is to test the LP approach in case of $|N| > 5$. The racing algorithm uses a Java-program implemented by Remco R. Bouckaert similar to the one presented in [9], while the LP approach uses the commercial optimisation software CPLEX 9.1 [36]. Both computations are done on a Sun Fire V440 Ultra Sparc IIIi processor with 16GB RAM and 4×1280 MHz.

6.4.1 Comparison for 5 variables

We use the same experimental setup as in [9]. A single experiment comprises the test for the independence implication of an input list containing 3 to 11 randomly chosen elementary CI statements to all other elementary CI statements. 1000 such experiments must be solved for 3 to 11 statements. The racing algorithm takes advantage of the fact, that elementary imsets of the input list do not have to be tested explicitly, as they are implicated for sure. The LP approach simply solves all problems. The choice of the elementary statements within the input list is random.

Since the racing algorithm tends to perform poorly for several problems, the computation of one single independence implication problem is aborted if the duration time is more than one second. We do not report explicitly on these cases, as the LP approach is tremendously faster anyway. But note, that the "real" duration times of the "Racer" might be much higher. Table 6.1, next page, shows the comparison of the running times of "Racer" vs. the LP approach. Total running times in seconds are depicted, including all 1000 experiments per size of an input list and all setups and additional steps within each program.

$\|L\|$	3	4	5	6	7	8	9	10	11
Racer	669.9	1133.1	1955.1	3152.0	5042.3	7314.7	10430.7	14687.7	18014.3
LP	12.9	13.0	13.7	14.2	14.8	15.3	15.8	16.2	16.8

Table 6.1: Total computation times for the "Racer" and LP method for $|N| = 5$.

Results and discussion In spite of not being able to finish every computation and therefore somehow biased computation times, the racing algorithm has no chance to beat the LP algorithm. Moreover it suffers from an significant increase when using input lists with more contained elementary statements, whereas the LP approach is nearly constant in time. This effect can be caused by various reasons. First of all, the LP approach which is written in C/C++ uses the commercial software package CPLEX 9.1 [36] being under permanent revision and using latest algorithms and heuristics. A second reason comes from the geometric understanding of both algorithms analysed in Section 6.6, page 132.

6.4.2 Experiments for a higher number of variables

To perform the experiments for $|N| > 5$, only the LP method presented in Section 6.3.2, page 123, can be used. The experiments are defined analogously to the initial ones in Section 6.4.1. If L is an input list of elementary CI statements, then one single experiment comprises the test of $|\mathcal{E}(N)|$ many independence implications $\mathbf{u}(L) \rightharpoonup \mathbf{u}$ for all $\mathbf{u} \in \mathcal{E}(N)$. For $|N| = 4, 5, 6$, we consider input lists L containing 3 up to 11 different elementary CI statements over N and perform 1000 such experiments for each combination of $|N|$ and $|L|$. We complete 9000 experiments in total for $|N| = 4, 5, 6$. For $|N| = 7, 8, 9, 10$ we only consider 1000 experiments with L containing three elementary CI statements.

The number of LP problems to be tested within a single experiment and the dimension of these problems depends on the number $|\mathcal{E}(N)| = \binom{|N|}{2} 2^{|N|-2}$ of elementary imsets and on the number $2^{|N|}$ of subsets of N. The running times are averaged over all 9000 experiments for $|N| = 4, 5, 6$ and over all 1000 experiments for $|N| = 7, 8, 9, 10$, respectively. The running times only include the setup of the LP method and the actual LP computation. The tests are done using the commercial optimisation software CPLEX 9.1 [36] on a Sun Fire V440 Ultra Sparc IIIi processor with 16GB RAM and 4×1280 MHz.

Table 6.2 below illustrates the growth of the running times of the computations for $|N| = 4$ up to $|N| = 10$. Therein, we relate the running times with the growth of $|N|$ and the amount $|\mathcal{E}(N)|$ of problems to be tested for each experiment.

$\|N\|$	4	5	6	7	8	9	10
dimension/LP	41	113	305	801	2049	5121	12545
LPs/experiment	24	80	240	672	1792	4608	11520
time/experiment	3 ms	15 ms	96 ms	490 ms	3533 ms	119 s	3789 s

Table 6.2: The growth of computation times for the LP method for $|N| = 4$ to $|N| = 10$ related to the number of LPs and the dimension.

Results and discussion Table 6.2 illustrates a natural increase in time needed for one experiment as $|N|$ increases. This is clear as both more LPs have to be solved and they are of higher dimension. For small values of $|N|$ the increase in time occurs mainly because of the LP setup, rather than because of the computation itself. For higher $|N|$ this proportion changes. But in spite of this, even for $|N| = 7$ all 1000 experiments together can be done within minutes. For $|N| = 10$ it takes more than one month to perform all 1000 experiments. However, it takes only about one hour for each single experiment. More specifically,

on average, only about one third of a second is enough to test $\mathbf{u}(L) \rightharpoonup \mathbf{u}$, $\mathbf{u} \in \mathcal{E}(N)$, for $|N| = 10$.

Just looking at the increase of the dimensions of the LPs, we can assume ourselves to be able to test a single independence implication for up to $|N| = 15$ by solving the corresponding LP directly with CPLEX. However, testing all $|\mathcal{E}(N)|$ inference problems for one experiment is too expensive. Recall that one experiment already requires about one hour of computation time for $|N| = 10$. To go beyond $|N| = 15$ for testing $\mathbf{u}(L) \rightharpoonup \mathbf{u}$, $\mathbf{u} \in \mathcal{E}(N)$, the known structure of the matrix of the LP can still be exploited or column generation techniques can be implemented in order to solve these much bigger LPs to optimality. The special structure of the matrix is given as its columns are elementary imsets. Otherwise, characteristic imsets, more precisely portraits of standard imsets, generally provide more structural insights and a binary representation. We refer to the next Section 6.4.3 for a reformulated test comparing the representation using standard imsets and their portrait.

6.4.3 Testing independence implication with upper portraits

The result of the tests presented in Section 6.4.1 is a clear domination of the LP method, over the racing algorithm, using standard imsets to test $\mathbf{u} \rightharpoonup \mathbf{v}$ via

$$\begin{aligned} -\mathbf{v} &= \sum_{\mathbf{w} \in \mathcal{E}(N)} \lambda_{\mathbf{w}} \mathbf{w} - k \cdot \mathbf{u} \\ \lambda_{\mathbf{w}} &\geq 0, \ \forall \mathbf{w} \in \mathcal{E}(N) \\ k &\geq 0. \end{aligned} \tag{6.4.1}$$

If and only if this system has a solution, then $\mathbf{u}$ conditionally implies $\mathbf{v}$.

Portraits originally used to define characteristic imsets provide a linear transformation of this problem into a structurally simpler problem.

Conclusion 6.4.1 *The implication $\mathbf{u} \rightharpoonup \mathbf{v}$ is valid if and only if the respective implication of portraits is valid. On the other hand, this is valid if and only if the system*

$$\begin{aligned} -\operatorname{portrait}(\mathbf{v}) &= \sum_{\mathbf{w} \in \mathcal{E}(N)} \lambda_{\mathbf{w}} \operatorname{portrait}(\mathbf{w}) - k \cdot \operatorname{portrait}(\mathbf{u}) \\ \lambda_{\mathbf{w}} &\geq 0, \ \forall \mathbf{w} \in \mathcal{E}(N) \\ k &\geq 0. \end{aligned} \tag{6.4.2}$$

has a solution.

The advantage of this formulation is visible in the new coefficients of the matrix of Problem (6.4.2) above. All entries of the vector $\operatorname{portrait}(\mathbf{w})$, $\mathbf{w} \in \mathcal{E}(N)$, are zero or one, while all entries of $-\operatorname{portrait}(\mathbf{v})$ and $-\operatorname{portrait}(\mathbf{u})$ are zero or minus one. If $-\operatorname{portrait}(\mathbf{v})$ is negative for one entry, then the only possibility to necessarily satisfy the above problem is $-\operatorname{portrait}(\mathbf{u})$ to have a negative entry at the same position. From this a necessary condition on $\mathbf{u} \rightharpoonup \mathbf{v}$ can be concluded, which can also be used to implement a quick heuristic.

Remark 6.4.2 *Let* $\operatorname{supp}(\mathbf{x})$ *denote the support of a vector* $\mathbf{x}$*, i.e. the vector of its non-zero entries. If* $\operatorname{supp}(\operatorname{portrait}(\mathbf{v})) \not\subseteq \operatorname{supp}(\operatorname{portrait}(\mathbf{u}))$*, then* $\mathbf{u} \rightharpoonup \mathbf{v}$ *is not valid.*

Note that this implication is indeed only a necessary condition. The following example shows an independence implication which is not valid, but the conditions on the support of both portraits are satisfied.

Example 31. Let $N = \{a, b, c, d\}$ be a set of nodes and $\mathcal{E}(N)$ be the corresponding set of portraits of elementary statements. We choose the input list to contain statements $\langle a, c|\{b, d\}\rangle$, $\langle b, d|a\rangle$ and $\langle a, c|b\rangle$. Additionally, the single elementary statement $\langle a, c|d\rangle$ is given. We can directly observe, that the support of the portrait of $\langle a, c|d\rangle$ is a subset of the portrait of $\langle a, c|\{b, d\}\rangle$. This single inclusion already implies the support of the given elementary statements to be a subset of the portrait of the complete input list.

