On the restrictions of Pair-Copula Bayesian Networks for integration-free computations

by

Niels Horsman

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Friday November 17, 2023 at 10:00 AM.

Student number:	4480864	
MSc programme:	Applied Mathematics	
Specialization:	Stochastics	
Thesis committee:	Dr. D. Kurowicka,	TU Delft, supervisor
	Dr. A. Derumigny,	TU Delft, co-supervisor
	Dr. G. F. Nane,	TU Delft

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft Delft University of Technology

Abstract

The pair-copula Bayesian network (PCBN) is a Bayesian network (BN) where the conditional probability functions are modeled using pair-copula constructions. By assigning bivariate conditional copulas to the arcs of the BN, one finds a proper joint density which can flexibly model all kinds of dependence structures. It is a known problem that the PCBN may require numerical integration to perform computations such as sampling and likelihood-inference. To address this issue we propose novel restrictions on the graphical structure and assignment of copulas such that integration will not be required. The resulting restricted PCBN offers significant computational benefits. We establish how to estimate and conduct a structure search for the restricted PCBN. A simulation study shows that a restricted PCBN is able to model non-Gaussian dependence structures more accurately than the widely used Gaussian Bayesian network.

Glossary

Notation	Description	Page List
V	Nodes	page 8
E	Edges	page 8
$\mathcal{G}$	Graph	page 8
$ad(A)$	Adjacency-set corresponding to A	page 11
$pa(v)$	Parents of v	page 11
$ch(v)$	Children of v	page 11
$an(v)$	Ancestors of v	page 11
$de(v)$	Descendants of v	page 11
$pa(v \downarrow w)$	Parents of v lower than w according to $<_v$	page 12
$pa(v \uparrow w)$	Parents of v higher than w according to $<_v$	page 12
$\underline{pa}(v \downarrow w)$	$pa(v \downarrow w) \sqcup \{v\}$	page 12
$\overline{pa}(v \downarrow w)$	$pa(v \downarrow w) \sqcup \{w\}$	page 12
$d-sep_{\mathcal{G}}$	d-separation in graph $\mathcal{G}$	page 12
$\not{d-sep}_{\mathcal{G}}$	No d-separation in graph $\mathcal{G}$	page 13
$\boldsymbol{\theta}$	Parameter vector	page 18
$\mathcal{D}$	Data	page 18
$\mathcal{O}$	Set of orders	page 33
O_v^k	Partial order corresponding to v of cardinality k	page 51
$B(O_v^k)$	Smallest B-set strictly larger than O_v^k	page 51
$Poss.Cand(O)$	Possible candidates of O	page 60
$Poss.Cand_{Ind}(O)$	Possible candidates of O by independence	page 61
$Poss.Cand_{In}(O)$	Possible candidates of O by an incoming arc	page 61
$Poss.Cand_{Out}(O)$	Possible candidates of O by an outgoing arc	page 61
$\mathfrak{P}$	General property of a trail	page 72
$TRAILS(X, Y \mid Z)$	Set of trails from X to Y activated by Z	page 86
$ConvCon(T)$	Converging connections in trail T	page 86
$\mathfrak{A}$	Particular property of a subset of nodes	page 87
$\mathcal{P}$	Collection of densities	page 113
$\boldsymbol{\Theta}$	Parameter space	page 113

Contents

1

Introduction

A **Bayesian network** (BN) is a graphical model of high dimensional random vectors. BNs are composed of a direct acyclic graph (DAG) where the nodes correspond to the univariate random variables and the arcs encode the dependence structure of these variables. An extremely attractive feature of these models is their ability to represent complex dependencies in an intuitive way. This is especially important for practitioners, who can easily describe their problems and rely on a solid mathematical theory and many computer implementations of BNs. These models have been applied in a wide variety of fields including medicine, finance, genetics, and forensic science ([29]).

For an in-depth introduction to Bayesian networks, it is advised to read [27]. Other books concerning BNs are [19], [15], and [8]. Let us start with a simple example.

Suppose that we wish to model the stock price of the following car manufacturers; Tesla, Ford, General Motors, Toyota, Honda, and Nissan. A plausible assumption is that the largest companies in terms of current stock price, Tesla and Toyota (as of 29-08-2023), are dependent on each other. Thus, in a graph, there is a direct arc Tesla→Toyota. We will assume that the stock prices of other American companies are influenced by the stock price of Tesla, and the stock prices of Japanese companies are influenced by Toyota, giving us the DAG in Figure 1.1.

Relationships between all variables in the DAG are included. For example, Tesla→Toyota→Nissan implies that Nissan depends on Toyota, which in turn is dependent on Tesla, and hence the stock price of Nissan depends on the stock price of Tesla. Intuitively, the stock price of Tesla influences the Nissan stock price "through" Toyota. If we were to know the stock price of Toyota, then the Tesla stock would not influence the price of the Nissan stock anymore. Nissan and Tesla are conditionally independent given Toyota. For more examples of how conditional independence is represented in the BN see [27, ch. 3], [19, ch. 3] and [15, ch. 2].

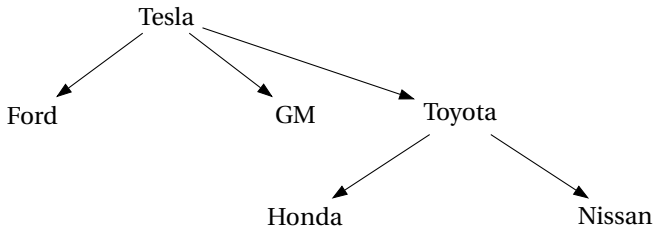


Figure 1.1: Simple BN for prices of stocks of six car companies, where General Motors is abbreviated by GM.

A key property of the BN is that the conditional independencies encoded by the graph, $\mathcal{G} = (V, E)$, allow for a factorization of the joint probability into a product of conditional probabilities:

$$f_V(\mathbf{x}_V) = \prod_{v \in V} f_{v|pa(v)}(x_v|\mathbf{x}_{pa(v)}).$$

Here, f_V is the joint density over all the nodes, and $f_{v|pa(v)}$ is the conditional density of a node v given its parents $pa(v)$, where node w is said to be a parent of node v if the graph contains the arc $w \to v$. In the example of the six stocks, the joint probability density function (PDF) may be factorized by the formula below where arguments are suppressed.

$$f_V = f_{Tesla} \cdot f_{Ford|Tesla} \cdot f_{GM|Tesla} \cdot f_{Toyota|Tesla} \cdot f_{Honda|Toyota} \cdot f_{Nissan|Toyota}.$$

This factorization allows us to represent a multi-dimensional problem (for all nodes) into a set of lower-dimensional problems (a node and its parents). Instead of estimating a complex function f_V, we are tasked with finding simpler functions of the form $f_{v|pa(v)}$.

BNs can be used to represent purely discrete, purely continuous, or mixed (discrete and continuous with more restrictions, see [25]) distributions.

This thesis will only consider the BNs with continuous nodes. Given a graphical structure, we can additionally assume that all conditional PDFs are Gaussian which are linear functions of parents and the constant variances:

$$X_v \sim N(\mu_v + \sum_{w \in pa(v)} \phi_w x_w, \sigma_v^2).$$

In this case, we get the **Gaussian Bayesian network** (GBN). GBNs are well-studied (see [19], [22], [8] and [33]), have been implemented (for example in the R package bnlearn, see [32]) and found many applications.

An important property of the GBN is that its joint density is multivariate Gaussian ([34]). Hence, conditioning on a subset of the variables in a GBN will again provide a Gaussian joint density. This allows us to find analytical expressions of densities such as $f_{Nissan|Tesla}$ or $f_{Ford,Tesla|Honda,Nissan}$, which is convenient when performing inference and propagating the evidences.

Although the GBN provides clear computational benefits, it also has an obvious disadvantage. These models can only represent Gaussian dependence structures and all nodes have Gaussian marginal distributions. To highlight this, we consider a smaller version of the car stock example concerning the Tesla and Toyota stocks. In particular, we investigate a simulated data set containing observations of the logarithmic returns of said stocks where the dependence structure is non-Gaussian.

