university of groningen / faculty of science and engineering / mathematics and applied mathematics

Bayesian Network Structure Learning with a Focus on Networks with Background Information and Incomplete Information

Bachelor's Project Mathematics

July 2024

Student: Jovan Andreevski S4964004

First Supervisor: Prof. Dr. Marco A. Grzegorczyk

Second Assessor: Dr. Alef E. Sterk

Abstract

Bayesian networks are a widely-used probabilistic graphical model in which random variables, and the conditional dependencies between these variables, are represented using a directed acyclic graph (DAG). Bayesian network structure learning refers to the highly sought-after, yet challenging, task of inferring Bayesian networks from data. In this paper, we formally introduce the task of structure learning and focus on structure learning in networks with background information and networks with incomplete information. First, we provide a formal definition of networks with background information through the treatment of maximally oriented partially directed acyclic graphs (MPDAGs). Moreover, we study the differences between regular networks and networks with background information, and develop theory regarding dependencies in MPDAGs as a starting point for performing structure learning in networks with background information. Second, we introduce networks with incomplete information, and provide a detailed exposition of the belief propagation algorithm as an efficient approach for calculating marginal distributions of unobserved variables. Moreover, we provide an adaptation of the belief propagation algorithm to Bayesian networks, discuss the exactness of the algorithm, and propose an approach for using the algorithm to sample missing data in discrete Bayesian networks. Finally, we offer a comprehensive implementation of the algorithm in the programming language R.

Keywords: *Bayesian Network, Structure Learning, Background Information, CPDAG, MPDAG, Incomplete Information, Propagation*

Acknowledgements

I would like to thank everyone who has directly or indirectly assisted me in the writing of this thesis. First, I would like to thank my first supervisor, Prof. Dr. Marco A. Grzegorczyk, without whose support the successful completion of my thesis would not have been possible. His kindness, dedication, research ideas and knowledge, and genuine interest in my topic have been instrumental throughout the entire writing process, and I am eternally thankful to him for that. I would also like to thank my second assessor, Dr. Alef E. Sterk, who has showed tremendous understanding and support regarding the completion of my thesis. Finally, I would like to thank my family, partner and friends for supporting me emotionally and encouraging me throughout my entire formal education, and especially during the last few months. Their love towards me and willingness to help me have been nothing short of extraordinary.

Contents

1 Introduction

In many natural and social processes, in which there are interacting variables (e.g. factors, agents, objects) involved, one is often interested in exploring the different influence and causal relationships that exist in the given processes. Discovering causal relations is extremely desirable from a practical perspective, whether that is the stock market, for which we are interested in learning how the price fluctuations in certain financial instruments (e.g. stocks, bonds or market indices) influence the price of other financial instruments, or an entire ecosystem, for which we want to know how the changes in predator and prey populations influence each other. However, in practice this is often not possible due to the many possible explanatory factors for an identified correlation [28]. To that end, one might colloquially state that the "next best thing" is to discover dependence relations in a given process. For instance, when studying predator-prey relations, we might observe that a decline in the predator population correlates with the increase in the prey population size. We most likely are not able to attribute this observation to direct causality due to the many environmental factors involved, such as grazing opportunities and climate change, but we are far more likely to deliberate that given a decrease in the predator population size, the prey population size is more likely to increase, thus establishing a dependence relation.

The formal study and modelling of such relations can be done in a multitude of ways, ranging from the use of *neural networks* to the application of classical *rule-based systems* founded upon first-order logic principles [21]. Currently, a popular approach for studying dependence relations, which garnered a considerable surge in interest in the late 20^{th} century, is the use of *probabilistic graphical models* (PGMs) [21]. Namely, if we introduce uncertainty and some form of logical structure to a process involving interacting variables, we are able to assume the interacting variables to be *random variables* and the dependence relations between them to be given as *conditional probabilities*. In particular, PGMs are graphical models, meaning that they are represented by graphs, in which the *nodes* of the graph are represented by random variables and the *edges* of the graph encode the dependence relations between the variables and the associated conditional probabilities [17]. For instance, our predator-prey example from the first paragraph could be rather simply represented as a PGM (see figure 1.1), and we could interpret it as the event of the random variable "predator population" being low increases the probability of the event of the random variable "prey population" being high.

There are two main types of probabilistic graphical models, *Bayesian networks*, based on directed acyclic graphs (also known as belief networks), and *Markov networks*, based on undirected graphs (also known as Markov random fields (MRFs)) [17].

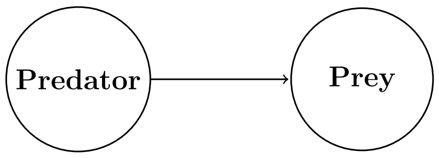

Figure 1.1: Predator Prey PGM.

There exist also PGMs in which a mixture of directed and undirected representation is used such as *chain graphs* (see [2]). The focus of our research is on Bayesian networks.

Bayesian networks are a type of PGM, typically represented through a directed acyclic graph (DAG) (e.g. the graph in figure 1.1 is a DAG) [21]. DAGs visually depict the different dependencies among variables. As mentioned, in a DAG nodes represent random variables, while directed edges indicate probabilistic dependencies between them. Crucially, the acyclic property ensures that no cycles exist in the graph, preventing feedback loops and maintaining a clear directionality in the dependence relationships [21]. Bayesian networks offer an intuitive framework for representing and reasoning under uncertainty, finding applications across diverse domains including artificial intelligence, machine learning and genetics [27].

On the one hand, from the description of Bayesian networks, it is rather natural and obvious to use them to describe dependence relations among interacting variables. However, from a practical perspective, a much more beneficial use of Bayesian networks would be to infer a network from collected data. This is commonly known as *Bayesian network structure learning*, and in essence it represents the task of developing statistical approaches with which we can analyze collected data regarding some interacting variables and infer the underlying network structure that these variables are a part of [28]. Uncovering the underlying network structure of a process involving interacting variables by simply analyzing collected data allows us to better understand how the dependence relations among these variables look like, which is an extremely powerful result given the required input. For instance, in the field of biology, studying gene expression to understand how certain genes act as regulators for other genes in different human cells is of paramount importance for drug development, therapy development and targeted medical care [12]. One way to study and understand how genes regulate and influence each other is through the use of structure learning approaches for gene expression data [27].

There exist two main types of structure learning approaches: *constraint-based* and *score-based* approaches [13]. Constraint-based Bayesian network structure learning identifies the network structure by leveraging conditional independence tests on the data [5]. These tests determine whether two variables are independent given a set of other variables. The process begins with constructing an undirected graph where nodes are connected if they fail the independence tests. Edges are then oriented to satisfy the identified conditional independencies, adhering to the rules of directed acyclic graphs (DAGs). Algorithms like the PC algorithm and the GS algorithm are commonly used, iterating over increasing conditioning sets to refine the graph [20]. On the other hand, score-based approaches involve identifying the optimal network structure by evaluating different structures using a scoring metric [5, 13]. This metric measures how well a given structure fits the data, typically balancing model complexity and data fit. Common scores include the Gaussian BGe score from Geiger and Heckerman [11] and the discrete BDe score from Madigan and York [23]. These approaches often employ search algorithms, such as the hill climbing algorithm [10], to explore the space of possible network structures, aiming to find the structure that best represents the underlying probabilistic relationships among variables.

The main assumption of the majority of classical constraint-based and score-based structure learning approaches is that the data they are taking as input is complete [13]. However, from a practical perspective, having complete data is not always the case. For instance, we could have missing data in our dataset due to technical errors, equipment malfunction or documentation oversight [5]. Alternatively, we could also have additional information that extends our collected data due to past observations, expert knowledge or model constraints [5]. Therefore, we would like to be able to perform structure learning on datasets with *background information* and *incomplete information.*

It is important to note that there exist structure learning approaches for Bayesian networks with background information and incomplete information. However, in the case of networks with background information, these approaches aim to circumvent the background information and use classical approaches, thus they are typically either very restrictive or computationally inefficient [35, 18]. On the other hand, in the case of networks with incomplete information, the majority of approaches typically aim to find one "best" network fitting the data [20]. A recent paper promotes a different route by developing a so-called *Bayesian model averaging* (BMA) approach for inferring Gaussian Bayesian networks from incomplete data by simultaneously sampling DAGs and missing data values from a posterior distribution [13]. This approach turns out to be highly competitive in terms of network reconstruction accuracy compared to classical approaches such as the structural EM algorithm and the NAL approach [13, 1, 6]. Nevertheless, one potential limitation of this approach is that it is currently tailored to Gaussian networks with no established adaptation for discrete Bayesian networks [13].

Therefore this paper has three main objectives. First, we formally introduce Bayesian networks and the task of Bayesian network structure learning. Second, we introduce Bayesian networks with background information, explore differences to regular networks, and develop new theory on dependence relations with the aim of performing structure learning directly on these networks without having to avoid incorporating the background information at hand. Third, we formally introduce networks with incomplete information and adapt the so-called *belief propagation* algorithm to Bayesian networks to propose an approach for sampling missing values with the aim of developing an adaptation of the introduced BMA approach to discrete Bayesian networks. Additionally, we explore the convergence of this algorithm and provide an implementation of the algorithm in the programming language R.

The remainder of this thesis is structured as follows. Section 2 provides a comprehensive theoretical background from the fields of graph theory, measure theory and probability theory (sections 2.1, 2.2 and 2.3). Moreover, section 2.4 is devoted to the formal introduction of Bayesian networks, alongside key results related to the study and use of Bayesian networks. In section 3 we introduce the task of Bayesian network structure learning, and discuss the difficulty of conducting structure learning by looking at the space of possible DAGs on $n \in \mathbb{N}$ nodes (section 3.1), as well as how we can define an equivalence relation on DAGs to simplify structure learning (section 3.2). We conclude section 3 by providing a concise overview of structure learning approaches, with more attention devoted to the so-called *score-based* approaches (section 3.3). Section 4 introduces Bayesian networks with background information in the form of maximally oriented partially directed acyclic graphs (MPDAGs) (section 4.1), presents key differences with respect to regular networks (section 4.2), and adapts dependence relations to networks with background information (section 4.3). Section 5 introduces Bayesian networks with incomplete information, provides an overview of the belief propagation algorithm and an adaption of the algorithm to Bayesian networks (sections 5.1 and 5.2), discusses convergence results, proposes an approach for sampling the missing values in a network, and concludes with an implementation of belief propagation in Bayesian networks (section 5.3). Finally, section 6 concludes the thesis, discusses venues for future research, and provides a personal reflection on the writing experience of the author.

2 Theoretical Background

In order to provide a detailed discussion of the theoretical results presented in this paper, we first must present some preliminary results related to Bayesian networks. Bayesian networks are a type of probabilistic model, belonging to the class of *probabilistic graphical models*, which focus on modelling the dependence and independence relationships among random variables in a graph-theoretic manner [21]. In particular, to formalize concepts related to independence relationships in a probabilistic sense, theoretical background is needed from the fields of graph theory, measure theory and probability theory. As a remark, please note that the proofs of all statements in this section are self-developed, except where stated otherwise.

2.1 Graph Theory

Given that Bayesian networks are at core models which rely upon graphical representation, we present several important results and concepts from graph theory, some of which are very general and broad in their applicability, while other results are specific to the study of Bayesian networks. The terminology, notation and results presented in this section closely follow [21] and [7].

Definition 1 (Graph and Directed Graph).
A graph is a pair $G = (V(G), E(G))$, where $V(G)$ is a set of vertices (nodes) and $E(G) = \{\{u, v\} | \, u \neq v, \, u, v \in V\}$ is a set of pairs of vertices, called edges, with an edge typically denoted in a shorthand way as $e = uv$. A directed graph is a graph in which the edges have a direction, i.e. if $v, w \in V(G)$, an undirected edge is an unordered pair $e = \{v, w\}$, while a directed edge is an ordered pair (v, w) or (w, v), indicating the direction of the edge.

To introduce further terminology, we say that $v \in V(G)$ is *incident* to $e \in E(G)$ if v is on the edge e. Moreover, if two vertices in a graph are connected by an edge, we say the vertices are *adjacent*. If we have two graphs G_1 and G_2 such that $G_1 \subseteq G_2$, then we say that G_1 is a *subgraph* of G_2, i.e. $V(G_1) \subseteq V(G_2)$ and $E(G_1) \subseteq E(G_2)$. Moreover, a *walk* w is a possibly infinite sequence of vertices $w = \langle v_1, v_2, \ldots \rangle$ with $v_i \in V(G)$ such that the edges $e_i = v_i v_{i+1} \in E(G)$ for all $i \in \mathbb{N}$. Note that the same vertex can occur several times in a walk. In particular, a *path* p is a walk in which all vertices are distinct. The length of a walk/path is $n \in \mathbb{N}$, the number of edges traversed by it. Finally, a *cycle* c is a walk which starts and ends at the same vertex and all the other vertices are distinct. It is conventional, through abuse of notation, to also let the edges between subsequent nodes be part of the path/walk/cycle, typically to indicate if we have undirected or directed edges. For instance, if we formally have an edge $e_i = v_i v_{i+1} \in E(G)$, one would typically write $e_i = v_i - v_{i+1}$ in case the edge is undirected, and $e_i = v_i \rightarrow v_{i+1}$ or $e_i = v_i \leftarrow v_{i+1}$ to indicate that the edge is directed.

In our treatment of Bayesian networks, the notion of a *directed acyclic graph* has a key role. In particular, when treating networks with background information in section 4, we make use of *directed walks* (also known as *causal paths*), which are walks in which the involved edges are directed, and of *directed paths* (also known as *possibly causal paths*), which are paths in which the involved edges are directed [21], [29]. Formally, let $p = \langle v_1, \dots, v_n \rangle$ be a path in a graph G ($n > 1$), then we say that p is a directed path if no edge $v_i \leftarrow v_j$ with $1 \leq i < j \leq n$ is in G. Moreover, in the discussion of Bayesian networks with incomplete information in section 5, the concepts of *tree* and *polytree* are important.

Definition 2 (Directed Acyclic Graph).
A directed acyclic graph (DAG) is a graph G in which there are no cycles (acyclic) and all edges are directed.

We remark that it is conventional to use the term "node" instead of "vertex" when discussing DAGs, in particular Bayesian networks, as this reflects the fact that we are dealing with networks. Moreover, we adopt the convention to denote the nodes of DAGs with capital letters, as is customary in literature.

Definition 3 (Trees and Polytrees).
A tree is a connected acyclic graph, typically denoted by T. Namely, there are no cycles in T (acyclic) and there is a path between any two vertices in T (connected). A polytree is a DAG whose underlying undirected graph is a tree.

We remark that the underlying undirected graph of a DAG G is also called the *skeleton* of G, obtained by removing the direction of directed edges. An important aspect of DAGs, which is necessary for the study of dependencies and independencies of random variables, is the notion of *ancestor-descendant relationships*. Given a directed edge between two nodes $V_i \rightarrow V_j$, we say that the node V_i is a *parent* of V_j and V_j is a *child* of V_i. The set of all parent nodes of a node V_i, which could be empty, is called the *parent nodes* set of V_i and we denote it by $Pa(V_i)$. The corresponding set of all child nodes of V_i, which could be also empty, is called the *child nodes* set of V_i and we denote it by $Ch(V_i)$. In terms of paths in DAGs, a *directed path* is a path in which every subsequent node in the path sequence is a child of the previous node in the path sequence, thus respecting the ancestor-descendant relationships in the DAG. Moreover, If there is a directed path of parent-child relations from V_i to V_j, then we call V_i an *ancestor* of V_j and we call V_j a *descendant* of V_i. Therefore, we observe that given a DAG G and any two nodes V_i and V_j in G, there is a directed path from V_i to V_j if and only if V_i is an ancestor of V_j.

Upon closer examination of a given node V_i in a DAG G and its ancestor-descendant relationships with other nodes, by solely focusing on the "neighboring" nodes of V_i we obtain an important structure. The following definition formalizes what "neighboring" means and which structure we have in mind.

Definition 4 (Markov Blanket)**.**
Consider a DAG G with nodes $V_1, \dots, V_n$. The Markov Blanket of node V_i, $i \in \{1, \dots, n\}$, denoted by $MB(V_i)$ is a set consisting of:

1. *the parent nodes of V_i,*
2. *the children nodes of V_i,*
3. *the nodes that share a child with V_i.*

Example 1. *Consider the following DAG G consisting of 7 nodes.*

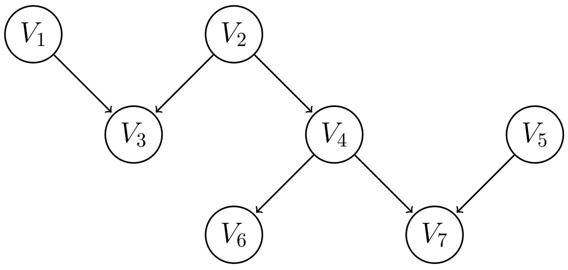

Figure 2.1: DAG G.

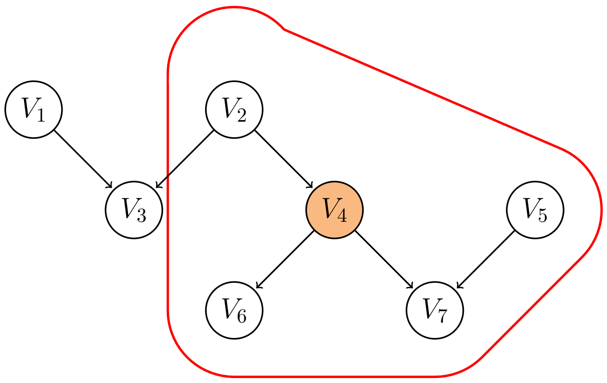

Figure 2.2: Markov Blanket of node V_4.

Following definition 4, we obtain that the nodes encircled in red (without V_4) form the Markov blanket of node V_4, and thus $MB(V_4) = \{V_2, V_5, V_6, V_7\}$.

As the reader might recognize, we implicitly gave a form of numbering (ordering) of the nodes in example 1. While typically, the way in which we order nodes in a given graph G is not an essential property of the graph itself, when it comes to DAGs, we would like to have the nodes ordered in such a way that we respect the ancestor-descendant relationships in the DAG.

Definition 5 (Topological Ordering)**.**
A topological ordering of a DAG G is an ordering of all the nodes in the graph such that each node has a position behind all of its parents, and therefore also behind all of its ancestors.

In example 1, for instance, two possible topological orderings are $(V_1, V_2, V_3, V_4, V_5, V_6, V_7)$, which is the one we used implicitly in our discussion of $MB(V_4)$, or $(V_2, V_1, V_4, V_3, V_6, V_5, V_7)$. In case of non-uniqueness of the topological ordering, as in example 1, one is allowed to choose a valid topological ordering freely.

In our treatment of (in-)dependencies in Bayesian networks, the ability to infer these dependence relations from the DAG is of crucial importance. In particular, we aim to discover how paths between nodes might imply independence and how conditioning on some nodes in the DAG might create or remove independence relations, in the probabilistic sense. Therefore, we introduce the concepts of a $v-structure$ and $d-separation$.

Definition 6 (V-Structure)**.**
In a given DAG G, a v-structure, also known as an immorality or as an unshielded collider, is a set of three nodes, of which one node is the child of the other two nodes, with the two parent nodes having no directed edge between them.

For instance, in example 1, we have two v-structures: $V_1 \to V_3 \leftarrow V_2$ and $V_4 \to V_7 \leftarrow V_5$. Note that if there were a directed edge between V_1 and V_2, then $V_1 \to V_3 \leftarrow V_2$ would no longer be a v-structure.

In an arbitrary DAG G, consider a path $\langle V_1, \dots, V_n \rangle$ in the underlying undirected graph of G. A node V_i in the given path is a called a *collider* if $V_{i-1} \to V_i \leftarrow V_{i+1}$ is in G. Note that a collider is not necessarily a v-structure, as there might be an edge between V_{i+1} and V_{i-1} in the DAG G, but this edge is not considered in the path $\langle V_1, \dots, V_n \rangle$. Moreover, given a path $\langle V_1, \dots, V_n \rangle$ in the underlying undirected graph of G, let Z be a subset of $V(G)$, which might overlap with the chosen path, but crucially it must not contain V_1 and V_n. We say that the path $\langle V_1, \dots, V_n \rangle$ is *blocked conditional on* Z if:

1. There is a node V_j in $\{V_2, \dots, V_{n-1}\}$ which is not a collider in the path and $V_j \in Z$.
2. There is a node V_j in $\{V_2, \dots, V_{n-1}\}$ which is a collider in the path, but $V_j \notin Z$ nor any of its descendants are in Z.

In example 1, for instance, we notice that the path $\langle V_2, V_4, V_7 \rangle$ is blocked conditional on $Z = \{V_4, V_6\}$ since $V_4 \in Z$ and V_4 is not a collider.

Definition 7 (d-separation)**.**
Given a DAG G, two nodes V_i and V_j in G, and a subset Z of nodes in G such that $V_i \notin Z$, $V_j \notin Z$, we say V_i and V_j are d-separated with respect to Z if every path between V_i and V_j is blocked conditional on Z.

One can extend this definition to say that two subsets of nodes X and Y are d-separated with respect to a subset of nodes Z, if for any node in X and any node in Y, every path between these nodes is blocked conditional on Z. Fundamentally, if V_i and V_j are d-separated with respect to Z, then it turns out that V_i and V_j are independent conditional on Z in the probabilistic sense. This is further explored in section 2.3.

2.2 Measure Theory

The formal introduction of key probability concepts relies upon definitions and results from measure theory. For our purposes, a less formal treatment of probability theory will suffice, so to that end, we limit the discussion of measure theory to the very fundamentals needed for rigorous treatment of Bayesian networks. The terminology, notation and results in this section closely follow [34].

Definition 8 (σ-algebra)**.**
A collection $\mathcal{A}$ of subsets of a set Ω is called a σ-algebra if:

1. $\Omega \in \mathcal{A}$
2. $A \in \mathcal{A} \implies A^c \in \mathcal{A}$
3. $A_n \in \mathcal{A}, n \in \mathbb{N} \implies \bigcup_{n\in\mathbb{N}} A_n \in \mathcal{A}$

The pair $(\Omega, \mathcal{A})$ is referred to as a *measurable space*, and the sets in $\mathcal{A}$ are called *measurable sets*.

Definition 9 (Measure)**.**
A measure μ on a σ-algebra $\mathcal{A}$ is an extended real-valued function $\mu : \mathcal{A} \to [0, \infty]$ such that:

1. $\mu(\emptyset) = 0$
2. $A_n \in \mathcal{A}, n \in \mathbb{N}$, *pairwise disjoint* $\implies \mu(\bigcup_{n\in\mathbb{N}} A_n) = \sum_{n=1}^{\infty} \mu(A_n)$

Definition 10 (Measure and Probability Space)**.**
A measure space is a triple $(\Omega, \mathcal{A}, \mu)$ such that Ω is a set, $\mathcal{A}$ is a σ-algebra on Ω and $\mu : \mathcal{A} \to [0, \infty]$ is a well-defined measure on $(\Omega, \mathcal{A})$. In particular, a measure space for which $\mu(\Omega) = 1$ is called a probability space.

Moreover, to introduce further terminology from probability theory, if we are given a probability space $(\Omega, \mathcal{A}, \mu)$, the set Ω is called the *sample space*, the measurable sets $A \in \mathcal{A}$ are called *events*, and the elements $\omega \in \Omega$ are called *outcomes*. When dealing with probability spaces, one typically denotes the probability measure by $\mathbb{P}$. Additionally, when discussing probability spaces, one has to provide a formal definition of *random variables* and *probability distributions*.

Definition 11 (Measurable Function)**.**
Given two pairs of sets with their respective σ-algebras, $(\Omega, \mathcal{A})$ and $(\Omega', \mathcal{A}')$, a function $f : \Omega \to \Omega'$ is called a measurable function (more precisely $(\mathcal{A}, \mathcal{A}')$-measurable) if for every set $A' \in \mathcal{A}'$, the preimage $f^{-1}(A') \in \mathcal{A}$.

A random variable $X : \Omega \to E$ is a measurable function from a sample space Ω to a measurable space $(E, \mathcal{F})$, where $(\Omega, \mathcal{A}, \mathbb{P})$ is a probability space. Moreover, the probability that X "takes on" a measurable subset $S \subset E$ is given as:

$$\mathbb{P}(\{\omega \in \Omega | \, X(\omega) \in S\})$$

For our purposes, we will only consider $E = \mathbb{R}$ and $E = \overline{\mathbb{R}}$, where $\overline{\mathbb{R}} = [-\infty, \infty]$, which is obtained by extending the real line by adjoining the two symbols $-\infty$ and ∞ with the obvious ordering. Typically, we equip $\overline{\mathbb{R}}$ with the *Borel* σ-algebra, denoted by $\overline{\mathcal{B}}$ which is the σ-algebra generated by all open sets in $\overline{\mathbb{R}}$. For clarity, we also state that by a "generated σ-algebra" for a given collection of subsets $\mathcal{E}$ of a set Ω (e.g. collection of open sets in $\overline{\mathbb{R}}$ or $\mathbb{R}$), we mean the smallest σ-algebra containing $\mathcal{E}$. For an arbitrary collection of subsets $\mathcal{E}$ of a set Ω, we typically denote this σ-algebra by $\sigma(\mathcal{E})$. A standard result in measure theory guarantees the existence of this generated σ-algebra [34]. Note that, if we have $\mathbb{R}$, then the σ-algebra generated by the open sets of $\mathbb{R}$ is the Borel σ-algebra on $\mathbb{R}$, denoted by $\mathcal{B}$.

Furthermore, we distinguish between *discrete random variables* and *(absolutely) continuous random variables.* A random variable is discrete if it only attains countably many values. Let $\mathcal{X} \subset \mathbb{R}$ be the set of values attained by X. We define the *probability mass function* $f_X : \mathcal{X} \to [0, 1]$ of X as:

$$f_X(x) = \mathbb{P}(X^{-1}(\{x\}) = \mathbb{P}(X = x) \tag{2.1}$$

with $\sum_{x \in \mathcal{X}} f_X(x) = 1$. On the other hand, a random variable is (absolutely) continuous if there exists a function $f_X : \mathbb{R} \to [0, \infty)$ such that:

$$\mathbb{P}(X \in A) = \mathbb{P}(X^{-1}(A)) = \int_A f_X(x) \, dx \qquad \forall A \in \mathcal{B} \tag{2.2}$$

Such a function f_X is called a probability density function of X.