Analysing the linear program to solve, we observe the following elementary statements to be all candidates for the decomposition.

$$\langle a, c|\{b, d\}\rangle,\ \langle a, c|b\rangle,\ \langle a, c|d\rangle,\ \langle b, d|a\rangle,\ \langle a, c|\emptyset\rangle,\ \langle b, d|\emptyset\rangle$$

We can analyse the feasibility of the LP by considering the entries of the portraits of the elementary statements, the input list and the additional statement. The portrait of $\langle a, c|d\rangle$ has only two non-zero entries. If the portrait of the input list has non-zero entries in the other coordinates, then these entries have to be compensated with an addition of portraits of the elementary statements, e.g. $k \cdot \text{portrait}(\langle a, c|\{b, d\}\rangle) = \lambda_{\langle a,c|\{b,d\}\rangle}\, \text{portrait}(\langle a, c|\{b, d\}\rangle)$. Therefore, we can reduce the candidate list to statements $\langle a, c|d\rangle$, $\langle a, c|\emptyset\rangle$, $\langle b, d|\emptyset\rangle$ and

$$\begin{aligned} -\,\text{portrait}(\langle a, c|d\rangle) = {} & \lambda_{\langle a,c|d\rangle}\, \text{portrait}(\langle a, c|d\rangle) + \lambda_{\langle a,c|\emptyset\rangle}\, \text{portrait}(\langle a, c|\emptyset\rangle) \\ & + \lambda_{\langle b,d|\emptyset\rangle}\, \text{portrait}(\langle b, d|\emptyset\rangle),\ \lambda_{\langle a,c|d\rangle} \geq 0,\ \lambda_{\langle a,c|\emptyset\rangle} \geq 0,\ \lambda_{\langle b,d|\emptyset\rangle} \geq 0. \end{aligned}$$

Hence, no element of the input list is left and the resulting linear problem is not feasible due to non-positive entries on the left hand side. ⋆

Moreover, LP solvers can take advantage of these simpler coefficients, as well. Extensive presolving techniques may lead to necessary conditions similar to above reducing the time needed for the optimisation step.

But this formulation has also a disadvantage. The higher dimensional experiments of Section 6.4 show the actual solving of the LP to be of small percentage of the overall computational process. The higher percentage comes from the setup of the LP and from initialising CPLEX [36]. With standard imsets we can use an efficient sparse encoding of the vectors and the coefficient matrix. Each elementary imset has exactly four non-zero entries of values one and minus one. In each step of the experiment only these four non-zero values have to be changed. If we use the formulation of the LP using the portraits (Problem (6.4.2)), then up to $2^{|N|-2}$ values have to be changed per LP. Although there exist portraits of elementary imsets which have less entries, the overall expected effect is a more extensive setup step necessary to formulate and to solve one single LP within the experiments. The experiments in the next section show the new trade off between the setup and the solving of the LPs.

In Example 31 above we already use a further simplification of the problem leading into a smaller problem description. Portraits of elementary statements only have non-negative entries. One of them is allowed to participate in the decomposition if it has an entry one only at a position at which also the portrait of the input list has a non-zero entry. Otherwise, we conclude the corresponding row in the left hand side of System (6.4.2) to be positive and the right hand side to be non-positive. We reduce Problem (6.4.2) to the following system

$$\begin{aligned} -\,\text{portrait}(\mathbf{v}) = {} & \sum_{\substack{\mathbf{w}\in\mathcal{E}(N),\\ \text{supp}(\text{portrait}(\mathbf{w}))\subseteq\text{supp}(\text{portrait}(\mathbf{u}))}} \lambda_{\mathbf{w}}\, \text{portrait}(\mathbf{w}) - k \cdot \text{portrait}(\mathbf{u}) \\ & \lambda_{\mathbf{w}} \geq 0,\ \forall \mathbf{w} \in \mathcal{E}(N) \\ & k \geq 0. \end{aligned} \tag{6.4.3}$$

This reduction of the problem comprises the tests: $\text{supp}(\text{portrait}(\mathbf{w})) \subseteq \text{supp}(\text{portrait}(\mathbf{u}))$, for all possible elementary statements $\mathbf{w} \in \mathcal{E}(N)$. Besides this overhead, the time actually

needed to solve the LP should decrease. Especially for higher dimensions, this problem reduction has a positive effect on the performance of the LP method.

6.4.3.1 Computational experiments

We use the same setup of the experiments as presented in Section 6.4.2. For different values of $|N|$, we solve all 1000 types of implication problems with successively increasing sets L for $|N| = 4, 5, 6$ and report the mean value of a single one. We compare the LP method using a formulation with standard imsets with a formulation using their portraits by just solving the LPs directly with CPLEX. The third compared method consists of a LP formulation using portraits but instead of simply solving it with CPLEX we implement Heuristic 6.4.2 in advance and additionally reduce the system according to (6.4.3). Computations are done on a machine with 4 AMD-Opteron 6174 12-core CPUs with 2.2 GHz and a total of 128GB RAM using the commercial optimisation software CPLEX 12.1 [36].

$\|N\|$	4	5	6	7	8	9	10
dimension/LP	41	113	305	801	2049	5121	12545
LPs/experiment	24	80	240	672	1792	4608	11520
	time/experiment						
Standard imsets	0.947 ms	5.326 ms	36.426 ms	175.983 ms	2.490 s	omitted	
Portraits	0.689 ms	3.679 ms	22.842 ms	187.044 ms	2.674 s	omitted	
Portraits, reduced	1.762 ms	4.604 ms	12.809 ms	21.216 ms	0.094 s	0.641 s	4.540 s

Table 6.3: The growth of computation times for the LP method for $|N| = 4$ to $|N| = 10$ using different representations of elementary statements.

6.4.3.2 Results and Discussion

Comparing the standard imsets and their portrait formulations, we can observe a faster computation using portraits for smaller sizes of $|N|$ (up to 6). This changes for bigger $|N|$. One reason is that the amount of non-zero entries in the portraits is exponential in $|N|$. It simply becomes too costly to copy the corresponding entries in the coefficient matrix. But for smaller $|N|$, CPLEX is able to solve the corresponding problems faster and there is no reason why this should not also be the case for larger $|N|$.

As expected, an additional implementation of the heuristic to check a necessary condition and a reduction of the system is quite effective and causes a huge improvement of running times. The quick support test of the heuristic causes both a LP computation and a LP setup to be skipped. Of course, the support still is a vector of exponential dimension in $|N|$. This causes the overhead for smaller problem sizes. However, non-valid implications occur quite often and a falsification with the help of the support can be given quickly in most cases. The same reason causes an additional benefit from using the reduction of the system. In fact, this reduction has the main influence on the tremendous speed-up.

The transformation into portraits shows this problem to inherit structural properties which enable us to solve the implication problem fast. Moreover, the necessary conditions above imply a reduction of the problem to a significantly smaller problem. A naturally raised conjecture is:

Conjecture 6.4.3 *Let L be a list of elementary statements and $\mathbf{u}$ be a single elementary imset over N. The independence implication problem $\mathbf{u}(L) \rightharpoonup \mathbf{u}$ can be solved in polynomial time in $|N|$ and $|L|$.*

To show the validity of this conjecture, we have to show that the conditions above lead into only polynomial many elementary imsets as candidates for the decomposition.

6.5 Related work

6.5.1 Representation due to Niepert [48]

In [48] and [50] an alternative representation is introduced. Niepert defines representatives of CI statements as 0/1-vectors to obtain a formulation of the CI implication problem based on linear programming. We identify his vector representation as related to the characteristic imset representation.

Definition 6.5.1 *Let $s = \langle A, B|C\rangle$ be a CI statement over node set N and*

$$\mathcal{L}(A, B|C) := [C, N] \setminus ([A, N] \cup [B, N])$$

*be its corresponding **semi-lattice** with $[A, B] := \{T : A \subseteq T \subseteq B\}$. Then $\mathbf{v}(s)$ with $v_T(s) = 1$ if $T \in \mathcal{L}(A, B|C)$, and zero otherwise, defines the **vector representative** of s.*

Modifying this definition to a direct characterisation avoiding the semi-lattice notation yields:

$$\begin{aligned}\mathcal{L}(A, B|C) &= [C, N] \setminus ([A, N] \cup [B, N]) = ([C, N] \setminus [A, N]) \cap ([C, N] \setminus [B, N]) \\ &= (\{T : C \subseteq T \subseteq N\} \setminus \{T : A \subseteq T \subseteq N\}) \\ &\qquad \cap (\{T : C \subseteq T \subseteq N\} \setminus \{T : B \subseteq T \subseteq N\}) \\ &= \{T : C \subseteq U, A \nsubseteq T\} \cap \{T : C \subseteq T, B \nsubseteq T\} \;=\; \{T : C \subseteq T, A, B \nsubseteq T\}.\end{aligned}$$

in terms of Definition 6.5.1 above this means

$$v_T(s) = 1 \iff C \subseteq T \text{ and } A, B \nsubseteq T. \tag{6.5.1}$$

Since $A, B \neq \emptyset$, we can suppose $v_T(s) = 0$ if $|T| \geq |N| - 1$, for all CI statements s. Therefore $\mathbf{v}(s)$ can be projected onto its first $2^{|N|} - |N| - 1$ components.

6.5.2 Testing independence implication with Niepert [48]

Analogous to the approach presented in Section 6.3, page 122, Niepert [48] introduces a falsification and a verification algorithm to decide independence implication. Moreover, the falsification is also used to compute the actual necessary input to the validation algorithm.

The question for implication remains the same. Let L be a set of given CI statements and t a single one. The question is, whether t is implicated by L or not.

To falsify this statement Niepert uses the following property:

Proposition 6.5.2 (Proposition 4.2 in [48], and references therein) *Let L be a set of CI statements and let t be a single CI statement. If $\mathcal{L}(L) \not\supseteq \mathcal{L}(t)$, then t is not implied by L.*

A semi-lattice of several CI statements $\mathcal{L}(L)$ is defined to be the union of the semi-lattices of each of the CI statements.

Example 32. *Example 30, page 121, continued.* We use the Example 30 presented in Section 6.2.4, page 119. Recall, $N = \{a, b, c, d\}$ to be the set of nodes, L a list of CI

statements $\langle a, b|c\rangle$, $\langle a, c|d\rangle$, $\langle a, d|b\rangle$ and $\langle b, c|\{a, d\}\rangle$ and t a single CI statement $\langle b, c|a\rangle$. We get,

$$\begin{aligned}\mathcal{L}(L) &= \mathcal{L}(a, b|c) \cup \mathcal{L}(a, c|d) \cup \mathcal{L}(a, d|b) \cup \mathcal{L}(b, c|\{a, d\}) \\ &= \{\{b\}, \{c\}, \{d\}, \{c, d\}, \{b, d\}, \{b, c\}, \{a, d\}\}\end{aligned}$$

and, contrarily,

$$\mathcal{L}(t) = \{\{a\}, \{a, d\}\},$$

which is no subset of $\mathcal{L}(L)$ and t is not implied by L. ⋆

For validation Niepert first observes that not all CI statements in L are necessary to implicate t. In fact, only the elementary statements within L are important. As already defined $\mathcal{E}(N)$ denotes the set of elementary statements over node set N.