Consider the data set in Figure 1.2. This data was generated[1] such that $Tesla \sim N(0, 0.02^2)$ and $Toyota \sim N(0, 0.025^2)$ with a Kendall's τ equal to 0.7. We can observe the asymmetric behavior of the data, in the

[1] The data was generated such that it roughly resembles an actual data set of stock returns.

sense that observations in the bottom-left corner are more correlated than the observations in the top-right corner. Such a property cannot be captured by the multivariate Gaussian distribution, and therefore cannot be captured by a GBN. If we fit a bivariate Gaussian to the data and simulate from the fitted model, we obtain the scatter plot displayed in Figure 1.3. Indeed, this generated data looks very different as compared to the original data, even though the marginal distributions in both models are Gaussian.

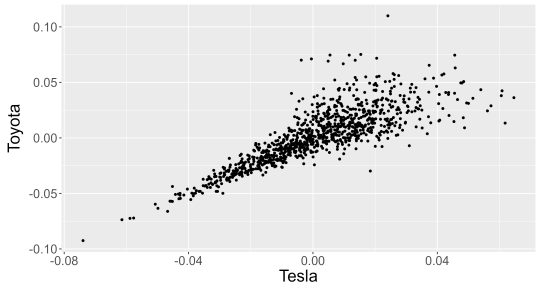


Figure 1.2: Simulated data for the log-returns of Tesla and Toyota.

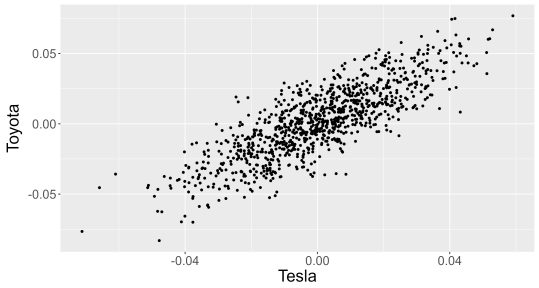


Figure 1.3: Data simulated from a bivariate Gaussian fitted to the data in Figure 1.2.

Suppose now that the log-returns of Toyota are exponentially distributed[2] with rate 1, and let the dependence structure be as in the data set above [3] . The scatter plot of the obtained data is plotted in Figure 1.4. Fitting a bivariate Gaussian distribution to this data leads to the simulated data set displayed in Figure 1.5. Again, it is clear that a joint Gaussian distribution is not suitable to model this data set.

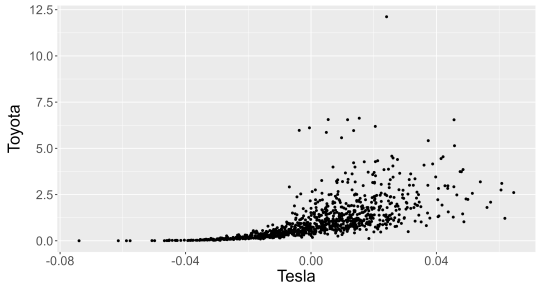


Figure 1.4: Simulated data for the log-returns of Tesla and Toyota with $Toyota \sim Exp(1)$.

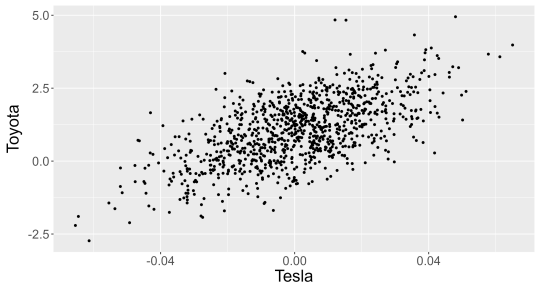


Figure 1.5: Data simulated from a bivariate Gaussian fitted to the data in Figure 1.4.

The distribution of a multivariate random vector is determined by the univariate marginal distributions and the dependence function, called **copula** of the random vector. The copula is the distribution function on the unit hypercube with uniform marginal distributions. To see which copula should be used to model the dependence of the data, this data can be transformed into the uniform distribution (uniform scale) by applying the cumulative distribution functions to the margins. The data set obtained by transforming Figures 1.2 and 1.4 to the uniform scale is displayed in Figure 1.6.

[2] Of course, this assumption is not very realistic.

[3] Meaning that the data sets were sampled from the same copula, Clayton with Kendall's τ equal to 0.7, which will be defined in the next two paragraphs.

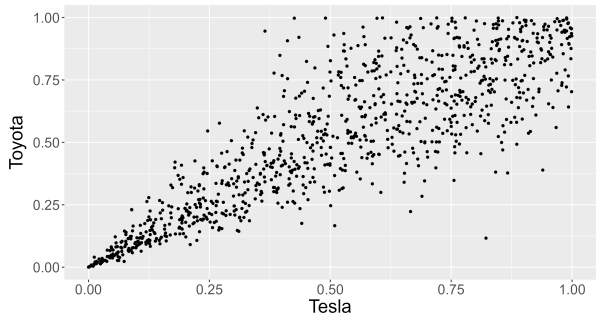


Figure 1.6: Scatter plot of the data in the uniform scale for the Tesla and Toyota stocks.

There are many copula families with different properties that are able to model different dependence structures for bivariate distributions. Figure 1.7 displays data sets simulated from 5 different copulas; Gaussian, Clayton, Gumbel, Frank, and Joe. Even though all data sets were simulated with Kendall's τ equal to 0.7, the dependence structures are different. For example, the Clayton, Gumbel, and Joe copulas are asymmetric whereas the Gaussian and Frank copulas are symmetric. The GBN is not be able to capture asymmetric dependence between random variables, as we have seen for the data set in Figure 1.2. It would be very interesting to be able to incorporate the possibility of more flexible dependence into the BN than the GBN.

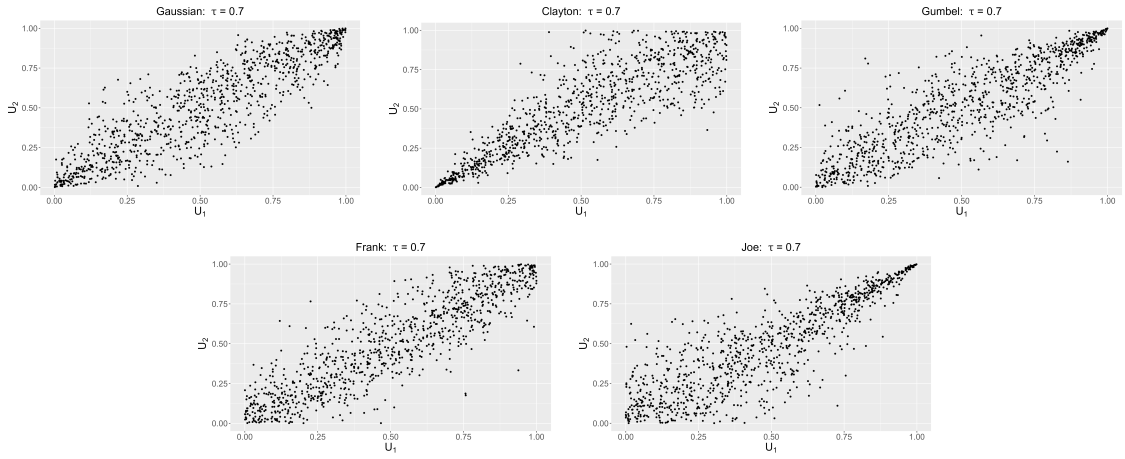


Figure 1.7: Scatter-plots of 1000 samples from different copulas simulated with $\tau = 0.7$

Instead of assuming that conditional PDFs $f_{v|pa(v)}$ are Gaussian, they can be decomposed as a product of bivariate copulas. Such a decomposition is called a **pair-copula Bayesian network** (PCBN), and has been proposed by [21], and was further investigated in [11], [12], [4] and [3].