Proposition 1. *Given a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and a random variable $X : \Omega \to \overline{\mathbb{R}}$, the composition $\mathbb{P}_X = \mathbb{P} \circ X^{-1}$ defines a measure on $(\overline{\mathbb{R}}, \overline{\mathcal{B}})$, called the probability distribution of X.*

Proof. We need to check the two defining properties of measures:

1. $\mathbb{P}_X(\emptyset) = \mathbb{P}(X^{-1}(\emptyset)) = \mathbb{P}(\emptyset) = 0$

2. $A_n \in \mathcal{A}, n \in \mathbb{N}$, pairwise disjoint $\implies \mathbb{P}_X(\bigcup_{n\in\mathbb{N}} A_n) = \mathbb{P}(X^{-1}(\bigcup_{n\in\mathbb{N}} A_n)) = \mathbb{P}((\bigcup_{n\in\mathbb{N}} X^{-1}(A_n)) = \sum_{n=1}^{\infty} \mathbb{P}(X^{-1}(A_n)) = \sum_{n=1}^{\infty} \mathbb{P}_X(A_n)$

Finally, $\mathbb{P}_X(\overline{\mathbb{R}}) = \mathbb{P}(X^{-1}(\overline{\mathbb{R}})) = \mathbb{P}(\Omega) = 1$, which indeed concludes that $\mathbb{P}_X$ defines a probability measure on $(\overline{\mathbb{R}}, \overline{\mathcal{B}})$. The claim also holds for $(\mathbb{R}, \mathcal{B})$. □

Proposition 2. *Given a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and $n \in \mathbb{N}$ random variables $X_i : \Omega \to \overline{\mathbb{R}}$ for $i = 1, \ldots, n$, the map $\boldsymbol{X} := (X_1, \ldots, X_n) : \Omega \to \overline{\mathbb{R}^n}$, defines a measurable function, which is called a random vector. The probability distribution $\mathbb{P}_{\boldsymbol{X}}$ of a random vector defines a measure, called the joint (probability) distribution of $\boldsymbol{X} = (X_1, \ldots, X_n)$. Moreover, the probability distribution $\mathbb{P}_{X_i}$ of the random variables X_i is called the i-th marginal distribution of $\boldsymbol{X}$.*

For conciseness purposes and due to the fact that the proof is similar to the one in the previous proposition, we offer no proof of the given proposition, and a detailed proof can be found in [34].

In the study of Bayesian networks, joint probability distributions and marginal distributions play an important role, in particular, there is an intricate relationship between the two distributions that we would like to utilize. Consider a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and $n \in \mathbb{N}$ discrete random variables $X_i : \Omega \to \mathbb{R}$ for $i \in \{1, \ldots, n\}$, where $\mathcal{X}_i \subset \mathbb{R}$ is the set of values attained by X_i. Then, for any random variable X_i, consider the set $X_i^{-1}(\{x_i\})$ for $x_i \in \mathcal{X}_i$. Notice that:

$$X_i^{-1}(\{x_i\}) = \bigcup_{(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n):\, x_i\in\mathcal{X}_i} (X_1^{-1}(\{x_1\}) \cap \ldots \cap X_n^{-1}(\{x_n\})) \tag{2.3}$$

which is a disjoint union since $X_i^{-1}(\{x_i\})$ are disjoint for all $x_i \in \mathcal{X}_i$. Therefore, we can apply the countable additivity property of the probability measure to (2.3) and obtain:

$$\begin{aligned}
\mathbb{P}_{X_i}(x_i) &= \mathbb{P}(X_i = x_i) \\
&= \mathbb{P}(X_i^{-1}(\{x_i\})) \\
&= \mathbb{P}\Big(\bigcup_{(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n):\, x_i\in\mathcal{X}_i} (X_1^{-1}(\{x_1\}) \cap \ldots \cap X_n^{-1}(\{x_n\}))\Big) \\
&= \sum_{(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n):\, x_i\in\mathcal{X}_i} \mathbb{P}\big(X_1^{-1}(\{x_1\}) \cap \ldots \cap X_n^{-1}(\{x_n\})\big) \\
&= \sum_{x_1\in\mathcal{X}_1} \cdots \sum_{x_{i-1}\in\mathcal{X}_{i-1}} \sum_{x_{i+1}\in\mathcal{X}_{i+1}} \cdots \sum_{x_n\in\mathcal{X}_n} \mathbb{P}\big(X_1^{-1}(\{x_1\}) \cap \ldots \cap X_n^{-1}(\{x_n\})\big) \\
&= \sum_{x_1\in\mathcal{X}_1} \cdots \sum_{x_{i-1}\in\mathcal{X}_{i-1}} \sum_{x_{i+1}\in\mathcal{X}_{i+1}} \cdots \sum_{x_n\in\mathcal{X}_n} \mathbb{P}\big((X_1, \ldots, X_n) = (x_1, \ldots, x_n)\big) \\
&= \sum_{x_1\in\mathcal{X}_1} \cdots \sum_{x_{i-1}\in\mathcal{X}_{i-1}} \sum_{x_{i+1}\in\mathcal{X}_{i+1}} \cdots \sum_{x_n\in\mathcal{X}_n} \mathbb{P}_{\boldsymbol{X}}((x_1, \ldots, x_n)).
\end{aligned} \tag{2.4}$$

Thus, from (2.4) we have obtained the marginal distribution $\mathbb{P}_{X_i}$ of X_i as an iterative sum of the joint distribution $\mathbb{P}_{\boldsymbol{X}}$. This is also known as *marginalizing*. A very similar derivation follows for joint distributions and marginal distributions of absolutely continuous variables, except that the summations are exchanged with iterative integration.

2.3 Probability Theory

One of the central themes in the study of Bayesian networks, in particular structure learning, is the notion of *conditional independence* [21]. To better understand the notion of conditional independence and its usefulness, we provide a motivating example.

Consider the following oversimplified scenario. Two neighbors, Jack and John live in the same building, but different apartments. Their building possesses a shared alarm. In the case of a burglary or a fire in their building, the chance of the alarm turning on is very high. Note that we say "the chance ..." to account for situation of a technical defect when the alarm does not turn on, regardless of how unrealistic such a situation is. In turn, if the alarm turns on, if either Jack or John are at home, it increases the chance that they call the emergency services. Given the presented scenario, we can model this situation by considering "Burglary", "Fire", "Alarm", "Jack" and "John" as binary random variables, meaning that they take on two values only, i.e. either there is a burglary or not, either there is a fire or not, either the alarm rings or not, either Jack calls the emergency services or not, and either John calls the emergency services or not. Graphically, one can represent the given scenario as in figure 2.3, with the nodes representing the five random variables we consider, and each directed edge representing the relationship between the two random variables, with the direction of the edge indicating the direction of influence.

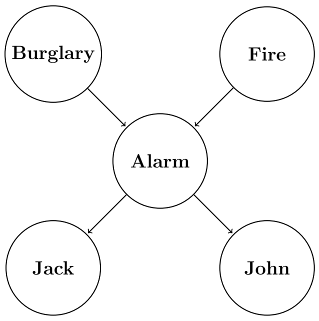

Figure 2.3: Graphical Representation of "Alarm" Example.

To illustrate the idea of conditional independence, we consider three different relationships among the random variables, presented in figure 2.4.

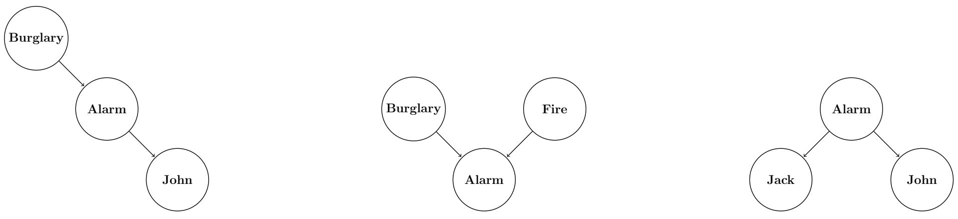

Figure 2.4: Three different relationships in “Alarm” Example.

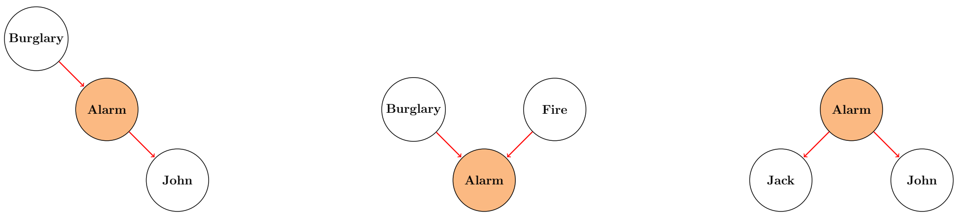

Figure 2.5: Conditioning on the variable “Alarm” in “Alarm” Example.

In the first case of figure 2.4, we see that “Burglary” influences “Alarm”, which in turn influences “John”, which means that in the event of a burglary, there is a higher probability that the alarm turns on, and thus a higher probability that John calls the emergency services. Thus, one is able to state that the random variables “Burglary” and “John” are dependent as “John” is indirectly influenced by “Burglary”. In the second case, we see that “Burglary” influences “Alarm” and “Fire” influences “Alarm”. However, there seems to be no apparent connection between the event of a burglary and the event of a fire. Therefore, one is able to state that “Burglary” and “Fire” are independent of each other. Finally, in the third case, we observe that “Alarm” influences both “John” and “Jack”, as the event of there being an alarm increases the chance of both John and Jack calling the emergency services. Therefore, we are able to state that “Jack” and “John” are dependent as they are both influenced by the same cause, and thus indirectly correlated.

However, if we *condition* on the variable “Alarm”, meaning that we assume that the alarm turns on (or not), the dependencies in each situation change (see figure 2.5). In the first case, conditioning on alarm “Alarm” makes “Burglary” and “John” independent. If we know that the alarm turns on, the event of a burglary does not have any effect on John calling the emergency services, and vice versa, if John calls, that does not influence the event of a burglary. In the second situation, conditioning on alarm “Alarm” makes “Burglary” and “Fire” dependent. Namely, if we know that the alarm turns on, then there being no fire increases the probability of there being a burglary, and if there is no burglary, the chances of a fire increase, thus these variables become dependent. Finally, conditioning on alarm “Alarm” makes “Jack” and “John” independent, since if we know that the alarm has turned on, then John calling for emergency has no influence on whether Jack calls the services.

While highly oversimplified and likely not very realistic, the provided example and its relatively lengthy discussion provide us with intuition and clear understanding of the idea behind conditional independencies, and moreover, it gives us an initial idea of how powerful such conditional independencies can be in much more complicated real-world scenarios, such as stock trading on international financial markets and gene expression in human cells. With the aid of measure theory, we are able to also succinctly formalize the concepts of independence and conditional independence in a probabilistic sense. This section closely follows the notation, terminology and results presented in [34].

Lemma 1. *Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space and $A, B \in \mathcal{A}$ be events such that $\mathbb{P}(B) > 0$. We define the conditional probability of A given B as:*

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}. \tag{2.5}$$

Define $\mathbb{P}_B : \mathcal{A} \to [0, 1]$ by $\mathbb{P}_B(A) = \mathbb{P}(A|B)$. $\mathbb{P}_B$ is a probability measure on $(\Omega, \mathcal{A})$.

Proof. We need to check the two defining properties of measures:

1. $\mathbb{P}_B(\emptyset) = \frac{\mathbb{P}(\emptyset \cap B)}{\mathbb{P}(B)} = \frac{\mathbb{P}(\emptyset)}{\mathbb{P}(B)} = 0$
2. $A_n \in \mathcal{A}, n \in \mathbb{N}$, pairwise disjoint $\implies \mathbb{P}_B(\bigcup_{n\in\mathbb{N}} A_n) = \frac{\mathbb{P}\big(\big(\bigcup_{n\in\mathbb{N}} A_n\big)\cap B\big)}{\mathbb{P}(B)} = \frac{\mathbb{P}(\bigcup_{n\in\mathbb{N}}(A_n \cap B))}{\mathbb{P}(B)} = \frac{\sum_{n=1}^{\infty}\mathbb{P}(A_n\cap B)}{\mathbb{P}(B)} = \sum_{n=1}^{\infty}\frac{\mathbb{P}(A_n\cap B)}{\mathbb{P}(B)} = \sum_{n=1}^{\infty}\mathbb{P}_B(A_n)$

Finally, $\mathbb{P}_B(\Omega) = \frac{\mathbb{P}(\Omega\cap B)}{\mathbb{P}(B)} = \frac{\mathbb{P}(B)}{\mathbb{P}(B)} = 1$, which indeed concludes that $\mathbb{P}_B$ defines a probability measure on $(\Omega, \mathcal{A})$ as given in (2.5). □

The conditional probability measure is instrumental in the notion of conditional independence. We distinguish independence concepts between events and random variables.

Definition 12 (Independence of Events and Collection of Events)**.**
Given a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and a collection of events $\{A_\lambda\}_{\lambda\in\Lambda}$ in $\mathcal{A}$ for some indexing set Λ, the events A_λ are called independent if for any finite subset of indices $\lambda_1, \dots, \lambda_n \in \Lambda$ we have:

$$\mathbb{P}(\cap_{i=1}^{n} A_{\lambda_i}) = \prod_{i=1}^{n} \mathbb{P}(A_{\lambda_i}).$$

Moreover, given a sequence of collection of events $\mathcal{F}_1, \mathcal{F}_2, \dots \subset \mathcal{A}$, we say $\mathcal{F}_1, \mathcal{F}_2, \dots$ are independent if for any $i_1, \dots, i_n$ with $1 \leq i_1 < i_2 < \dots < i_n$, and any $A_1 \in \mathcal{F}_{i_1}, A_2 \in \mathcal{F}_{i_2}, \dots, A_n \in \mathcal{F}_{i_n}$, we have:

$$\mathbb{P}(\cap_{i=1}^{n} A_i) = \prod_{i=1}^{n} \mathbb{P}(A_i).$$

This definition allows us to extend to the conditional probability measure, with which we are able to define conditional independence of events and collection of events. Namely, the events $\{A_\lambda\}_{\lambda\in\Lambda}$ in $\mathcal{A}$ for some indexing set Λ are called *conditionally independent* given an event $B \in \mathcal{A}$ with $\mathbb{P}(B) > 0$ if they are independent with respect to the conditional probability measure $\mathbb{P}_B$, i.e. $\mathbb{P}_B(\cap_{i=1}^n A_{\lambda_i}) = \prod_{i=1}^n \mathbb{P}_B(A_{\lambda_i}) \iff \mathbb{P}(\cap_{i=1}^n A_{\lambda_i}|B) = \prod_{i=1}^n \mathbb{P}(A_{\lambda_i}|B)$.

In similar fashion, a sequence of collection of events $\mathcal{F}_1, \mathcal{F}_2, \ldots \subset \mathcal{A}$ are called *conditionally independent* given an event $B \in \mathcal{A}$ with $\mathbb{P}(B) > 0$ if they are independent with respect to the conditional probability measure $\mathbb{P}_B$, i.e. if for any $i_1, \ldots, i_n$ with $1 \leq i_1 < i_2 < \ldots < i_n$, and any $A_1 \in \mathcal{F}_{i_1}, A_2 \in \mathcal{F}_{i_2}, \ldots, A_n \in \mathcal{F}_{i_n}$, we have $\mathbb{P}_B(\cap_{i=1}^n A_i) = \prod_{i=1}^n \mathbb{P}_B(A_i) \iff \mathbb{P}(\cap_{i=1}^n A_i|B) = \prod_{i=1}^n \mathbb{P}(A_i|B)$.

We shift our attention to random variables and aim to emulate the same idea in order to consider independence of random variables.

Definition 13 (The σ-algebra $\sigma(X)$)**.**
Given a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and the measurable space $(\mathbb{R}, \mathcal{B})$, for a random variable $X : \Omega \to \mathbb{R}$, we define the σ-algebra:

$$\sigma(X) = \sigma(\{X^{-1}(A)|\, A \in \mathcal{B}\})$$

A simple observation relying upon the properties of preimages yields that $\{X^{-1}(A)|\, A \in \mathcal{B}\}$ is a σ-algebra, which in turn shows that $\sigma(X) = \{X^{-1}(A)|\, A \in \mathcal{B}\}$.

Definition 14 (Independence of Random Variables)**.**
Given a probability space $(\Omega, \mathcal{A}, \mathbb{P})$, the measurable space $(\mathbb{R}, \mathcal{B})$ and a sequence of random variables $X_1, X_2, \ldots$ with $X_i : \Omega \to \mathbb{R}$, we say that the random variables are independent if the corresponding σ - algebras $\sigma(X_1), \sigma(X_2), \ldots$ are independent.

Using our previous observation and a few additional details, one is able to give an equivalent characterization of this definition, which states that two random variables X_1 and X_2 with $X_i : \Omega \to \mathbb{R}$ are independent if :

$$\mathbb{P}(X_1^{-1}(A_1) \cap X_2^{-1}(A_2)) = \mathbb{P}(X_1^{-1}(A_1)) \cdot \mathbb{P}(X_2^{-1}(A_2)) \qquad \forall A_1, A_2 \in \mathcal{B}. \tag{2.6}$$

For conciseness we omit the details of the proof, and refer the reader to [34]. The characterization of independence of random variables in (2.6) is the one commonly used in probability theory, and will be the one that we rely upon in our further discussion. One can inductively extend the definition to the case of independence of more than two random variables.

As in the case for independence of events, the presented definition together with the conditional probability measure allow us to define the notion of *conditional independence of random variables.*

Definition 15 (Conditional Independence of Random Variables)**.**
Given a probability space $(\Omega, \mathcal{A}, \mathbb{P})$, the measurable space $(\mathbb{R}, \mathcal{B})$, and the random variables X_1, X_2 and X_3 with $X_i : \Omega \to \mathbb{R}$, we say that X_1 and X_2 are conditionally independent given X_3 if:

$$\begin{aligned}
&\mathbb{P}_{X_3^{-1}(A_3)}(X_1^{-1}(A_1) \cap X_2^{-1}(A_2)) = \mathbb{P}_{X_3^{-1}(A_3)}(X_1^{-1}(A_1)) \cdot \mathbb{P}_{X_3^{-1}(A_3)}(X_2^{-1}(A_2)) \\
\iff &\mathbb{P}(X_1^{-1}(A_1) \cap X_2^{-1}(A_2)|X_3^{-1}(A_3)) = \mathbb{P}(X_1^{-1}(A_1)|X_3^{-1}(A_3)) \cdot \mathbb{P}(X_2^{-1}(A_2)|X_3^{-1}(A_3))
\end{aligned}$$

for all $A_1, A_2, A_3 \in \mathcal{B}$, with $\mathbb{P}(X_3^{-1}(A_3)) > 0$.

We note that one can inductively extend this definition to the case of conditional independence of more than two random variables.

Notation: With abuse of notation, one can use a shorthand way to write the set $X^{-1}(A) = \{\omega \in \Omega | \, X(w) \in A\}$ simply as $X \in A$. This allows us to simplify our expressions from before, and thus two random variables X_1 and X_2 are independent if:

$$\mathbb{P}(X_1 \in A_1, X_2 \in A_2) = \mathbb{P}(X_1 \in A_1) \cdot \mathbb{P}(X_2 \in A_2) \qquad \forall A_1, A_2 \in \mathcal{B}.$$

Furthermore, we say that X_1 and X_2 are conditionally independent given X_3 if:

$$\mathbb{P}(X_1 \in A_1, X_2 \in A_2 | X_3 \in A_3) = \mathbb{P}(X_1 \in A_1 | X_3 \in A_3) \cdot \mathbb{P}(X_2 \in A_2 | X_3 \in A_3)$$

for all $A_1, A_2, A_3 \in \mathcal{B}$ with $\mathbb{P}(X_3 \in A_3) > 0$. In fact, one could go even further, and with the fine understanding of the delicate formalities behind the definitions of independence and conditional independence, we can use further abuse of notation and state that X_1 and X_2 are independent if:

$$\mathbb{P}(X_1, X_2) = \mathbb{P}(X_1) \cdot \mathbb{P}(X_2),$$

and X_1 and X_2 are conditionally independent given X_3 if:

$$\mathbb{P}(X_1, X_2 | X_3) = \mathbb{P}(X_1 | X_3) \cdot \mathbb{P}(X_2 | X_3)$$

with $\mathbb{P}(X_3) > 0$. Moreover, a shorthand notation to indicate that X_1 and X_2 are conditionally independent given X_3 is to write $X_1 \perp\!\!\!\perp X_2 | \, X_3$. Finally, we refer to the definition of a joint probability distribution, and remark that with the same shorthand notation, one typically writes $\mathbb{P}(X_1, \ldots, X_n)$ for the joint distribution of the random vector $\boldsymbol{X} = (X_1, \ldots, X_n)$, and moreover, for the marginal distribution of X_i, we write $\mathbb{P}(X_i) = \sum_{X_1} \cdots \sum_{X_n} \mathbb{P}(\boldsymbol{X}) = \sum_{X_1} \cdots \sum_{X_n} \mathbb{P}\big(X_1, \ldots, X_n\big)$, in case we have discrete random variables, where in the iterative summation we do not sum over the values of X_i. A similar expression follows for absolutely continuous random variables with the summation symbols replaced by integration symbols.

Using this notation is standard in literature on Bayesian networks, and prevents cluttered text. Nevertheless, it is the author's personal conviction, that one must understand what one is doing when using such shorthand notation. To that end, the presented formal treatment of independence and conditional independence justifies our use of shorthand notation.

We introduce one final result from probability theory needed in section 2.4.

Lemma 2. *Consider the probability space* $(\Omega, \mathcal{A}, \mathbb{P})$*, the measurable space* $(\mathbb{R}, \mathcal{B})$*, and the random variables* X_1, X_2 *and* X_3 *with* $X_i : \Omega \to \mathbb{R}$*.* X_1 *and* X_2 *are conditionally independent given* X_3 *if and only if there exist (set) functions* f *and* g *such that* $\mathbb{P}(X_1 \in A_1, X_2 \in A_2 | X_3 \in A_3) = f(X_1 \in A_1, X_3 \in A_3) g(X_2 \in A_2, X_3 \in A_3)$ *for all* $A_1, A_2, A_3 \in \mathcal{B}$ *with* $\mathbb{P}(X_3 \in A_3) > 0$*.*

Proof. ($\Rightarrow$) This follows by the definition of conditional independence by taking f and g to be the conditional probability measure.
($\Leftarrow$) Suppose there exist (set) functions f and g such that $\mathbb{P}(X_1 \in A_1, X_2 \in A_2 | X_3 \in A_3) = f(X_1 \in A_1, X_3 \in A_3) g(X_2 \in A_2, X_3 \in A_3)$ for all $A_1, A_2, A_3 \in \mathcal{B}$ with $\mathbb{P}(X_3 \in A_3) > 0$.

Note that $\mathbb{P}(X_1 \in \mathbb{R}, X_2 \in \mathbb{R} |\, X_3 \in A_3) = 1$ for any A_3 with $\mathbb{P}(X_3 \in A_3) > 0$, since $\mathbb{P}$ is a probability measure. On the other hand, by assumption, $\mathbb{P}(X_1 \in \mathbb{R}, X_2 \in \mathbb{R} |\, X_3 \in A_3) = f(X_1 \in \mathbb{R}, X_3 \in A_3) g(X_2 \in \mathbb{R}, X_3 \in A_3)$. Therefore, we conclude that $f(X_1 \in \mathbb{R}, X_3 \in A_3) g(X_2 \in \mathbb{R}, X_3 \in A_3) = 1$.

Furthermore, note that since $\mathbb{P}$ is a probability measure, we obtain:

$$\begin{aligned}
\mathbb{P}(X_1 \in A_1 |\, X_3 \in A_3) &= \mathbb{P}(X_1 \in A_1, X_2 \in \mathbb{R} |\, X_3 \in A_3) \\
&= f(X_1 \in A_1, X_3 \in A_3) g(X_2 \in \mathbb{R}, X_3 \in A_3) \qquad (2.7) \\
\mathbb{P}(X_2 \in A_2 |\, X_3 \in A_3) &= \mathbb{P}(X_1 \in \mathbb{R}, X_2 \in A_2 |\, X_3 \in A_3) \\
&= f(X_1 \in \mathbb{R}, X_3 \in A_3) g(X_2 \in A_2, X_3 \in A_3) \qquad (2.8)
\end{aligned}$$

for all $A_1, A_2, A_3 \in \mathcal{B}$ with $\mathbb{P}(X_3 \in A_3) > 0$. By combining our initial remark with (2.7) and (2.8), we obtain:

$$\begin{aligned}
&\mathbb{P}(X_1 \in A_1 |\, X_3 \in A_3) \mathbb{P}(X_2 \in A_2 |\, X_3 \in A_3) \\
&= f(X_1 \in A_1, X_3 \in A_3) g(X_2 \in \mathbb{R}, X_3 \in A_3) f(X_1 \in \mathbb{R}, X_3 \in A_3) g(X_2 \in A_2, X_3 \in A_3) \\
&= f(X_1 \in A_1, X_3 \in A_3) g(X_2 \in A_2, X_3 \in A_3) \\
&= \mathbb{P}(X_1 \in A_1, X_2 \in A_2 | X_3 \in A_3)
\end{aligned}$$

for all $A_1, A_2, A_3 \in \mathcal{B}$ with $\mathbb{P}(X_3 \in A_3) > 0$. The second equality follows from the observation that $f(X_1 \in \mathbb{R}, X_3 \in A_3) g(X_2 \in \mathbb{R}, X_3 \in A_3) = 1$ and the final equality follows by assumption.

This shows that X_1 and X_2 are conditionally independent given X_3. This proves the lemma as we have shown both directions of the claim. $\square$

With our introduced shorthand notation, through abuse of notation, one can formulate the claim succinctly by stating that $X_1 \perp\!\!\!\perp X_2 | X_3 \iff \exists f, g$ such that $\mathbb{P}(X_1, X_2 | X_3) = f(X_1, X_3)g(X_2, X_3)$ with $\mathbb{P}(X_3) > 0$.

2.4 Bayesian Networks

Having formally defined the notion of conditional independence, the concepts of a Markov blanket and a parent set for a node in a graph, we take one additional step and introduce the *local Markov property* and the *probability chain rule*. With these we formally define *Bayesian networks*, and state and prove results related to conditional independence of random variables in a DAG G. These results take a central role in the study of Bayesian networks, Bayesian network structure learning, and in our treatment of Bayesian networks with background information and incomplete information.

Definition 16 (Local Markov Property)**.**
Consider a DAG G with n nodes represented and given by the random variables $X_1, \dots, X_n$. We say that G satisfies the local Markov property if for $i \in \{1, \dots, n\}$, X_i is conditionally independent of its non-descendants given $Pa(X_i)$.

If we consider example 1 and focus on the node V_4, then $pa(V_4) = \{V_2\}$, and by the local Markov property, $V_4 \perp\!\!\!\perp \{V_1, V_3, V_5\} | V_2$.

Note that in section 2.1 we used the capital letter V to indicate a node in a graph, whereas in section 2.3, we used the capital letter X to indicate a random variable. In the study of Bayesian networks, the nodes of a DAG are random variables, so throughout the remainder of this paper, we denote both graph nodes in a DAG and random variables with the capital letter X, except where appropriate to do otherwise.

With the statement of the local Markov property, we are able to provide a formal definition of a Bayesian network.

Definition 17 (Bayesian Network)**.**
A Bayesian network is a DAG G with n nodes represented and given by the random variables $X_1, \dots, X_n$ that satisfies the local Markov property.

While this definition is rather simple to understand, we are able to go one step further and provide equivalent characterizations of the local Markov property in a directed acylic graph. These characterizations are typically of great aid when one engages in the task of learning the structure of a Bayesian network from a given dataset.

Proposition 3 (Probability Chain Rule)**.**
Given a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and $n \in \mathbb{N}$ random variables $X_1, \dots, X_n$, the joint probability distribution of $\boldsymbol{X} = (X_1, \dots, X_n)$ satisfies:

$$\mathbb{P}(X_1, \dots, X_n) = \prod_{i=1}^{n} \mathbb{P}(X_i | \, X_1, \dots, X_{i-1}) \tag{2.9}$$

Proof. The proof of (2.9) follows by induction on n for $n \geq 2$.