Definition 6.5.3 *Let L be a set of CI statements over N. The set of **relevant elementary CI statements** $\mathcal{R}(L)$ is defined as follows:*

$$\mathcal{R}(L) = \{A \perp\!\!\!\perp B \,|\, K \in \mathcal{E}(N) : \mathcal{L}(A, B|K) \subseteq \mathcal{L}(L)\}.$$

The reduced set $\mathcal{R}(L)$ can now be computed using the falsification algorithm to test the inclusion $\mathcal{L}(A, B|K) \subseteq \mathcal{L}(L)$.

Using his vector representation and setting $\mathbf{v}(L) := \sum_{l \in L} \mathbf{v}(l)$, the validation of L to imply t turns into the following characterisation:

Proposition 6.5.4 (Proposition 4.9 in [48]) *Let L be a set of CI statements over N and let $\mathbb{Q}_+$ be the non-negative rational numbers. Then, L implies t if*

$$\mathbf{v}(L) = \mathbf{v}(t) + \sum_{e \in \mathcal{E}(N)} k_e \cdot \mathbf{v}(e) \text{ with } k_e \in \mathbb{Q}_+ \tag{6.5.2}$$

has a solution.

We skip the corresponding proof and observe this characterisation to coincide with the definition of $\mathbf{u}(L) \rightharpoonup \mathbf{u}(t)$ instead. In particular, the condition above only solves the CI implication problem approximately, as well. The following section specifies the relation between standard imsets and the vector representation.

For testing Criterion (6.5.2) computationally a LP method is used similar to the one in Section 6.3.2, page 123. The columns of the coefficient matrix now only consist of those elementary statements surviving the initial falsification step, i.e. those in $\mathcal{R}(L)$. Niepert compares the racing method of [9] with his LP approach up to 15 variables. The corresponding computational results can be seen in [48]. They show a tremendous simplification of the method, which we also observe in [8] and in the previous sections.

Moreover to the computational method presented in this section, Niepert shows the co$\mathcal{NP}$-completeness of the implication problem restricted to a certain class of CI statements which are called *stable CI statements* in [49].

6.5.3 Relation of Niepert [48] and characteristic imsets

We analyse the relation of this representative with the definition of a standard imset describing a CI statement and get the following corollary similar to the results from Lemma 4.7 in [48].

Lemma 6.5.5 *Let $s = \langle A, B|C\rangle$ be a CI statement, $\mathbf{u}(s)$ the corresponding standard imset and $\mathbf{v}(s)$ its vector representation according to [48], then*

$$v_T(s) = \sum_{U:U\subseteq T} u_U(s),\ \forall T \subseteq N. \tag{6.5.3}$$

Proof. We show this by transforming Equation (6.5.3) above into Equation (6.5.1).

First of all, by definition $\mathbf{u}(s) = \mathbf{u}(\langle A, B|C\rangle) = \delta_{A\cup B\cup C} + \delta_C - \delta_{A\cup C} - \delta_{B\cup C}$. We suppose $A, B \neq \emptyset$. If set T in Equation (6.5.3) above has A or B as its subset and $C \nsubseteq T$, then we can distinguish between $C = \emptyset$ or not. If $C = \emptyset$, then we have to add $\delta_C - \delta_{A\cup C} = 0$, $\delta_C - \delta_{B\cup C} = 0$ or $\delta_C - \delta_{B\cup C} - \delta_{A\cup C} + \delta_{A\cup B\cup C} = 0$, respectively. If $C \neq \emptyset$, then all summands are zero. In both cases we have $v_T(s) = 0$. If $C \subseteq T$ and $A, B \nsubseteq T$, then only δ_C appears in the sum and $v_T(s) = 1$. Analogously, $A \cup C \subseteq T$, $B \cup C \subseteq T$ and $A \cup B \cup C \subseteq T$ yield $v_T(s) = 0$. Summing up, only in case $C \subseteq T$ with $A, B \nsubseteq T$, we have $v_T(s) = 1$, which is equivalent to Equation (6.5.1) implying both formulas to be equivalent. □

Note that, with respect to our upper portrait notation, this definition of a vector representation can be seen as taking the **lower portrait** of a standard imset. On the other hand, this representation shows the existence of an affine linear transformation between elementary imsets and the vector representation of Niepert. As the characteristic imset is also an affine transformation of imsets, we can now look for a direct relation of characteristic imsets and the Vector representation (6.5.3).

Corollary 6.5.6 *Let $s = \langle A, B|C\rangle$ be a CI statement, $\mathbf{u}(s)$ the corresponding standard imset, $\mathbf{v}(s)$ its vector representation according to [48] and $\mathbf{c}(s)$ the characteristic imset of s. There exists an affine transformation between $\mathbf{c}(s)$ and $\mathbf{v}(s)$ using $\mathbf{u}(s)$:*

$$\mathbf{c}(s) = \mathbf{1} - \mathbf{u}(s) + \mathbf{v}(s). \tag{6.5.4}$$

Proof. For all $T \subseteq N$ and $\sum_{U:U\subseteq N} u_U(c) = 0$ we have

$$\begin{aligned} c_T(s) = 1 - \sum_{U:T\subseteq U\subseteq N} u_U(s) &= 1 - \left(\sum_{U:U\subseteq N} u_U(s) - \sum_{U:U\subseteq T} u_U(s) + u_T(s) \right) \\ &= 1 + v_T(s) - u_T(s). \end{aligned}$$

Equation (6.5.4) is implied. Slight modifications need to be made if both $\mathbf{c}(s)$ and $\mathbf{v}(s)$ are given in their projected version. Then $c_T(s) = 1 - u_T(s) + 0$ if $|T| \geq |N| - 1$ and $v_T(s) = u_T(s)$ for $|T| \leq 2$. □

In contrast to the characteristic imset the vector representation of Niepert is a 0/1-vector only in case of standard imsets of CI statements, whereas the entries of characteristic imsets are in $\{0, 1\}$ for all standard imsets.

Example 33. Consider G a DAG over N with four nodes and the corresponding imset $\mathbf{u}(G) = 2 \cdot \delta_\emptyset - \delta_a - \delta_b - \delta_d + \delta_{a\cup b\cup d}$, then $\mathbf{v}(G) = (2\,1\,1\,2\,1\,0\,1\,0\,1\,0\,0\,1\,0\,0\,0\,0)$ which is not a 0/1-vector. ⋆

In spite of this, many properties valid for characteristic imsets can also be proven for the vector representation. This duality can also be used for the converse implication. Characteristic imsets, more precisely the portrait map, can be used to computer testing the independence implication analogous to the vector representation. Likewise to the results by Niepert, the

computer testing with portraits of standard imsets (see Section 6.4.3, page 126) leads to a tremendous simplification and smaller computation times. In fact, due to Lemma 6.5.5, Corollary 6.5.6 and Heuristic 6.5.2 both formulations are equivalent.

6.6 Geometric interpretation of the independence implication problem

6.6.1 Independence implication and representations of cones

The direct characterisation, i.e. the verification part in the racing algorithm, translates into the characterisation of elements of cones as already presented in Section 1.2.2.1, page 12. On the one hand, we know every elementary CI statement to be an elementary imset and moreover an extreme ray of a cone. On the other hand, we know every other CI statement to be a conic combination of these elementary imsets. Therefore, talking about CI statements means talking about membership of cones. With respect to the direct characterisation this cone is given in its inner representation. The geometric interpretation of the direct characterisation can be seen in Figure 6.4 below.

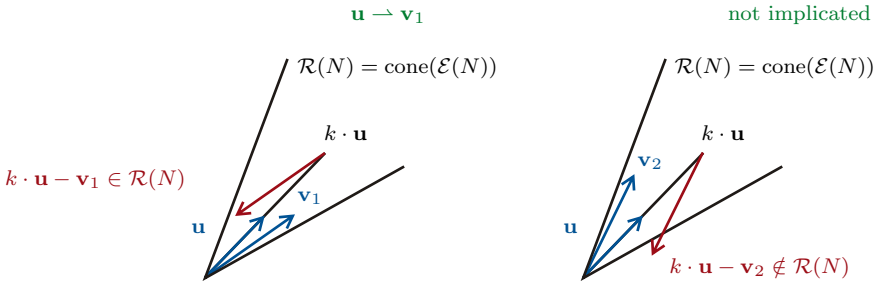

Figure 6.4: Geometric interpretation of the direct characterisation of the independence implication problem for valid and for invalid implications.

If $\mathcal{R}(N)$ is given via its outer description $\mathcal{R}(N) = \mathrm{cone}(\mathcal{E}(N)) = \{\mathbf{u} : \langle m, \mathbf{u}\rangle \geq 0\}$, then we talk about the skeletal characterisation. The supermodular functions are, in fact, the faces m of the cone $\mathcal{R}(N)$. The skeletal characterisation requires every face generated by $\mathbf{u}$ to contain the face generated by $\mathbf{v}$. Analogously, Figure 6.5 below shows the skeletal characterisation via the faces of the cone.

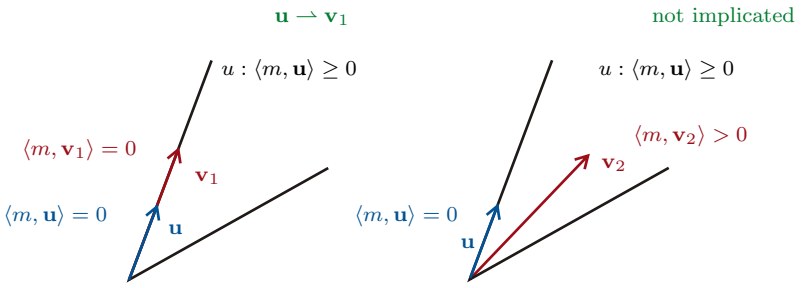

Figure 6.5: Geometric interpretation of the skeletal characterisation of the independence implication problem for valid and for invalid implications.