In PCBNs the copulas must be assigned in a specific manner. If a node v has more than one parent, then a total order $<_v$ is defined over the parental set $pa(v)$. The order provides us with an ordered set of parents $(p_1,\dots,p_n) := pa(v)$ such that $i < j$ implies that $p_i <_v p_j$. The copulas are then assigned as follows; the arc from the first parent p_1 to v is assigned the copula $c_{p_1 v}$, the arc from the second parent p_2 to v is assigned the copula $c_{p_2 v|p_1}$, the arc from the third parent p_3 to v is assigned the copula $c_{p_3 v|p_1,p_2}$, etc. Thus, each arc $w \to v$ is assigned the copula $c_{wv|pa(v\downarrow w)}$, where $pa(v \downarrow w)$ is the set consisting of all parents of v, which are lower than w according to $<_v$. It has been shown in [21] that such an assignment of copulas to the arcs of a BN will provide us with a proper joint density function. Furthermore, if all copulas and margins in the PCBN are Gaussian, then the PCBN is equivalent to the GBN.

Parental orders for all nodes are collected in the set $\mathcal{O} := \{<_v; v \in V\}$. A PCBN consists of the tuple $(\mathcal{G},\mathcal{O})$ where the graph determines direct dependencies and independencies between elements of the random vector, and the parental orders indicate (conditional) copula assignments. Additionally, the copula types have to be determined and their parameters estimated as well as the marginal distributions of all nodes. PCBNs are much more expensive computationally as compared to GBNs, but we can represent a much more flexible set of dependencies in this way ([4]).

In Chapter 3, it will be seen that the joint density of a PCBN corresponding to a multivariate random vector $U_V := (u_v)_{v\in V}$ with uniform margins, i.e. $u_v \sim Unif(0,1)$ can be written as

$$c(\mathbf{u}_V) = \prod_{v\in V}\prod_{w\in pa(v)} c_{wv|pa(v\downarrow w)}\big(u_{w|pa(v\downarrow w)}, u_{v|pa(v\downarrow w)}\big).$$

To compute $c(\mathbf{u}_V)$, the terms $u_{w|pa(v\downarrow w)}$ and $u_{v|pa(v\downarrow w)}$ are required. These conditional margins may need to be computed with integration. For example, in [3], the graph in Figure 1.8 was found to require integration for any assignment of parental orders $\mathcal{O}$. Note that for this graph we have two possible choices of orders for node 4; $2 <_4 3$ and $3 <_4 2$. Suppose that we pick $2 <_4 3$. Then, the joint density can be factorized by

$$c(u_1,u_2,u_3,u_4) = c_{12}(u_1,u_2)\cdot c_{13}(u_1,u_3)\cdot c_{24}(u_2,u_3)\cdot c_{34|2}(u_{3|2},u_{4|2}).$$

Here, the conditional margin $u_{3|2}$, which depend on u_2 and u_3 must be computed using integration, as will be demonstrated in Chapter 3.

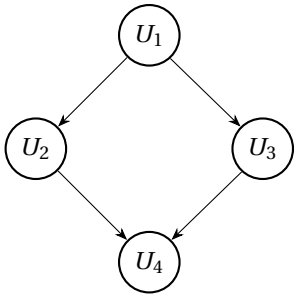


Figure 1.8: Graphical structure for which the joint density will require integration for every assignment of copulas.

The one-dimensional integrals can still be computed numerically in a reasonable amount of time with good accuracy. However, graphs can be constructed for which integration in an arbitrary dimension is required. Hence, the PCBN in its current state is not scalable in general to larger graphs. Therefore, it would be beneficial to find necessary and sufficient conditions under which one can evaluate the density for PCBNs without the need to integrate, giving us the first goal of this thesis.

Goal 1. Construct a subclass of DAGs for which we can choose an assignment of copulas such that the computation of the joint density does not require integration.

Even if we have a DAG for which there exists a suitable assignment of (conditional) copulas, we must still assign the copulas in an intelligent manner. Therefore, we will construct an algorithm that finds a set of orders $\mathcal{O}$ given a restricted graph.

Goal 2. Construct an algorithm that assigns copulas to the arcs of a restricted graph such that the joint density does not require integration.

We will prove that this algorithm is always able to find a suitable $\mathcal{O}$ given a restricted DAG. This requires a

substantial amount of work, compelling us to prove many other results which take up a large part of this thesis.

Goal 3. Prove results concerning restricted DAGs which are needed to show that the steps in the algorithm are necessary and sufficient to find a suitable $\mathcal{O}$.

With the restrictions on the graph and the algorithm in place, we define a restricted PCBN model for which we can estimate parameters without expensive integration. Furthermore, we illustrate how to fit such restricted PCBNs to a data set. We start our presentation by finding the parameters of a fixed graph $\mathcal{G}$, hence we only have to find the optimal assignment of copulas and parameters of the assigned copula families.

Goal 4. Establish how to fit a restricted PCBN to a data set for a fixed graph.

In real-world applications the graph is generally not known. Hence, we are also tasked with finding the graphical structure. This structure can be chosen by means of expert judgment. However, if the sufficiently good data set is available it will be of interest to construct the graph given a particular data set. This process is called structure learning.

Structure learning algorithms can be roughly divided into two categories; score-based and constraint-based algorithms [19, ch. 18]. We will concentrate on a score-based algorithm. Here, the graph is assigned a score, most of the time a likelihood-based score function ([18]). The algorithm will move through the search space of graphs making slight adjustments to the current graph at each iteration. The adjustments are chosen such that the resulting graph will have a higher score than the current graph. The end result will be a graph for which no adjustment will provide a higher score. In particular, we will be implementing the Hill climbing algorithm which is one of the most elementary structure learning algorithms ([31]). Despite its simplicity, the algorithm is remarkably competitive ([33]).

Goal 5. Illustrate how to apply a structure learning algorithm to find the optimal graph corresponding to a restricted PCBN given a data set.

An outline of the thesis is as follows. First, all basic concepts which are necessary for the study of PCBNs are discussed in Chapter 2. This chapter will not contain any novel results with the exception of Theorem 2.25. Hereafter, we introduce the PCBN in Chapter 3, and construct the subclass of graphs not needing integration. The algorithm which assigns the copulas to the found restricted graph will be presented in Chapter 4. Chapter 5 contains the necessary results concerning restricted graphs which are needed to prove the necessity and sufficiency of the steps in the aforementioned algorithm. Finally, we establish how to perform estimation and structure learning on the subclass of restricted PCBNs in Chapter 6.

2

Preliminaries

This chapter contains an exposition of basic concepts used later in this thesis. The necessary notation is established and some useful results available in the literature are given. The following subjects are treated; graph theory, BNs, GBNs, structure learning in GBNs and copulas.

We will not provide background information regarding basic probability theory. This means that all elementary topics are assumed to be known by the reader. For a good introduction we refer to [10]. All used notation will be straightforward and inline with the regular expressions found throughout the literature. A few examples are the following.

- Univariate random variables are denoted with capitol letters; X, Y and Z, and multivariate random variables are in bold; $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$. All are assumed to be real-valued[1].
- Observations of X are denoted by a small x, and similarly $\mathbf{x}$ is an observation of $\mathbf{X}$.
- The probability density function (PDF) of X is denoted by f_X, where the subscript is omitted if X is clear from context.
- The cumulative distribution (CDF) function of X is denoted by F_X, where the subscript may be omitted.
- Independence between two random variables X and Y is denoted by $X \perp\!\!\!\perp Y$. If they are dependent, then we write $X \not\perp\!\!\!\perp Y$.
- Conditional independence is denoted by $X \perp\!\!\!\perp Y \,|\, Z$ which means that X and Y are independent given Z.

Finally, we assume that all conditional densities exist; i.e. all joint densities are assumed to be positive.

[1] Sets are also denoted by capitol letters; e.g. A, K or X. In this thesis, sets will often be subsets of nodes in a graph, but they may also consist of univariate random variables.

2.1. Graph theory

The idea behind Bayesian networks is to represent key characteristics of a probabilistic model in an understandable graphical structure. Therefore, the usage of graph theory will be required throughout this report. All necessary definitions will be discussed in this section.

Graphs are mathematical objects which consist of nodes which may or may not be connected by edges. The nodes and edges can represent many things, in the context of Bayesian networks the nodes will be associated with univariate random variables and the edges between them will imply dependence.

Definition 2.1 (Graph)**.** Let $V \neq \emptyset$ be a finite set and let $E \subseteq \{(v,w); v,w \in V, v \neq w\}$. The pair $\mathcal{G} = (V,E)$ is called a ***graph*** with vertices/nodes V and edges E. Furthermore, we assume that if E contains an edge (w,v), then the edge (v,w) is not in E.