1. $n = 2$: For two random variables X_1, X_2 using the conditional probability measure, we simply obtain:

   $$\mathbb{P}(X_1, X_2) = \mathbb{P}(X_1)\mathbb{P}(X_2 | \, X_1)$$

   as desired.

2. $n = k$: Assume that the statement holds for some $k \in \mathbb{N}$ with $k > 2$, i.e. for any k random variables $X_1, \dots, X_k$:

   $$\mathbb{P}(X_1, \dots, X_k) = \prod_{i=1}^{k} \mathbb{P}(X_i | \, X_1, \dots, X_{i-1}) \tag{2.10}$$

3. $n = k + 1$: For any $k + 1$ random variables, we make use of the conditional probability measure to obtain:

   $$\begin{aligned} \mathbb{P}(X_1, \dots, X_{k+1}) &= \mathbb{P}(X_1, \dots, X_k)\mathbb{P}(X_{k+1} | \, X_1, \dots, X_k) \\ &= \prod_{i=1}^{k} \mathbb{P}(X_i | \, X_1, \dots, X_{i-1})\mathbb{P}(X_{k+1} | \, X_1, \dots, X_k) \\ &= \prod_{i=1}^{k+1} \mathbb{P}(X_i | \, X_1, \dots, X_{i-1}) \end{aligned} \tag{2.11}$$

   where in the second equality of (2.11) we made use of the induction hypothesis in (2.10). Therefore, the result follows by induction.

□

We state and prove two further results regarding conditional independence, already hinted upon in section 2.3.

Lemma 3. *Given a Bayesian network represented by a DAG G with $n \in \mathbb{N}$ random variables $X_1, \dots, X_n$ as nodes, the joint distribution of $X_1, \dots, X_n$ factorizes as:*

$$\mathbb{P}(X_1, \dots, X_n) = \prod_{i=1}^{n} \mathbb{P}(X_i | Pa(X_i)) \tag{2.12}$$

Proof. According to the probability chain rule, it holds that $\mathbb{P}(X_1, \dots, X_n) = \prod_{i=1}^{n} \mathbb{P}(X_i | X_1, \dots, X_{i-1})$. In particular this holds for a given topological ordering of G, which then implies that for a node X_i, the non-descendants of X_i are a subset of $X_1, \dots, X_{i-1}$. Therefore, using the local Markov property, we obtain:

$$\begin{aligned}
\mathbb{P}(X_i | X_1, \dots, X_{i-1}) &= \frac{\mathbb{P}(X_1, \dots, X_i)}{\mathbb{P}(X_1, \dots, X_{i-1})} \\
&= \frac{\mathbb{P}(\{X_1, \dots, X_i\} \setminus Pa(X_i), Pa(X_i))}{\mathbb{P}(\{X_1, \dots, X_{i-1}\} \setminus Pa(X_i), Pa(X_i))} \\
&= \frac{\mathbb{P}(\{X_1, \dots, X_i\} \setminus Pa(X_i) | Pa(X_i))}{\mathbb{P}(\{X_1, \dots, X_{i-1}\} \setminus Pa(X_i) | Pa(X_i))} \\
&= \frac{\mathbb{P}(X_i | Pa(X_i)) \cdot \mathbb{P}(\{X_1, \dots, X_{i-1}\} \setminus Pa(X_i) | Pa(X_i))}{\mathbb{P}(\{X_1, \dots, X_{i-1}\} \setminus Pa(X_i) | Pa(X_i))} \\
&= \mathbb{P}(X_i | Pa(X_i))
\end{aligned} \tag{2.13}$$

which yields the desired result in (2.12). Note that in the second equality of (2.13), we used that $\{X_1, \dots, X_i\} = \{X_1, \dots, X_{i-1}\} \setminus Pa(X_i) \cup Pa(X_i)$, in the third equality we used the definition of conditional probability, and in the fourth equality we used the local Markov property. $\square$

Note that in lemma 4, we use bold notation for the subsets of nodes to clearly distinguish them from the single nodes involved in the result.

Lemma 4. *Consider a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and a Bayesian network G with $n \in \mathbb{N}$ random variables $\{X_1, \dots, X_n\}$ as nodes. If $\boldsymbol{S_1}$, $\boldsymbol{S_2}$ and $\boldsymbol{Z}$ are subsets of nodes in G such that $\boldsymbol{S_1}$ and $\boldsymbol{S_2}$ are d-separated with respect to $\boldsymbol{Z}$, then $\boldsymbol{S_1} \perp\!\!\!\perp \boldsymbol{S_2} | \boldsymbol{Z}$.*

Proof. We distinguish two cases.

1. $\boldsymbol{S_1} \cup \boldsymbol{S_2} \cup \boldsymbol{Z} = \{X_1, \dots, X_n\}$
   Let $\boldsymbol{Z} = \boldsymbol{Z_1} \cup \boldsymbol{Z_2}$ such that $\boldsymbol{Z_1}$ is the set of nodes in $\boldsymbol{Z}$ with parent nodes in $\boldsymbol{S_1}$ and $\boldsymbol{Z_2} = \boldsymbol{Z} \setminus \boldsymbol{Z_1}$. Since $\boldsymbol{Z}$ d-separates $\boldsymbol{S_1}$ and $\boldsymbol{S_2}$, then we observe that:

$$\forall\, X \in \boldsymbol{S_1} \cup \boldsymbol{Z_1} \implies Pa(X) \subseteq \boldsymbol{S_1} \cup \boldsymbol{Z} \tag{2.14}$$
$$\forall\, X \in \boldsymbol{S_2} \cup \boldsymbol{Z_2} \implies Pa(X) \subseteq \boldsymbol{S_2} \cup \boldsymbol{Z} \tag{2.15}$$

   Since G is a Bayesian network, it satisfies the local Markov property, so using lemma 3 together with our previous observation in (2.14) and (2.15) allows us to factorize the joint probability distribution as follows:

$$\begin{aligned}
\mathbb{P}(X_1, \dots, X_n) &= \mathbb{P}(\boldsymbol{S_1}, \boldsymbol{S_2}, \boldsymbol{Z}) \\
&= \prod_{X \in \boldsymbol{S_1} \cup \boldsymbol{S_2} \cup \boldsymbol{Z}} \mathbb{P}(X | Pa(X)) \\
&= \prod_{X \in \boldsymbol{S_1} \cup \boldsymbol{Z_1}} \mathbb{P}(X | Pa(X)) \prod_{X \in \boldsymbol{S_2} \cup \boldsymbol{Z_2}} \mathbb{P}(X | Pa(X)) \\
&= f(\boldsymbol{S_1}, \boldsymbol{Z}) g(\boldsymbol{S_2}, \boldsymbol{Z})
\end{aligned} \tag{2.16}$$

where $f(\boldsymbol{S_1}, \boldsymbol{Z}) = \prod_{X \in \boldsymbol{S_1} \cup \boldsymbol{Z_1}} \mathbb{P}(X|Pa(X))$ and $g(\boldsymbol{S_2}, \boldsymbol{Z}) = \prod_{X \in \boldsymbol{S_2} \cup \boldsymbol{Z_2}} \mathbb{P}(X|Pa(X))$ in (2.16). Moreover, from this we obtain that for $\mathbb{P}(\boldsymbol{Z}) > 0$:

$$\begin{aligned} \mathbb{P}(\boldsymbol{S_1}, \boldsymbol{S_2}|\, \boldsymbol{Z}) &= \frac{\mathbb{P}(\boldsymbol{S_1}, \boldsymbol{S_2}, \boldsymbol{Z})}{\mathbb{P}(\boldsymbol{Z})} \\ &= \frac{f(\boldsymbol{S_1}, \boldsymbol{Z}) g(\boldsymbol{S_2}, \boldsymbol{Z})}{\mathbb{P}(\boldsymbol{Z})} \\ &= \frac{f(\boldsymbol{S_1}, \boldsymbol{Z})}{\mathbb{P}(\boldsymbol{Z})} g(\boldsymbol{S_2}, \boldsymbol{Z}) \\ &= h(\boldsymbol{S_1}, \boldsymbol{Z}) g(\boldsymbol{S_2}, \boldsymbol{Z}) \end{aligned} \tag{2.17}$$

where in (2.17) we set $h(\boldsymbol{S_1}, \boldsymbol{Z}) = \frac{f(\boldsymbol{S_1}, \boldsymbol{Z})}{\mathbb{P}(\boldsymbol{Z})}$. By lemma 2, this implies that $\boldsymbol{S_1} \perp\!\!\!\perp \boldsymbol{S_2}|\, \boldsymbol{Z}$, as desired.

2. $\boldsymbol{S_1} \cup \boldsymbol{S_2} \cup \boldsymbol{Z} \subset \{X_1, \ldots, X_n\}$
   Let $\boldsymbol{M} = \boldsymbol{S_1} \cup \boldsymbol{S_2} \cup \boldsymbol{Z}$ and $\tilde{\boldsymbol{M}} = \boldsymbol{M} \cup An(\boldsymbol{M})$, where $An(\boldsymbol{M})$ denotes the set of ancestor nodes of $\boldsymbol{M}$. We denote the subgraph of G formed by the nodes in $\tilde{\boldsymbol{M}}$ as $\tilde{G}$. Furthermore, we make the claim that the joint distribution of $\boldsymbol{M}$ is equal in both G and $\tilde{G}$, i.e. symbolically $\mathbb{P}_G(\boldsymbol{M}) = \mathbb{P}_{\tilde{G}}(\boldsymbol{M})$, where the subscript indicates the graph over which we consider the joint distribution of $\boldsymbol{M}$.

   To see this, we first prove the following intermediate result. Given a *leaf* node (a node with no children) L in G, let G^* be the subgraph of G obtained by removing L. Let $\boldsymbol{K}$ be a subset of nodes in G^*. Then, the joint distribution of $\boldsymbol{K}$ is equal in G and G^*, i.e. symbolically as before, $\mathbb{P}_G(\boldsymbol{K}) = \mathbb{P}_{G^*}(\boldsymbol{K})$. Indeed, using the marginalization of K, we obtain:

$$\begin{aligned} \mathbb{P}_G(\boldsymbol{K}) &= \sum_L \mathbb{P}_G(\boldsymbol{K}, L) \\ &= \sum_L \Bigg(\prod_{W \in \boldsymbol{K}} \mathbb{P}_G(W|\, Pa(W)) \Bigg) \mathbb{P}_G(L|Pa(L)) \\ &= \prod_{W \in \boldsymbol{K}} \mathbb{P}_G(W|\, Pa(W)) \sum_L \mathbb{P}_G(L|Pa(L)) \\ &= \prod_{W \in \boldsymbol{K}} \mathbb{P}_G(W|\, Pa(W)) \\ &= \mathbb{P}_{G^*}(\boldsymbol{K}) \end{aligned} \tag{2.18}$$

   where in the second equality of (2.18) we used the fact that G is a Bayesian network and satisfies the local Markov property (lemma 3), and in the fourth equality we use the fact that $\sum_L \mathbb{P}_G(L|Pa(L)) = 1$. Note that we assumed G to be a discrete network for simplicity. The claim follows analogously for continuous Bayesian networks.

Returning back to our original problem, we again consider the sets $\boldsymbol{M} = \boldsymbol{S_1} \cup \boldsymbol{S_2} \cup \boldsymbol{Z}$ and $\tilde{\boldsymbol{M}} = \boldsymbol{M} \cup An(\boldsymbol{M})$. Namely, using our previous claim, we can take the following procedure to obtain that $\mathbb{P}_G(\boldsymbol{M}) = \mathbb{P}_{\tilde{G}}(\boldsymbol{M})$:

a) Find a leaf in G and remove it.

b) Repeat until there are nodes outside of $\tilde{\boldsymbol{M}}$.

This procedure results in $\tilde{\boldsymbol{M}}$, and according to our intermediate result, the joint probability of M remains the same throughout the procedure, thus it follows that $\mathbb{P}_G(\boldsymbol{M}) = \mathbb{P}_{\tilde{G}}(\boldsymbol{M})$.

We proceed with a similar approach as in the first case, since now $\tilde{\boldsymbol{M}}$ is a complete set of nodes in $\tilde{G}$. To that end, let $\boldsymbol{S_1}^*$ be the set of nodes in $\tilde{\boldsymbol{M}}$ that are not d-separated from $\boldsymbol{S_1}$ by $\boldsymbol{Z}$, and let $\boldsymbol{S_2}^* = \tilde{\boldsymbol{M}} \setminus (\boldsymbol{S_1}^* \cup \boldsymbol{Z})$. By doing so, we have split $\tilde{\boldsymbol{M}}$ as $\tilde{\boldsymbol{M}} = \boldsymbol{S_1}^* \cup \boldsymbol{S_2}^* \cup \boldsymbol{Z}$, such that $\boldsymbol{S_1}^*$ and $\boldsymbol{S_2}^*$ are d-separated given $\boldsymbol{Z}$. This is the exact situation that we had in case 1. Therefore, we know that there exist functions f, g such that:

$$\begin{aligned}
\mathbb{P}_{\tilde{G}}(\tilde{\boldsymbol{M}}) &= \mathbb{P}_{\tilde{G}}(\boldsymbol{S_1}^*, \boldsymbol{S_2}^*, \boldsymbol{Z}) \\
&= f(\boldsymbol{S_1}^*, \boldsymbol{Z}) g(\boldsymbol{S_2}^*, \boldsymbol{Z})
\end{aligned}$$

Moreover, by noticing that $\boldsymbol{S_1} \subset \boldsymbol{S_1}^*$ and $\boldsymbol{S_2} \subset \boldsymbol{S_2}^*$, we set $\tilde{\boldsymbol{S_1}} = \boldsymbol{S_1}^* \setminus \boldsymbol{S_1}$ and $\tilde{\boldsymbol{S_2}} = \boldsymbol{S_2}^* \setminus \boldsymbol{S_2}$ to obtain using marginalization of $\boldsymbol{S_1}$ and $\boldsymbol{S_2}$:

$$\begin{aligned}
\mathbb{P}_{\tilde{G}}(\boldsymbol{M}) &= \mathbb{P}_{\tilde{G}}(\boldsymbol{S_1}, \boldsymbol{S_2}, \boldsymbol{Z}) \\
&= \sum_{\tilde{\boldsymbol{S_1}}, \tilde{\boldsymbol{S_2}}} \mathbb{P}_{\tilde{G}}(\boldsymbol{S_1}, \tilde{\boldsymbol{S_1}}, \boldsymbol{S_2}, \tilde{\boldsymbol{S_2}}, \boldsymbol{Z}) \\
&= \sum_{\tilde{\boldsymbol{S_1}}, \tilde{\boldsymbol{S_2}}} \mathbb{P}_{\tilde{G}}(\boldsymbol{S_1}^*, \boldsymbol{S_2}^*, \boldsymbol{Z}) \\
&= \sum_{\tilde{\boldsymbol{S_1}}, \tilde{\boldsymbol{S_2}}} f(\boldsymbol{S_1}^*, \boldsymbol{Z}) g(\boldsymbol{S_2}^*, \boldsymbol{Z}) \\
&= \sum_{\tilde{\boldsymbol{S_1}}, \tilde{\boldsymbol{S_2}}} f(\boldsymbol{S_1}, \tilde{\boldsymbol{S_1}}, \boldsymbol{Z}) g(\boldsymbol{S_2}, \tilde{\boldsymbol{S_2}}, \boldsymbol{Z}) \\
&= \sum_{\tilde{\boldsymbol{S_1}}} f(\boldsymbol{S_1}, \tilde{\boldsymbol{S_1}}, \boldsymbol{Z}) \sum_{\tilde{\boldsymbol{S_2}}} g(\boldsymbol{S_2}, \tilde{\boldsymbol{S_2}}, \boldsymbol{Z}) \\
&= h(\boldsymbol{S_1}, \boldsymbol{Z}) m(\boldsymbol{S_2}, \boldsymbol{Z})
\end{aligned} \tag{2.19}$$

where in (2.19) we define the set functions h, m as $h(\boldsymbol{S_1}, \boldsymbol{Z}) = \sum_{\tilde{\boldsymbol{S_1}}} f(\boldsymbol{S_1}, \tilde{\boldsymbol{S_1}}, \boldsymbol{Z})$ and $m(\boldsymbol{S_2}, \boldsymbol{Z}) = \sum_{\tilde{\boldsymbol{S_2}}} g(\boldsymbol{S_2}, \tilde{\boldsymbol{S_2}}, \boldsymbol{Z})$. Since we have shown that $\mathbb{P}_G(\boldsymbol{M}) = \mathbb{P}_{\tilde{G}}(\boldsymbol{M})$, this implies that there exist functions h, m such that $\mathbb{P}(\boldsymbol{S_1}, \boldsymbol{S_2}, \boldsymbol{Z}) = h(\boldsymbol{S_1}, \boldsymbol{Z}) m(\boldsymbol{S_2}, \boldsymbol{Z})$. The rest follows analogously as case 1 through the utilization of lemma 2.

Since we proved both cases, this concludes the proof of the lemma. □

Combined together, these two results lead to the main theorem related to conditional independence in a Bayesian network.

Theorem 1. *Consider a probability space* $(\Omega, \mathcal{A}, \mathbb{P})$ *and a DAG* G *with* $n \in \mathbb{N}$ *random variables* $X_1, \dots, X_n$ *as nodes. The following statements are equivalent:*

1. $\mathbb{P}(X_i|\, X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n) = \mathbb{P}(X_i|\, MB(X_i)) \qquad \forall i \in \{1, \dots, n\}$
2. $\mathbb{P}(X_1, \dots, X_n) = \prod_{i=1}^{n} \mathbb{P}(X_i|Pa(X_i))$
3. *G satisfies the local Markov property.*

Proof of Theorem 1.
((1) ⇒ (2)) This implication follows by induction on the number of nodes in G.

1. $n = 2$
   We have a DAG with 2 nodes, denoted by X_1, X_2. Regardless of what the dependence relation between these two nodes is, we have that $MB(X_1) = \{X_2\}$ and $MB(X_2) = \{X_1\}$, which proves the result immediately, using the definition of conditional probability. In case the nodes are not connected, then the claim follows immediately by definition of independence.

2. $n = k$ for $k \geq 3$
   Assume the claim holds for any DAG with $k \geq 3$ nodes, i.e. $\mathbb{P}(X_1, \dots, X_k) = \prod_{i=1}^{k} \mathbb{P}(X_i|Pa(X_i))$. In particular, notice that this holds for any DAG with an imposed topological ordering.

3. $n = k + 1$
   We need to prove the claim for a DAG with $k+1$ nodes. By our previous remark, we impose a topological ordering on the DAG, such that each node X_i is preceded by its ancestors in the ordering, and moreover the last node in the topological ordering has no children. If we denote this topological ordering as $(X_1, \dots, X_{k+1})$, then by the last remark, $MB(X_{k+1}) = Pa(X_{k+1})$. Using the definition of conditional probability, we obtain:

$$\begin{aligned}\mathbb{P}(X_1,\dots,X_{k+1}) &= \mathbb{P}(X_{k+1}|\, X_1,\dots,X_k)\mathbb{P}(X_1,\dots,X_k)\\ &= \mathbb{P}(X_{k+1}|\, X_1,\dots,X_k)\prod_{i=1}^{k}\mathbb{P}(X_i|Pa(X_i))\\ &= \mathbb{P}(X_{k+1}|\, MB(X_{k+1}))\prod_{i=1}^{k}\mathbb{P}(X_i|Pa(X_i))\\ &= \mathbb{P}(X_{k+1}|\, Pa(X_{k+1}))\prod_{i=1}^{k}\mathbb{P}(X_i|Pa(X_i))\\ &= \prod_{i=1}^{k+1}\mathbb{P}(X_i|Pa(X_i))\end{aligned} \tag{2.20}$$

where in the second equality of (2.20) we used the induction hypothesis, and in the third equality we used the given assumption.

This proves (1) $\Rightarrow$ (2) by induction.

((2) $\Rightarrow$ (3)) On the one hand, by assumption $\mathbb{P}(X_1,\dots,X_n) = \prod_{i=1}^{n}\mathbb{P}(X_i|Pa(X_i))$. On the other hand, noting that $Pa(X_i)$ d-separates X_i from its non-descendants, together with lemma 4 proves the claim. To see this consider a path between X_i and a non-descendant node Z, with a node Y being adjacent to X_i on the path to Z. We have two possibilities. If $Y \in Pa(X_i)$, then Y is not a collider on the path, and the path is block conditional on $Pa(X_i)$. If $Y \notin Pa(X_i)$, then moving along the path we reach a v-structure, since this is a path from X_i to a non-descendant, but the node forming this v-structure alongside its descendants are not elements of $Pa(X_i)$, thus the path is block conditional on $Pa(X_i)$. In both cases, we see that $Pa(X_i)$ d-separates X_i from its non-descendants, and thus G satisfies the local Markov property, as desired.

((3) $\Rightarrow$ (1)) Since G satisfies the local Markov property, it is a Bayesian network, so by lemma 4, if we are able to show that any node X_i in G is d-separated from all other nodes in the network given its Markov blanket $MB(X_i)$, we are done. Indeed, this is true. Observe that any path between X and a node outside of $MB(X_i)$ passes either through a child node or a parent node or a parent of a child node, and each one of these nodes is not a collider in the path and it is a part of the Markov blanket of X_i. Therefore, each path from X_i to a node outside of $MB(X_i)$ is block conditional given $MB(X_i)$, thus by definition of d-separation, this implies that X_i is d-separated from all other nodes in the network given its Markov blanket $MB(X_i)$. This completes the proof of (3) $\Rightarrow$ (1).

By showing (1) $\Rightarrow$ (2) $\Rightarrow$ (3) $\Rightarrow$ (1), the proof of the theorem is complete. □

3 Bayesian Network Structure Learning

A common goal in the study of Bayesian networks is to infer the structure of the underlying Bayesian network given a dataset. This is commonly known as *Bayesian network (BN) structure learning* [12]. Namely, if we have measured or collected data about a natural or social process in which different variables (factors or actors) interact among each other, we are interested in learning which variables are connected to each other, i.e. we want to identify the (in-)dependencies between variables. This helps in understanding the underlying probabilistic relationships within the data, which in turn allows us to factorize the joint distribution of all the involved variables in a convenient way, leading to more efficient computation and inference.

3.1 The Space of DAGs

While structure learning is a task of great interest, it is often a challenging task to complete due to the fact that the number of possible DAGs, denoted by D_n, grows super-exponentially in the number of labelled nodes $n \in \mathbb{N}$.

Theorem 2. *For $D_0 = 1$, D_n $(n \geq 1)$ satisfies the recurrence relation*

$$D_n = \sum_{k=1}^{n} (-1)^{k+1} \binom{n}{k} 2^{k(n-k)} D_{n-k}. \tag{3.1}$$

A simple iteration of (3.1) reveals that already in the case when we have $n = 6$, $D_6 = 3781503 \approx 4 \times 10^6$. Given that most processes, which are of real-world interest and applicability, typically involve hundreds or even thousands of interacting variables, the already large number of possible DAGs for a single-digit number of variables indicates that estimating a fitting Bayesian network for a given dataset is a computationally heavy task. The result in theorem 2 was first proved by Robinson in 1973 [30]. The proof by Robinson is rather involved and beyond the scope of this thesis as it heavily relies upon notions from *group theory*. To that end, we offer a different proof, building on the combinatorial ideas in [30] and *ordinary generating functions* (OGFs) from the field of *discrete mathematics* [33]. To the best of our knowledge and conducted literature review, this is also a new proof of the result in theorem 2 (see [30, 25, 15]).

Definition 18 (Ordinary Generating Function)**.**
Given a sequence of complex numbers $(a_n)_{n=0}^{\infty}$, the ordinary generating function (OGF) associated with this sequence is:

$$A(x) = \sum_{n=0}^{\infty} a_n x^n.$$

For our purposes, we limit the discussion to sequences of real numbers, in particular positive integers, as we are interested in the number of directed acyclic graphs G on $n \in \mathbb{N}$ nodes. One might recognize that the expression provided for an OGF is the same as the one for a power series. A crucial analytic aspect of studying power series, from an analysis point of view, is the notion of convergence. In particular, we aim at finding the *radius of convergence* R. However, when discussing OGFs we are mainly interested in the algebraic manipulation of the series with little consideration for convergence properties. This is because the main goal of generating functions is that they allow us to turn counting problems and recurrence relations, or more generally questions about sequences of numbers, into questions about certain functions to which we can apply the powerful tools one learns in calculus [7, 33].

To equip this idea with mathematical rigour, one introduces the *formal theory* of power series, as opposed to the well-known *analytic theory* of power series. In particular, we introduce an algebraic structure called the *ring of formal power series* [33]. In the context of generating functions as an algebraic object, the terms “formal power series” and “ordinary generating function” are used interchangeably.

Definition 19 (Formal Power Series)**.**
A formal power series is an expression of the form:

$$a_0 + a_1x + a_2x^2 + \dots$$

where the sequence $(a_n)_{n=0}^{\infty}$ is called the sequence of coefficients. On the collection of formal power series, we define an additive and a multiplicative operation:

$$\sum_{n=0}^{\infty} a_nx^n \pm \sum_{n=0}^{\infty} b_nx^n = \sum_{n=0}^{\infty}(a_n \pm b_n)x^n$$
$$\sum_{n=0}^{\infty} a_nx^n \cdot \sum_{n=0}^{\infty} b_nx^n = \sum_{n=0}^{\infty} c_nx^n \qquad (c_n = \sum_{k} a_kb_{n-k})$$

Proposition 4. *A formal power series $f = \sum_{n=0}^{\infty} a_nx^n$ is invertible if and only if $a_0 \neq 0$.*

Proof. ($\Rightarrow$) Suppose f is invertible, and thus an inverse element $\frac{1}{f} = \sum_{n=0}^{\infty} b_nx^n$ exists such that $f \cdot \frac{1}{f} = 1$. Following the product operation we defined on the collection of formal power series, we obtain $c_0 = a_0b_0$ and moreover $c_0 = 1$. Therefore, it follows that $a_0 \neq 0$. A simple observation using the multiplicative operation yields for $n \geq 1$:

$$b_n = (-\frac{1}{a_0}) \sum_{k=1} a_nb_{n-k}$$

which determines $\frac{1}{f}$ uniquely as desired.
($\Leftarrow$) Suppose $a_0 \neq 0$. In that case, we can use the previously derived expression for b_n to obtain the inverse element $\frac{1}{f}$ of f, which shows that f is invertible. $\square$

With the provided additive and multiplicative operations, with the common additive inverse 0 and multiplicative inverse 1 on the one hand, and the invertible elements given as per proposition 4, one obtains a ring structure, commonly known as the ring of formal power series. To that end, one can make an analogy with the formal, algebraic definition of a polynomial. In that sense, a formal power series is a potentially infinite ordered list of coefficients. Consequently, this also justifies the algebraic manipulation of formal power series, in particular OGFs, without considering aspects of convergence [33].

With the formal consideration of generating functions, we are able to use them in the proof of theorem 2. Note that to avoid cluttered notation in the proof of theorem 2, we divert from our convention of denoting vertices with the capital letter X, and instead use the lowercase letter v with an appropriate subscript to denote vertices.