The pure skeletal characterisation implies to compute the outer description of the cone explicitly. This is computationally "impossible" for $|N| > 5$, for this reason the falsification algorithm creates supermodular functions, i.e. potential faces of the cone, randomly and tries to find one containing $\mathbf{u}$ but not $\mathbf{v}$.

6.6.2 Direct characterisation vs. primal of LP and skeletal characterisation vs. dual of LP

We get an immediate correspondence of the LP approach obtained by the decomposition into elementary imsets and of the dual of the LP obtained by the skeletal description. Moreover, both the verification and the falsification algorithm are much easier to solve when represented as a LP. This can easily be seen when analysing the LP formulations of both characterisations and their consequences for the algebraic characterisation.

6.6.2.1 Duality of the LP formulations

We already analysed the duality of the direct characterisation and the skeletal characterisation. This can, of course, likewise be seen in the LP problems, and more specifically, in their corresponding polytopes as already mentioned in [59] and references therein. We show both problems to be dual if the skeletal characterisation is given in a non-standardised form and analyse the corresponding reformulated polytopes with respect to duality and relaxation, otherwise.

We start by recalling the explicit LP formulation of Problem (6.3.3), page 124, and the one given by the skeletal characterisation in [59] and references therein.

$$\max\{\langle m, \mathbf{v}\rangle : \langle m, \mathbf{u}\rangle = 0, m(S) = 0 \text{ for } |S| \leq 1, \langle m, \mathbf{w}\rangle \geq 0,\ \forall w \in \mathcal{E}(N)\} \quad \text{and} \tag{6.6.1}$$

$$\min\{0 : k \cdot \mathbf{u} - \mathbf{v} = \sum_{\mathbf{w}\in\mathcal{E}(N)} \lambda_{\mathbf{w}}\mathbf{w}, \lambda_{\mathbf{w}} \geq 0,\ \forall \mathbf{w} \in \mathcal{E}(N)\}. \tag{6.6.2}$$

The implication $\mathbf{u} \rightharpoonup \mathbf{v}$ is valid if and only if Problem (6.6.1) above has the optimal value zero and, analogously, $\mathbf{u} \rightharpoonup \mathbf{v}$ if and only if Problem (6.6.2) above has a solution. Note that the red-shaded conditions vanish in the non-standardised version. We call the respective objective values M for Problem (6.6.1) and $k_{\mathbf{u}\rightharpoonup\mathbf{v}}$ for Problem (6.6.2).

We can now investigate both problems to be "nearly" dual. We call them "nearly" dual as there exist modifications, with respect to the l-standardisation, of both problems leading to dual formulations, i.e. we consider new problems.

$$\min\{0 : k \cdot \mathbf{u} + \begin{pmatrix}\mathbf{s}\\ \mathbf{0}\end{pmatrix} - \mathbf{v} = \sum_{\mathbf{w}\in\mathcal{E}(N)} \lambda_{\mathbf{w}}\mathbf{w}, \lambda_{\mathbf{w}} \geq 0,\ \forall \mathbf{w} \in \mathcal{E}(N), \mathbf{s} \in \mathbb{R}^{|N|+1}\} \quad \text{and} \tag{6.6.3}$$

$$\max\{m^{\intercal}\mathbf{v} : m^{\intercal}\mathbf{u} = 0, m^{\intercal}\mathbf{w} \geq 0,\ \forall \mathbf{w} \in \mathcal{E}(N)\} \tag{6.6.4}$$

and call their objective values $\bar{k}_{\mathbf{u}\rightharpoonup\mathbf{v}}$ and $\bar{M}$. With these additional problems we can get a strong duality conclusion for the two initial problems.

Lemma 6.6.1 *Let Problems (6.6.1), (6.6.2), (6.6.3) and (6.6.4) be given. The following relations are valid.*

1. *Problems (6.6.1) and (6.6.3) are dual problems and Problems (6.6.2) and (6.6.4) are dual problems, as well.*
2. *The optimal objective values of all problems coincide, i.e. $M = k_{\mathbf{u}\rightharpoonup\mathbf{v}} = \bar{k}_{\mathbf{u}\rightharpoonup\mathbf{v}} = \bar{M} = 0$ if and only if $\mathbf{u} \rightharpoonup \mathbf{v}$.*
3. *If Problem (6.6.2) is feasible, then Problem (6.6.3) as well, and if Problem (6.6.1) is feasible, then Problem (6.6.4) is feasible, too.*

Proof. We show the first statement by explicitly constructing the dual formulation of the initial Problems (6.6.1) and (6.6.2).

For the sake of simplicity in indexing, let the sets $S \subseteq N$ be ordered, according to increasing $|S|$. To apply the LP duality we reformulate Problem (6.6.1) to get an inequality system in standardised form: $\max\{\mathbf{c}^\intercal\mathbf{m} : \mathbf{A}\mathbf{m} \leq \mathbf{b}\}$. To obtain this, we define an objective vector $\mathbf{c} := \mathbf{v}$, a right hand side $\mathbf{b} := \mathbf{0}$ and a coefficient matrix

$$\mathbf{A}^\intercal := \left(\begin{array}{cc|cccc|ccc} \mathbf{u} & -\mathbf{u} & -I_{|N|+1} & \mathbf{0} & I_{|N|+1} & \mathbf{0} & -\mathbf{w}_1 & \dots & -\mathbf{w}_{|\mathcal{E}|} \\ & & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & & & \end{array}\right)$$

with $(|N|+1)\times(|N|+1)$-identity matrices $I_{|N|+1}$. Then $\mathbf{A}$ is a $(2\cdot 2^{|N|}+|\mathcal{E}(N)|+2)\times(2^{|N|})$-dimensional matrix. The corresponding LP dual is

$$\min\{\mathbf{y}^\intercal\mathbf{b} : \mathbf{y}^\intercal\mathbf{A} = \mathbf{c}, \mathbf{y} \geq \mathbf{0}\}.$$

We now have to replace the defined shortenings and reformulate again, to get a shorter dual formulation. Therefore we substitute the difference of two bounded variables $-y_1 + y_2$ with an unbounded variable k and likewise the differences $-y_{2+j} + y_{2^{|N|}+2+j}$, for $j = 1, \dots, |N|+1$, with unbounded variables s_j. Furthermore we rename each variable $y_{2+2^{|N|}+i}$ as λ_i, for all $i = 1, \dots |\mathcal{E}|$. This simplifies the system to

$$\begin{array}{lllll} \min & 0 & & & \\ s.t. & -u_1 k & +s_1 & -\sum_{\mathbf{w}\in\mathcal{E}(N)} w^1\lambda_{\mathbf{w}} & = v_1 \\ & \dots & & & \\ & -u_{|N|+1}k & +s_{|N|+1} & -\sum_{\mathbf{w}\in\mathcal{E}(N)} w^{|N|+1}\lambda_{\mathbf{w}} & = v_{|N|+1} \\ & -u_{|N|+2}k & & -\sum_{\mathbf{w}\in\mathcal{E}(N)} w^{|N|+2}\lambda_{\mathbf{w}} & = v_{|N|+2} \\ & \dots & & & \\ & -u_{2^{|N|}}k & & -\sum_{\mathbf{w}\in\mathcal{E}(N)} w^{2^{|N|}}\lambda_{\mathbf{w}} & = v_{2^{|N|}} \\ & & & & \lambda_{\mathbf{w}} \geq 0,\ \forall \mathbf{w} \in \mathcal{E}(N) \end{array}$$

which we can then reduce to Problem (6.6.3).

To show the second duality, we start with Problem (6.6.2) and compute its corresponding dual. The reformulation techniques are similar to the previous case such that we can directly state Problem (6.6.4) to be the dual one of (6.6.2).

The second statement follows from the definition of the problems. Both M and $k_{\mathbf{u}\rightharpoonup\mathbf{v}}$ are defined to be zero if the implication is valid, i.e. the implication problem is feasible and the LPs are attaining optimal values. Since Problems (6.6.3) and (6.6.4) are their dual problems, their optimal objective values coincide. We likewise get $\bar{M}$ and $\bar{k}_{\mathbf{u}\rightharpoonup\mathbf{v}}$ to be zero and also the other way around: If $\bar{M} = 0$ and $\bar{k}_{\mathbf{u}\rightharpoonup\mathbf{v}} = 0$, then $M = 0$ and $k_{\mathbf{u}\rightharpoonup\mathbf{v}} = 0$.

To show the third fact we observe the LP formulations to differ in an addition of $\binom{\mathbf{s}}{\mathbf{0}}$ to Problem (6.6.3) and a removal of $m(S) = 0$, for all $|S| \leq 1$, in Problem (6.6.1), i.e. the red shaded l-standardisation. This leads in both cases in a formal increase in the amount of feasible solutions. Note that, in spite of this formal increase, the optimal objective values are still bounded by zero due to the duality. □

The l-standardisation is useful for the skeletal characterisation, because it reduces the possible set of supermodular functions to be tested. But it does not change the implication problem.

By the duality of the LP approaches, we can now find a corresponding consequence for the direct characterisation. Problem (6.6.3) is equal to Problem (6.6.2) with the additional summation of the vector $\binom{\mathbf{s}}{\mathbf{0}}$, where $\mathbf{s}$ is not limited in its sign. For all choices of $\mathbf{u}$ and $\mathbf{v}$ the first entries corresponding to sets $|S| \leq 1$ are irrelevant. Of course, there always exists a value s_S which can be added to satisfy the conic representation of extreme rays. This means, we can limit the direct characterisation to a projection of (elementary, structural, combinatorial, etc.) imsets onto their remaining entries for $S \subseteq N$ with $|S| \geq 2$. In simpler terms, $\mathbf{u} \rightharpoonup \mathbf{v}$ if and only if the projection of $k \cdot \mathbf{u} - \mathbf{v}$ lies in the corresponding projection of the cone generated by elementary imsets, see above. This fact comes from the definition of standard imsets which are especially structural imsets. The entries corresponding to single nodes and the empty set are a direct consequence of the remaining entries.

This fact can likewise be derived by means of the portrait formulation of this problem (compare with Section 6.4.3, page 126). With respect to the Conclusion 6.4.1, page 126, we get a direct linear(!) transformation of the LP formulations mentioned above. This shows entries of sets corresponding to single nodes to be irrelevant for deciding the implication problem. Statements 2 and 3 of Lemma 6.6.1 can now be concluded directly as the relaxed problems are equal to the original ones.