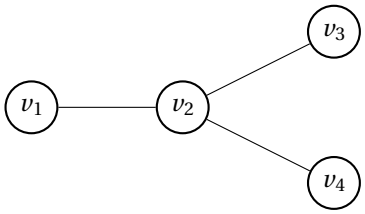


Figure 2.1: Example of a simple graph.

The graph in Figure 2.1 shows four nodes which are linked by three edges.

Definition 2.2 (Directed/undirected edge)**.** Let $\mathcal{G} = (V,E)$ be a graph and $(w,v) \in E$. The set E can contain sets $\{w,v\}$ and ordered pairs (w,v).

- An ordered pair (w,v) is called a ***directed edge***, denoted by $w \to v$.
- A set $\{w,v\}$ is called an ***undirected edge***, denoted by $w - v$.

If an edge $w \to v$ is not present in E, then we write $w \not\to v$.

A directed edge is also referred to as an arc. The undirected edges will be represented by lines and the directed edges by arrows. For example, if the edge between nodes v_2 and v_3 in Figure 2.1 is to be replaced by a directed edge $v_2 \to v_3$, the graph below is obtained.

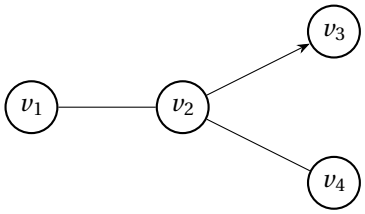


Figure 2.2: Example of a simple graph with one directed edge.

Note that Figure 2.2 contains a mixture of directed and undirected edges. Such graphs will not be used in this thesis. All graphs will only consist of either directed or undirected edges. For example, the graph corresponding to a Bayesian network only has directed edges. Figures 2.3a and 2.3b show an undirected and a directed graph respectively.

Definition 2.3 (Directed/undirected graph)**.** Let $\mathcal{G} = (V, E)$ be a graph.

- If E only contains directed edges, then $\mathcal{G}$ is a ***directed graph***.
- If E only contains undirected edges, then $\mathcal{G}$ is an ***undirected graph***.

(a) Undirected graph

(b) Directed graph

Figure 2.3: Example of an undirected and directed graph.

Each directed graph is associated with an undirected graph called a skeleton, which is obtained by removing the directions of the arcs. These undirected graphs will prove to be useful in revealing certain properties of Bayesian networks. For example, the graph in Figure 2.3a is the skeleton of the graph in Figure 2.3b.

Definition 2.4 (Skeleton)**.** Let $\mathcal{G} = (V, E)$ be a directed graph. Then, the ***skeleton*** of $\mathcal{G}$ is defined as the undirected graph obtained by replacing each arc $w \to v \in E$ with an undirected edge $w - v$. The skeleton is denoted by $S(\mathcal{G}) = (V, S(E))$.

The directed graph in Figure 2.3b contains the arcs $v_1 \to v_2$ and $v_2 \to v_4$. Thus, it is possible to travel from node v_1 to node v_4 via the two aforementioned arcs. Such a traversal is called a path. For example, the graphs in Figures 2.3a and 2.3b contain the paths $v_1 - v_2 - v_3$ and $v_1 \to v_2 \to v_4$, respectively.

Definition 2.5 (Path)**.** Let $\mathcal{G} = (V, E)$ be a graph. A ***path*** is a sequence of nodes $(v_1, v_2, \ldots, v_n)$ such that $\{v_1, v_2, \ldots, v_n\} \subseteq V$ and $\{(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n)\} \subseteq E$ for some integer $n > 0$ called the length of the path. Paths which only contain directed/undirected edges are denoted distinctly:

- Directed edges: $v_1 \to v_2 \to \cdots \to v_n$.
- Undirected edges: $v_1 - v_2 - \cdots - v_n$.

Note that Figure 2.3b contains no path from node v_1 to node v_3. Traveling from v_1 to v_3 would only be possible by violating the directions of the arcs. Such traversals are called trails. For example, the graph in Figure 2.3b contains the trail $v_1 \to v_2 \leftarrow v_3$.

Definition 2.6 (Trail)**.** Let $\mathcal{G} = (V, E)$ be a directed graph. A ***trail*** is a sequence of nodes $(v_1, v_2, \ldots, v_n)$ such that $v_1 -, v_2 - \cdots - v_n$ forms a path in the skeleton of $\mathcal{G}$.

When considering a general graph, the direction of an arc may not be specified, meaning that an arc between v_i and v_{i+1} can either point towards the left or right, denoted by $v_i \rightleftharpoons v_{i+1}$. If the direction of the arcs is not specified, a trail is denoted by $v_1 \rightleftharpoons v_2 \rightleftharpoons \cdots \rightleftharpoons v_n$. If the direction of the arc between v_i and v_{i+1} is known, we may replace $v_i \rightleftharpoons v_{i+1}$ by $v_i \to v_{i+1}$ or $v_i \leftarrow v_{i+1}$, depending on the direction.

Each node (v_i) not corresponding to an end-point of the trail has two neighbours (v_{i-1} and v_{i+1}). The arcs between the neighbours and the node itself can both point in different directions. For instance, consider the trail $v_1 \to v_2 \leftarrow v_3$ from Figure 2.3b. Here, both v_1 and v_3 point towards v_2. Such a graphical structure is called a converging connection. We define three distinct types of connections.

Definition 2.7 (Connections)**.** Let $\mathcal{G} = (V, E)$ be a directed graph and $v_1 \rightleftharpoons \cdots \rightleftharpoons v_n$ a trail in $\mathcal{G}$. For an integer $i \in \{2, \ldots, n-1\}$ we say that $v_{i-1} \rightleftharpoons v_i \rightleftharpoons v_{i+1}$ is a

- ***serial connection*** if $v_{i-1} \leftarrow v_i \leftarrow v_{i+1}$ or $v_{i-1} \rightarrow v_i \rightarrow v_{i+1}$,
- ***diverging connection*** if $v_{i-1} \leftarrow v_i \rightarrow v_{i+1}$,
- ***converging connection*** if $v_{i-1} \rightarrow v_i \leftarrow v_{i+1}$.

The node v_i is referred to as a serial, diverging or converging node, respectively.

It will be of particular interest if a trail contains nodes which are not located next to each other in the sequence of the trail, but have an arc between them. Such arcs are called chords. For example, the trail $v_1 \rightarrow v_2 \leftarrow v_3 \leftarrow v_4$ in Figure 2.3b contains the chord $v_2 \rightarrow v_4$.

Definition 2.8 (Chord)**.** Let $\mathcal{G} = (V, E)$ be a directed graph and $v_1 \rightleftharpoons \ldots \rightleftharpoons v_n$ a trail in $\mathcal{G}$. An arc between non-consecutive nodes in the trail is referred to as a ***chord***. That is, for $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, i-2, i+2, \ldots, n\}$ the arcs $v_i \rightarrow v_j$ and $v_j \rightarrow v_i$ are chords.

Taking a subset of the nodes and arcs of a graph produces a smaller graph called a subgraph.

Definition 2.9 (Subgraph)**.** Let $\mathcal{G} = (V, E)$ be a graph. Then, $\mathcal{G}' = (V', E')$ is a ***subgraph*** of $\mathcal{G}$ if

- $V' \subseteq V$,
- $E' \subseteq E$ and for all arcs $w \rightarrow v \in E'$ the nodes w and v are in V'.

If the arc set E' of a subgraph $\mathcal{G}' = (V', E')$ contains all arcs between nodes in V' in the original graph $\mathcal{G}$, then it is said to be induced by the subset V'.

Definition 2.10 (Induced subgraph)**.** Let $\mathcal{G} = (V, E)$ be a graph and $K \subseteq V$ a subset of nodes. Then, $\mathcal{G}(K) = (K, E(K))$ is a subgraph ***induced*** by K with $E(K) := \{w \rightarrow v;\ w \rightarrow v \in E,\ w, v \in K\}$.

A special type of path is one which follows a circular pattern. These paths are called cycles. For example, if we reverse the arc $v_3 \rightarrow v_4$ in Figure 2.3b, resulting in the graph in Figure 2.4, the cycle $v_2 \rightarrow v_4 \rightarrow v_3 \rightarrow v_2$ is found.