Proof of Theorem 2.
Given $n \in \mathbb{N}$ nodes, we observe that the number of directed acyclic graphs on these n nodes can also be written as a sum of the number of directed acyclic graphs on n nodes with r edges, denoted by $D_{n,r}$. Namely, $D_n = \sum_{r=0} D_{n,r}$. Given the sequence $(D_{n,r})_{r=0}^{\infty}$, we define the ordinary generating function:

$$A_n(x) = \sum_{r=0}^{\infty} D_{n,r} x^r.$$

The treatment of generating functions as formal power series allows us to observe that $D_n = A_n(1)$. Note that terms of the sequence $(D_{n,r})_{r=0}^{\infty}$, for which r exceeds the number of possible number of edges in the directed acyclic graph, are set to 0 which confirms our previous observation.

Therefore, to prove that D_n satisfies the recurrence relation in theorem 2, we are required to prove that:

$$A_n(x) = \sum_{k=1}^{n} (-1)^{k+1} \binom{n}{k} (1+x)^{k(n-k)} A_{n-k}(x)$$

since an evaluation for $x = 1$ yields the desired result. To that end, let $n \in \mathbb{N}$ and define the set $N = \{1, \ldots, n\}$. Given an arbitrary subset $\emptyset \neq S \subseteq N$, let $D_{n,r}^{S}$ denote the set of directed acylic graphs on n nodes, with r edges and with all nodes v_j with $j \in S$ having no edges coming into them, i.e. the number of incoming edges into the node v_j is 0.

If we have another subset $\widetilde{S} \neq \emptyset$ such that $\widetilde{S} \subseteq S$, then for an arbitrary graph in $D_{n,r}^{\widetilde{S}}$, the same graph is also in $D_{n,r}^{S}$, as the graph has n nodes, r edges and the nodes with 0 incoming edges belong to the edges with 0 incoming edges in $D_{n,r}^{S}$. This gives the inclusion $D_{n,r}^{\widetilde{S}} \subseteq D_{n,r}^{S}$. In fact, this inclusion allows us to use the *inclusion-exclusion principle*, which we assume to be known and do not explicitly state for conciseness. For more details on the inclusion-exclusion principle, we refer the reader to [7].

Namely, we observe that the set of directed acylic graphs with n nodes and r edges can be written as the union of subsets of directed acylic graphs with n nodes, r edges and a subset of nodes with 0 incoming edges. Using the inclusion-exclusion principle, this yields:

$$D_{n,r} = \sum_{\emptyset \neq S \subseteq N} (-1)^{|S|+1} |D^S_{n,r}| \tag{3.2}$$

$$= \sum_{|S|=1, S\subseteq N}^{n} (-1)^{|S|+1} \binom{n}{|S|} |D^S_{n,r}| \tag{3.3}$$

where we recall that for the set N of n elements, the binomial coefficient $\binom{n}{|S|}$ is the number of $|S|$-element subsets of N, with $|\cdot|$ indicating the cardinality of a set. For simplicity, let us denote $|S| = k \in \mathbb{N}$ for a subset $S \subseteq N$. We make two final claims for $S \subseteq N, \widetilde{S} \subseteq N, |\widetilde{S}| = |S| = k$:

$$|D^S_{n,r}| = |D^{\widetilde{S}}_{n,r}| \tag{3.4}$$

$$|D^S_{n,r}| = \sum_{m=0}^{r} \binom{k(n-k)}{r-m} D_{n-k,m}. \tag{3.5}$$

The first claim follows by the fact that both subsets S and $\widetilde{S}$ have the same cardinality, so in that sense, they just provide a different labelling for the same number of nodes with 0 incoming edges, which indeed shows that $|D^S_{n,r}| = |D^{\widetilde{S}}_{n,r}|$, given that both S and $\widetilde{S}$ are subsets with the same cardinality.

Regarding the second claim, for any graph $G \in D^S_{n,r}$, we can divide the edges in G in two different sets. Namely, one is the set of edges which are incident to nodes v_i and v_j such that $i, j \notin S$, i.e. the nodes v_i and v_j have incoming edges. The second is the set of edges incident to v_i and v_j with $i \in S$ and $j \notin S$, which also indicates that the edge is directed from v_i to v_j as v_i has 0 incoming edges. Consequently, if there are m edges in the first set, then there are $r - m$ edges in the second set. Consequently, for each graph $G \in D^S_{n,r}$, if we have m edges in the first set, then these m edges belong to a subgraph of G with $n-k$ nodes, since these edges are incident to the nodes v_i with $i \in N \setminus S$ with $|N \setminus S| = n - k$ as $|S| = k$. If we consider all possible subgraphs of G with $n - k$ nodes and m edges, and the fact that there are $r - m$ edges in the other set, on subgraphs of m nodes, then given the definition of the binomial coefficient, we obtain the desired identity in (3.5).
Plugging in (3.5) in (3.2) yields:

$$D_{n,r} = \sum_{k=1}^{n} (-1)^{k+1} \binom{n}{k} \sum_{m=0}^{r} \binom{k(n-k)}{r-m} D_{n-k,m} \tag{3.6}$$

and plugging in the obtained expression for $D_{n,r}$ from (3.6) in the ordinary generating function $A_n(x)$ yields:

$$
\begin{aligned}
A_n(x) &= \sum_{r=0}^{\infty}\sum_{k=1}^{n}(-1)^{k+1}\binom{n}{k}\sum_{m=0}^{r}\binom{k(n-k)}{r-m}D_{n-k,m}x^r \\
&= \sum_{k=1}^{n}(-1)^{k+1}\binom{n}{k}\sum_{r=0}^{\infty}\sum_{m=0}^{r}\binom{k(n-k)}{r-m}D_{n-k,m}x^r \\
&= \sum_{k=1}^{n}(-1)^{k+1}\binom{n}{k}\sum_{m=0}^{\infty}\left(\sum_{r=m}^{\infty}\binom{k(n-k)}{r-m}x^{r-m}\right)D_{n-k,m}x^m \\
&= \sum_{k=1}^{n}(-1)^{k+1}\binom{n}{k}\sum_{m=0}^{\infty}(1+x)^{k(n-k)}D_{n-k,m}x^m \\
&= \sum_{k=1}^{n}(-1)^{k+1}\binom{n}{k}(1+x)^{k(n-k)}\sum_{m=0}^{\infty}D_{n-k,m}x^m \\
&= \sum_{k=1}^{n}(-1)^{k+1}\binom{n}{k}(1+x)^{k(n-k)}A_{n-k}(x)
\end{aligned}
\tag{3.7}
$$

which proves the required claim. We remark that in the fourth equality of (3.7) we use the binomial expansion of $(1+x)^{k(n-k)}$ and in the last equality we use the definition of the generating function $A_n(x)$. This completes the proof. $\square$

The recurrence relation in (3.1) hints upon the difficulty of structure learning. To that end, one might be naturally inclined to devise ways of simplifying the task, which is in fact the main purpose of doing research in structure learning. As such, we pinpoint two main aspects of simplifying this task. First, in section 3.2 we explore structural similarities between different DAGs and introduce an equivalence relation on the set of DAGs. In section 3.3 we provide an overview of different structure learning approaches, and in particular, we provide a comprehensive exposition of the so-called "*score-based*" approaches, in particular the BDe score for discrete networks, where we build upon the introduced DAG equivalence classes to simplify the task of structure learning.

3.2 Equivalence Classes of DAGs

We recall that an *equivalence relation* on a set S is a binary relation which satisfies the reflexive, symmetric and transitive property of a binary relation [21]. We typically use the "$\sim$" symbol to denote that two elements of S are equivalent with respect to an equivalence relation. Moreover, for a given element $s \in S$ we define the *equivalence class* of s as the set $\{x \in S |\, x \sim s\}$ and denote it by $[s]$. A simple, yet important, observation is the fact that for a given equivalence relation on a set S, every element in S belongs to one and only one equivalence class.

When it comes to Bayesian network structure learning, our goal is to define an equivalence relation on the set of DAGs, and develop algorithms for structure learning which require us to only look at a representative of an equivalence class, and thus reduce the number of graphs we need to study in order to find the underlying Bayesian network given a dataset [21]. we provide a motivating example for how to define an equivalence relation.

Example 2. *Consider a probability space* $(\Omega, \mathcal{A}, \mathbb{P})$*, the following 4 DAGs consisting of 3 nodes* X_1, X_2, X_3 *and the factorization they impose on the joint distribution* $\mathbb{P}(X_1, X_2, X_3)$*. We assume that* $\mathbb{P}(X_1) > 0$*,* $\mathbb{P}(X_2) > 0$ *and* $\mathbb{P}(X_3) > 0$ *for simplicity.*

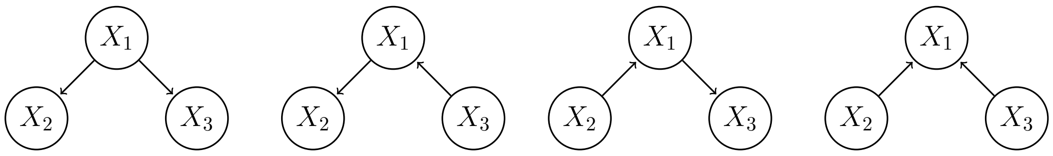

Figure 3.1: 4 DAGs with 3 nodes.

The leftmost DAG imposes the factorization:

$$\mathbb{P}(X_1, X_2, X_3) = \mathbb{P}(X_1)\mathbb{P}(X_2| X_1)\mathbb{P}(X_3| X_1) = \frac{\mathbb{P}(X_1, X_2)\mathbb{P}(X_1, X_3)}{\mathbb{P}(X_1)}. \tag{3.8}$$

The second DAG imposes the factorization:

$$\mathbb{P}(X_1, X_2, X_3) = \mathbb{P}(X_3)\mathbb{P}(X_2| X_1)\mathbb{P}(X_1| X_3) = \frac{\mathbb{P}(X_1, X_2)\mathbb{P}(X_1, X_3)}{\mathbb{P}(X_1)}. \tag{3.9}$$

The third DAG as one might already suspect yields the same factorization as the first two:

$$\mathbb{P}(X_1, X_2, X_3) = \mathbb{P}(X_2)\mathbb{P}(X_1| X_2)\mathbb{P}(X_3| X_1) = \frac{\mathbb{P}(X_1, X_2)\mathbb{P}(X_1, X_3)}{\mathbb{P}(X_1)}. \tag{3.10}$$

However, the rightmost DAG produces a different factorization:

$$\mathbb{P}(X_1, X_2, X_3) = \mathbb{P}(X_2)\mathbb{P}(X_3)\mathbb{P}(X_1| X_2, X_3), \tag{3.11}$$

which in no way simplifies to the factorization obtained in (3.8), (3.9) *and* (3.10).

Intuitively, we observe that "having the same joint distribution factorization" is indeed a relation which is reflexive, symmetric and transitive. To that end, we formally introduce an equivalence relation on the set of DAGs on $n \in \mathbb{N}$ nodes.

Definition 20 (Equivalence Relation on DAGs).
Given a probability space $(\Omega, \mathcal{A}, \mathbb{P})$, we say that two DAGs G_1 and G_2 on $n \in \mathbb{N}$ nodes are equivalent if:

$$\mathbb{P}(X_1, \dots, X_n) = \prod_{i=1}^{n} \mathbb{P}(X_i | Pa_{G_1}(X_i)) = \prod_{i=1}^{n} \mathbb{P}(X_i | Pa_{G_2}(X_i)) \tag{3.12}$$

where in (3.12), $Pa_{G_1}(X_i)$ denotes the parent set of X_i in the DAG G_1 and $Pa_{G_2}(X_i)$ denotes the parent set of X_i in the DAG G_2.

To denote that G_1 and G_2 are equivalent, we write $G_1 \sim G_2$. In the literature, this equivalence is also known as a *Markov equivalence* [21]. As discussed before, this defines a valid equivalence relation on the set of DAGs on n nodes.

Considering example 2 again, if one focuses on the graph structure of each DAG, we make another interesting observation. First, all four DAGs have the same skeleton $X_2 - X_1 - X_3$. However, besides the first three DAGs imposing a different joint distribution factorization compared to the last DAG, we also notice that the last DAG has the v-structure $X_2 \to X_1 \leftarrow X_3$, whereas the other three do not have this particular v-structure, and in general, they contain no v-structures. We state the following key theorem, first proved by Verma and Pearl [32].

Theorem 3. *Two DAGs G_1 and G_2 on $n \in \mathbb{N}$ nodes, represented by random variables $X_1, \dots, X_n$, are Markov equivalent if and only if they have the same skeleton and share the same v-structures.*

In their original paper, Verma and Pearl provide a constructive approach to the proof of this theorem, and thus, we refer the reader to [32] for the details of the proof. Despite the author's personal preference for constructive proofs, we opt not to provide the full details of this constructive approach as it is beyond the scope of this paper. Nevertheless, to provide a clear understanding of why this theorem holds, we offer a slightly informal argument by contradiction.

Proof of Theorem 3.
($\Rightarrow$) First, suppose that two equivalent DAGs G_1 and G_2 do not have the same skeleton. In simplest terms, without loss of generality, assume that a particular edge $X_i - X_j$ is present in the skeleton of G_1 and not a part of G_2. This in turn implies either $Pa_{G_1}(X_i) \neq Pa_{G_2}(X_i)$ or $Pa_{G_1}(X_j) \neq Pa_{G_2}(X_j)$. Matching all the other common factors in (3.12) shows that G_1 and G_2 do not factorize $\mathbb{P}(X_1, \dots, X_n)$ the same way, which is a contradiction. Therefore, two equivalent DAGs G_1 and G_2 must have the same skeleton. Second, suppose that two equivalent DAGs G_1 and G_2 do not share the same v-structures. Without loss of generality, assume that $X_i \to X_j \leftarrow X_k$ is a v-structure in G_1 and not a v-structure in G_2. Therefore, we have one of the three following connections in G_2: $X_i \leftarrow X_j \leftarrow X_k$, $X_i \to X_j \to X_k$ or $X_i \leftarrow X_j \to X_k$.

In either case, if we consider X_j not to be observed, meaning we do not condition on X_j, by our initial discussion of the "Alarm" example in section 2.3, we observe that X_i and X_k are dependent in G_2 regardless of which other nodes we condition on. In G_1 on the other hand, X_i and X_k are either dependent or independent given the structure of G_1. If we condition on a set of nodes M which includes $MB(X_i) \setminus \{X_j\}$ and $MB(X_k) \setminus \{X_j\}$, then it follows that $X_i \perp\!\!\!\perp X_k | \, M$ in G_1, but $X_i \perp\!\!\!\perp X_k | \, M$ does not hold in G_2, which gives a contradiction. Therefore, this proves the first direction of theorem 3.

($\Leftarrow$) Aiming for contradiction once more, suppose that G_1 and G_2 are two DAGs on n nodes that share the same skeleton and the same v-structures, but are not Markov equivalent. This implies that there exist nodes X_i, X_j and X_k such that $X_i \perp\!\!\!\perp X_j | \, X_k$ in G_1 and not in G_2. One can show that this in turn implies that X_i and X_j are not d-separated given X_k in G_2 [5], thus there exists at least one path between X_i and X_j in G_2 which is not block conditional given X_k. Consider the shortest such path and denote it by p. Given that both graphs share the same skeleton, this implies that the only difference in terms of p is in the orientation of the edges in each graph. To that end, we can try to re-orient each edge that is part of p in G_2 to obtain the same orientations that the edges of p have in G_1. Such re-orientations will not make the path block conditional on X_k unless one of them results in the removal of a v-structure. However, if a v-structure is removed, then this would imply that G_1 and G_2 no longer share the same v-structures. This leads to a contradiction. On the other hand, if we re-orient the edges of p such that they have the same orientations as the edges in G_1 without making p block conditional given X_k in G_2, this implies that p is also not block conditional given X_k in G_1 which leads to a contradiction, as we assumed that $X_i \perp\!\!\!\perp X_j | \, X_k$ in G_1. Therefore, in both cases we get a contradiction, and thus the second direction of theorem 3 is proved.

This also concludes the proof of theorem 3 in its entirety.

$\square$

In order to reach our mentioned goal related to structure learning, we would like to utilize theorem 3 in order to determine the equivalence class of a DAG G. To do this, we provide a graphical representation of the equivalence class of a DAG G. First, we state some needed terminology and concepts.

Definition 21 (Compelled and Reversible Edges)**.**
A directed edge is called compelled if all $G' \in [G]$, for some DAG G, contain the directed edge with the same direction as in G. If an edge is not compelled, then it is called reversible.

Definition 22 (Partially Directed Acyclic Graph)**.**
A partially directed acyclic graph (PDAG) $\tilde{\mathcal{G}}$ is a partially directed graph without directed cycles, i.e. it contains both directed and undirected edges, but it does not contain directed cycles, akin to a DAG.

One observes that a DAG is a PDAG with no directed edges. Moreover, we note that PDAGs are typically denoted by cursive letters to distinguish them from DAGs.

Definition 23 (Completed Partially Directed Acyclic Graph).
A completed partially directed acyclic graph (CPDAG), typically denoted by $\mathcal{C}$, for the equivalence class $[G]$ of a DAG G is a PDAG such that the directed edges are precisely the compelled edges of G and the undirected edges replace the reversible edges of G.

Proposition 5. *Two DAGs G_1 and G_2 on $n \in \mathbb{N}$ nodes are equivalent if and only if they have the same CPDAG $\mathcal{C}$.*

Proof. By theorem 3, we know that $G_1 \sim G_2$ if and only if G_1 and G_2 have the same skeleton and share the same v-structures. By the definition of reversible and compelled edges, we observe that G_1 and G_2 have the same skeleton and share the same v-structures if and only if the edges forming the v-structures are compelled, and all the other edges are either reversible or compelled depending on the equivalence class G_1 and G_2 belong to. Following the definition of a CPDAG, we observe that the last claim implies that the two DAGs share the same CPDAG $\mathcal{C}$. Moreover, if the two DAGs share the same CPDAG $\mathcal{C}$, then by definition they share the same skeleton and the same v-structures. Therefore, we obtain one final equivalence, which proves the claim, as desired. □

The result of proposition 5 allows us to represent the equivalence class of a DAG G using its corresponding CPDAG $\mathcal{C}$ [3]. This also establishes a bijection between the equivalence classes $[G]$ of all DAGs on n nodes and the collection of all CPDAGs. Therefore, for a given equivalence class $[G]$, each $G^{'} \in [G]$ corresponds to one and only one CPDAG $\mathcal{C}$.

Given our ability to represent equivalence classes of DAGs using CPDAGs, we proceed to develop an approach for determining a CPDAG from a DAG. Referring back to theorem 3, we see that as an initial step we must have that a CPDAG has the same skeleton and v-structures as the DAG in question. However, in order to precisely determine the compelled and reversible edges for a given equivalence class, we define the so-called *orientation rules* developed by Meek in 1995 [24].

The orientation rules for determining the direction of undirected edges in a CPDAG are the four rules presented graphically in figure 3.2, denoted by R1, R2, R3 and R4, that replace an existing subgraph configuration (on the left) with another subgraph configuration (on the right). Crucially, we make the assumption that the two nodes in the subgraphs on the left that do not have an edge between them, also do not have a parent-child relationship in the DAG.

One has to note that these orientation rules are by no means the only orientation rules one can define for a CPDAG. For instance, a composition of these rules would also be an orientation rule for a CPDAG. However, what is important is that any introduced orientation rule respects the structure of all DAGs in a given equivalence class. To that end, we say that an orientation rule is *valid* if the produced orientation by the rule in question does not introduce a new v-structure nor a directed cycle in the CPDAG.

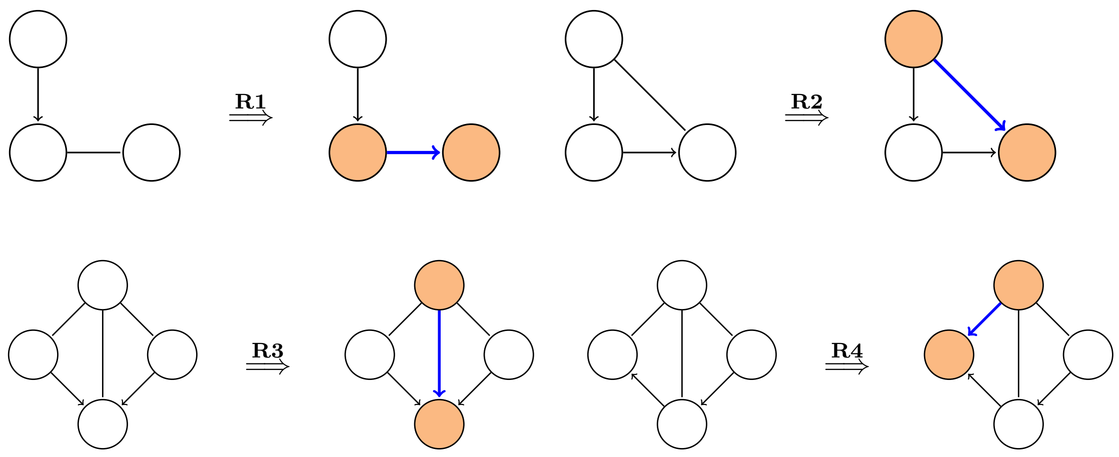

Figure 3.2: Orientation Rules for CPDAGs.

Theorem 4. *The four orientation rules provided in figure 3.2 are valid orientation rules.*

Proof of theorem 4.
We treat the rules on a case-by-case basis:

1. Rule R1: If the blue directed edge in the right subgraph configuration were oriented in the opposite direction there would be a new v-structure in the CPDAG.

2. Rule R2: If the blue directed edge in the right subgraph configuration were oriented in the opposite direction there would be a directed cycle in the CPDAG.

3. If the blue directed edge in the right subgraph configuration were oriented in the opposite direction then by two subsequent applications of rule R2 there would be a new v-structure in the CPDAG (see figure 3.3).

4. If the blue directed edge in the right subgraph configuration were oriented in the opposite direction then by two subsequent applications of rule R2 there would be a new v-structure in the CPDAG (see figure 3.4).

□

Equipped with the results from theorems 3 and theorem 4, we provide a comprehensive algorithmic procedure for determining the CPDAG of a given DAG G. The algorithm runs as follows for a given DAG G:

1. Replace all directed edges in G with undirected edges. This yields the skeleton G_S.

2. Identify the v-structures of G and for all edges participating in a v-structure in G, replace the corresponding undirected edge in G_S by the directed edge from G. We denote this graph by G_0.

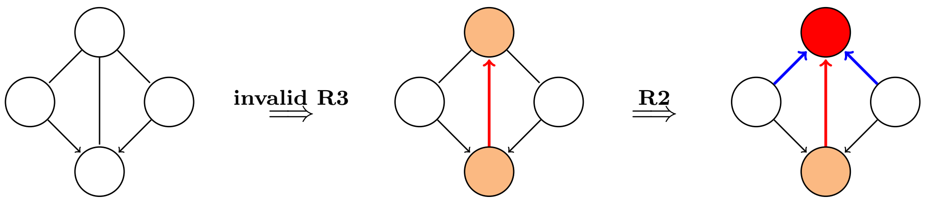

Figure 3.3: Validity of Orientation Rule R3.

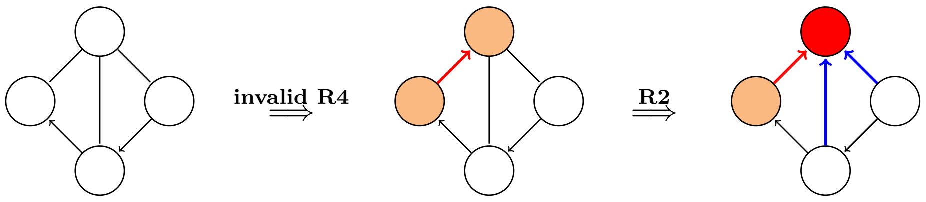

Figure 3.4: Validity of Orientation Rule R4.

3. G_0 is a PDAG. At each step $t \in \mathbb{N}$ identify all subgraph configurations as in figure 3.2 and apply the corresponding orientation rule to obtain a new subgraph configuration.

4. Repeat step 3 until the graph is closed under the orientation rules in figure 3.2, which means that there are no remaining subgraph configurations as in figure 3.2.

It is noteworthy to say that this algorithm is not the most efficient algorithm for identifying the CPDAG for a given DAG. More efficient algorithms can be found in [4, 21]. For our purposes, the efficiency of such an algorithm does not play an important role, as in section 4 the main focus is on exploring Bayesian networks with background information and studying what differences there are between such networks and regular networks in terms of the presented orientation rules and structural properties for CPDAGs.

In section 3.3 we provide an overview of structure learning approaches and in particular, focus on the BDe score for discrete Bayesian networks as a scoring metric that takes into account Markov equivalent DAGs that we introduced to simplify the task of Bayesian network structure learning.

3.3 Structure Learning Approaches

The goal of BN structure learning is to infer DAGs from data. As discussed in the introduction, there exist two main types of structure learning approaches: constraint-based and score-based approaches [20]. Constraint-based approaches utilize conditional independence tests to determine the dependence relations between variables. By constructing an undirected graph based on these tests and then orienting edges to form a directed acyclic graph (DAG), these methods capture the underlying dependencies that are consistent with observed data. On the other hand, score-based approaches evaluate different network structures using a scoring metric that balances data fit and model complexity. A popular way to assign scores to DAGs is using the so-called *Bayesian paradigm*:

$$\begin{aligned}\mathbb{P}(G|\, D) &= \frac{\mathbb{P}(D|\, G)\mathbb{P}(G)}{\mathbb{P}(D)} \\ &\propto \mathbb{P}(D|\, G)\mathbb{P}(G)\end{aligned} \tag{3.13}$$

where D denotes the observed dataset, $\mathbb{P}(D|\, G)$ denotes the so-called *marginal likelihood* of D that is graph-specific, $\mathbb{P}(G)$ denotes the so-called *prior distribution* of a specific DAG G, which represents the initial "belief" about G before observing any data, and $\mathbb{P}(G|\, D)$ denotes the so-called *posterior distribution* of a graph G, which represents our updated "belief" about G after observing the data D [21]. Typically, we assume that each DAG is equally likely, thus we set the prior distribution to be a uniform distribution. Nevertheless, for specific applications, one can also use different priors that fit the given scenario more appropriately [20]. Finally, $\mathbb{P}(D)$ is considered to be a normalization constant, thus leading to the proportionality in (3.13). Prominent scores that achieve score equivalence among Markov equivalent DAGs include the Gaussian BGe score from Geiger and Heckerman [11] and the discrete BDe score from Madigan and York [23].

It is important to note, that there are also *hybrid* approaches for structure learning which combine score-based and constraint-based methods, leveraging the strengths of both [5]. They typically use constraint-based techniques to identify a skeleton and some initial orientations, and then apply score-based methods to refine and optimize the network structure [20]. Moreover, there are also the so-called *Bayesian model averaging* (BMA) approaches which consider multiple possible network structures, weighting them by their posterior probabilities to account for model uncertainty [5, 20]. These approaches integrate over all potential DAGs rather than selecting a single best one, aimed at providing more reliable inferences. One such recently developed BMA approach, building upon the BGe score, for inferring Gaussian (continuous) DAGs from datasets with incomplete data is one of the central aspects of section 5 [13]. Since one of the objectives of this paper is to provide an approach for sampling missing data in discrete networks with incomplete data, with the aim of leading to an adaptation of the mentioned novel BMA approach to discrete networks, the rest of this section provides a detailed exposition of the BDe score, which would be a central component of the said adaptation of the novel BMA approach. The presented terminology, notation and results closely follow [21, 13].