6.6.2.2 Consequences for the presented algorithms

Both the verification part of the racing algorithm and the LP method try to find a decomposition of $k \cdot \mathbf{u} - \mathbf{v}$ with respect to the extreme rays $\mathcal{E}(N)$. The verification part performs a depth search over $\mathcal{E}(N)$ trying to find a sum of elementary imsets yielding $k \cdot \mathbf{u} - \mathbf{v}$ for a previously fixed k. In worst case, this means a complete enumeration, i.e. a construction of all sums of elementary imsets. Furthermore k is chosen to be fixed to its upper bound. The higher this value and hence $k \cdot \mathbf{u}$, the more possible decompositions exist and the higher the coefficients of the single elementary imsets are.

In contrast to this the LP approach performs several of these steps simultaneously. First of all, k is not fixed and is chosen appropriately during the computation. Secondly, since for example the primal simplex method only uses feasible steps, only these elementary imsets leading to a feasible solution are subtracted if the original solution has been feasible.

Analogously, the falsification has to search for an appropriate face, whereas the LP approach can answer this by applying for example the first phase of the simplex method and hence, again, by constructing a LP to be solved.

Conclusion

Summary of results and contributions

Based on an affine linear transformation of well-known algebraic representatives of structures of Bayesian networks we have derived a new binary representation. We call these representatives characteristic imsets to follow the notion of their algebraic counterparts, so-called standard imsets [60]. We have shown these characteristic imsets to be a binary but exponential representation of Markov equivalent graphs, i.e. structures of Bayesian networks.

We have used the affine transformation between standard imsets and characteristic imsets to derive properties of both representatives. The advantage of the use of characteristic imsets is the easier derivation of theoretical results. By the use of binary representatives we have shown that the proof of the set of standard imsets to constitute a polytope can be reduced to a note. Furthermore, we have shown this binary representation to imply the polytopes of both types of imsets not to contain other integral vectors than the imsets. We have used the other direction of the transformation to show that a decomposable and score equivalent score function is linear when applied to characteristic imsets. We have summarised the relation of both polytopes as an isomorphic relation.

Another advantage of characteristic imsets is their direct graphical translation. The edges of the underlying graph, i.e. its characteristic vector, are directly represented by entries of the characteristic imsets. But also other entries have a direct consequence on the graphical representative of Markov equivalent graphs. If the graph is chordal, then the characteristic imset has entry one if and only if the corresponding set induces a clique in the graph. We have concluded the graphical interpretation of particular entries corresponding to sets of cardinality two and three of the characteristic imsets to result in a direct reconstruction of the pattern graph, a representation of Markov equivalent graphs. All other entries of characteristic imsets can be derived iteratively, that we have illustrated to be a non-linear map. Other graphical representatives of Bayesian network structures can be obtained using so-called orientation rules, as well.

For the standard imset polytope and for the characteristic imset polytope we have derived and analysed inequalities describing their relaxation. Any inequality description of both polytopes uses an exponential, in the number of nodes, amount of variables and inequalities. Therefore, from a practical point of view, heuristics similar to the Simplex method and edges of the polytope are far more interesting than solution algorithms based on a direct description of the polytopes. Edges in the polytope lead to the notion of geometric neighbours of vertices. We have contributed several types of new geometric neighbours of the characteristic imset polytope. We have shown these geometric neighbours to be a direct consequence of the binary encoding and we have shown the overall polytope to decompose into a Minkowski sum of smaller polytopes. The same geometric neighbours and the same decompositions are valid for the standard imset polytope. But the easier derivation with respect to the characteristic imset polytope again shows the introduction of characteristic imsets to enable us to derive results more easily than via standard imsets.

For the characteristic imset polytope we have derived a complete description and a LP-relaxation of an extended formulation of it. For the derivation of both descriptions we have

used graphical properties of characteristic imsets. These properties define a binary vector to satisfy the notion of a characteristic imset and a characteristic imset to define a valid Markov equivalence class of a directed acyclic graph. The extended formulation moreover uses a direct encoding of the directed acyclic graph given by the acyclic subgraph polytope [25].

The introduced inequality description and characteristic imsets have allowed us to derive LP-relaxations of subpolytopes in an easy way. The description of undirected forests moreover directly leads into the polytope of the well-known cycle matroid for which the description is known to be both a facet and a TDI (totally dual integral) description. Moreover, we have derived descriptions of polytopes of undirected forests and spanning trees with additional bounds on the degrees of all nodes and of chordal graphs with and without bounds on the cardinality of cliques.

Combinatorial representatives like characteristic imsets make it possible to analyse complexity results of learning restricted Bayesian network structures theoretically. For this analysis, we have only supposed minor and common assumptions on the score function and the representation with characteristic imsets. We have both extended and reproduced already known results.

Based on the derived inequality descriptions of learning Bayesian network structures we have implemented computational experiments verifying our polyhedral approach to be powerful for solving several structure learning tasks, unrestricted or not. First of all, we have followed known results about the incorporation of the score function to get tremendous simplifications of the problem description ([17] and [37]). These simplifications we have shown to be applicable for the characteristic imsets polytope for general directed acyclic graphs but not for chordal graphs. For the first problem, we have used the simplifications to reduce both the number inequalities and variables of our description. Other methods have been introduced to influence the performance of the computations, such as problem specific row generation techniques.

For several freely available databases we have been able to compute optimal structures which we have additionally analysed with respect to the used descriptions and computational experiments. The results of these experiments show in particular the simplification of the description to be necessary and useful. We have analysed the row generation with additional problem specific parameters and observed concrete examples to yield further simplifications. Together with efficient programming techniques these simplifications may enable us to compute optimal structures also for a higher number of variables. This we have validated by analysing fixed databases. For these special databases the structure is more important than the actual size of the description. Using again state-of-the-art software like CPLEX, quick optimisation procedures, especially fitted to binary problems, can be exploited. These preliminary tests show the applicability of our approach and the strength of state-of-the-art optimisation software for learning Bayesian network structures.

In this thesis we have analysed and presented a new method for solving the conditional independence implication problem sufficiently. This method reduces the conditional independence implication problem to the feasibility test of a linear program. We call the transformed problem the implication problem.

Computational experiments for problems of five variables have been done to compare an already known method “Racer” [9] with the new linear programming based algorithm to solve the implication problem. These computations show “Racer” to be much slower than the linear programming based method in solving the implication problem. In contrast to “Racer” the linear programming based method is not sensitive to the size of the input list, as well. This positive result is mainly due to the use of the state-of-the-art optimisation software CPLEX [36].

To show the applicability of the linear programming method tests for higher number of variables are done, as well. For these tests "Racer" has not been used as it is practically infeasible. The reason are both an imposed bound and a facet description, which are unknown for more than five variables. The higher dimensional computations show this particular bound in applicability not to exist for the linear programming based method. Although the linear programming based method uses an exponential representation in the number of variables, single feasibility tests of linear programs can still be done fast, also for a larger amount of variables, e.g. 10 variables. In spite of this, higher dimensional problems comprise to solve many linear programs. The overall computation time of the feasibility tests of the linear programs occurs mainly because of the setup of these programs and because of the amount of them to solve. The proportion between time necessary for the computation and for the setup changes with an increasing amount of variables. To go even beyond the presented dimension of 10 variables, the linear programming based method provides the possibility to use structural information and to apply column generation techniques.

The definition of characteristic imsets makes it possible to furthermore reformulate the already mentioned linear programming approach. We have used inverses of characteristic imsets, so-called portraits, to derive an equivalent linear programming formulation of the implication problem. The new problem formulation is structurally easier and we have given necessary conditions of its feasibility on the base of the inclusion of statements and imsets. These conditions we have used as a heuristic for falsifying the implication problem. Other structural properties have led into a general reduction of the problem. This structural benefit has turned out to be highly effective on the computational experiments.

We have related our results to two alternative references. The first reference [48] deals with the computer testing of independence implication using a linear programming based algorithm. We have shown this approach to be equivalent to the implication problem using portraits of standard imsets. This equivalence comes from an affine transformation between their representation of conditional independence statements and our characteristic imsets. The second reference [37] presents a structure learning algorithm which uses an integer linear program to solve and a binary representation of the set of possible structures in exponential dimension. We have shown the characteristic imsets to be linearly transformable to the ones presented in [37], but no linear reverse direction to exist. As the main difference between both ways of representation we have shown that characteristic imsets encode Markov equivalent graphs and that the representation of [37] encodes all directed acyclic graphs.

Future work and open problems

Several open problems and topics of future work have emerged during the development of this thesis. We have shown the linear programming approach in both formulations, with standard imsets and with portraits, to be a powerful approach for computer testing conditional independence implication. The combination of both representations may lead to further necessary and sufficient conditions again reducing the overall complexity of this task. We believe these necessary and sufficient conditions to result in a classification of the problem, which is, it is polynomial time solvable. This classification should include a specification of polynomially many elementary imsets as sufficient candidates for the solution of the linear program.

We believe two other related questions of conditional independence implication to be answerable with linear and non-linear programming formulations. Both questions deal with the standard imset representation of statements and the derivation of bounds on factors. The first bound comes from the hilbert basis of the cone of elementary statements. This bound is the maximal factor which a hilbert basis element must be multiplied with to be an element

of the corresponding monoid. The second bound is the maximal factor which is needed in "Racer".

From a practical point of view it is of major interest to solve structure learning tasks for real life instances in a competitive way. The use of better, i.e. recursive, parallel computations of the objective values and the incorporation of the objective function in a problem specific row generation step may lead to better running times and larger solvable problems. Furthermore, concrete real life examples may provide the possibility to evaluate the objective function more carefully and to incorporate additional structural information. These information can be incorporated into the optimisation procedure to solve the problems in an efficient way. This improved algorithm can then be compared to other algorithms obtaining optimal solutions.