Definition 2.11 (Cycle)**.** Let $\mathcal{G} = (V, E)$ be a directed/undirected graph. A directed/undirected path in $\mathcal{G}$ is a ***cycle*** if it starts and ends with the same node and if it is of length at least 3.

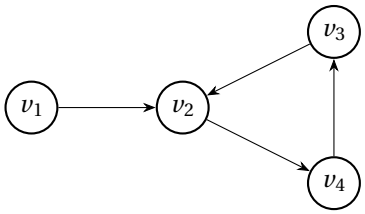


Figure 2.4: A cyclic graph.

Definition 2.12 (Acyclic)**.** A graph $\mathcal{G} = (V, E)$ is ***acyclic*** if it does not contain any cycles.

For example, the graph in Figure 2.3b is acyclic whereas the graph in Figure 2.4 is not. Graphs corresponding to Bayesian networks are both directed and acyclic. Thus, it is convenient to construct a definition which

merges the two properties.

Definition 2.13 (Directed acyclic graph)**.** A graph $\mathcal{G} = (V, E)$ is a ***directed acyclic graph*** (DAG) if it is directed and acyclic.

Another useful term is the concept of adjacency, which provides information about the direct connections in a graph.

Definition 2.14 (Adjacency)**.** Let $\mathcal{G} = (V, E)$ be a graph. Nodes $w, v \in V$ are ***adjacent*** if (w, v) or (v, w) is in E. Furthermore, we define the ***adjacency set*** corresponding to a subset $A \subseteq V$ by

$$ad(A) := \{w \in V \setminus A; \exists a \in A : (w, a) \in E \text{ or } (a, w) \in E\}.$$

In a directed graph, the adjacency of two nodes v and w reveals that the graph contains either $v \to w$ or $w \to v$. This warrants definitions which provide insight into the relation between adjacent nodes.

Definition 2.15 (Parents and children)**.** Let $\mathcal{G} = (V, E)$ be a directed graph. For each arc $w \to v \in E$ the node w is said to be the ***parent*** of v and v is said to be the ***child*** of w. For a node $v \in V$ the sets containing all its parents and children are denoted by $pa(v)$ and $ch(v)$ respectively.

For example, in Figure 2.3b, v_2 is the child of v_1, and nodes v_2 and v_3 are the parents of v_4.

It has been seen that two nodes can be indirectly linked by a path, this give rise to a definition which describes these indirectly connected nodes. .

Definition 2.16 (Ancestors and descendants)**.** Let $\mathcal{G} = (V, E)$ be a directed graph and $w, v \in V$. If there exists a path from w to v, then w is said to be the ***ancestor*** of v and v is said to be the ***descendant*** of w. For a node $v \in V$ the sets containing all its ancestors and descendants are denoted by $an(v)$ and $de(v)$, respectively.

For a directed graph which models a dependence structure, special attention must be paid to cases where two arcs point towards the same node; i.e. converging connections. In the literature these constructions are often called v-structures, where a special distinction is made between coupled and uncoupled v-structures. Examples of graphs with a coupled and uncoupled v-structure are displayed in Figure 2.5. In this thesis, we will use the the definitions converging connection and uncoupled v-structure interchangeably.

Definition 2.17 (v-structures)**.** Let $\mathcal{G} = (V, E)$ be a directed graph. Three nodes u, v and w form a ***v-structure*** around v if $u \to v$ and $w \to v$ are both in E. The v-structure is called ***coupled*** if u and w are adjacent and it is called ***uncoupled*** if they are not adjacent.

(a) Uncoupled v-structure.

(b) Coupled v-structure.

Figure 2.5: Example of both types of v-structures around node 3.

It will be desirable to order the nodes of a graph. Therefore, the definition of a total order on an arbitrary set K is introduced.

Definition 2.18 (Total order)**.** The binary relation $<$ on a set K is a total order if for all $a, b, c \in K$ the following

conditions are satisfied:

1. $a \nless a$.
2. $a < b \Rightarrow b \nless a$.
3. $a < b$ and $b < c \Rightarrow a < c$.
4. $a \neq b \Rightarrow a < b$ or $b < a$.

By using the directions of arcs in a DAG, we can define a particular total order on the nodes of the graph.

Definition 2.19 (Well-ordering)**.** Let $\mathcal{G} = (V, E)$ be a DAG. A total order $<$ on V is called a ***well-ordering*** of $\mathcal{G}$ if

$$\forall a, b \in V : a \to b \in E \Rightarrow a < b.$$

Note that the well-ordering of a DAG is not unique. For example, in Figure 2.5a both $v_1 < v_2 < v_3$ and $v_2 < v_1 < v_3$ are well-orderings. For all nodes v in a DAG, we will also need to define a total order on the parents of v. Therefore, the concept of parental order is defined.

Definition 2.20 (Parental order)**.** Let $\mathcal{G} = (V, E)$ be a directed graph and $v \in V$ be a node with $\left|pa(v)\right| > 1$. A ***parental order*** of v is a total order on the set $pa(v)$ denoted by $<_v$. Furthermore, for all $w \in pa(v)$, the set of parents of v strictly up to w is defined by

$$pa(v \downarrow w) := \{z \in pa(v);\, z <_v w\}.$$

Moreover, the set of parents of v strictly after w is defined as

$$pa(v \uparrow w) := \{z \in pa(v);\, w <_v z\}.$$

In the same way, we define

$$\underline{pa}(v \downarrow w) := pa(v \downarrow w) \sqcup \{v\} \text{ and } \quad \overline{pa}(v \downarrow w) := pa(v \downarrow w) \sqcup \{w\}.$$

Remark 2.21. The two definitions $\overline{pa}(v \downarrow w)$ and $\underline{pa}(v \downarrow w)$ are needed later in this thesis. The motivation behind the notation is that $pa(v \downarrow w)$ has corresponding arc $w \to v$, where w is the parent and v the child. In all graphs, parents are typically displayed above children. Therefore, when including the parent w, we use an overline ($\overline{pa}$) while an underline is used ($\underline{pa}$) to include the child v.

Another way to think about is that an overline implies the inclusion of the right node (which is w) in $pa(v \downarrow w)$, whereas an underline implies the inclusion of the left node (which is v).

An important concept for directed graphs is that two subsets of nodes can be connected through trails. These trails can be either blocked or activated given another subset.

Definition 2.22 (d-separation)**.** Let $\mathcal{G} = (V, E)$ be a directed graph and let $X, Y, Z \subseteq V$ be disjoint. Then, Z is said to ***d-separate*** X and Y in $\mathcal{G}$, denoted by $d - sep_{\mathcal{G}}(X, Y \,|\, Z)$, if every trail $v_1 \rightleftharpoons v_2 \rightleftharpoons \cdots \rightleftharpoons v_n$ with $v_1 \in X$ and $v_n \in Y$ contains at least one node v_i satisfying one of the following conditions:

- The trail forms a v-structure around v_i, i.e. $v_{i-1} \to v_i \leftarrow v_{i+1}$, and the set $\{v_i\} \sqcup de(v_i)$ is disjoint from Z.
- The trail does not contain a v-structure around v_i and $v_i \in Z$.

If a trail satisfies one of the conditions above, it is said to be ***blocked*** by Z, else it is ***activated*** by Z. Furthermore, if X and Y are not d-separated by Z, we use the notation $\not{d-sep}_{\mathcal{G}}(X, Y \mid Z)$. Moreover, if X or Y is equal to the empty set, then we take the convention that $d-sep_{\mathcal{G}}(X, Y \mid Z)$ always holds.

We make the following remark regarding the definition above.

Remark 2.23. In the literature, d-separation is usually not well-defined if X or Y is equal to the empty-set. However, we will use the convention that if X or Y or both are empty then they are d-separated.

To better explain the concept of d-separation, three examples will be given. Here, the d-separations in the graph in Figure 2.6 are examined. For each example, two subsets of nodes, X and Y, are given and we will examine which subsets, Z, d-separate them.