To that end, we return to the expression in (3.13). Given a DAG G with nodes represented as random variables, each random variables is distributed in a certain way, and this distribution is typically characterized by *parameters*. For instance, if a random variable is normally distributed, then the parameters of this distribution are the mean and variance. Therefore, let q denote the vector of all parameters for the random variables in a DAG G, which are also considered to be random variables. Using marginalization and conditional probability, we are able to write the marginal likelihood of D as:

$$\mathbb{P}(D|\,G) = \int \mathbb{P}(D, q|\,G)dq = \int \mathbb{P}(D|\,q, G)\mathbb{P}(q|\,G)dq. \tag{3.14}$$

Our goal is to determine exact expressions for $\mathbb{P}(D|\,q, G)$ and $\mathbb{P}(q|\,G)$, as this will lead to the BDe score. Therefore, let D be an $n \times m$ real dataset matrix, where n indicates the number of variables in a single observation and m indicates the number of observations. In terms of notation, let D_{ij} be the j-th realization of the random variable X_i, q_i be the parameters of X_i, and let $D_{Pa(X_i),j}$ denote the j-th realization of $Pa(X_i)$. Given that G is supposed to represent a Bayesian network, using the local Markov property, we obtain:

$$\begin{aligned}
\mathbb{P}(D|\,q, G) &= \prod_{j=1}^{m} \mathbb{P}(X_1 = D_{1j}, \ldots, X_n = D_{nj}|\,q, G) \\
&= \prod_{j=1}^{m}\prod_{i=1}^{n} \mathbb{P}(X_i = D_{ij}|\,Pa(X_i) = D_{Pa(X_i),j}, q) \\
&= \prod_{j=1}^{m}\prod_{i=1}^{n} \mathbb{P}(X_i = D_{ij}|\,Pa(X_i) = D_{Pa(X_i),j}, q_i) \\
&= \prod_{i=1}^{n}\prod_{j=1}^{m} \mathbb{P}(X_i = D_{ij}|\,Pa(X_i) = D_{Pa(X_i),j}, q_i).
\end{aligned} \tag{3.15}$$

where in the second equality of (3.15) we used theorem 1, and in the third equality we utilized the introduced notation in the preceding paragraph.

When it comes to $\mathbb{P}(q|\,G)$, we make two important assumptions. First, we assume that $\mathbb{P}(q|\,G) = \prod_{i=1}^{n} \mathbb{P}(q_i|\,G)$, which means that the parameters associated with different nodes in G are conditionally independent given G. Second, we assume that $\mathbb{P}(q_i|\,G) = \mathbb{P}(q_i|\,Pa(X_i))$, which means that the distribution of q_i solely depends on the parent nodes of X_i. One typically refers to these assumptions as the *parameter independence assumption*, and the *parameter modularity assumption*, respectively. Combining these two assumptions leads to:

$$\mathbb{P}(q|\,G) = \prod_{i=1}^{n} \mathbb{P}(q_i|\,Pa(X_i)). \tag{3.16}$$

Finally, plugging in the expressions from (3.15) and (3.16) in (3.14) yields the explicit expression:

$$\begin{aligned}
\mathbb{P}(D|\,G) &= \int \mathbb{P}(D|\,q, G)\mathbb{P}(q|\,G)dq \\
&= \int \Bigg(\prod_{i=1}^{n}\prod_{j=1}^{m} \mathbb{P}(X_i = D_{ij}|\,Pa(X_i) = D_{Pa(X_i),j}, q_i) \prod_{i=1}^{n} \mathbb{P}(q_i|\,Pa(X_i))\Bigg) dq \\
&= \int \dots \int \prod_{i=1}^{n} \Bigg(\mathbb{P}(q_i|\,Pa(X_i)) \prod_{j=1}^{m} \mathbb{P}(X_i = D_{ij}|\,Pa(X_i) = D_{Pa(X_i),j}, q_i)\Bigg) dq_1 \dots dq_n \\
&= \prod_{i=1}^{n} \int \Bigg(\mathbb{P}(q_i|\,Pa(X_i)) \prod_{j=1}^{m} \mathbb{P}(X_i = D_{ij}|\,Pa(X_i) = D_{Pa(X_i),j}, q_i)\Bigg) dq_i \qquad (3.17)
\end{aligned}$$

where in the last equality of (3.17) we used the fact that each factor in the integrand product is a function of q_i only. The expression in (3.17) plays a key role in the derivation of the BDe score, as we can use it in the proportionality in (3.13), from which the posterior distribution $\mathbb{P}(G|\,D)$ serves as an indicator of how well a given DAG describes the dataset D.

Since the BDe score is used as a metric for discrete Bayesian networks, we proceed to describe a typical model representing a discrete network. Given our assumptions, we will obtain an explicit expression for (3.17), which in the end will be our BDe score for the given DAG G.

To that end, let G be a DAG with n nodes given as discrete random variables $X_1, \dots, X_n$. We assume that X_i can take $r_i \in \mathbb{N}$ values and the total number of different realizations of $Pa(X_i)$ can take $s_i \in \mathbb{N}$ values. Furthermore, we assume that there are a total of m independent observations of G, out of which m_{ij} have the j-th realization of $Pa(X_i)$. Given these assumptions, we say that X_i has a multinomial distribution with m_{ij} *trial* parameters and probability parameters, denoted by θ_{ijk}, with $i \in \{1, \dots, n\}$, $j \in \{1, \dots, s_i\}$ and $k \in \{1, \dots, r_i\}$, such that $\sum_{k=1}^{r_i} \theta_{ijk} = 1$ for all i and all j. Finally, if through abuse of notation, m_{ijk} denotes the number of realizations in which $Pa(X_i)$ obtains the j-th realization (from s_i) and X_i obtains the k-th realization (from r_i), then:

$$\mathbb{P}(m_{ij1}, \dots, m_{ijk}|\,m_{ij}, \theta_{ij1}, \dots, \theta_{ijr_i}) = \frac{m_{ij}!}{\prod_{k=1}^{r_i} m_{ijk}!} \prod_{k=1}^{r_i} (\theta_{ijk})^{m_{ijk}}. \qquad (3.18)$$

To fully specify the model, we also need to set how the parameters θ_{ijk} are distributed. In particular, we need to explain how the parameter vectors $(\theta_{ij1}, \dots, \theta_{ijr_i})$ are distributed. In the formulation of the BDe score, we assume that $(\theta_{ij1}, \dots, \theta_{ijr_i})$ are *Dirichlet* distributed with parameters $(\alpha_{ij1}, \dots, \alpha_{ijr_i})$. In total, given our assumption regarding X_i from the preceding paragraph, we have $\prod_{i=1}^{n} r_i s_i$ such parameters α_{ijk}. Without delving into the specifics of a Dirichlet distribution, we state that this assumption yields the following distribution of $(\theta_{ij1}, \dots, \theta_{ijr_i})$:

$$\mathbb{P}(\theta_{ij1}, \dots, \theta_{ijr_i}) = \frac{\Gamma(\sum_{k=1}^{r_i} \alpha_{ijk})}{\prod_{k=1}^{r_i} \Gamma(\alpha_{ijk})} \prod_{k=1}^{r_i} \theta_{ijk}^{\alpha_{ijk}-1}. \tag{3.19}$$

For the sake of conciseness, by omitting the intermediate derivations and technical details, and plugging in the expressions from (3.18) and (3.19) into (3.17), we state that for $m_{ij} = \sum_{\tilde{k}=1}^{r_i} m_{ij\tilde{k}}$ and $\alpha_{ij} = \sum_{\tilde{k}=1}^{r_i} \alpha_{ij\tilde{k}}$, we obtain the BDe score for G:

$$\mathbb{P}(D|\, G) = \prod_{i=1}^{n} \prod_{j=1}^{s_i} \left(\frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + m_{ij})} \right) \prod_{k=1}^{r_i} \left(\frac{\Gamma(\alpha_{ijk} + m_{ijk})}{\Gamma(\alpha_{ijk})} \right). \tag{3.20}$$

where to ensure score equivalence among Markov equivalent networks, we also require that $\alpha_{ijk} = \frac{\alpha}{s_i \cdot r_i}$, where α is some to-be-determined constant, commonly referred to as *total precision*. We refer the reader to [23] for a detailed derivation of the intermediate steps leading to (3.20).

As a concluding remark to section 3, we note that the discussion in section 3.3 serves a relevant role in section 5, whereas section 4 refers back to the results discussed in section 3.2.

4 Bayesian Networks with Background Information

In many natural and social processes, one might encounter the situation in which two or more interacting variables have a causal relationship that is well understood, yet we are still interested in what kind of relationships and dependencies they might have with other variables involved in the process.

For instance, consider the example of gene expression in biology. Gene expression, in particular, the collection of genes, transcription factors and the regulatory interactions among them, responsible for controlling the gene expression levels of mRNA and proteins, are a key application area of Bayesian networks [12]. The study of gene regulatory networks with the use of Bayesian networks is instrumental in drug development. For our purposes, we are interested in the causal relationship between two particular transcription factors, the p53 transcription factor and the p21 protein [9]. It is a scientifically-confirmed fact that in a regular human cell, the p53 protein binds to DNA, which activates another gene to produce a protein called p21. This protein interacts with cdk2, a cell division-stimulating protein. When p21 forms a complex with cdk2, the cell is unable to proceed to the next stage of division. However, the presented causal relationship between the proteins p53 and p21 is not the only one of relevant interest. For instance, in the study of tumor repressors, the p53-p21-RB signaling pathway (see figure 4.1) with the retinoblastoma protein RB is also of great interest [9]. This signals that in the case of structure learning for gene regulatory networks, one would like to incorporate the known causal relationship p53-p21, and then be able to study further any other conditional dependencies.

While the presented example is a very specific, real-world scenario, it opens the way for a purely formal and mathematical study of scenarios when one possesses background information about specific dependence relationships in the network of interest. This section develops a theoretical understanding of such *Bayesian networks with background information*, ensures consistency with the developed theory on regular Bayesian networks and discerns the relevant differences with regular Bayesian networks. It is important to remark, that besides the common, real-world occurrence of social and natural processes with background information, having such networks also simplifies the task of structure learning, since knowing surely about the existence of a conditional dependency between two random variables already reduces the number of possible DAGs one has to study. This in turn reduces computational efforts and potentially reduces computational time of well-established approaches for Bayesian network structure learning.

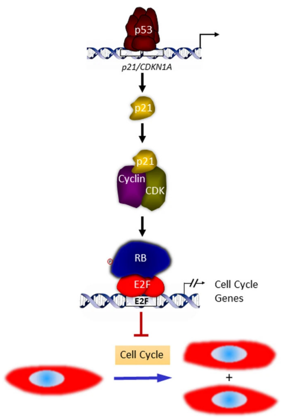

Figure 4.1: The p53-p21-RB signaling pathway.

4.1 Maximally Oriented PDAGs

The described Bayesian networks with background information have already appeared in literature, albeit under various names. For instance, they are referred to as "maximally oriented graphs" in [24], as "interventional essential graphs" in [16] or "aggregated PDAGs (APDAGs)" in [8]. A recent trend in literature shows that the name "maximally oriented PDAG" is being used on a wider scale [19, 29]. To that end, for consistency and currency purposes, we adopt the name "maximally oriented PDAG" and refer to such networks simply as MPDAGs. In [24], the author explores background information about edge orientations. This complements the idea we convey in our example from gene regulatory networks. For completeness, in [16], the authors explore background information in the sense of observational data, and in [8], the authors explore background knowledge by imposing parameter restrictions on the models under elucidation. Additionally, one also encounters background information in the sense of interventional data [22], which typically occurs when we regulate one of the variables in some realizations of the variables represented in a DAG. In such cases, we typically assume that different realizations are still independent, but the probability distribution of the random vectors with regulated variables are not the same as those for the realizations without regulated variables. The prescribed impact interventional data has on DAGs and CPDAGs is typically handled with the introduction of so-called *dummy* variables to the regulated nodes to form v-structures and then apply an algorithm that creates a CPDAG from a DAG, such as the one we introduce in section 3.2.

Since our understanding of background information is closest to the one in [24], our discussion of fundamental definitions related to MPDAGs follows the treatment of Bayesian networks with background information in [24]. Namely, we study MPDAGs that arise as a result of adding background information to an existing CPDAG, by orienting an undirected edge in the CPDAG. For alternative cases, such as MPDAGs arising as a result of adding background information before structure learning, setting parameters constraints or introducing interventional data, we refer the reader to [16, 8, 29].

On a different note, one has to recognize, that the terminology, notation and formulation of results used in [24] are not an integral part of modern-day research literature on Bayesian networks. To that end, in our treatment of MPDAGs we incorporate current scientific standards, notation and terminology with already established theory. Despite being relatively minor in significance, this contributes to the current body of knowledge by providing the reader with a formal treatment of MPDAGs consistent with current standards, thus preventing the need for additional effort spent on reconciling mismatching terminology, notation and statement of results.

To motivate the study of networks with background information we consider an introductory example.

Example 3. *Suppose we have* 4 *random variables denoted by* X_1, X_2, X_3 *and* X_4 *for which we have collected some observations. Through the use of a structure learning algorithm (e.g. PC algorithm) we are able to estimate the CPDAG* $\mathcal{C}$ *for the network, as given in figure 4.2. Consequently, by applying the orientation rules provided in 3, we identify all 10 DAGs represented by* $\mathcal{C}$ *(right-hand side of figure 4.2). A typical scenario would be to proceed with a certain score-based approach and check all the DAGs to determine the best-fitting one. One has to admit that in this case, that is not a computationally heavy task, so it can be done with ease.*

However, if for example we had a CPDAG on many more nodes, representing many more DAGs, then naturally this becomes a much more demanding problem. Now suppose either through previous experimentation or expert knowledge, we are certain that there is a conditional dependence relation between X_1 *and* X_3 *given by the directed edge* $X_1 \to X_3$ *as seen on the left-hand side of figure 4.3. By just adding this one edge, the number of DAGs represented by this new PDAG* $\mathcal{M}$ *reduces to* 5*. Again, this is an oversimplified example, so the benefit might not seem extraordinary, but for CPDAGs that represent more than a million DAGs for instance, reducing the problem in this way could be quite beneficial in terms of computational effort.*

We remark that one must remain cautious when taking a deeper look at the presented example. Namely, the reader might be inclined to mistakenly attribute the added edge as nothing else but an "added edge", and thus try to use established results and approaches for CPDAGs. However, in this case, the PDAG $\mathcal{M}$ is not even a CPDAG. To simply see this, in case $\mathcal{M}$ were a CPDAG, notice that the DAGs represented by $\mathcal{M}$, given on the right-hand side of figure 4.3 would belong to two distinct equivalence classes, which is impossible. To that end, we devote the remainder of this section to provide a sound definition of a PDAG with background information, and in sections 4.2 and 4.3 we elaborate on the difference with respect to regular CPDAGs and develop elementary approaches for establishing dependence relations in PDAGs with background information.

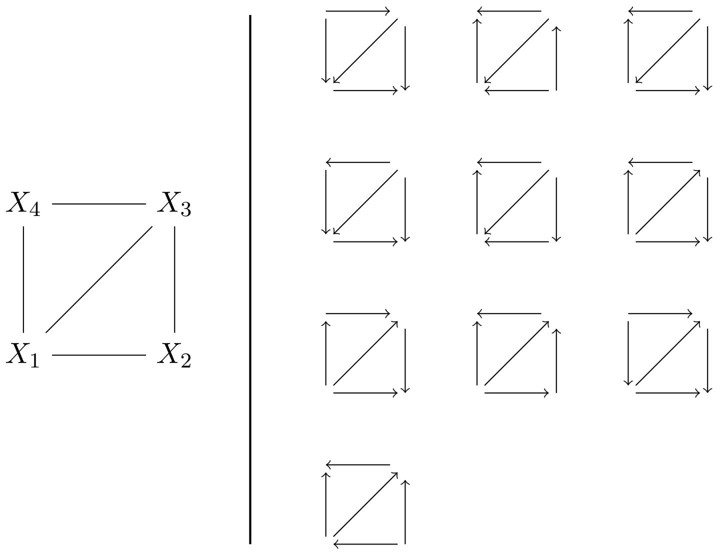

Figure 4.2: CPDAG $\mathcal{C}$ (left) and DAGs represented by $\mathcal{C}$ (right).

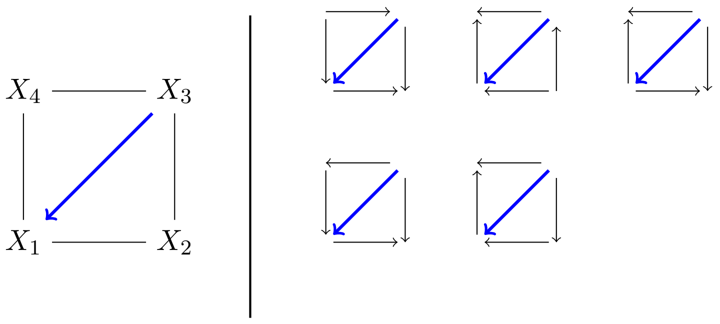

Figure 4.3: PDAG $\mathcal{M}$ (left) and DAGs represent by $\mathcal{M}$ (right).

Definition 24 (Extension of a DAG and a PDAG).
A DAG G extends a DAG H if G and H share the same skeleton and moreover, for every directed edge $X \to Y$ in $E(H)$, we have that $X \to Y$ is in $E(G)$. Applying the same rule to a PDAG defines an extension for a PDAG. Moreover, we say that G is a consistent extension of H, if it extends H and they belong to the same equivalence class.

In order to formally define a MPDAG, we concretize the notion of background information. Since we adopt the convention of using the term "background information" to refer to known edge orientations, we define *background information* as a set of oriented edges consistent with the v-structures in the CPDAG of G, and we denote it by $\mathcal{K}$. By consistent, we mean that in case we introduce an edge from $\mathcal{K}$ into the corresponding CPDAG, then that must not produce a new v-structure.

Definition 25 (Maximally Oriented PDAG).
A maximally oriented PDAG (MPDAG), denoted by $\mathcal{M}$, is a PDAG such that for each unoriented edge $X - Y$ in $\mathcal{M}$, there exist two distinct DAGs G_1 and G_2 which extend $\mathcal{M}$, belong to the same CPDAG $\mathcal{C}$, and such that $X \to Y \in E(G_1)$, $X \leftarrow Y \in E(G_2)$.

A straightforward observation is that every CPDAG is a MPDAG, but not vice versa. To see that the inclusion is strict, consider the following example.

Example 4. *Consider the following PDAG $\mathcal{M}$ consisting of 3 nodes and one directed edge $X_3 \leftarrow X_2$.*

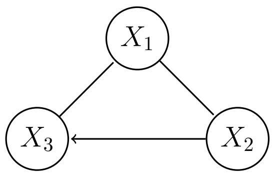

Figure 4.4: A MPDAG which is not a CPDAG.

By the definition of a MPDAG, we observe that the PDAG $\mathcal{M}$ in the example is indeed an MPDAG. For the undirected edge $X_1 - X_2$ consider the two DAGs in figure 4.5. These two DAGs precisely satisfy the requirements in the definition of a MPDAG. For the undirected edge $X_1 - X_2$, $X_1 \to X_2$ is in the DAG on the right and $X_1 \leftarrow X_2$ is in the DAG on the left. Both DAGs extend $\mathcal{M}$ and belong to the CPDAG given in figure 4.6. A similar observation follows for the undirected edge $X_1 - X_3$ in $\mathcal{M}$. However, we notice that the PDAG $\mathcal{M}$ is not a CPDAG as a CPDAG cannot contain partially oriented cycles. A formal treatment of this observation is covered in section 4.2. More simply, if $\mathcal{M}$ were a CPDAG, then the DAGs in figure 4.5 would belong to it, but we also saw that they belong to the CPDAG $\mathcal{C}$ in figure 4.6, which yields a contradiction.

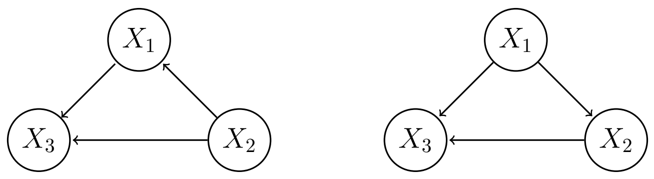

Figure 4.5: $\mathcal{M}$ is a MPDAG.

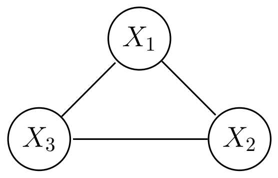

Figure 4.6: The CPDAG for the two DAGs in example 4.

Definition 26 (MPDAG with respect to Background Information $\mathcal{K}$)**.**
A maximally oriented PDAG (MPDAG) with respect to background information $\mathcal{K}$, denoted by $\mathcal{M}$, is a PDAG such that for each unoriented edge $X - Y$ in $\mathcal{M}$, there exist two distinct DAGs G_1 and G_2 which extend $\mathcal{M}$, belong to the same CPDAG $\mathcal{C}$, and such that $X \to Y \in E(G_1)$, $X \leftarrow Y \in E(G_2)$. Moreover, each directed edge from $\mathcal{K}$ needs to be oriented the same way in $\mathcal{M}$.

The provided definitions, while being clear, are relatively cumbersome to work with, as one would have to check the existence of such DAGs G_1 and G_2, which could be quite inefficient for DAGs with many nodes. However, we state the following claim, which allows us to obtain a simple equivalent characterization for a MPDAG.

Proposition 6. *For a given PDAG G, applying the orientation rules presented in section 3.2 to G results in a MPDAG M.*

We omit the proof of proposition 6, as the main argument relies upon the application of theorems 5 and 6 to PDAGs. We state and prove these theorems in section 4.2. The technical details of the proof can also be found in [24]. Most importantly, this statement allows us to characterize MPDAGs as PDAGs which are closed under the orientation rules from section 3.2. Moreover, this characterization also offers a structured way of determining whether a PDAG is a MPDAG, as one could check if any of the 4 subgraph configurations presented in section 3.2 are present in a given PDAG, and check if they are closed under their respective orientation rule. For instance, one could check that the PDAGs in figures 4.3 and 4.4 are closed under the 4 orientation rules introduced in section 3.2, and thus they are MPDAGs.

4.2 Differences between CPDAGs and MPDAGs

A rather naive approach to structure learning for networks with background information would be to use developed theory for CPDAGs and just apply it to MPDAGs. However, using our observation that not every MPDAG is a CPDAG, and by proving structural properties for CPDAGs which are not shared by MPDAGs in general, we examine 3 key differences between a MPDAG and a CPDAG, which in turn require us to develop new theory needed for structure learning in the case of networks with background information.

To that end, we first prove the following result for CPDAGs.

Theorem 5. *Let $\mathcal{C}$ be an arbitrary CPDAG. If for any three nodes X_1, X_2 and X_3 in $\mathcal{C}$, the directed edge $X_1 \rightarrow X_2$ and the undirected edge $X_2 - X_3$ are in $\mathcal{C}$, then the directed edge $X_1 \rightarrow X_3$ is also in $\mathcal{C}$.*

Proof of Theorem 5.
Without loss of generality, according to definition 5, the 4 orientation rules for CPDAGs provide a topological ordering in which every node has a position behind all of its ancestors. Furthermore, this introduces a *partial order* on the set of nodes in a CPDAG. Namely, for any two nodes X and Y in $\mathcal{C}$, $X < Y$ if X is an ancestor of Y. As a small remark, we note that we have a strict partial order since there may be pairs of nodes for which neither element precedes the other, and we cannot pairs of nodes such that they are each other's ancestors due to the acyclicity constraint for DAGs and CPDAGs.

We aim for a proof by contradiction. Thus, let X_2 be a node in $\mathcal{C}$ such that $X_1 \rightarrow X_2$ and the undirected edge $X_2 - X_3$ are in $\mathcal{C}$, but assume that the directed edge $X_1 \rightarrow X_3$ is not in $\mathcal{C}$. Moreover, we choose X_2 to be the *minimal* such node with respect to the introduced partial order on the set of nodes, meaning that if there is another node $\tilde{X_2}$ such that $\tilde{X_1} \rightarrow \tilde{X_2}$ and the undirected edge $\tilde{X_2} - \tilde{X_3}$ are in $\mathcal{C}$, but the directed edge $\tilde{X_1} \rightarrow \tilde{X_3}$ is not in $\mathcal{C}$, then we must have $X_2 < \tilde{X_2}$. Note that the existence of such a minimal node X_2 is guaranteed by the existence of a node that satisfies the assumptions in theorem 5.

First, note that X_1 and X_3 must be adjacent in $\mathcal{C}$, otherwise we would have the first subgraph configuration from figure 3.2, and thus we would orient $X_2 - X_3$ using rule R1. Moreover, we must have the undirected edge $X_1 - X_3$, as having the directed edge $X_1 \rightarrow X_3$ immediately yield a contradiction, whereas $X_1 \leftarrow X_3$ yields the second subgraph configuration from figure 3.2 and thus we would orient $X_2 - X_3$ using rule R2. To summarize, given the assumption we made in the preceding paragraph, we currently have the subgraph configuration as presented in figure 4.7.

We complete the proof by case distinction regarding the orientation of $X_1 \rightarrow X_2$.

1. $X_1 \rightarrow X_2$ is oriented using rule R1.

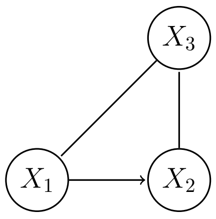

Figure 4.7: Subgraph configuration according to assumption.

This means that there exists a node Y such that there is a directed edge $Y \to X_1$, such that Y is not adjacent to X_2. Notice that Y and X_3 must be adjacent, otherwise, by applying rule R1 to the subgraph configuration $Y \to X_1 - X_3$, we would obtain the directed edge $X_1 \to X_3$, which yields a contradiction. Moreover, we cannot have $Y \to X_3$, as by applying rule R1 to the subgraph configuration $Y \to X_3 - X_2$, we would obtain the directed edge $X_3 \to X_2$, which yields a contradiction once again. However, then X_1 is a node such that $Y \to X_1$ and $X_1 - X_3$ are in $\mathcal{C}$, but the directed edge $Y \to X_3$ is not in $\mathcal{C}$. However, since X_1 is an ancestor of X_2, we have $X_1 < X_2$, which yields a contradiction as X_2 is assumed to be the minimal such node in $\mathcal{C}$ (see figure 4.8).

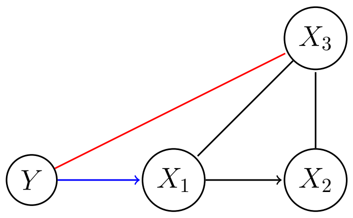

Figure 4.8: Case 1: $X_1 \to X_2$ oriented using rule R1.

2. $X_1 \to X_2$ is oriented using rule R2.