The reduction of the possible structure the network can have is a good way of incorporating additional structural information and obtaining better problem descriptions. More classes of structures may be derivable with properties enabling us to solve the structure learning task easily or at least for higher dimensional problems. For example, inequality descriptions of less complex subproblems may be provable to be facet (or TDI) descriptions. For this, additional inequalities must be derived. A candidate subproblem is the learning of chordal graphs. Another way for deriving reduced computational costs is to search for subproblems yielding provable polynomial time solvable problems. This includes the limitation of the possible structure as well as the limitation on certain decomposable and score equivalent objective functions.

The characterisation of all geometric neighbours of the characteristic and standard imset polytope, respectively, in terms of a complete inequality description or of a graphical characterisation is interesting both from a computational and from a theoretical point of view. Computationally, quick graphical based algorithms can be applied to obtain good heuristics on the learning of Bayesian network structures. Theoretically easy, because graphical based, characterisations of geometric neighbours may lead to compact inequality descriptions improving the performance of the polyhedral based learning of Bayesian network structures. An open question is hence to completely characterise all families of geometric neighbours in the restricted and unrestricted imset polytopes.

Closing words

Discrete optimisation can be applied effectively to answer several questions appearing in the field of Bayesian networks. First of all, we have shown conditional independence implication to be approximately solvable with linear programming formulations. Results and methods from discrete optimisation have been successfully applied even for problem sizes which could not have been solved with other approaches.

Second, we have shown the change of representatives of Bayesian network structures to be an efficient way of deriving new theoretical results which gain from a close graphical connection. The ability to directly interpret the representatives graphically passes the way for a necessary and sufficient description of the polytope of network structures. Moreover, we introduce the notion of restricted learning based on methods from discrete optimisation to derive valid inequalities. With a polytopal description and a linear objective function to be optimised, state-of-the-art software has been applied to present preliminary tests on learning both restricted and unrestricted Bayesian network structures.

Last but not least, we believe discrete optimisation to be further applicable to similar questions of Machine learning whenever discrete representations occur.

List of symbols

General notations

$\mathbb{N} := \{0, 1, 2, \ldots\}$	-	the natural numbers
$\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$	-	integers, rational numbers, real numbers
$\mathcal{P}(N)$	-	the power set of N
$\mathcal{P}(N)_{\geq k}$	-	the power set of N restricted to subsets with at least k elements
$G = (N, A)$	-	a directed graph with nodes N and arcs A
$a \to b$	-	an arc $\{a, b\} \in A$
$G = (N, E)$	-	an undirected graph with nodes N and edges E
$E(G)$	-	the set of edges of a graph G
$E(T)$	-	the edges between nodes of set T
$a - b$	-	an edge $\{a, b\} \in E$
$\chi(G) \in \{0, 1\}^{\lvert E \rvert}$	-	the characteristic vector of edges E of a graph G
DAG G	-	a directed acyclic graph G
DAGs(N)	-	the set of all directed acyclic graphs with nodes N
DATA(N, d)	-	the set of all databases with nodes N and length d
$D \in \text{DATA}(N, d)$	-	a database with nodes N and length d
$Q(G, D) \in \mathbb{R}$, $Q(G)$	-	the score of a graph G and a database D
BIC(G, D)	-	the score of a graph G and a database D with respect to the Bayesian information criterion
$\mathbf{x}$, $\mathbf{A}$	-	a vector and a matrix

Symbols used in the Introduction

Ω	-	a sample space
$A \subseteq \Omega$	-	an event A
$\Omega = \{e_1, \ldots, e_n\}$	-	a discrete sample space with elementary events e_i, for $i = 1, \ldots, n$
$P : \mathcal{P}(\Omega) \to [0, 1]$	-	a probability function
$P(A)$	-	the probability of event A
(Ω, P)	-	a probability space
$P(E\|F)$	-	the conditional probability of event E given event F
$X : \Omega \to S := \{x_1, \ldots, x_l\}$	-	a discrete random variable with a finite and countable space S
$x \in S$	-	a state x of a random variable
$P(X = x)$, $P(x)$, $p(x)$	-	the probability of (a state x of) a random variable X
$P(X)$, $P(\neg X)$	-	the probability of a random variable X with binary states
$P(X = x, Y = y)$	-	the joint probability of (states of) two random variables X and Y

Symbols used in Chapter 1

$a(s)$	-	an action a in a state s
$u(a)$	-	the utility u of an action a
$EU(a)$	-	the expected utility of an action a
$\mathrm{pa}_G(i)$, $\mathrm{pa}(i)$, G_i	-	the set of parents of a node i in a graph G
$P(X_N)$, $P(X_1, \ldots, X_{\lvert N\rvert})$	-	the joint probability distribution of the random variables corresponding to nodes N in a graph
$P(X_i = x_i \lvert \bigcup_{j \neq i} (X_j = x_j))$, $P(X_i \lvert \mathrm{pa}_G(i))$	-	the local probability distribution of node i in a graph G
$A \perp\!\!\!\perp B \mid C$	-	a conditional independence statement of pairwise disjoint sets $A, B, C \subseteq N$
$\langle A, B \lvert C \rangle$	-	a triplet of pairwise disjoint sets $A, B, C \subseteq N$; also used to identify the corresponding conditional independence statement
$A \perp\!\!\!\perp B \mid C \; [P]$	-	a conditional independence statement valid for a probability distribution
$\mathcal{T}(N)$	-	the set of pairwise disjoint triplets of N
$\mathcal{M}_P$	-	a conditional independence model defined by a probability distribution
$A \perp\!\!\!\perp B \mid C \; [\mathcal{M}]$	-	a conditional independence statement valid for the triplets in $\mathcal{M}$
$A \perp\!\!\!\perp B \mid C \; [o]$	-	a conditional independence statement valid for an object o
$a \perp\!\!\!\perp b \mid C$	-	an elementary conditional independence statement
$A \perp\!\!\!\perp B \mid C \; [G]$	-	a conditional independence statement valid for a graph G
$\mathcal{M}_G$	-	a conditional independence model defined by a graph G
$\mathrm{sink}(T)$	-	the set of sinks of nodes T
$\bar{G}$	-	the underlying undirected graph of a directed or mixed graph G
$\mathrm{pat}(G)$	-	the pattern graph of a graph G
G^*	-	the essential graph of a graph G
$\delta_A \in \{0,1\}^{2^{\lvert N\rvert}}$	-	the identifier of subset A of N
$\mathbf{u}(G) \in \mathbb{Z}^{2^{\lvert N\rvert}}$	-	a standard imset of a graph G
$\mathbf{u}(\langle A, B \lvert C \rangle)$	-	a standard imset of a conditional independence statement
$\mathbf{u}(\langle a, b \lvert C \rangle)$	-	an elementary imset
$\mathcal{E}(N)$	-	the set of all elementary imsets of nodes N
$\mathcal{R}(N)$	-	the pointed rational polyhedral cone generated by elementary imsets of nodes N
$\mathcal{C}(N)$	-	the set of all combinatorial imsets of nodes N
$\mathcal{S}(N)$	-	the set of all structural imsets of nodes N
y, $d(y)$	-	an element of the joint sample space of random variables and its number of occurrences in a database
$r(i)$	-	the number of states/elements of a random variable X_i
y_i^k	-	the k-th state/element of a random variable X_i
$q(i, G)$	-	the number of possible parent set configurations of a random variable X_i in a graph G
z_i^j	-	the j-th possible parent set configurations of a random variable X_i in a graph G

$k(i,x)$	-	the state/element of random variable X_i corresponding to x in the database
$j(i,x)$	-	the parent set configuration of random variable X_i corresponding to x in the database
d_{ij}	-	the frequency of a parent set configuration j of random variable X_i in the database
d_{ijk}	-	the frequency of a state k of random variable X_i together with a parent set configuration j in the database
$q_{i,\mathrm{pa}(i)}$, $\mathrm{score}_i(G_i)$, $Q_i(\mathrm{pa}(i))$	-	the local score of a node i and its parents
ord_i	-	the i-th element of an ordering of nodes
$CFT(v,W)$	-	a contingency frequency table of a node v and a set of other nodes W
$g_i^*(C)$	-	the score-maximal parent set of node i among nodes C
$\mathrm{sink}^*(W)$	-	the score-maximal sink in node set W
$\mathrm{ord}_i^*(N)$	-	the order of node i in the score-maximal ordering of nodes N
$G^*_{\mathrm{ord}_i^*(N)}$, $i=1,\ldots,\lvert N\rvert$	-	the score-maximal network given the score-maximal ordering of nodes N
$\mathbf{t}^Q(D)\in\mathbb{R}^{2^{\lvert N\rvert}}$, $\mathbf{t}(D)$	-	the data vector with respect to a quality criterion Q and a database D
S	-	the set of standard imsets over the same nodes
$P(S)\subseteq\mathbb{R}^{2^{\lvert N\rvert}}$	-	the standard imset polytope over nodes N
$\mathrm{neigh}_G(i)$	-	the neighbours of a node i in a graph G
$I(x_i,x_j)$	-	the mutual dependence of nodes i and j

Symbols used in Chapter 2

$\mathbf{y}\in\{0,1\}^{\lvert N\rvert(\lvert N\rvert-1)}$	-	the incidence vector of arcs in a directed graph
P_{AC}	-	the acyclic subgraph polytope
$\mathrm{pa}_G(C)$	-	the parents of nodes C in a graph G
$\mathrm{portrait}(\mathbf{u})\in\{0,1\}^{\lvert\mathcal{P}(N)_{\geq 2}\rvert}$	-	the portrait of a standard imset $\mathbf{u}$
$\mathbf{c}(G)\in\{0,1\}^{\lvert\mathcal{P}(N)_{\geq 2}\rvert}$	-	the characteristic imset of a graph G
$\mathbf{c}(\mathbf{u})$	-	the characteristic imset of a standard imset $\mathbf{u}$
$\mathrm{P}_{\mathbf{c}}\subseteq\mathbb{R}^{\lvert\mathcal{P}(N)_{\geq 2}\rvert}$	-	the characteristic imset polytope
$\mathbf{r}^Q(D)\in\mathbb{R}^{\lvert\mathcal{P}(N)_{\geq 2}\rvert}$, $\mathbf{r}(D)$	-	the revised data vector with respect to a quality criterion Q and a database D
$\mathbf{c}\lvert_{\leq k}$	-	a characteristic imset restricted to entries corresponding to sets S with $\lvert S\rvert\leq k$
$Pa(i)\in\mathcal{P}(N\setminus\{i\})$	-	the set of all possible parent sets of a node i in $N\setminus\{i\}$
$s_i\in Pa(i)$	-	the parent set of a node i
$\eta_i\in\{0,1\}^{2^{\lvert N\rvert-1}}$	-	a vector indicating all parent sets $Pa(i)$ of a node i
$\eta=(\eta_1,\ldots,\eta_{\lvert N\rvert})\in\{0,1\}^{\lvert N\rvert 2^{\lvert N\rvert-1}}$	-	a vector indicating all possible parent sets of all nodes in N
$P_\eta\subseteq\mathbb{R}^{\lvert N\rvert 2^{\lvert N\rvert-1}}$	-	the polytope of the vectors indicating all parent sets of nodes