1. $X = \{v_1\}$, $Y = \{v_6, v_7\}$:
   To d-separate X and Y, we must block all trails between v_1 and v_6. This immediately implies that all trails from v_1 to v_7 are blocked as well. The nodes v_2 and v_3 must be included in the set Z, since omitting them would activate the trails; $v_1 \to v_2 \to v_6$ or $v_1 \to v_3 \to v_6$. Other nodes which are optional but not required are v_4 and v_5. Thus, the possible choices for Z are: $\{v_2, v_3\}$, $\{v_2, v_3, v_4\}$, $\{v_2, v_3, v_5\}$ and $\{v_2, v_3, v_4, v_5\}$.

2. $X = \{v_1\}$, $Y = \{v_4\}$:
   The nodes v_1 and v_4 are connected by the following trails:

   (i) $v_1 \to v_2 \to v_5 \leftarrow v_4$

   (ii) $v_1 \to v_2 \to v_6 \leftarrow v_4$

   (iii) $v_1 \to v_3 \to v_6 \leftarrow v_4$

   Since all three trails contain a v-structure, v_1 and v_4 are d-separated by the empty set. The addition of v_2 and v_3 to Z will maintain the d-separation. However, the nodes v_5, v_6 or v_7 can be included in Z, but they imply the inclusion of other nodes as well. If node v_5 is present in Z, then v_2 must also be included. Indeed, v_5 activates the v-structure in trail (i), and hence to block the trail, v_2 must be in Z as well. For v_6 to be in Z, we must have $v_2, v_3 \in Z$ for the other two trails to be blocked. The same holds for v_7 since it is a descendant of v_6. Thus, the possible choices for Z are: $\emptyset$, $\{v_2\}$, $\{v_3\}$, $\{v_2, v_3\}$, $\{v_2, v_5\}$ and the set $\{v_2, v_3\}$ plus any combination of v_5, v_6 and v_7.

3. $X = \{v_4, v_5\}$, $Y = \{v_1, v_3\}$:
   The node v_6 cannot be included in Z, otherwise the trail $v_5 \to v_6 \leftarrow v_3$ would be active. Moreover, v_7 cannot be included in Z, since it is a descendant of v_6. In order to block the trail $v_5 \leftarrow v_2 \leftarrow v_1$, v_2 must be included in Z. Thus, the only possible choice for Z is: $\{v_2\}$.

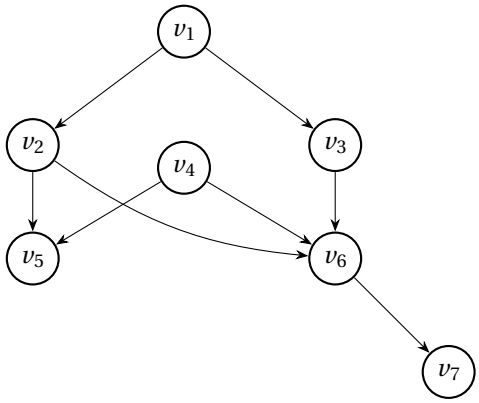


Figure 2.6: A DAG consisting of 7 nodes.

A subclass of DAGs are multitrees.

Definition 2.24 (Multitree)**.** A ***multitree*** is a DAG with at most one directed path between two nodes.

For example, the graph in Figure 2.6 is not a multitree since nodes v_1 and v_6 are joined by two distinct paths; $v_1 \to v_2 \to v_6$ and $v_1 \to v_3 \to v_6$. An example of a multitree is displayed in Figure 2.7.

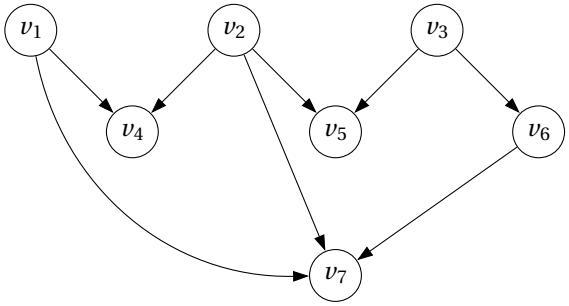


Figure 2.7: Example of a multitree.

Multitrees have the unique property that each pair of parents are d-separated given the empty set. This property will prove to be desirable in the context of the pair-copula Bayesian network. Therefore, we propose the following theorem.

Theorem 2.25. Let $\mathcal{G} = (V, E)$ be a DAG. The following are equivalent:

(i) $\mathcal{G}$ is a multitree.

(ii) $\forall v \in V : \forall w, z \in pa(v)$ with $w \neq z$ we have that $d - sep_{\mathcal{G}}(w, z \,|\, \emptyset)$.

Proof.

- **(i)** $\Rightarrow$ **(ii)** : Proof by contraposition. There exist $v \in V$ and $w \neq z \in pa(v)$ such that $\not{d - sep}_{\mathcal{G}}(w, z \,|\, \emptyset)$. Hence, there exists an active trail between w and z. If these nodes are connected, e.g. $w \to z \in E$, then there are obviously two distinct paths from w to v. In case when these nodes are not directly connected, hence $w \to z \notin E$ and $z \to w \notin E$, then the active trail between w and z can only correspond to one of the following paths:

1. $w \to v_1 \to \cdots \to v_n \to z$
2. $w \leftarrow v_1 \leftarrow \cdots \leftarrow v_n \leftarrow z$
3. $w \leftarrow \cdots \leftarrow v_i \to \cdots \to z$

For all three cases we can find two nodes which are connected by two distinct directed paths. For case 1, we have that w and v are connected by

$$w \to v,$$
$$w \to v_1 \to \cdots \to v_n \to z \to v.$$

Similarly, for the second case we have that z and v are joined by:

$$z \to v,$$
$$z \to v_n \to \cdots \to v_1 \to w \to v.$$

In the third case, the nodes v_i and v are connected by the paths

$$v_i \to v_{i+1} \to \cdots \to v_n \to z \to v,$$
$$v_i \to v_{i-1} \to \cdots \to v_1 \to w \to v.$$

Hence, we get that this graph is not a multitree.

- (**ii**) $\Rightarrow$ (**i**) : Again proof by contraposition. If $\mathcal{G}$ is not a multitree, then there exist two nodes $x, y \in V$ with two distinct directed paths between them. We can assume that these paths start at x and end with y. A path between x and y is either direct link $x \to y$ or it is of the form $x \to v_1 \to \cdots \to v_n \to y$. Thus, two cases need to be considered.

  1. Suppose that the two paths are of the form

     $$x \to v_1 \to \cdots \to v_n \to y,$$
     $$x \to y.$$

     It is clear that $x, v_n \in pa(y)$ and that $\not{d\text{-}sep}_{\mathcal{G}}\big(x, v_n \,\big|\, \emptyset\big)$ since $x \to v_1 \to \cdots \to v_n$ is an active trail.

  2. Let the two paths be as follows

     $$x \to v_1 \to \cdots \to v_n \to y,$$
     $$x \to w_1 \to \cdots \to w_m \to y.$$

     We can assume that these paths do not share a node (except for x and y), i.e. $\{v_1, \ldots, v_n\} \cap \{w_1, \ldots, w_m\} = \emptyset$. If this is not the case, we can simply pick the smallest i and j such that $v_i = w_j$, and set this node as our y. The resulting two paths will clearly not share a node. Moreover, if this operation reduces one of the paths to a direct link we are in case 1. Else, we have that $v_n, w_m \in pa(y)$ and $\not{d\text{-}sep}_{\mathcal{G}}\big(v_n, v_m \,\big|\, \emptyset\big)$ since $v_n \leftarrow \cdots \leftarrow v_1 \leftarrow x \to w_1 \to \cdots \to w_n$ is an active trail.

□

2.2. Bayesian network

This section will provide the necessary background knowledge on Bayesian networks, for more information on the topic it is advised to read [27, 22, 15, 19].

First, we must establish the necessary notation concerning graphical models. A graphical model is a graphical representation of a multivariate random variable. Here, the nodes of the graph correspond to univariate random variables and the arcs express the direct dependence between them. Consider a random vector $\mathbf{X}$ which is represented by a graph $\mathcal{G} = (V, E)$. Each node $v \in V$ corresponds to a univariate random variable X_i. A random variable X_i will also be referred to as X_v. Moreover, we will denote realizations of X_i by either x_i or x_v and the PDF of X_i will be written as f_i or f_v. This notation is easily extended to subsets $K \subseteq V$, i.e. $\mathbf{X}_K = (X_v)_{v \in K}$ with PDF f_K. Furthermore, the PDF of a random variable X_v conditional on $\mathbf{X}_K$ with $K \subseteq V \setminus \{v\}$ is denoted by $f_{v|K}$.