This means there exists a node Y such that there is a directed edge $X_1 \to Y$ and a directed edge $Y \to X_2$. Moreover, Y and X_3 must be adjacent, otherwise we would have the directed edge $X_2 \to X_3$ by orienting $X_2 - X_3$ using rule R1 in the subgraph configuration $Y \to X_2 - X_3$. Regarding the edge $Y - X_3$, if we have the direction $Y \to X_3$, then we must have $X_1 \to X_3$ according to rule R2 applied to the subgraph configuration $X_1 \to Y \to X_3 - X_1$, which yields a contradiction, as $X_1 \to X_3$ is assumed not to be in $\mathcal{C}$. On the other hand, if we have the direction $Y \leftarrow X_3$, then we must have $X_3 \to X_2$ by applying rule R2 to the subgraph configuration $X_3 \to Y \to X_2 - X_3$, which yields a contradiction, as $X_2 - X_3$ is assumed to be undirected (see figure 4.9).

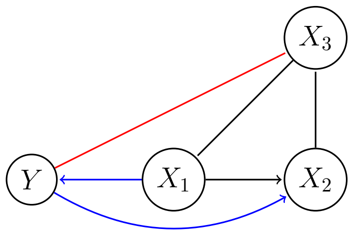

Figure 4.9: Case 2: $X_1 \to X_2$ oriented using rule R2.

3. $X_1 \to X_2$ is oriented using rule R3.

   This means that there exist nodes Y and Z such that we have the undirected edges $X_1 - Y$, $X_1 - Z$ and the directed edges $Y \to X_2$, $Z \to X_2$ in $\mathcal{C}$. Moreover, Y and Z must be adjacent to X_3, as otherwise, by applying rule R1 to the subgraph configurations $Y \to X_2 - X_3$ and $Z \to X_2 - X_3$, we would get the directed edge $X_2 \to X_3$, which yields a contradiction. Furthermore, if we have the unoriented edges $Y - X_3$ and $Z - X_3$, then by applying rule R3 to the subgraph configuration $X_3 - Y \to X_2 \leftarrow Z - X_3 - X_2$ we obtain the directed edge $X_3 \to X_2$, which yields a contradiction, as $X_2 - X_3$ is assumed to be undirected. Therefore, if we have the orientation $Y \to X_3$, then by rule R1 applied to the subgraph configuration $Y \to X_3 - Z$, we must have the directed edge $X_3 \to Z$, which in turn yields the directed edge $X_3 \to X_2$ to preserve acyclicity, which yields a contradiction in turn. On the other hand, if we have the orientation $Y \leftarrow X_3$, to preserve acyclicity we again must have $X_3 \to X_2$, which yields a contradiction. An analogous treatment of the edge $Z - X_3$ shows that regardless of how we orient the edge, we get a contradiction (see figure 4.10).

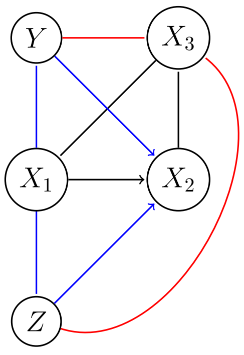

Figure 4.10: Case 3: $X_1 \to X_2$ oriented using rule R3.

4. $X_1 \to X_2$ is oriented using rule R4.

   This means that there exists a node Y such that the undirected edge $Y - X_1$ and the directed edges $Y \to X_3$ and $X_3 \to X_2$ are in $\mathcal{C}$. This yields an immediate contradiction, as $X_2 - X_3$ is assumed to be undirected (see figure 4.11).

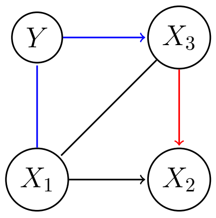

Figure 4.11: Case 4: $X_1 \to X_2$ oriented using rule R4.

5. $X_1 \to X_2$ is oriented as part of a v-structure.

   This means there is a node Y such that X_1, X_2 and Y form a v-structure, meaning that the directed edge $Y \to X_2$ is in $\mathcal{C}$, and X_1 and Y are not adjacent. Notice that, Y and X_3 must be adjacent, as otherwise, by applying rule R1 to the subgraph configuration $Y \to X_2 - X_3$, we would have the directed edge $X_2 \to X_3$, which yields a contradiction. Moreover, the edge between X_3 and Y must be oriented, as otherwise, by applying rule R3 to the subgraph configuration $X_3 - X_1 \to X_2 \leftarrow Y - X_3 - X_2$, we would have the directed edge $X_3 \to X_2$, which again yields a contradiction. The rest of the argument is completely analogous to case 3 (see figure 4.12).

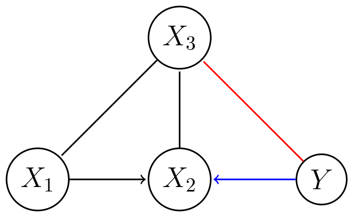

Figure 4.12: Case 5: $X_1 \to X_2$ oriented as part of a v-structure.

As we have exhausted all possible orientation cases for how one could have obtained the directed edge $X_1 \to X_2$ in $\mathcal{C}$ following the given assumptions, and obtained a contradiction in each one of them, we conclude that in a CPDAG $\mathcal{C}$, if there are three nodes X_1, X_2 and X_3 such that the directed edge $X_1 \to X_2$ and the undirected edge $X_2 - X_3$ are in $\mathcal{C}$, then the directed edge $X_1 \to X_3$ is also in $\mathcal{C}$, as desired. □

As a result of this theorem, we observe that a paramount difference between MPDAGs and CPDAGs, is that MPDAGs are allowed to contain partially directed cycles, whereas

CPDAGs cannot. To observe this, consider the CPDAG and MPDAG given in example 3. Indeed, we observe that the MPDAG in figure 4.3 contains the partially directed cycles $X_3 \to X_1 - X_2 - X_3$ and $X_3 \to X_1 - X_4 - X_3$. However, this is not possible in the CPDAG in figure 4.2, as it would contradict the result of theorem 5.

To introduce another interesting and important difference we define the concept of a *triangulated* graph.

Definition 27 (Triangulated Graph)**.**
An undirected graph H is called triangulated if every cycle in H with length greater than or equal to 4 has an edge between two nonconsecutive nodes on the cycle.

Theorem 6. *Consider a CPDAG $\mathcal{C}$. The undirected subgraph $\tilde{\mathcal{C}}$ of $\mathcal{C}$, obtained by removing the directed edges in $\mathcal{C}$, is triangulated.*

Proof of Theorem 6.
We proceed with a proof by contradiction. Assume that $\tilde{\mathcal{C}}$ is not triangulated, meaning there is a cycle of length greater than or equal to 4 in which there are no non-adjacent nodes with an edge between them. Formally, there is a cycle $c = \langle X_1, \dots, X_n \rangle$ in $\tilde{\mathcal{C}}$ with $n \geq 4$ such that for any pair of non-adjacent nodes X_i and X_j, $X_i - X_j$ is not in $\tilde{\mathcal{C}}$.

Since c is an undirected cycle, if one would like to orient the edges, then to preserve acyclicity, there must be a node X_i in c such that $X_{i-1} \to X_i \leftarrow X_{i+1}$. Note that if $i = n$, then $i + 1 = 1$ and if $i = 1$, then $i - 1 = n$. Importantly, this creates a v-structure in $\tilde{\mathcal{C}}$. By theorem 5, we also obtain a v-structure in $\mathcal{C}$, which yields a contradiction, and proves the claim.

To see this, suppose that $X_{i-1} \to X_i \leftarrow X_{i+1}$ is not a v-structure in $\mathcal{C}$, thus we have $X_{i-1} - X_i - X_{i+1}$ and there is an edge between X_{i-1} and X_{i+1}. If this edge is undirected, then $\tilde{\mathcal{C}}$ is triangulated, which is a contradiction. If this edge is directed and we have the edge $X_{i-1} \to X_{i+1}$, then by theorem 5 we would have the directed edge $X_{i-1} \to X_i$ in $\mathcal{C}$, which yields a contradiction. Similarly, if we have the edge $X_{i-1} \leftarrow X_{i+1}$, then by theorem 5 we would have the directed edge $X_{i+1} \to X_i$ in $\mathcal{C}$, which again yields a contradiction. This shows that if we have a v-structure in $\tilde{\mathcal{C}}$, then we have the same v-structure in $\mathcal{C}$, as desired. □

As a result of this theorem, we observe that another key difference between MPDAGs and CPDAGs, is that the undirected subgraph of a CPDAG $\mathcal{C}$ must be triangulated, whereas this is not the case for MPDAGs. To observe this, consider the CPDAG and MPDAG given in example 3. Indeed, we observe that for the MPDAG $\mathcal{M}$, the undirected subgraph $\tilde{\mathcal{M}}$ forms a cycle, which is not triangulated (see figure 4.13).

We observe one additional difference, again related to the undirected subgraph $\tilde{\mathcal{C}}$ of a CPDAG $\mathcal{C}$. This difference between CPDAGs and MPDAGs is the result of applying a theorem in [24] to MPDAGs. For clarity, we state this theorem. Moreover, we call a

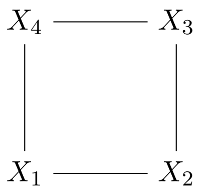

Figure 4.13: The undirected subgraph $\tilde{\mathcal{M}}$ of the MPDAG $\mathcal{M}$ from example 3.

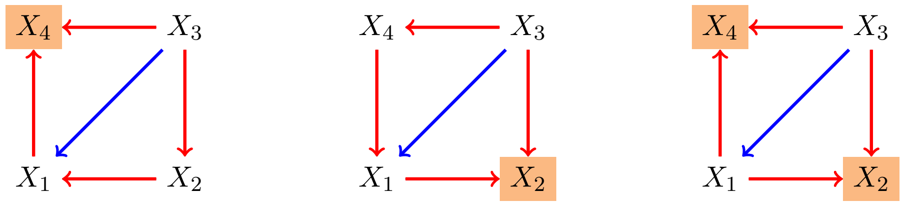

Figure 4.14: No independent orientation $\tilde{\mathcal{M}}$.

subgraph of an undirected graph G *connected* if there is a path between any two nodes in the subgraph [7].

Theorem 7. *Given a CPDAG $\mathcal{C}$ and its underlying undirected subgraph $\tilde{\mathcal{C}}$, the connected subgraphs of $\tilde{\mathcal{C}}$ can be oriented independently into DAGs without any v-structures, to form all DAGs represented by $\mathcal{C}$.*

A detailed proof of this claim can be found in [24]. We omit the proof and its technicalities as our main concern is what this theorem implies for MPDAGs. It is important to note that “oriented independently” in theorem 7 refers to being able to orient the edges in each connected subgraph of $\tilde{\mathcal{C}}$ independently of the other connected subgraphs and independently of the orientations of the directed edges in $\mathcal{C}$.

Therefore, as a result of this theorem, we observe one additional difference between MPDAGs and CPDAGs, best illustrated by considering the CPDAG and MPDAG given in example 3. Namely, from theorem 7 we know that for a CPDAG $\mathcal{C}$, the connected subgraphs of $\tilde{\mathcal{C}}$ can be oriented independently into DAGs without any v-structures, as one can observe in figure 4.2, whereas this is not true for MPDAGs, as one can observe from the undirected subgraph of the MPDAG $\mathcal{M}$ from figures 4.3 and 4.13. Namely, to preserve acyclicity, according to theorem 6, any possible orientation of the edges in $\tilde{\mathcal{M}}$ will result in at least one v-structure in $\tilde{\mathcal{M}}$. To avoid forming a v-structure in $\mathcal{M}$, in every DAG G represented by $\mathcal{M}$, we must have the subgraph configuration $X_1 \rightarrow X_2 \leftarrow X_3$ or the subgraph configuration $X_1 \rightarrow X_4 \leftarrow X_3$, which comes from the directed edge $X_1 \leftarrow X_3$, thus prohibiting independent orientation of edges. This is illustrated in figure 4.14 where the edges of $\tilde{\mathcal{M}}$ are coloured in red to indicate that we must consider $X_1 \leftarrow X_3$ when orienting the edges in a DAG represented by $\mathcal{M}$.

In summary, we have identified three key differences between a MPDAG and a CPDAG. Namely, a MPDAG can have partially directed cycles, whereas a CPDAG cannot (theorem 5). Moreover, the underlying subgraph $\tilde{\mathcal{M}}$ of a MPDAG $\mathcal{M}$ does not have to be triangulated, where for a CPDAG $\mathcal{C}$, $\tilde{\mathcal{C}}$ must be triangulated (theorem 6). Finally, for a CPDAG $\mathcal{C}$, we can independently orient the edges in $\tilde{\mathcal{C}}$ to obtain all the DAGs represented by $\mathcal{C}$ (theorem 7), which is not necessarily true for MPDAGs.

To circumvent these differences and be able to apply established structure learning approaches for CPDAGs and DAGs (such as the PC algorithm), one could think of several simple solutions. First, one could list all the DAGs represented by a MPDAG. Second, one could exclude MPDAGs with partially directed cycles. While these solutions are rather simple to implement, one must recognize that the first solution is extremely inefficient in cases of MPDAGs with a lot of nodes, whereas the second solution is simply too restrictive [29]. To that end, we devote the next section to develop specific theory related to dependence relations in MPDAGs.

4.3 Dependence Relations in MPDAGs

By our observation that every CPDAG is a MPDAG, we are able to modify existing theory on dependence relations in CPDAGs and adapt it to MPDAGs. In particular, we focus on defining what *directed paths, ancestors* and *descendants* are in a MPDAG. It turns out that the definitions of these notions are rather similar to their counterparts for CPDAGs, but more complex [29]. Moreover, their formulation allows us to state and prove useful results regarding dependence relations in MPDAGs.

Definition 28 (Possibly Directed Path).
A path $p = \langle X_1, \dots, X_n \rangle$ ($n \in \mathbb{N}$) in a MPDAG $\mathcal{M}$ is called a possibly directed path in $\mathcal{M}$ if for any pair of nodes X_i and X_j in p for $1 \leq i < j \leq n$, there is no directed edge $X_i \leftarrow X_j$.

Notice that this definition has the same formulation as the definition of a directed path in a DAG G given in section 2.1. However, this definition also considers edges which are not part of the path p and in the case of MPDAGs, this adds an additional layer of complexity due to the presence of both directed and undirected edges. Consider the CPDAG $\mathcal{C}$ from figure 4.2, the MPDAG $\mathcal{M}$ from figure 4.3, and the undirected path $X_1 - X_2 - X_3$. In $\mathcal{C}$, according to definition 28, this is a possibly directed path. However, in $\mathcal{M}$, $X_1 - X_2 - X_3$ is not a possibly directed path as $X_1 \leftarrow X_3$ is a directed edge in $\mathcal{M}$. We observe a similar situation when it comes to defining ancestors and descendants in MPDAGs (see figure 4.15).

Definition 29 (Possible Ancestors and Descendants).
For a given node Y in a MPDAG $\mathcal{M}$, a node X is called a possible ancestor of Y in $\mathcal{M}$ if there is a possibly directed path from X to Y. Similarly, a node Z is called a possible descendant of Y in $\mathcal{M}$ if there is a possibly directed path from Y to Z.

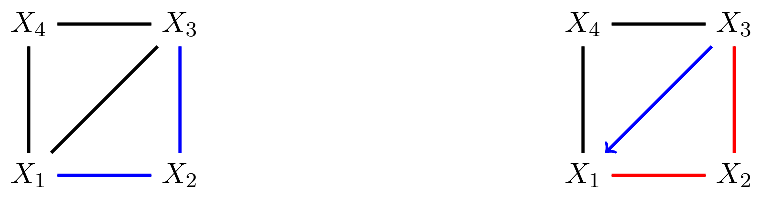

Figure 4.15: A possibly directed path from X_1 to X_3 in the CPDAG C (left) and not a possibly directed path from X_1 to X_3 in the CPDAG M (right).

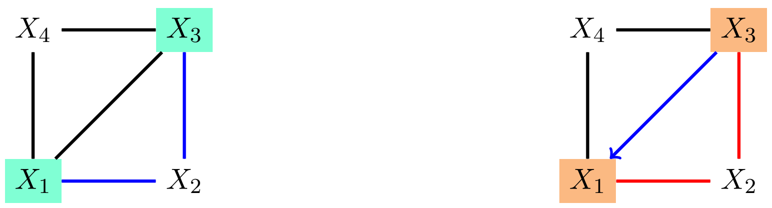

Figure 4.16: X_3 is a possible descendant of X_1 in the CPDAG C (left) and not a possible descendant of X_1 in the MPDAG M (right).

We again observe that this definition is similar to the definition of a descendant and an ancestor in a DAG G given in section 2.1. However, considering the CPDAG $\mathcal{C}$ from figure 4.2, and the MPDAG $\mathcal{M}$ from figure 4.3, we observe that X_3 is a possible descendant of X_1 in $\mathcal{C}$ since $X_1 - X_2 - X_3$ is a possibly directed path from X_1 to X_3 in $\mathcal{C}$. Analogously, X_1 is a possible ancestor of X_3 in $\mathcal{C}$. However, when it comes to the MPDAG $\mathcal{M}$, X_3 is not a possible descendant of X_1 in $\mathcal{M}$ since the only paths from X_1 to X_3 are $X_1 - X_2 - X_3$ and $X_1 - X_4 - X_3$, and these paths are not possibly directed paths from X_1 to X_3 in $\mathcal{M}$. Similarly, X_1 is not a possible ancestor of X_3 in $\mathcal{M}$ (see figure 4.16).

Following the definition of a possibly directed path, we present the first interesting result regarding dependence relations in MPDAGs. In particular we show how possibly directed paths in a MPDAG affect directed paths in the DAGs represented by the same MPDAG.

Proposition 7. *For a given path p in a MPDAG $\mathcal{M}$ which is not possibly directed, the corresponding path in each DAG G represented by M is not a directed path.*

Proof. Let $p = \langle X_1, \dots, X_n \rangle$ be a path in $\mathcal{M}$ which is not possibly directed. Therefore, there exist two nodes X_i and X_j in p with $i < j$ such that $X_i \leftarrow X_j$ is in $\mathcal{M}$. Then, for every DAG G represented by $\mathcal{M}$, by definition 26, the corresponding path $\tilde{p}$ in G also contains the directed edge $X_i \leftarrow X_j$. Hence, $\tilde{p} = \langle X_1, \dots, X_n \rangle$ in G is not a directed path, as there exist two nodes X_i and X_j in $\tilde{p}$ with $i < j$ such that $X_i \leftarrow X_j$ is in G. $\square$

Additionally, the contrapositive statement of proposition 7 shows that if $\tilde{p}$ is a directed path in any DAG G represented by $\mathcal{M}$, then the corresponding path p in $\mathcal{M}$ is a possibly directed path.

The following two results rely upon the notions of a *fully determined node* and a *fully determined path* [29]. We say that a node X_i on an undirected path $p = \langle X_1, \dots, X_n \rangle$ ($1 \leq i \leq n$, $n \in \mathbb{N}$) in a DAG G is a fully determined node if satisfies one of the following:

1. X_i is a collider in p.
2. X_i is an end-point of p.
3. X_i has an outgoing edge in p
4. $X_k - X_i - X_j$ is part of p, and X_k and X_j are not adjacent.

Furthermore, we say that an undirected path $p = \langle X_1, \dots, X_n \rangle$ ($1 \leq i \leq n$, $n \in \mathbb{N}$) in DAG G is a fully determined path if every node X_i on p is a fully determined node.

Proposition 8. *Given a fully determined path $p = \langle X_1, \dots, X_n \rangle$ ($1 \leq i \leq n$, $n \in \mathbb{N}$) in a MPDAG $\mathcal{M}$, p is a possibly directed path if and only if there is no edge $X_i \leftarrow X_{i+1}$ for $i \in \{1, \dots, n-1\}$ in $\mathcal{M}$.*

Proof. ($\Rightarrow$) This direction follows by the definition of a possibly directed path in a MPDAG $\mathcal{M}$.
($\Leftarrow$) We proceed by contradiction. Assume that p is not possibly directed path in $\mathcal{M}$, and thus there exists a directed edge $X_k \leftarrow X_t$ such that $1 \leq k < t \leq n$ and $k+1 < t$. By assumption, the MPDAG $\mathcal{M}$ contains either the edge $X_i - X_{i+1}$ or the directed edge $X_i \rightarrow X_{i+1}$ for all $i \in \{1, \dots, n-1\}$. Since p is a fully determined path, all the nodes on p are fully determined, and given that no edge $X_i \leftarrow X_{i+1}$ for $i \in \{1, \dots, n-1\}$ is in $\mathcal{M}$, all nodes on p either have an outgoing edge in p or they are part of a triple where the other two nodes are not adjacent. In particular, consider a DAG G represented by $\mathcal{M}$ that contains the directed edge $X_1 \rightarrow X_2$. Therefore, due to p being fully determined, it means that $\tilde{p}$ contains all the directed edges $X_i \rightarrow X_{i+1}$ for $i \in \{1, \dots, n-1\}$. However, this produces the directed cycle $X_k \rightarrow \dots \rightarrow X_t \rightarrow X_k$ in G, which yields a contradiction.

This proves the proposition, as we have proved both directions of the claim. □

Proposition 9. *Given a possibly directed path $p = \langle X_1, \dots, X_n \rangle$ ($1 \leq i \leq n$, $n \in \mathbb{N}$) in a MPDAG $\mathcal{M}$, there exists a subset of the nodes $\{X_1, \dots, X_n\}$ which forms a fully determined, possibly directed path in which every node is not a collider.*

Proof. The last sentence of the claim means that every node in the mentioned subset satisfies either condition 2, 3 or condition 4 of the definition of a fully determined node. We prove the claim by induction on the number of nodes in p.

1. $n = 3$
   In this case, $p = \langle X_1, X_2, X_3 \rangle$. Since p is possibly directed, the edge $X_1 \leftarrow X_3$ is not in $\mathcal{M}$, so either X_1 and X_3 are not adjacent, thus X_2 satisfying condition 4 of the definition of a fully determined node, or we have either $X_1 - X_3$ or $X_1 \rightarrow X_3$

in $\mathcal{M}$, in which the path $\tilde{p} = \langle X_1, X_3 \rangle$ satisfies the claim.

2. $n = k - 1$ for $k > 4$
   Assume that the claim holds for all paths p of length $n = k - 1$ for $k > 4$.

3. $n = k$ for $k > 4$
   If all nodes in a path $p = \langle X_1, \ldots, X_{k+1} \rangle$ of length k satisfy condition 4 of the definition of a fully determined node, then we are done. To that end, suppose there is a triple $X_{i-1} - X_i - X_{i+1}$ in p ($2 \leq i \leq k - 1$) such that we either have the edge $X_{i-1} - X_{i+1}$ or the edge $X_{i-1} \rightarrow X_{i+1}$ in $\mathcal{M}$. Note that since p is possibly directed, we cannot have the edge $X_{i-1} \leftarrow X_{i+1}$ in $\mathcal{M}$. Then, in a sense we "skip" the node X_i, and thus the concatenation (adjoining) of the paths $\langle X_1, \ldots, X_{i-1} \rangle$, $\langle X_{i-1}, X_{i+1} \rangle$ and $\langle X_{i+1}, \ldots, X_{k+1} \rangle$ is a possibly directed path in $\mathcal{M}$ of length $(k+1) - (i+1) + 1 + (i-1) - 1 = k - 1$, which by the induction hypothesis implies that there exists a subset of the nodes $\{X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_{k+1}\}$, and thus a subset of $\{X_1, \ldots, X_{k+1}\}$, which forms a fully determined, possibly directed path in which every node is not a collider, as desired.

Therefore, by induction, the claim is proved. □

We observe that by properly modifying the existing theory on directed paths and ancestor-descendant relationships in regular DAGs, we are able to provide sound definitions of the same concepts in terms of MPDAGs. This further allows us to prove important claims related to determining dependence relations in MPDAGs, which is an important and necessary precursor for developing structure learning approaches tailored to Bayesian networks with background information.

5 Bayesian Networks with Incomplete Information

In this section we explore Bayesian networks with incomplete information, to which one might refer to as the opposite of Bayesian networks with background information. Namely, as discussed in section 4, in cases of established dependence relations, previous observations or model restrictions, one modifies the structure of a given CPDAG by adding an oriented edge from the set of background information, denoted by $\mathcal{K}$, thus obtaining a so-called MPDAG. However, in many practical applications, especially applications involving scientific or corporate measurement, one often encounters the situation of having incomplete observations, i.e. a dataset in which certain values are missing [13].

Due to the fact that incomplete observations are prevalent in many real-world situations, one would like to be able to successfully perform the task of structure learning even when data is incomplete. However, the approaches discussed in section 3.3 assume data to be complete. As such, a naive attempt to try to somehow incorporate these approaches for networks with incomplete data would not yield an actual outcome. Therefore, structure learning of Bayesian networks from incomplete data is an even more challenging task as one has to infer both the missing data values and the network structure from the observed data [13].

Bayesian network structure learning from incomplete data is an active line of research in the field of structure learning. One renowned approach for handling networks with missing data is *data imputation*, which refers to filling in missing values with statistical measures from the observed data such as the mean of the observed data [31]. This approach is straightforward but may introduce bias. Another widely used approach is the so-called EM algorithm and its extension, the structural EM algorithm [6, 20]. These algorithms typically iterate between two steps, the *E-step*, during which the missing values are sampled, and the *M-step*, during which the model parameters are updated based on the estimated complete data. In particular, the structural EM-algorithm searches for a fitting DAG by considering a *penalized likelihood* in an expectation-minimization algorithm [6]. Furthermore, a more recent approach is the so-called NAL approach which relies upon penalized node-average log-likelihoods (NALs), which one can compute from locally complete data [1]. Further approaches include the use of partial data, implementing data augmentation, as well as model averaging [20].

In this section, we focus on a recent paper from 2023 [13], which provides a novel approach for handling the task of structure learning with incomplete data. Namely, the author develops a so-called *Bayesian model averaging* (BMA) approach for *Gaussian Bayesian networks*, which are Bayesian networks in which the variables in the network are continuous and follow a multivariate Gaussian distribution [21]. Given the graph structure of the network, the joint probability distribution of all variables in the network can be factorized into *conditional Gaussian distributions*. The proposed BMA approach builds upon the BGe score developed by Geiger and Heckerman [11] as a metric for score-based approaches aimed at structure learning for Gaussian Bayesian networks. Fundamentally, it proposes a so-called *Markov Chain Monte Carlo* (MCMC) sampling algorithm for sampling DAGs and missing data values from posterior distributions [13]. Crucially, this approach achieved higher network reconstruction accuracy compared to two of the most prominent approaches for handling missing data in Bayesian networks, the structural EM algorithm and the NAL approach [13]. Nevertheless, the proposed approach is predominantly tailored to Gaussian Bayesian networks as it relies upon the fact that one is able to sample missing values from a multivariate conditional Gaussian distribution. This is not possible in discrete Bayesian networks, as the conditional distributions required for the proposed BMA approach are not of a well-known form in discrete networks. Nevertheless, from a conceptual standpoint, one can adapt the proposed BMA approach to discrete networks, if one were able to find a suitable approach for sampling the missing values in the dataset.