Symbols used in Chapter 3

$\mathrm{P}_{\mathbf{c},\mathrm{sink}}(i)$	-	the polytope of characteristic imsets of all DAGs containing only edges/arcs to node i
G^T	-	an induced subgraph with nodes T
G^i_{sink}	-	the induced subgraph of a graph G containing all its edges/arcs to node i
$\pi \in S_N$	-	an ordering of nodes N, an element of the permutation group S_N
$\mathrm{P}^{\pi}_{\mathbf{c}}$	-	the polytope of characteristic imsets of all DAGs containing only edges/arcs from smaller nodes to larger nodes with respect to an ordering π
$\mathrm{P}^{\pi}_{\mathbf{c},\mathrm{sink}}(i)$	-	the polytope of characteristic imsets of all DAGs containing only edges/arcs from smaller nodes to node i with respect to an ordering π
$\mathrm{P}_{\mathrm{ext}}$	-	an extension of $\mathrm{P}_{\mathbf{c}}$
$\mathrm{P}_{\mathrm{ext}}\,\vert_{\mathbf{c}}$	-	an extension of $\mathrm{P}_{\mathbf{c}}$ projected onto the characteristic imsets
$\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$	-	a LP-relaxation of $\mathrm{P}_{\mathrm{ext}}$
$G(\mathbf{x}\vert_{\leq k})$	-	the pattern interpretation of a vector $\mathbf{x}\vert_{\leq k}$
$\mathbf{x}(\mathbf{y})$	-	a vector $\mathbf{x}$ defined on the base of an incidence vector $\mathbf{y}$ of a directed graph
P_{C}	-	a LP-relaxation of the acyclic subgraph polytope
$\mathrm{P}^{\mathrm{rel}\,\prime}_{\mathrm{ext}}$	-	a modification of $\mathrm{P}^{\mathrm{rel}}_{\mathrm{ext}}$ with special integrality properties
$\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}$	-	a LP-relaxation of $\mathrm{P}_{\mathbf{c}}$
$\mathrm{P}^{\mathrm{rel}(.)}_{\mathbf{c}}$	-	an inequality descriptions of $\mathrm{P}_{\mathbf{c}}$
$\overline{\mathrm{P}^{\mathrm{rel}}_{\mathbf{c}}}$	-	a conjectured LP-relaxation of $\mathrm{P}_{\mathbf{c}}$
$\mathrm{DAGs}(K)$	-	the set of DAGs whose underlying undirected graph is a subgraphs of graph K
$\mathrm{DAGs}_{\mathrm{UF}}(K)$	-	the set of DAGs whose underlying undirected graph is an undirected forest in graph K
$\mathrm{P}_{\mathrm{UF}}(K)$	-	the polytope of characteristic imsets of undirected forests in graph K
$\hat{\mathrm{P}}_{\mathrm{UF}}(K)$	-	the polytope of characteristic vectors of undirected forests in graph K
$\vert k(G)\vert$	-	the number of components (connected subgraphs) in a graph G
$\mathrm{DAGs}_{\mathrm{UST}}(K)$	-	the set of DAGs whose underlying undirected graph is an undirected spanning tree in a connected graph K
$\mathrm{P}_{\mathrm{UST}}(K)$	-	the polytope of characteristic imsets of undirected spanning trees in a connected graph K
$\hat{\mathrm{P}}_{\mathrm{UST}}(K)$	-	the polytope of characteristic vectors of undirected spanning trees in a connected graph K
$\mathrm{DAGs}_{\mathrm{USLT}^2}(K)$	-	the set of DAGs whose underlying undirected graph is an undirected spanning tree with maximum degree two in a connected graph K
$\mathrm{DAGs}_{\mathrm{ULF}^2}(K)$	-	the set of DAGs whose underlying undirected graph is an undirected forest with maximum degree two of all nodes in graph K
$\mathrm{P}_{\mathrm{ULF}}{}^2(K)$	-	the polytope of characteristic imsets of undirected forests with maximum degree two of all nodes in graph K

$\mathrm{P_{USLT}}^2(K)$	-	the polytope of characteristic imsets of undirected spanning trees with maximum degree two of all nodes in a connected graph K
$\delta(i)$	-	the set of all incident edges of node i
$d(i)$	-	the degree of a node i
$\mathrm{P_{ULF}}^k(K)$	-	the polytope of characteristic imsets of undirected forests with the maximum degree k of all nodes in graph K
$\mathrm{P_{USLT}}^k(K)$	-	the polytope of characteristic imsets of undirected spanning trees with the maximum degree k of all nodes in a connected graph K
$\mathrm{DAGs_{CH}}(K)$	-	the set of DAGs whose underlying undirected graph is a chordal graph in graph K
$\mathrm{P_{CH}}$	-	the polytope of characteristic imsets of chordal graphs
$\mathrm{P_{CH}^{rel}}$, $\mathrm{P_{CH}^{rel\,\prime}}$	-	two LP-relaxations of $\mathrm{P_{CH}}$
$\mathbf{u}(\emptyset)$	-	the standard imset of the empty graph
$\mathbf{u}(e)$	-	the standard imset of a graph containing only the edge e
$\mathrm{P_{sink}}(i)$	-	the polytope of standard imsets of all DAGs containing only edges/arcs to node i
$\mathrm{P_{sink}^{\pi}}(i)$	-	the polytope of standard imsets of all DAGs containing only edges/arcs from smaller nodes to node i with respect to an ordering π
$\mathcal{P}_{cluster}$	-	a LP-relaxation of P_η

Symbols used in Chapter 4

$r_U(F)$	-	the cardinality of an inclusion-minimal base of nodes F
$r_M(F)$	-	the cardinality of an inclusion-maximal base of nodes F

Symbols used in Chapter 6

$\mathcal{M}_{\mathbf{u}}$	-	a conditional independence model defined by a standard imset
$m : \mathcal{P}(N) \to \mathbb{R}^{2^{\lvert N\rvert}}$	-	a supermodular function m
$\mathcal{K}(N)$, $\mathcal{K}_l(N)$	-	the set of all supermodular and l-standardised supermodular functions over N
$\mathcal{K}_l^{\diamond}(N)$	-	the l-skeleton over N
$\mathbf{u}(L)$	-	a standard imset of a list of conditional independence statements L
$\mathcal{I}_G$	-	the collection of all conditional independence statements valid for a graph G
$\mathrm{supp}(\mathbf{x})$	-	the support of a vector $\mathbf{x}$
$\mathcal{L}(A, B\lvert C)$	-	the semi-lattice of $\langle A, B\lvert C\rangle$
$\mathbf{v}(\langle A, B\lvert C\rangle)$	-	the vector representative of $\langle A, B\lvert C\rangle$
$\mathcal{L}(L)$	-	the semi-lattice of a set of conditional independence statements L
$\mathbf{v}(L)$	-	the vector representative of a set of conditional independence statements L

Index of definitions

Bibliography

[1] S. A. Andersson, D. Madigan, and M. D. Perlman, *A characterization of Markov equivalence classes for acyclic digraphs*, Annals of Statistics, 25 (1997), pp. 505–541.

[2] ——, *On the Markov Equivalence of Chain Graphs, Undirected Graphs, and Acyclic Digraphs,*, Scandinavian Journal of Statistics, 24 (1997), pp. 81–102.

[3] V. Auvray and L. Wehenkel, *Learning Inclusion-Optimal Chordal Graphs*, in Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence, D. A. McAllester and P. Myllymäki, eds., UAI '08, AUAI Press, 2008, pp. 18–25.

[4] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance, *Branch-and-Price: Column Generation for Solving Huge Integer Programs*, Operations Research, 46 (1996), pp. 316–329.

[5] I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper, *The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks*, in Second European Conference on Artificial Intelligence in Medicine, J. Hunter, J. Cookson, and J. Wyatt, eds., vol. 38, Springer-Verlag, Berlin, 1989, pp. 247–256.

[6] E. A. Bender and J. R. Goldman, *On the Applications of Mobius Inversion in Combinatorial Analysis*, The American Mathematical Monthly, 82 (1975), pp. 789–803.

[7] R. R. Bouckaert, *Bayesian Belief Networks: From Construction to Inference*, PhD thesis, University of Utrech, 1995.

[8] R. R. Bouckaert, R. Hemmecke, S. Lindner, and M. Studený, *Efficient Algorithms for Conditional Independence Inference*, Journal of Machine Learning Research, 11 (2010), pp. 3453–3479.

[9] R. R. Bouckaert and M. Studený, *Racing Algorithms for Conditional Independence Inference*, International Journal of Approximate Reasoning, 45 (2007), pp. 386–401.

[10] W. Bruns, R. Hemmecke, B. Ichim, M. Köppe, and C. Söger, *Challenging computations of Hilbert bases of cones associated with algebraic statistics*, Experimental Mathematics, 20 (2011).

[11] D. M. Chickering, *Learning Bayesian Networks is NP-Complete*, in Learning from Data: Artificial Intelligence and Statistics V, D. Fisher and H. Lenz, eds., Springer-Verlag, 1996, pp. 121–130.

[12] ——, *Optimal Structure Identification with Greedy Search*, Journal of Machine Learning Research, 3 (2002), pp. 507–554.

[13] D. M. Chickering, D. Heckerman, and C. Meek, *Large-Sample Learning of Bayesian Networks is NP-Hard*, Journal of Machine Learning Research, 5 (2004), pp. 1287–1330.