Bayesian networks are graphical models where the graph is a DAG, see Definition 2.13. The arcs induce direct independencies between random variables and conditional independencies between sets of random variables through d-separation ([27]), see Definition 2.22.

Definition 2.26 (Bayesian network)**.** A ***Bayesian network*** (BN) is a graphical model composed of

- a DAG $\mathcal{G} = (V, E)$ where the nodes correspond to univariate random variables and the arcs describe the conditional independencies through d-separation,
- a sequence of conditional densities $\{f_{v|pa(v)};\ v \in V\}$.

The set of conditional independencies allows for the decomposition of the joint density as a product of the specified conditional densities;

$$f_V(\mathbf{X}_V) = \prod_{v \in V} f_{v|pa(v)}(x_v|\mathbf{x}_{pa(v)}). \tag{2.1}$$

A set of random variables corresponding to a certain BN must have a probability measure that is in compliance with the set of conditional independencies. Such measures are called Markovian w.r.t. the BN.

Definition 2.27 (Markovian)**.** A probability measure P over the nodes of a directed graph $\mathcal{G} = (V, E)$ is said to be ***Markovian*** w.r.t. $\mathcal{G}$ if:

$$\forall X, Y, Z \subseteq V : d - sep_{\mathcal{G}}(X, Y \,|\, Z) \Rightarrow X \perp\!\!\!\perp_P Y \,|\, Z.$$

It is easily seen that a probability measure P which is Markovian w.r.t a directed graph $\mathcal{G}$ and absolutely continuous w.r.t. the Lebesgue measure allows for the factorization in Equation (2.1). Since all Markovian probabilities can be factorized according to Equation (2.1), we can construct the high-dimensional densities f_V by using the conditional distributions $f_{v|pa(v)}$.

Consider a multivariate random variable $\mathbf{X}$ which is Markovian w.r.t. a DAG $\mathcal{G}$. Remark that ordering the univariate random variables X_i according to a well-ordering of $\mathcal{G}$ (see Definition 2.19) allows for the factorization

$$f_{\mathbf{X}}(\mathbf{X}) = \prod_{i=1}^{n} f_{X_i|X_1,\ldots,X_{i-1}}(x_i|x_1,\ldots,x_{i-1}).$$

An important observation is that BNs with varying graphical structures can induce the exact same collection of Markovian probabilities, such networks are called equivalent.

Definition 2.28 (Equivalence class)**.** Two Bayesian networks $\mathcal{G}_1$ and $\mathcal{G}_2$ are ***equivalent*** if they induce the same set of Markovian probability measures. For a network $\mathcal{G}$, the set of all networks which are equivalent to $\mathcal{G}$ is called the ***equivalence class*** of $\mathcal{G}$, denoted by $[\mathcal{G}]$.

To illustrate the equivalence between different BNs we examine the graphs in Figure 2.8.

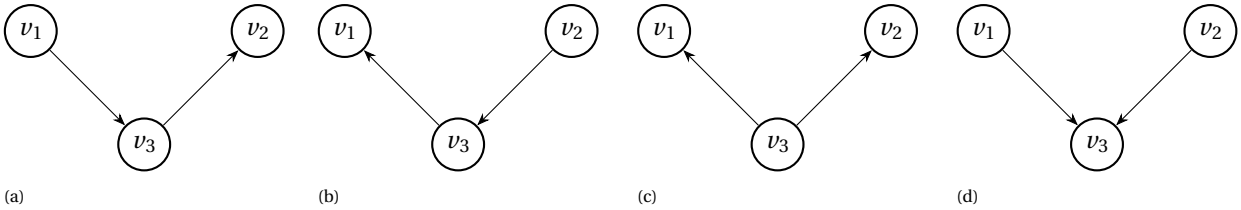


Figure 2.8: Example illustrating equivalence of graphs.

It is easily seen that the first three graphs are equivalent. For example, by rewriting the factorization of the first graph;

$$\begin{aligned} f(v_1, v_2, v_3) &= f(v_1) f(v_3|v_1) f(v_2|v_3) \\ &= f(v_1, v_3) f(v_2|v_3) \\ &= f(v_3) f(v_1|v_3) f(v_2|v_3), \end{aligned}$$

we can find the factorization of the third graph. However, the fourth graph is not equivalent to the first three. The v-structure is a special case which implies different conditional independencies. Indeed, the first three graphs encode the d-separation $d-sep_{\mathcal{G}}\big(v_1, v_2 \,\big|\, v_3\big)$ whereas the fourth graph encodes the d-separation $d-sep_{\mathcal{G}}\big(v_1, v_2 \,\big|\, \emptyset\big)$. In [28] it was shown that two DAGs are equivalent if and only if they have the same set of d-separations. An even more intuitive theorem was proven by Pearl and Verma a year later in [38].

Theorem 2.29. Two DAGs are equivalent if and only if they have the same skeletons and uncoupled v-structures.

Note that two equivalent DAGs can have different coupled v-structures. For example, the graphs in Figure 2.9 are equivalent, i.e. the d-separations following from both graphs are the same, yet they have different v-structures.

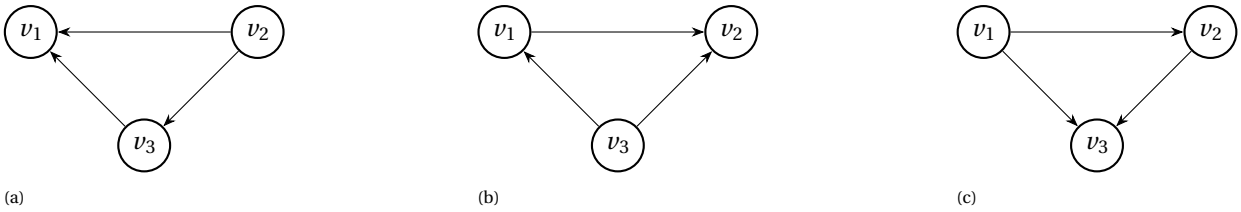


Figure 2.9: Example illustrating equivalent graphs with varying coupled v-structures.

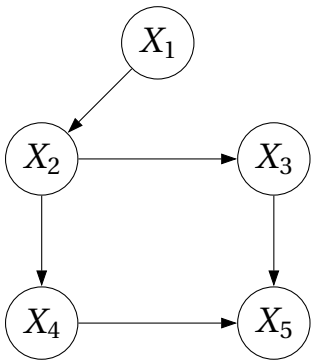
2.3. Gaussian Bayesian network

In this section we provide a brief introduction into the Gaussian Bayesian network (GBN). An extensive treatment of GBN models can be found in [19], [22], [8] and [33].

A GBN is a BN where all conditional PDFs in the decomposition of the joint density (Equation (2.1)) are assumed to be Gaussian.

Definition 2.30 (Gaussian Bayesian network)**.** A ***Gaussian Bayesian network*** is a BN such that all conditional PDFs of each node, $f_{v|pa(v)}$, are normally distributed with mean equal to a linear combination of its parents and a constant standard deviation. That is, $X_v \sim N\big(\mu_v + \sum_{w \in pa(v)} \phi_v^w X_w, \sigma_v^2\big)$, where μ_v, σ_v and ϕ_v^w are constants.

For example, we could define the following GBN.



$$X_1 \sim N(2, 1^2)$$
$$X_2 \sim N(1.2 + 0.5 \cdot X_1, 0.9^2)$$
$$X_3 \sim N(2 + 0.8 \cdot X_2, 0.75^2)$$
$$X_4 \sim N(0.5 - X_2, 0.3^2)$$
$$X_5 \sim N(0 - 0.75 \cdot X_3 + 0.6 \cdot X_4, 1.2^2)$$

Figure 2.10: GBN with the graph displayed on the left and the conditional distributions displayed on the right.