In this section, we provide an initial discussion of how one can tackle this issue in discrete networks. To that end, we introduce the concept of *belief propagation*, first developed as the "sum-product" algorithm for *factor graphs* by Pearl [26]. To the best of our knowledge, we contribute to this research field by providing a detailed adaptation of belief propagation to Bayesian networks and proposing an approach for efficient use of belief propagation in Bayesian networks relying upon the local Markov property of Bayesian networks. Additionally, we discuss the convergence of the belief propagation algorithm in different graphs, and provide a comprehensive implementation of belief propagation in Bayesian networks.

5.1 The Belief Propagation Algorithm

In this section we introduce the belief propagation algorithm in its original formulation. We first present an intuitive idea of what the goal of the belief propagation algorithm is, and how the algorithm aims to achieve this goal. Fundamentally, for a given network with both *unobserved* (missing) and observed random variables, the purpose of the belief propagation algorithm is to calculate the marginal distribution for each unobserved node, conditional on any observed nodes. To introduce needed terminology, an unobserved variable is one type of missing data, referring to a variable that we do not observe, but whose position in the network is known [31]. Other types of missing data include *latent variables*, which represent variables that we know nothing about, and *partially observed variables*, for which we have some observations, but not all [21]. Furthermore, we distinguish between three classes of missing data: data *missing completely at random* (MCAR), data *missing at random* (MAR), and data *missing not at random* (MNAR) [31]. In this section, we focus on random variables with the incomplete data being MCAR, meaning that the missing data points are a random subset of the dataset. This is in accordance with the made assumptions in the mentioned BMA approach [13].

The way these marginal distributions are calculated using the belief propagation algorithm relies upon the idea of *iterative message passing* in a network [26]. We formalize what message-passing means in the context of random variables and conditional probabilities in subsequent paragraphs. From an intuitive point of view, message-passing in a factor graph (network) refers to viewing the graph as a dynamic processing unit, rather than our usual understanding of a static representation of the dependence relations between random variables [26]. This means that the edges between nodes are viewed as links through which information flows between the nodes. In that sense, message-passing refers to iteratively sending and receiving information between adjacent nodes in a network. More concretely, in a Bayesian network, the mentioned messages will simply be defined to be the conditional probabilities imposed by the network structure.

In its original formulation, the belief propagation algorithm was tailored to factor graphs. To that end, we introduce *bipartite graphs*.

Definition 30 (Bipartite Graph)**.**
A bipartite graph G is a graph whose set of vertices $V(G)$ can be divided into two disjoint and independent sets $V(G) = U(G) \cup W(G)$, and whose edges connect a node in $U(G)$ to a node in $W(G)$.

Definition 31 (Factor Graph)**.**
A factor graph G is an undirected bipartite graph which represents the factorization of a real multivariate function $g(X_1, \dots, X_n)$ for $n \in \mathbb{N}$ as a product of real multivariate functions f_j:

$$g(X_1, \dots, X_n) = \prod_{j=1}^{m} f_j(N_j) \tag{5.1}$$

where $N_j \subseteq \{X_1, \dots, X_n\}$.

More concretely, being a bipartite graph, the nodes of a factor graph G are given as a disjoint union of the set of *variable* nodes $U(G) = \{X_1, \dots, X_n\}$ and the set of *function* nodes $W(G) = \{f_1, \dots, f_m\}$. Moreover, there is an undirected edge between a variable node X_i and a function variable f_j if $X_i \in N_j$. An example is given in figure 5.1, which represents the factorization of a function $g(X_1, X_2, X_3)$ as

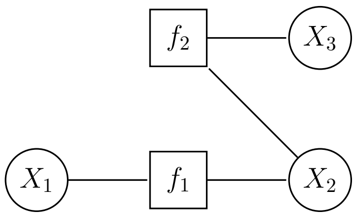

Figure 5.1: A factor graph G.

$g(X_1, X_2, X_3) = f_1(X_1, X_2)f_2(X_2, X_3)$.

Moreover, similar to probability distributions, we define the marginal of a variable X_i as $g(X_i) = \sum_{X_{\bar{i}}} g(X_1, \dots, X_n)$, where $X_{\bar{i}}$ indicates that we sum over all variables except X_i [28]. Notice that the notions we defined are incredibly similar, almost identical, to our treatment of Bayesian networks if one consider the function g to be a joint distribution, the variables X_i to be random variables, and the function variables to be conditional probabilities.

These concepts are sufficient for us to formulate the belief propagation algorithm for factor graphs. We do this constructively by considering the factor graph in figure 5.1. First, we assume that the variables X_i take on finitely many values from a finite set $\mathcal{X}$, akin to discrete random variables. Moreover, let $\lambda^{(t)}_{i\to f_j}(x_i)$ denote a message sent from a variable node X_i to a function node f_j at time step $t \in \mathbb{N}$, evaluated at the value point $x_i \in \mathcal{X}$. Similarly, let $\pi^{(t)}_{f_j\to i}(x_i)$ denote the message sent from a function node f_j to a variable node X_i at time step $t \in \mathbb{N}$, evaluated at the value point $x_i \in \mathcal{X}$. The belief propagation has three main steps:

1. Initialization step: In this step we initialize the messages at $t = 0$ for the leaf nodes of the factor graph G, in this case X_1 and X_3, as:

$$\begin{aligned} \lambda^{(0)}_{i\to f_j}(x_i) &= \frac{1}{|\mathcal{X}|} \\ \pi^{(0)}_{f_j\to i}(x_i) &= f_j(x_i) \end{aligned} \tag{5.2}$$

   which in our case are the two variable to function messages $\lambda^{(0)}_{1\to f_1}(x_1)$ and $\lambda^{(0)}_{3\to f_2}(x_3)$, both set to $\frac{1}{|\mathcal{X}|}$. Setting the initial value to 1 over the cardinality of the set of values the variables X_i take, comes from probability theory and having uniformly distributed variables. Note that, in alternative formulations of the algorithm it is also typical to set these initial messages to 1, and have them normalized in subsequent steps. Step 1 is illustrated in figure 5.2.

2. Message Passing: In this step we compute outgoing messages when incoming messages are available. In particular, a message from a variable node to a function

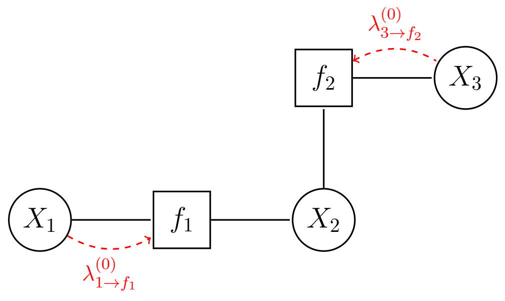

Figure 5.2: Step 1 of Belief Propagation in G at $t = 0$.

node is the product of the incoming messages, and a message from a function node to a variable node is the sum over the previous variables of the product of incoming messages and the function variable evaluated at its adjacent variables nodes. More precisely, at time step $t > 0$ and $t + 1$, these messages are given as:

$$\lambda^{(t+1)}_{i\to f_j}(x_i) = \prod_{f_k \in adj(X_i)\setminus\{f_j\}} \pi^{(t)}_{f_k\to i}(x_i) \tag{5.3}$$

$$\pi^{(t)}_{f_j\to i}(x_i) = \sum_{S_j\setminus\{X_i\}} f_j(S_j) \prod_{X_k\in adj(f_j)\setminus\{X_i\}} \lambda^{(t-1)}_{k\to f_j}(x_k) \tag{5.4}$$

where in (5.3) $adj(X_i)\setminus\{f_j\}$ denotes the adjacent function nodes to the variable X_i except the function node f_j to which the outgoing message is passed to. Similarly, in (5.4) $adj(f_j) \setminus \{X_i\}$ denotes the adjacent variable nodes of the function variable f_j except X_i to which the outgoing message is passed on. Note that in (5.3) the time step is set to $t + 1$ to distinguish that these messages are not computed simultaneously for a given edge. It is important to state, that the expressions in (5.3) and (5.4) are the so-called *fundamental* belief propagation equations, which govern the entire iterative procedure. In the factor graph from figure 5.1, the different time steps are illustrated in figure 5.3 and the obtained messages are given as:

$$\begin{aligned}
\pi^{(1)}_{f_2\to 2}(x_2) &= \sum_{X_3} f_2(X_2,X_3)\lambda^{(0)}_{3\to f_2}(x_3)\\
&= \sum_{X_3} f_2(X_2,X_3)\\
\pi^{(1)}_{f_1\to 2}(x_2) &= \sum_{X_1} f_1(X_1,X_2)\lambda^{(0)}_{1\to f_1}(x_1)\\
&= \sum_{X_1} f_1(X_1,X_2)\\
\lambda^{(2)}_{2\to f_2}(x_2) &= \pi^{(1)}_{f_1\to 2}(x_2)\\
&= \sum_{X_1} f_1(X_1,X_2)\\
\lambda^{(2)}_{2\to f_1}(x_1) &= \pi^{(1)}_{f_2\to 2}(x_2)\\
&= \sum_{X_3} f_2(X_2,X_3)\\
\pi^{(3)}_{f_2\to 3}(x_3) &= \sum_{X_2} f_2(X_2,X_3)\lambda^{(2)}_{2\to f_2}(x_2)\\
&= \sum_{X_2}\sum_{X_1} f_2(X_2,X_3)f_1(X_1,X_2)\\
\pi^{(3)}_{f_1\to 1}(x_1) &= \sum_{X_2} f_1(X_1,X_2)\lambda^{(2)}_{2\to f_1}(x_1)\\
&= \sum_{X_2}\sum_{X_3} f_1(X_1,X_2)f_2(X_2,X_3)
\end{aligned} \tag{5.5}$$

where we set the initialized values discussed in step 1 to 1 for simplicity and to avoid cluttered notation.

3. Termination: In this step we estimate the marginals of each variable $g(X_i) = \sum_{X_{\tilde{i}}} g(X_1,\dots,X_n)$, where $X_{\tilde{i}}$ indicates that we sum over all variables except X_i. According to the algorithm, each marginal at time step $t+1$ is equal to the product of all of its incoming messages:

$$g(X_i) = \prod_{f_j\in adj(X_i)} \pi^{(t)}_{f_j\to i}(x_i) \tag{5.6}$$

So, in our example, at $t=4$, we have $g(X_1) = \sum_{X_2}\sum_{X_3} f_1(X_1,X_2)f_2(X_2,X_3)$, $g(X_2) = \sum_{X_1}\sum_{X_3} f_1(X_1,X_2)f_2(X_2,X_3)$ and $g(X_3) = \sum_{X_1}\sum_{X_2} f_1(X_1,X_2)f_2(X_2,X_3)$.

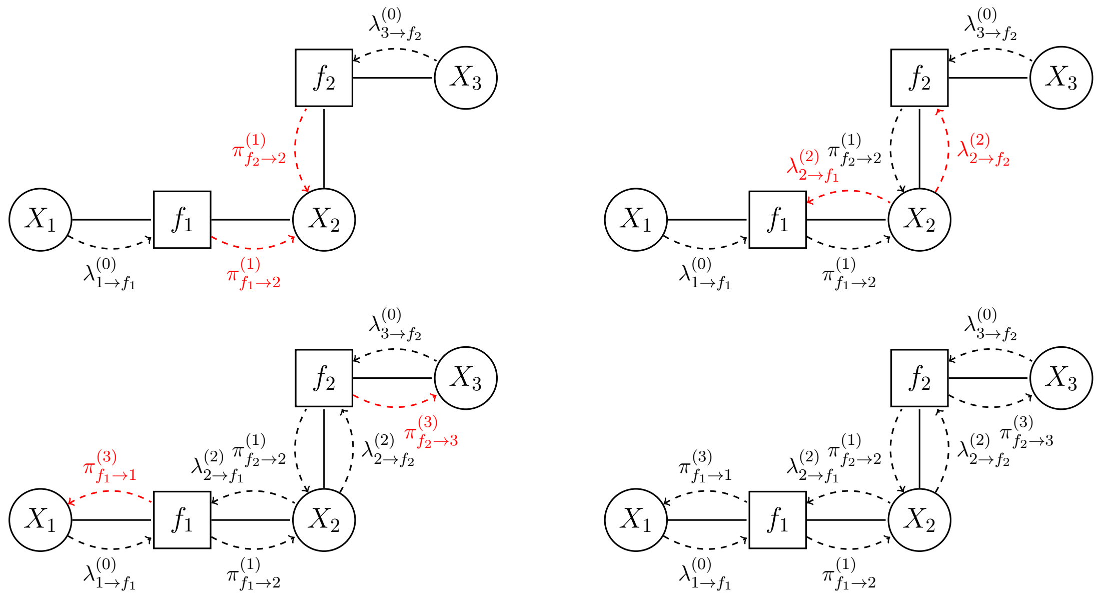
$\lambda^{(0)}_{3\to f_2}$
f_2
X_3
$\pi^{(1)}_{f_2\to 2}$
X_1
f_1
X_2
$\lambda^{(0)}_{1\to f_1}$
$\pi^{(1)}_{f_1\to 2}$
$\lambda^{(0)}_{3\to f_2}$
f_2
X_3
$\lambda^{(2)}_{2\to f_1}$
$\pi^{(1)}_{f_2\to 2}$
$\lambda^{(2)}_{2\to f_2}$
X_1
f_1
X_2
$\lambda^{(0)}_{1\to f_1}$
$\pi^{(1)}_{f_1\to 2}$
$\lambda^{(0)}_{3\to f_2}$
f_2
X_3
$\pi^{(3)}_{f_1\to 1}$
$\lambda^{(2)}_{2\to f_1}$
$\pi^{(1)}_{f_2\to 2}$
$\lambda^{(2)}_{2\to f_2}$
$\pi^{(3)}_{f_2\to 3}$
X_1
f_1
X_2
$\lambda^{(0)}_{1\to f_1}$
$\pi^{(1)}_{f_1\to 2}$
$\lambda^{(0)}_{3\to f_2}$
f_2
X_3
$\pi^{(3)}_{f_1\to 1}$
$\lambda^{(2)}_{2\to f_1}$
$\pi^{(1)}_{f_2\to 2}$
$\lambda^{(2)}_{2\to f_2}$
$\pi^{(3)}_{f_2\to 3}$
X_1
f_1
X_2
$\lambda^{(0)}_{1\to f_1}$
$\pi^{(1)}_{f_1\to 2}$

Figure 5.3: Steps 2 of Belief Propagation in G at $t = 1$ (top left), at $t = 2$ (top right), at $t = 3$ (bottom left) and Step 3 of Belief Propagation in G (bottom right).

In section 5.3, in the case of a factor graph representing a joint distribution, we observe that the estimate in (5.6) and the fundamental belief propagation equations in (5.3) and (5.4) are exact on polytrees and produce the exact marginal distributions. It is important to state that according to the Hammersley-Clifford fundamental theorem of Markov fields, any positive joint distribution $\mathbb{P}(\boldsymbol{X})$ can be represented via factor graphs, thus making equations (5.3) and (5.4) immediately applicable to Bayesian networks [14]. In section 5.2 we take a different route, and we provide an adaptation of the belief propagation algorithm to Bayesian networks without resorting to the Hammersley-Clifford theorem.

5.2 Belief Propagation for Bayesian Networks

In this section we provide a detailed account of how we can adapt the described belief propagation algorithm for factor graphs to the case of directed acyclic graphs, in particular Bayesian networks. This adaptation in turn serves as the base for our proposal for sampling missing values in discrete Bayesian networks with incomplete data, which is explored in section 5.3. We make the remark, that the presented algorithm is tailored to Bayesian networks represented by polytrees, meaning that the network is represented by a DAG whose underlying undirected graph does not possess any cycles (see section 2.1 for more details). We take this approach due to the fact that the belief propagation algorithm is exact on polytrees, which we discuss in section 5.3, and because belief propagation on polytrees fits our proposal for sampling missing values which we discuss in section 5.3 as well. Moreover, in our adaptation of the belief propagation algorithm to Bayesian networks, we assume all the random variables to be discrete, as we want to tailor our findings to discrete Bayesian networks as per our discussion of the results obtained in [13]. Nevertheless, the belief propagation algorithm can also be adapted for absolutely continuous variables in a very similar manner [26].

To avoid cluttered notation, in the derivation of the equations for belief propagation in Bayesian networks we do not add the superscript t to indicate the specific time step. Therefore, we remark that all the presented equations refer to a specific node in the network at an arbitrary time step. To that end, consider a network with $r \in \mathbb{N}$ random variables as nodes, out of which there is a subset of so-called observed nodes, for which we have complete data, and a subset of unobserved nodes. As a starting point, we state the problem we want to solve. Namely, we consider an *unobserved* (also known as latent or query) node, denoted by X, and the subset of observed (also known as evidence) nodes, denoted by E. Moreover, we denote the parent set of X by $Pa(X)$ and assume X to have $k < r$ parent nodes, thus $Pa(X) = \{Z_1, \dots, Z_k\}$. Similarly, we denote the child set of X by $Ch(X)$ and assume X to have $n < r$ children nodes, thus $Ch(X) = \{Y_1, \dots, Y_n\}$. Moreover, to avoid cumbersome notation, we treat the evidence nodes as *virtual*, meaning obtained from dummy children of variables whose values are known. Furthermore, the set of evidence nodes E can be written as a disjoint union $E = A_X \cup D_X$, where A_X denotes the subset of evidence nodes which has a path to X through the parent nodes of X, and D_X denotes the subset of evidence nodes which has a path to X through the children nodes of X. Note that both A_X and D_X could be empty.

As in the standard premise of belief propagation, we treat X as a *processing* node which receives and sends "messages" through its parents (top-down propagation) and through its children (bottom-up propagation). We denote a message from a parent node Z_i ($i \in \{1, \dots, k\}$) to X by $m^+_X(Z_i)$, and a message from X to Z_i by $m^-_X(Z_i)$. Similarly, we denote a message from a child node Y_j ($j \in \{1, \dots, n\}$) to X by $m^-_{Y_j}(X)$, and a message from X to Y_j by $m^+_{Y_j}(X)$. In both cases, the "$+$" symbol denotes a top-down message and the "$-$" symbol denotes a bottom-up message. When it comes to the parent nodes of X, we have in total $2k$ messages (top-down and bottom-up messages), and similarly, considering the children nodes of X, we have in total $2n$ messages. Visually, one can represent the given process using figure 5.4.

Now, since we assume $E = A_X \cup D_X$ to be the evidence nodes, let D_X take on a value and A_X take on a value, i.e. $A_X = a_X$ and $D_X = d_X$, where we note that these values are vectors, as both A_X and D_X denote subsets of nodes. Therefore, to find the marginal distribution of X given the evidence nodes E, we want to solve:

$$\mathbb{P}(X \mid E) = \mathbb{P}(X \mid D_X, A_X) \tag{5.7}$$

where we again rely onto the abuse of notation introduced in section 2.3. Now, since we

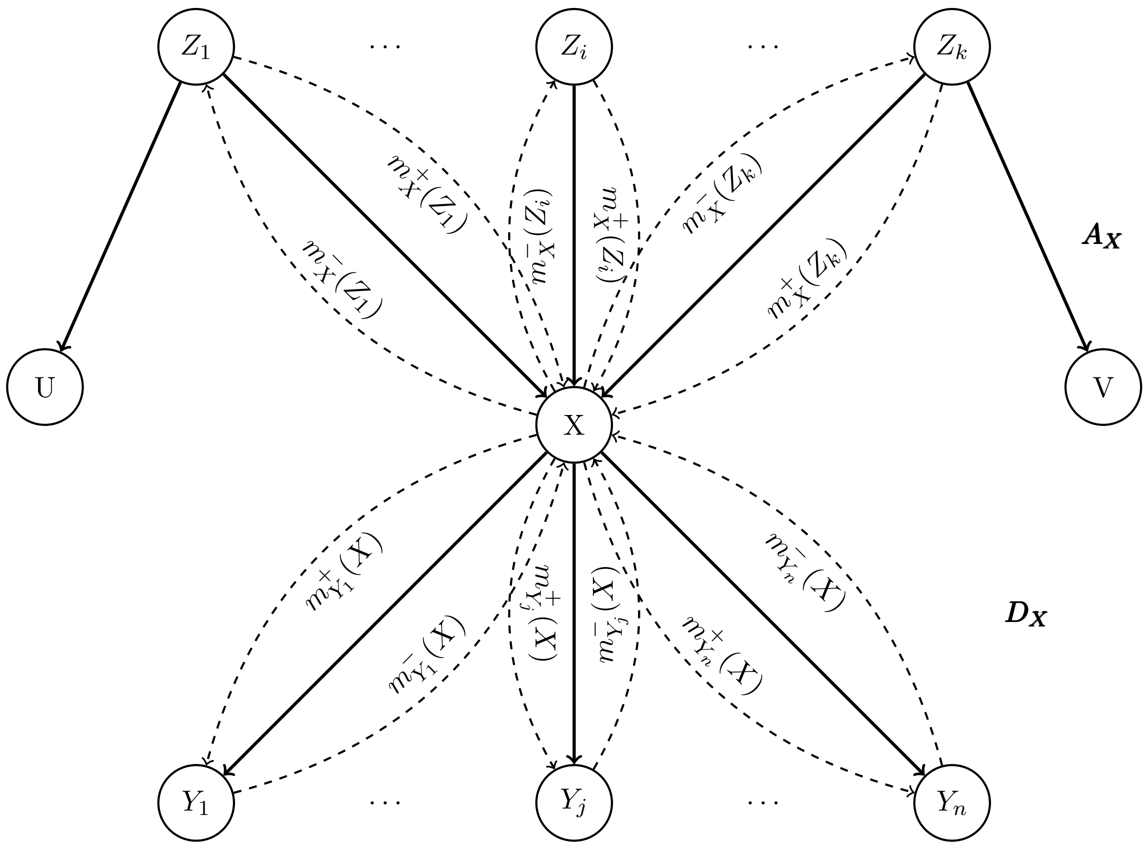

Figure 5.4: A visual representation of Belief Propagation in Bayesian networks.

know that each node d-separates its parents from its children, we have that D_X and A_X are d-separated with respect to X, so by lemma 4, we have that $A_X \perp\!\!\!\perp D_X | X$, thus:

$$\begin{aligned}
\mathbb{P}(X|E) &= \mathbb{P}(X|D_X, A_X) \\
&= \frac{\mathbb{P}(D_X, A_X|X)\mathbb{P}(X)}{\mathbb{P}(D_X, A_X)} \\
&= \frac{\mathbb{P}(A_X|X)\mathbb{P}(D_X|X)\mathbb{P}(X)}{\mathbb{P}(D_X, A_X)} \\
&= \frac{\mathbb{P}(X|A_X)\mathbb{P}(A_X)\mathbb{P}(D_X|X)\mathbb{P}(X)}{\mathbb{P}(D_X, A_X)\mathbb{P}(X)} \\
&= \frac{\mathbb{P}(X|A_X)\mathbb{P}(A_X)\mathbb{P}(D_X|X)}{\mathbb{P}(D_X, A_X)} \\
&= \kappa\mathbb{P}(X|A_X)\mathbb{P}(D_X|X) \\
&= \kappa m^+(X)m^-(X)
\end{aligned} \tag{5.8}$$

where $\kappa = \frac{\mathbb{P}(A_X)}{\mathbb{P}(D_X, A_X)}$ is a normalizing constant, and we denote $m^+(X) = \mathbb{P}(X|A_X)$,

$m^{-}(X) = \mathbb{P}(D_X|\, X)$, to be the messages that are propagated to X from its parents and then passed onto its children ($m^{+}(X)$), and the messages that are propagated from its children and then onto its parents ($m^{-}(X)$). The core of the belief propagation algorithm for Bayesian networks is summarized in (5.8). The remainder of this section is devoted to evaluating $m^{+}(X)$ and $m^{-}(X)$.

To that end, we write $D_X = \bigcup_{j=1}^{n} D_{X,Y_j}$ where D_{X,Y_j} is a subset of D_X that represents the evidence contained in the subgraph on the bottom (head) side of $X \to Y_j$, meaning it contains nodes which are also descendants of Y_j. Similarly, we write $A_X = \bigcup_{i=1}^{k} A_{X,Z_i}$ where A_{X,Z_i} is a subset of A_X that represents the evidence contained in the subgraph on the upper (tail) side of $Z_i \to X$, meaning it contains also ancestors of Z_i. This also prompts us to define the messages sent to X from a specific parent and child, and the messages from X sent to a specific parent and child:

$$\begin{aligned} m_X^{+}(Z_i) &= \mathbb{P}(Z_i|\, A_{X,Z_i}) \\ m_X^{-}(Z_i) &= \mathbb{P}(D_{X,Z_i}|\, Z_i) \\ m_{Y_j}^{+}(X) &= \mathbb{P}(X|\, A_{X,Y_j}) \\ m_{Y_j}^{-}(X) &= \mathbb{P}(D_{X,Y_j}|\, X) \end{aligned} \tag{5.9}$$

where in the second line of (5.9), D_{X,Z_i} represents the entire evidence E, excluding the evidence propagated through A_{X,Z_i}, and in the third line of (5.9), A_{X,Y_j} represents the entire evidence E, excluding the evidence propagated through D_{X,Y_j}. This leads to:

$$\begin{aligned} m^{-}(X) &= \mathbb{P}(D_X|\, X) \\ &= \mathbb{P}(D_{X,Y_1}, \ldots, D_{X,Y_n}|\, X) \\ &= \prod_{j=1}^{n} \mathbb{P}(D_{X,Y_j}|\, X) \\ &= \prod_{j=1}^{n} m_{Y_j}^{-}(X) \end{aligned} \tag{5.10}$$

where in the third equality of (5.10) we make use of the fact that our DAG is a polytree and thus X d-separates $Y_1, \ldots, Y_n$, and consequently, it d-separates $D_{X,Y_1}, \ldots, D_{X,Y_n}$. We observe the similarity between the expression in (5.10) and the expression in (5.3).

On the other hand, for the message received from X by its parents and passed onto its children we obtain:

$$\begin{aligned}
m^+(X) &= \mathbb{P}(X|\, A_X) \\
&= \mathbb{P}(X|\, A_{X,Z_1}, \dots, A_{X,Z_k}) \\
&= \sum_{Z_1} \dots \sum_{Z_k} \mathbb{P}(X, Z_1, \dots, Z_k|\, A_{X,Z_1}, \dots, A_{X,Z_k}) \\
&= \sum_{Z_1} \dots \sum_{Z_k} \mathbb{P}(X|\, Z_1, \dots, Z_k, A_{X,Z_1}, \dots, A_{X,Z_k}) \mathbb{P}(Z_1, \dots, Z_k|\, A_{X,Z_1}, \dots, A_{X,Z_k}) \\
&= \sum_{Z_1} \dots \sum_{Z_k} \mathbb{P}(X|\, Z_1, \dots, Z_k) \mathbb{P}(Z_1, \dots, Z_k|\, A_{X,Z_1}, \dots, A_{X,Z_k}) \\
&= \sum_{Z_1} \dots \sum_{Z_k} \mathbb{P}(X|\, Z_1, \dots, Z_k) \prod_{i=1}^{k} \mathbb{P}(Z_i|\, A_{X,Z_i}) \\
&= \sum_{Z_1} \dots \sum_{Z_k} \mathbb{P}(X|\, Z_1, \dots, Z_k) \prod_{i=1}^{k} m_X^+(Z_i)
\end{aligned} \tag{5.11}$$

where in the third equality of (5.20) we marginalized the distribution over $(Z_1, \dots, Z_n)$, in the fourth equality we used the definition of conditional probability to re-write the expression from before, in the fifth equality we used the fact that $(Z_1, \dots, Z_n)$ d-separates X and A_{X,Z_i} for $i = 1, \dots, k$, and finally in sixth equality we used the fact that $\{Z_i, A_{X,Z_i}\}$ are independent for $i = 1, \dots, k$. We again notice the similarity between the expression in (5.20) and the expression in (5.4). Finally, this yields an elegant expression for $\mathbb{P}(X|\, E)$:

$$\mathbb{P}(X|\, E) = \kappa m^+(X) m^-(X) \tag{5.12}$$

$$= \kappa \Big(\sum_{\boldsymbol{Z}} \mathbb{P}(X|\, \boldsymbol{Z}) \prod_{i=1}^{k} m_X^+(Z_i) \Big) \prod_{j=1}^{n} m_{Y_j}^-(X) \tag{5.13}$$

which shows that we can obtain the marginal distribution for X given the set of observed variables E by receiving messages from its child nodes and parent nodes, which are defined to be conditional probabilities. An important aspect is that to use (5.13), we must know $\mathbb{P}(X|\, \boldsymbol{Z})$ either through assumption or estimation [28].