[14] D. M. Chickering and C. Meek, *Finding Optimal Bayesian Networks*, in Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence, A. Darwiche and N. Friedman, eds., UAI '02, Morgan Kaufmann, 2002, pp. 94–102.

[15] C. Chow and C. Liu, *Approximating Discrete Probability Distributions with Dependence Trees*, IEEE Transactions on Information Theory, 14 (1968), pp. 462–467.

[16] S. Dasgupta, *Learning Polytrees*, in Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, K. B. Laskey and H. Prade, eds., UAI '99, Morgan Kaufmann, 1999, pp. 134–141.

[17] C. P. de Campos, Z. Zeng, and Q. Ji, *Structure Learning of Bayesian Networks using Constraints*, in Proceedings of the 26th Annual International Conference on Machine Learning, A. P. Danyluk, L. Bottou, and M. L. Littman, eds., vol. 382 of ICML 2009, ACM, 2009, pp. 113–120.

[18] L. M. de Campos and J. F. Huete, *Algorithms for Learning Decomposable Models and Chordal Graphs*, in Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence, D. Geiger and P. P. Shenoy, eds., UAI '97, Morgan Kaufmann, 1997, pp. 46–53.

[19] F. M. Dekking, C. Kraaikamp, H. P. Lopuhaä, and L. E. Meester, *A Modern Introduction to Probability and Statistics: Understanding Why and How (Springer Texts in Statistics)*, Springer, 2005.

[20] J. Desrosiers and M. E. Lübbecke, *Branch-Price-and-Cut Algorithms*, in Wiley Encyclopedia of Operations Research and Management Science, J. J. Cochran, L. A. Cox, P. Keskinocak, J. P. Kharoufeh, and J. C. Smith, eds., John Wiley & Sons, Inc., 2010.

[21] M. Frydenberg, *The chain graph Markov property*, Scandinavian Journal of Statistics, 17 (1990), pp. 333–353.

[22] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, 1979.

[23] D. Geiger, T. Verma, and J. Pearl, *Identifying Independence in Bayesian Networks*, Networks, 20 (1990), pp. 507–534.

[24] S. B. Gillispie and M. D. Perlman, *Enumerating Markov Equivalence Classes of Acyclic Digraph Models*, in Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, J. S. Breese and D. Koller, eds., UAI '01, Morgan Kaufmann, 2001, pp. 171–177.

[25] M. Grötschel, M. Jünger, and G. Reinelt, *On the acyclic subgraph polytope*, Mathematical Programming, 33 (1985), pp. 28–42.

[26] P. Haddawy, *An Overview of Some Recent Developments in Bayesian Problem-Solving Techniques*, AI Magazine, 20 (1999), pp. 11–19.

[27] F. Harary and E. M. Palmer, *Graphical enumeration*, Addison-Wesley, 1973.

[28] U.-U. Haus and R. Hemmecke, *Decomposition of reaction networks: the initial phase of the permanganate/oxalic acid reaction*, Journal of Mathematical Chemistry, 48 (2010), pp. 305–312.

[29] U.-U. Haus, K. Niermann, K. Truemper, and R. Weismantel, *Logic Integer Programming Models for Signaling Networks*, Journal of Computational Biology, 16 (2009), pp. 725–743.

[30] D. Hausmann, *Adjacency on polytopes in combinatorial optimization*, Verlag Anton Hain Meisenheim GmbH, 1980.

[31] D. Heckerman, *A Tutorial on Learning with Bayesian Networks*, tech. report, Microsoft Research, Redmond, Washington, 1995.

[32] D. Heckerman, D. Geiger, and D. M. Chickering, *Learning Bayesian Networks: The Combination of Knowledge and Statistical Data*, Machine Learning, 20 (1995), pp. 197–243.

[33] R. Hemmecke, S. Lindner, and M. Studený, *Learning restricted Bayesian network structures*, CoRR, abs/1011.6664 (2010).

[34] ——, *Characteristic imsets for learning Bayesian network structure*, (2011). submitted.

[35] R. Hemmecke, J. Morton, A. Shiu, B. Sturmfels, and O. Wienand, *Three Counter-Examples on Semi-Graphoids*, Combinatorics, Probability & Computing, 17 (2008), pp. 239–257.

[36] IBM Ilog team, *CPLEX - mathematical programming optimizer*. Available electronically at www-01.ibm.com/software/integration/optimization/cplex/.

[37] T. Jaakkola, D. Sontag, A. Globerson, and M. Meila, *Learning Bayesian Network Structure using LP Relaxations*, Journal of Machine Learning Research - Proceedings Track, 9 (2010), pp. 358–365.

[38] F. V. Jensen, *Introduction to Bayesian Networks*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st ed., 1996.

[39] F. V. Jensen and T. D. Nielsen, *Bayesian Networks and Decision Graphs*, Springer Publishing Company, Incorporated, 2nd ed., 2007.

[40] D. R. Karger and N. Srebro, *Learning Markov networks: maximum bounded tree-width graphs*, in Proceedings of the 12th Annual Symposium on Discrete Algorithms, SODA '01, Society for Industrial and Applied Mathematics, 2001, pp. 392–401.

[41] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*, Springer, 2rd ed., 2002.

[42] S. L. Lauritzen, *Graphical Models (Oxford Statistical Science Series)*, Oxford University Press, USA, 1996.

[43] J. Lu, G. Getz, E. A. Miska, E. Alvarez-Saavedra, J. Lamb, D. Peck, A. Sweet-Cordero, B. L. Ebert, R. H. Mak, A. A. Ferrando, J. R. Downing, T. Jacks, H. R. Horvitz, and T. R. Golub, *MicroRNA expression profiles classify human cancers*, Nature, 435 (2005), pp. 834–838.

[44] C. Meek, *Causal inference and causal explanation with background knowledge*, in Proceedings of the 11th Annual Conference on Uncertainty in Artificial Intelligence, P. Besnard and S. Hanks, eds., UAI '95, Morgan Kaufmann, 1995, pp. 403–410.

[45] ——, *Graphical Models: Selecting causal and statistical models*, PhD thesis, Carnegie Mellon University, 1997.

[46] ——, *Finding a Path is Harder than Finding a Tree*, Journal of Artificial Intelligence Research (JAIR), 15 (2001), pp. 383–389.

[47] R. E. Neapolitan, *Learning Bayesian Networks*, Prentice Hall, April 2003.

[48] M. Niepert, *Logical Inference Algorithms and Matrix Representations for Probabilistic Conditional Independence*, in Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI '09, AUAI Press, 2009, pp. 428–435.

[49] M. Niepert and D. V. Gucht, *Logical properties of stable conditional independence*, in Proceedings of the Fourth European Workshop on Probabilistic Graphical Models, M. Jaeger and T. D. Nielsen, eds., 2008, pp. 225–232.

[50] M. Niepert, D. V. Gucht, and M. Gyssens, *On the Conditional Independence Implication Problem: A Lattice-Theoretic Approach*, in Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence, D. A. McAllester and P. Myllymäki, eds., UAI '08, AUAI Press, 2008, pp. 435–443.

[51] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*, Morgan Kaufmann, 1997.

[52] L. J. Savage, *The Foundations of Statistics*, Dover Publications, Inc., New York, 2nd ed., 1972.

[53] A. Schrijver, *Theory of Linear and Integer Programming*, Wiley, 1998.

[54] ——, *Combinatorial Optimization - Polyhedra and Efficiency*, Springer, 2003.

[55] T. Silander and P. Myllymäki, *A Simple Approach for Finding the Globally Optimal Bayesian Network Structure*, in Proceedings of the 22nd Conference in Uncertainty in Artificial Intelligence, UAI '06, AUAI Press, 2006.

[56] N. Srebro, *Maximum Likelihood Bounded Tree-Width Markov Networks*, in Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence, J. S. Breese and D. Koller, eds., UAI '01, Morgan Kaufmann, 2001, pp. 504–511.

[57] M. Studený, *Multiinformation and the problem of characterization of conditional independence relations*, Problems of Control and Information Theory, 18 (1989), pp. 3–16.

[58] ——, *Conditional independence relations have no finite complete characterization*, in Information Theory, Statistical Decision Functions and Random Process, Transactions of the 11th Praque Conference, S. Kubík and J. A. Víšosek, eds., Kluwer, 1992, pp. 377–396.

[59] ——, *Structural imsets, an algebraic method for describing conditional independence structures*, in International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, B. Bouchon-Meunier, G.Goletti, and R. R. Yager, eds., IPMU 2004, 2004, pp. 1323–1330.

[60] ——, *Probabilistic Conditional Independence Structures*, Springer, London, 2005.

[61] M. Studený and D. Haws, *On polyhedral approximations of polytopes for learning Bayes nets*, CoRR, abs/1107.4708 (2011).

[62] M. Studený, D. Haws, R. Hemmecke, and S. Lindner, *Polyhedral approach to statistical learning graphical models*, (2011). submitted.

[63] M. Studený, R. Hemmecke, and S. Lindner, *Characteristic imset: a simple algebraic representative of a Bayesian network structure*, in Proceedings of the 5th European Workshop on Probabilistic Graphical Models, P. Myllymki, T. Roos, and T. Jaakkola, eds., PGM 2010, HIIT Publications, 2010, pp. 257–264.

[64] M. Studený and J. Vomlel, *On open questions in the geometric approach to structural learning Bayesian nets*, International Journal of Approximate Reasoning, 52 (2011), pp. 627–640.

[65] M. Studený, J. Vomlel, and R. Hemmecke, *A geometric view on learning Bayesian network structures*, International Journal of Approximate Reasoning, 51 (2010), pp. 573–586.

[66] T. Verma and J. Pearl, *Equivalence and synthesis of causal models*, in Proceedings of the 6th Annual Conference on Uncertainty in Artificial Intelligence, P. P. Bonissone, M. Henrion, L. N. Kanal, and J. F. Lemmer, eds., UAI '90, Elsevier, 1990, pp. 255–270.

[67] ——, *An Algorithm for Deciding if a Set of Observed Independencies Has a Causal Explanation*, in Proceedings of the 8th Annual Conference on Uncertainty in Artificial Intelligence, D. Dubois and M. P. Wellman, eds., UAI '92, Morgan Kaufmann, 1992, pp. 323–330.