The remaining question is to determine how will X compute its outgoing messages $m_X^-(Z_i)$ and $m_{Y_j}^+(X)$ from the incoming messages $m_X^+(Z_i)$ and $m_{Y_j}^-(X)$, i.e. we have to determine the next iterative step once X has received its messages from its parents and children. We show this for the variable X as per the DAG presented in figure 5.4.

We first derive an expression for $m_{Y_j}^+(X)$, where we remind ourselves of the definition of A_{X,Y_j}, which we can also write as $A_{X,Y_j} = E \setminus D_{X,Y_j} = A_X \cup \{D_X \setminus D_{X,Y_j}\}$. For conciseness, we define a normalization constant β beforehand as $\beta = \frac{1}{\mathbb{P}(D_X \backslash D_{X,Y_j} | A_X)}$. This leads to:

$$\begin{aligned} m^+_{Y_j}(X) &= \mathbb{P}(X | A_{X,Y_j}) \\ &= \mathbb{P}(X | A_X, \{D_X \setminus D_{X,Y_j}\}) \\ &= \frac{\mathbb{P}(D_X \setminus D_{X,Y_j} | A_X, X)\mathbb{P}(X | A_X)}{\mathbb{P}(D_X \setminus D_{X,Y_j} | A_X)} \\ &= \beta \cdot \mathbb{P}(D_X \setminus D_{X,Y_j} | A_X, X) m^+(X) \\ &= \beta \prod_{l \neq j} \mathbb{P}(D_{X,Y_l} | X) m^+(X) \\ &= \beta \prod_{l \neq j} m^-_{Y_l}(X) m^+(X) \end{aligned} \tag{5.14}$$

where in the third equality of (5.14) we used the definition of conditional probability and in the fifth equality we used the fact that X d-separates its children.

To derive an expression of $m^-_X(Z_i)$, we remind ourselves of the definition of D_{X,Z_i}, which we can write as $D_{X,Z_i} = E \setminus A_{X,Z_i} = D_X \cup \{A_X \setminus A_{X,Z_i}\}$. Again, for conciseness, we define a normalization constant γ beforehand as $\gamma = \mathbb{P}(A_X \setminus A_{X,Z_i})$. This leads to:

$$\begin{aligned} m^-_X(Z_i) &= \mathbb{P}(D_{X,Z_i} | Z_i) \\ &= \sum_X \sum_{Z_l : l \neq i} \mathbb{P}(D_{X,Z_i}, X, Z_l | Z_i) \\ &= \sum_X \sum_{Z_l : l \neq i} \mathbb{P}(D_X, A_X \setminus A_{X,Z_i}, X, Z_l | Z_i) \\ &= \sum_X \sum_{Z_l : l \neq i} \mathbb{P}(D_X, A_X \setminus A_{X,Z_i} | X, Z_l, Z_i)\mathbb{P}(X, Z_l | Z_i) \\ &= \sum_X \sum_{Z_l : l \neq i} \mathbb{P}(A_X \setminus A_{X,Z_i} | Z_l)\mathbb{P}(D_X | X)\mathbb{P}(X, Z_l | Z_i) \\ &= \sum_X \sum_{Z_l : l \neq i} \frac{\gamma \mathbb{P}(Z_l | A_X \setminus A_{X,Z_i})}{\mathbb{P}(Z_l)} \mathbb{P}(D_X | X)\mathbb{P}(X, Z_l | Z_i) \\ &= \gamma \sum_X \sum_{Z_l : l \neq i} \mathbb{P}(Z_l | A_X \setminus A_{X,Z_i})\mathbb{P}(D_X | X)\mathbb{P}(X | Z_l, Z_i) \\ &= \gamma \sum_X \sum_{Z_l : l \neq i} \left(\prod_{l \neq i} m^+_X(Z_l) \right) m^-(X)\mathbb{P}(X | Z_1, \ldots, Z_k) \\ &= \gamma \sum_X m^-(X) \sum_{Z_l : l \neq i} \mathbb{P}(X | \boldsymbol{Z}) \prod_{l \neq i} m^+_X(Z_l) \end{aligned} \tag{5.15}$$

where in the second equality of (5.15) we used marginalization over X and $\{Z_l : l \neq i\}$, in the fourth, fifth and sixth equalities we used the definition of conditional probability multiple times, and we also utilized the fact that X d-separates D_X and $A_X \setminus A_{X,Z_i}$, and $\{Z_l : l \neq i\}$ d-separates $A_X \setminus A_{X,Z_i}$ and Z_i. Finally, in the seventh equality we used the fact that Z_i and $\{Z_l : l \neq i\}$ are independent.

Therefore, we have obtained comprehensive expressions for all the messages sent to and from a random variable X, thus formalizing the iterative step of the belief propagation algorithm for a given node in a Bayesian network. As a concluding remark, we also specify how the messages coming from leaf nodes are initialized, so that the iterative procedure can commence. If X is a root node, meaning it has no parent nodes, with children $Y_1, \dots, Y_n$, then we initialize $m^+_{Y_j}(X) = \mathbb{P}(X)$, which we assume to be a known prior probability. If on the other hand, X is a leaf node, meaning it has no child nodes, with parents $Z_1, \dots, Z_k$, then we initialize $m^-_X(Z_i) = 1$. If X is an evidence node, meaning we have observed $X = \tilde{x}$, then we initialize all incoming and outgoing messages for $\tilde{x}$ to be 1, and for the other values, the messages are set to 0. This concludes the adaptation of the belief propagation algorithm to Bayesian networks, whose graph structure is a polytree.

5.3 Convergence, Implementation and Proposed Approach for Sampling Missing Data

When discussing the belief propagation algorithm for Bayesian networks as a desired approach for calculating marginal distributions of unobserved variables, one must understand what are the advantages of using this algorithm. One important aspect that highlights the strength of the belief propagation algorithm is computational complexity. To that end, consider a discrete Bayesian network represented by a DAG G with $n \in \mathbb{N}$ random variables as nodes, where each random variable X_i attains values from a set $\mathcal{X} \subset \mathbb{R}$ with cardinality $|\mathcal{X}|$. We recall that if we wish to compute the marginal distribution of an arbitrary node X_i using the joint distribution of the n random variables, we have the expression:

$$\mathbb{P}(X_i) = \sum_{X_1} \cdots \sum_{X_n} \mathbb{P}(\boldsymbol{X}) = \sum_{X_1} \cdots \sum_{X_n} \mathbb{P}\big(X_1, \dots, X_n\big) \tag{5.16}$$

where in (5.16) we sum over all variables except X_i. In order to calculate $\mathbb{P}(X_i)$ for a value X_i attains from $\mathcal{C}$, using (5.16) we would need to sum over $|\mathcal{X}|^{n-1}$ values. In order to calculate $\mathbb{P}(X_i)$ for all values X_i attains from $\mathcal{C}$, we would need to perform $|\mathcal{X}|^n$ calculations, which leads to $\mathcal{O}(|\mathcal{X}|^n)$ complexity. On the other hand, when using the belief propagation algorithm to calculate the marginal distribution of an unobserved node, by observing the expression in (5.13), we see that we have $n-1$ summations over one variable, and with $|\mathcal{X}|$ values that X_i can attain, we obtain $\mathcal{O}(n \cdot |\mathcal{X}|^2)$ complexity. To get an impression of how drastic this difference is, it is enough to consider 100 binary random variables, meaning $n = 100$ and $|\mathcal{X}| = 2$, which gives $|\mathcal{X}|^n = 2^{100} \approx 10^{30}$, whereas $n \cdot |\mathcal{X}|^2 = 400$. Therefore, using the belief propagation makes a lot of difference in terms of computational effort.

In terms of exactness, we state the following result concerning factor graphs with a tree structure and the fundamental belief propagation equations in (5.3) and (5.4), first proved by Pearl [28].

Theorem 8. *Consider a tree factor graph G with a maximum distance $d^* \in \mathbb{N}$ between any two nodes, meaning that the longest walk in G is of length d^*. Moreover, let the factor graph represent a joint distribution of n random variables. Then,*

1. *For any initial condition set in (5.2), the iterative equations in (5.3) and (5.4) converge after at most d^* iterations, i.e. for $t > d^*$, $\lambda^{(t)}_{i \to f_j}(x_i) = \lambda^{(d^*)}_{i \to f_j}(x_i)$ and $\pi^{(t)}_{f_j \to i}(x_i) = \pi^{(d^*)}_{f_j \to i}(x_i)$.*
2. *The estimated marginal equation in (5.6) computes the exact marginal, i.e. $g(X_i) = \mathbb{P}(X_i)$.*

The technical details of the proof of this result are beyond the scope of this paper, and we refer the reader to [28] for details. Nevertheless, for clarity, we provide a brief, slightly informal, argument by induction on d^*. Given an arbitrary edge $U - V$ in G, let the edge be directed, i.e. $U \to V$, without loss of generality. Consider the tree subgraph T of G rooted at this edge, which we denote by $T_{U \to V}$, and which represents the subgraph containing all nodes which are connected to V via a directed path with the last edge being $U \to V$. If we only consider $T_{U \to V}$, with V removed if it is a factor node, and retained if it is a variable node, using induction on d^*, the maximum distance between any two nodes in $T_{U \to V}$, one is able to prove both claims in theorem 8. First assume $d^* = 1$, with U being a variable node and V a factor node. In this case $T_{U \to V}$ is just the node U. Convergence in this case is clear. Following the assumption, regarding the initialization of variable nodes made in section 5.1, that for X_i taking finitely many values from a set $\mathcal{X}$ with cardinality $|\mathcal{X}|$, we initialize the outgoing message of U to be $\frac{1}{|\mathcal{X}|}$ at $t = 0$. Given the structure of $T_{U \to V}$, the message of U stays $\frac{1}{|\mathcal{X}|}$ for all $t > 0$, which in turn coincides with the marginal distribution of a node X_i which is uniformly distributed. Second, assume the claim holds for $d^* \leq \tau \in \mathbb{N}$. We need to show that the claim holds for $t = \tau + 1$. Therefore, assume again U to be a variable node and V to be a factor node. For any $t > \tau + 1$, we can use the equations in (5.3) and (5.4) to compute $\lambda^{(t+1)}_{U \to V}(U)$ in terms of iterates/messages from time step t, which in turn, with a few additional steps proves convergences. Following the induction hypothesis, by combining the equations in (5.3) and (5.4), and noting that d^* is at most τ for each tree subgraph from the t-th iterate, one is able to show that the marginal estimate yields the true marginal distribution, thus proving the claim.

It is important to note that due to the Hammersley-Clifford theorem, the statements in theorem 8 show that also for Bayesian networks whose DAG is a polytree, the belief propagation algorithm would converge in finitely many steps to the exact marginals. However, in networks with undirected cycles in their underlying undirected graph, the belief propagation algorithm provides only approximate inference [26]. This stems from the fact that when cycles exist, a node X is no longer guaranteed to d-separate its ancestors and descendants. Therefore, this creates ambiguity in the iterative message-passing, which means that messages can be propagated through multiple paths. One common way of handling such networks is by converting them to polytrees through edge deletions and edge additions [20]. Moreover, there is an extension of the classical belief propagation algorithm, called *loopy* belief propagation, which handles networks with cycles, but typically provides only approximate inference [28].

For our purposes, the provided adaptation of the belief propagation algorithm will suffice for sampling missing data values in networks with incomplete data. Namely, with reference to the BMA approach for inferring Gaussian Bayesian networks [13], we propose the following approach for sampling the missing values, thus leading to a proposed strategy of how one can adapt the approach to discrete networks. First, given an incomplete dataset, we initialize the parameters of the random variables involved in the network. By parameters, we mean the local distribution of the root/leaf nodes and the conditional probabilities for the other nodes. For discrete networks, conditional probabilities are typically represented using so-called *conditional probability tables* (CPTs) which specify the probability distribution of a variable for each possible combination of its parents' states [21] via tables. Each row in the table corresponds to a unique combination of the states of the parent variables and each column represents a possible state of the variable for which the CPT is defined. For instance, consider two binary random variables with the dependence relation given as $X \to Y$. We make 10 observations, some of which are missing, completely at random (see table 5.1).

Observation	Obs. 1	Obs. 2	Obs. 3	Obs. 4	Obs. 5	Obs. 6	Obs. 7	Obs. 8	Obs. 9	Obs. 10
X	0	0	0	NA	NA	NA	1	1	1	1
Y	0	1	1	1	0	0	0	0	1	NA

Table 5.1: 10 observations of two binary variables X and Y.

We use only the complete observations, and thus discard observations $4, 5, 6$ and 10. From the complete observations, we estimate $\mathbb{P}(X = 0) = \frac{1}{2}$ and $\mathbb{P}(X = 1) = \frac{1}{2}$. On the other hand, given $X \to Y$, for Y we obtain the CPT given in table 5.2.

	$Y = 0$	$Y = 1$
$X = 0$	$\mathbb{P}(Y = 0 \vert X = 0) = \frac{1}{3}$	$\mathbb{P}(Y = 1 \vert X = 0) = \frac{2}{3}$
$X = 1$	$\mathbb{P}(Y = 0 \vert X = 1) = \frac{2}{3}$	$\mathbb{P}(Y = 1 \vert X = 1) = \frac{1}{3}$

Table 5.2: The CPT of Y given the complete observations.

Having initialized the needed parameters via CPTs, the next step is to perform the belief propagation algorithm. For any variable X_i in the network, to avoid convergence issues due to potential undirected cycles, identify $MB(X_i)$, extract the subgraph containing

X_i and $MB(X_i)$, and perform the message-passing on X_i using only the extracted subgraph. This utilizes the fact that X_i is a node in a Bayesian network, thus satisfying the local Markov property, which, using theorem 1, means that all the information we need to perform inference on X_i is contained in $MB(X_i)$. This approach reduces computational complexity and avoids convergence issues related to undirected cycles. This finalizes the proposed approach.

We refer the reader to the appendix section, in which one can find the code for an implementation of the belief propagation algorithm for Bayesian networks in the programming language R. Note that the implementation in R relies upon the gRain and gRbase R packages, thus requiring the user to only specify the DAG structure, and the initial local and conditional probabilities in order to be able to run the code. Additionally, code for example usage is also provided.

6 Conclusion and Discussion

This thesis provides a formal introduction to the task of Bayesian network structure learning. In particular it demonstrates why structure learning is a challenging task, provides a comprehensive treatment of equivalence classes of directed acyclic graphs as a way of simplifying this task, and details different structure learning approaches with a detailed overview of the BDe score used in score-based approaches for discrete Bayesian networks. The main focus of this paper is on networks with background information and networks with incomplete information.

In terms of networks with background information, we formalize the idea of having a dataset with background information through the introduction of MPDAGs. As a starting point to the development of structure learning for networks with background information, this paper carefully discerns three important differences between MPDAGs and CPDAGs, and builds upon existing theory on CPDAGs to develop new theory on dependence relations in MPDAGs.

Regarding networks with incomplete information, this thesis provides an adaptation of the well-known belief propagation algorithm to Bayesian networks with the aim of utilizing this algorithm to sample missing values in an adaptation to discrete Bayesian networks of a newly-developed Bayesian model averaging approach for inferring Gaussian Bayesian networks from incomplete data. Moreover, it discusses convergence aspects of the algorithm, proposes an approach for sampling missing values based on the Markov blanket of a network node, and provides an implementation of the algorithm in the programming language R.

6.1 Further Research

The present paper and its results offer several possibilities for further research.

First, the work done on networks with background information serves as a theoretical starting point to developing structure learning algorithms for MPDAGs. Namely, by considering the differences between MPDAGs and CPDAGs, one could try to modify existing constraint-based approaches such as the PC algorithm to be able to apply it to MPDAGs. This is partially done in [29], where an algorithmic approach is developed for identifying causal relations in a MPDAG. Additionally, one could also attempt to study further potential differences between MPDAGs and CPDAGs, as the three observed differences in this paper are not guaranteed to be the only ones which are significant.

Moreover, inspired from the developed theory on CPDAGs, further research could also focus on continuing to develop theory related to dependence relations in MPDAGs. For instance, given the structural similarities between DAGs that lead to equivalence classes of DAGs, one could try to explore potential structural similarities between MPDAGs or a specific kind of MPDAGs. Something similar to this is done in [16], in the context of interventional essential graphs.

Second, when it comes to discrete networks with background information, one logically natural venue for further research is to implement the developed exposition of the belief propagation algorithm in Bayesian networks to the Bayesian model averaging (BMA) approach presented in [13]. In particular, one could inspect if the proposed approach for sampling missing values results in an adaptation for discrete networks that is as competitive as the original approach for Gaussian networks in terms of network reconstruction accuracy. Additionally, one could compare the performance of the BMA approach when using the belief propagation algorithm to sample missing values and when using data imputation to fill in the missing values. This comparison has already been considered for the EM algorithm [31].

6.2 Personal Reflection

Reflecting on the completed thesis work, the author is satisfied with the produced outcome. Namely, the author is under the impression that by delving deep into a topic not known to him beforehand, he was able to learn a lot, but more importantly, to have the chance to experience what professional mathematical research looks like. In particular, the author was able to experience the associated upturns and downturns when doing research.

One particular downturn came as a result of the author not being able to develop quality work regarding his initial project idea to work on establishing bijections between directed acyclic graphs. Additionally, when working on networks with background information, the author was under the initial impression that he stumbled upon an almost completely new line of research, only to discover later in his literature review, that such networks have already been researched to a certain extent, albeit under a different name than the one the author formulated.

On a positive note, the author successfully managed to cover two different and difficult topics from the field of Bayesian network structure learning. This is in turn due to the fact that the author did not allow for any last-moment obstacles, thus allowing himself to invest substantial effort and develop the two topics to a high enough level. Therefore, this achievement instills a strong feeling of accomplishment in the author, and motivates him to further develop himself with the goal of becoming a researcher in mathematics.

7 References

[1] N. Balov. “Consistent model selection of discrete Bayesian networks from incomplete data”. In: *Electronic Journal of Statistics* 7 (2013), pp. 1047–1077. DOI: 10.1214/13-EJS802.

[2] W. L. Buntine. “Chain graphs for learning”. In: *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*. 1995, pp. 46–54.

[3] D. M. Chickering. “Learning equivalence classes of Bayesian-network structures”. In: *Journal of Machine Learning Research* 2 (2002), pp. 445–498.

[4] D. M. Chickering. “Optimal structure identification with greedy search”. In: *Journal of Machine Learning Research* 3 (2002), pp. 507–554.

[5] R. G. Cowell et al. *Probabilistic Networks and Expert Systems*. Springer New York, NY, 1999. DOI: https://doi.org/10.1007/b97670.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin. “Maximum likelihood from incomplete data via the EM algorithm”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39 (1 1977), pp. 1–22. DOI: https://doi.org/10.1111/j.2517-6161.1977.tb01600.x.

[7] R. Diestel. *Graph Theory*. 5th ed. Springer Berlin, Heidelberg, 2017. DOI: https://doi.org/10.1007/978-3-662-53622-3.

[8] M. F. Eigenmann, P. Nandy, and M. H. Maathuis. “Structure learning of linear Gaussian structural equation models with weak edges”. In: *Proceedings of UAI 2017*. 2017.

[9] K. Engeland. “Cell cycle regulation: p53-p21-RB signaling”. In: *Cell Death & Differentiation* 29 (2022), pp. 946–960. DOI: https://doi.org/10.1038/s41418-022-00988-z.

[10] J. A. Gámez, J. L. Mateo, and J. M. Puerta. “Learning Bayesian networks by hill climbing: efficient methods based on progressive restriction of the neighborhood”. In: *Data Mining and Knowledge Discovery* 22 (2011), pp. 106–148. DOI: https://doi.org/10.1007/s10618-010-0178-6.

[11] D. Geiger and D. Heckerman. “Learning Bayesian networks: a unification for discrete and Gaussian domains”. In: *Proceedings of the 11th Annual Conference on Uncertainty in Artificial Intelligence (UAI-95)*. 1995, pp. 274–284.

[12] M. Grzegorczyk. “An introduction to Gaussian Bayesian networks”. In: *Systems Biology in Drug Discovery and Development* 662 (2010), pp. 121–147. DOI: https://doi.org/10.1007/978-1-60761-800-3_6.

[13] M. Grzegorczyk. “Being Bayesian about learning Gaussian Bayesian networks from incomplete data”. In: *International Journal of Approximate Reasoning* 160 (2023). DOI: https://doi.org/10.1016/j.ijar.2023.108954.

[14] J. M. Hammersley and P. Clifford. “Markov fields on finite graphs and lattices”. In: *Unpublished manuscript* 46 (1971).

[15] F. Harary and E. M. Palmer. *Graphical Enumeration.* Academic Press, 1973.

[16] A. Hauser and P. Bühlmann. “Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs”. In: *Journal of Machine Learning Research* 13 (2012), pp. 2409–2464.

[17] D. Heckerman, D. Geiger, and D. M. Chickering. “Learning Bayesian networks: the combination of knowledge and statistical data”. In: *Machine Learning* 20.3 (1995), pp. 197–243. DOI: 10.1023/A:1022623210503.

[18] A. Hyttinen, F. Eberhardt, and M. Järvisalo. “Do-calculus when the true graph is unknown”. In: *UAI’15: Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence.* 2015, pp. 395–404.

[19] D. Katz et al. “Size of interventional Markov equivalence classes in random DAG models”. In: *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS).* 2019.

[20] N. K. Kitson et al. “A survey of Bayesian network structure learning”. In: *Artificial Intelligence Review* 56 (2023), pp. 8721–8814. DOI: https://doi.org/10.1007/s10462-022-10351-w.

[21] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques.* MIT press, 2009.

[22] M. H. Maathuis, M. Kalisch, and P. Bühlmann. “Estimating high-dimensional intervention effects from observational data”. In: *The Annals of Statistics* 37.6A (2009), pp. 3133–3164.

[23] D. Madigan and J. York. “Bayesian graphical models for discrete data”. In: *International Statistical Review* 63.2 (1995), pp. 215–232. DOI: https://doi.org/10.2307/1403615.

[24] C. Meek. “Causal inference and causal explanation with background knowledge”. In: *Proceedings of UAI 1995.* 1995, pp. 403–410.

[25] É. de Panafieu and S. Dovgal. “Counting directed acyclic and elementary digraphs”. In: *Proceedings of the 32nd Conference on Formal Power Series and Algebraic Combinatorics (Online).* 2020.

[26] J. Pearl. “Reverend Bayes on inference engines: a distributed hierarchical approach”. In: *Proceedings of the Second AAAI Conference on Artificial Intelligence.* 1982, pp. 133–136.

[27] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Elsevier, 1988. DOI: https://doi.org/10.1016/C2009-0-27609-4.

[28] J. Pearl. "Graphical Models for Probabilistic and Causal Reasoning". In: *Quantified Representation of Uncertainty and Imprecision*. Dordrecht: Springer Netherlands, 1998, pp. 367–389. DOI: 10.1007/978-94-017-1735-9_12.

[29] E. Perkovic. "Identifying causal effects in maximally oriented partially directed acyclic graphs". In: *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*. Vol. 124. 2020.

[30] R. W. Robinson. "Counting labeled acyclic digraphs". In: *New Directions in the Theory of Graphs* (1973), pp. 239–273.

[31] A Ruggieri et al. "Hard and soft EM in Bayesian network learning from incomplete data". In: *Algorithms* 13 (12 2020). DOI: https://doi.org/10.3390/a13120329.

[32] T. Verma and J. Pearl. "Causal networks: semantics and expressiveness". In: *Machine Intelligence and Pattern Recognition* 9 (1990), pp. 69–76. DOI: https://doi.org/10.1016/B978-0-444-88650-7.50011-1.

[33] H. S. Wilf. *Generatingfunctionology*. Elsevier, 1990. DOI: https://doi.org/10.1016/C2013-0-11714-X.

[34] D. Williams. *Probability with Martingales*. Cambridge University Press, 1991.

[35] B. van der Zander, M. Liśkiewicz, and J. Textor. "Constructing separators and adjustment sets in ancestral graphs". In: *CI'14: Proceedings of the UAI 2014 Conference on Causal Inference: Learning and Prediction*. Vol. 1274. 2014, pp. 11–24.

8 Appendix

Section 8.1 provides sample code for the implementation of the belief propagation algorithm for Bayesian networks in the programming language R.

8.1 Implementation of Belief Propagation in R

```r
# Load necessary libraries

install.packages("gRbase")
library(gRbase)
install.packages("gRain")
library(gRain)

# Function to create a Bayesian network from a given DAG and CPTs
create_bayesian_network <- function(dag_formula, cpt_list) {
  # Define the network structure
  dag <- dag(dag_formula)

  # Compile the CPTs into a probability table
  plist <- compileCPT(cpt_list)

  # Create a grain object (graphical independence network)
  network <- grain(plist)

  return(network)
}

# Function to perform belief propagation and update beliefs
perform_belief_propagation <- function(network) {
  # Propagate the beliefs
  network <- propagate(network)

  # Update the beliefs for all nodes
  beliefs <- querygrain(network)

  return(beliefs)
}

# Function to print the belief propagation results
print_beliefs <- function(beliefs) {
  for (node in names(beliefs)) {
    cat("Beliefs for", node, ":\n")
    print(beliefs[[node]])
    cat("\n")
  }
}

# Example usage:

# Define the network structure using a formula
# Example: A -> B, A -> C
dag_formula <- ~A + D + A:B + A:C

# Define the conditional probability tables (CPTs)
cpt_A <- cptable(~A, values = c(0.7, 0.3), levels = c("True", "False"))
cpt_B_A <- cptable(~B | A, values = c(0.8, 0.2, 0.1, 0.9),
levels = c("True", "False"))
cpt_C_A <- cptable(~C | A, values = c(0.6, 0.4, 0.3, 0.7),
levels = c("True", "False"))

# Combine the CPTs into a list
cpt_list <- list(cpt_A, cpt_B_A, cpt_C_A)

# Create the Bayesian network
net <- create_bayesian_network(dag_formula, cpt_list)

# Perform belief propagation
beliefs <- perform_belief_propagation(net)

# Print the beliefs for all nodes
print_beliefs(beliefs)
```