An important property of GBNs is their equivalency to multivariate Gaussians. This result is proven in Theorems 7.3 and 7.4 in [19]. When a GBN is specified it is easy to compute the parameters of corresponding joint Gaussian distributions (see [19, ch. 7]). This can be done for any graph structure. This means that GBNs inherit all nice properties of joint Gaussian distribution, that is, the marginal distributions of all variables can be easily computed. The conditionalization of GBNs on observed evidences can be computed analytically. For example, in Figure 2.10 the conditional density $f_{13|5}$ is not directly given by the GBN but can be easily computed, as we will see in Example 2.32.

All parameters of a GBN are contained in a parameter vector denoted by $\boldsymbol{\theta}$. This means that $\boldsymbol{\theta}$ contains μ_v, σ_v and ϕ_v^w for all $v \in V$ and $w \in pa(v)$. Observe that the GBN can be seen as a set of regression equations; for all $v \in V$,

$$X_{v|pa(v)} = \mu_v + \sum_{w \in pa(v)} \phi_v^w X_w + N(0, \sigma_v^2).$$

Consequently, $\boldsymbol{\theta}$ can be estimated very fast. In this thesis this will be done by optimizing the log-likelihood, AIC and BIC.

Definition 2.31 (Likelihood based selection criteria for GBNs)**.** Let $\mathcal{G}$ be a BN and f_V a density induced by $\mathcal{G}$ with corresponding parameter vector $\boldsymbol{\theta}$. For a random sample $\mathcal{D} = (\mathbf{x}_V^{(m)})_{m=1,\dots,M}$ of size M we define the

log-likelihood by

$$\ell(\boldsymbol{\theta};\mathcal{G},\mathcal{D}) = \log \prod_{m=1}^{M} f_V(\mathbf{x}_V^{(m)};\boldsymbol{\theta})$$
$$= \sum_{m=1}^{M}\sum_{v\in V} \log f_{v|pa(v)}(x_v^{(m)}|\mathbf{x}_{pa(v)}^{(m)};\boldsymbol{\theta}).$$

The ***AIC*** and ***BIC*** are defined as

$$AIC(\boldsymbol{\theta};\mathcal{D}) = -2\cdot\ell(\boldsymbol{\theta};\mathcal{G},\mathcal{D}) + 2k,$$
$$BIC(\boldsymbol{\theta};\mathcal{D}) = -2\cdot\ell(\boldsymbol{\theta};\mathcal{G},\mathcal{D}) + \log(M)k,$$

where k is the number of parameters in $\boldsymbol{\theta}$.

As an example, we apply the GBN on a data set generated from the GBN in Figure 2.10.

Example 2.32. Consider the data set in Figure 2.11.

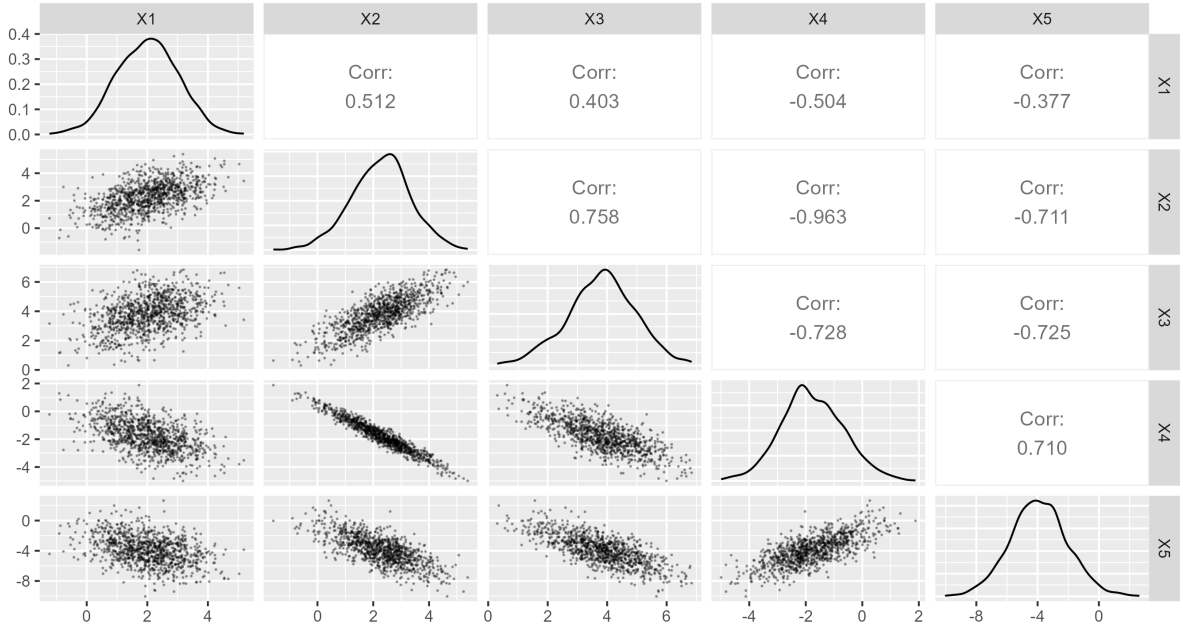


Figure 2.11: Data set of 1000 samples generated from the GBN in Figure 2.10.

We fit a GBN using the true graph in Figure 2.10 by minimizing the BIC with the function bn.fit() from the bnlearn package in R. The fitted model has the following conditional distributions:

$$X_1 \sim N(2.009, 1.008^2),$$
$$X_2 \sim N(1.188 + 0.537\cdot X_1, 0.908^2),$$
$$X_3 \sim N(1.942 + 0.822\cdot X_2, 0.748^2),$$
$$X_4 \sim N(0.502 - 1.000\cdot X_2, 0.298^2),$$
$$X_5 \sim N(0.009 - 0.725\cdot X_3 + 0.660\cdot X_4, 1.192^2).$$

The found parameters closely resemble the parameters of the true GBN in Figure 2.10. Moreover, the BIC of the fitted model is equal to 11445.13 which is close to the BIC of the true distribution; 11453.3. Therefore, we can conclude that the estimated GBN fits the data rather well.

With the function gbn2mvnorm() from bnlearn, we can convert this GBN into an equivalent multivariate Gaussian, which has mean vector and covariance matrix equal to

$$\boldsymbol{\mu} = \begin{pmatrix} 2.009 \\ 2.268 \\ 3.806 \\ -1.767 \\ -3.919 \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{pmatrix} 1.017 & 0.546 & 0.449 & -0.547 & -0.687 \\ 0.546 & 1.118 & 0.919 & -1.119 & -1.406 \\ 0.449 & 0.919 & 1.315 & -0.920 & -1.562 \\ -0.547 & -1.119 & -0.920 & 1.208 & 1.465 \\ -0.687 & -1.406 & -1.562 & 1.465 & 3.520 \end{pmatrix}.$$

We see what are the mean and the variance of Gaussian distribution of each node and we are able to determine any desired conditional distribution in closed form. Suppose that we wish to find $(X_1, X_3)|X_5 = x_5$. First, we observe that the joint distribution of (X_1, X_3, X_5) is joint normal with

$$\boldsymbol{\mu}_{135} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_3 \\ \boldsymbol{\mu}_5 \end{pmatrix} \quad \text{and} \quad \Sigma_{135} = \begin{pmatrix} \Sigma_{11} & \Sigma_{13} & \Sigma_{15} \\ \Sigma_{31} & \Sigma_{33} & \Sigma_{35} \\ \Sigma_{51} & \Sigma_{53} & \Sigma_{55} \end{pmatrix}.$$

Now, it follows (see for instance [9, p. 116] that the conditional random vector $(X_1, X_3)|X_5$ is joint normally distributed with mean vector and covariance matrix equal to

$$\begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_3 \end{pmatrix} + \begin{pmatrix} \Sigma_{15} \\ \Sigma_{35} \end{pmatrix} \frac{1}{\Sigma_{55}} (x_5 - \boldsymbol{\mu}_5) \quad \text{and} \quad \begin{pmatrix} \Sigma_{11} & \Sigma_{13} \\ \Sigma_{31} & \Sigma_{33} \end{pmatrix} - \begin{pmatrix} \Sigma_{15} \\ \Sigma_{35} \end{pmatrix} \frac{1}{\Sigma_{55}} \begin{pmatrix} \Sigma_{15} & \Sigma_{35} \end{pmatrix}$$

which in this case are equal to

$$\begin{pmatrix} -0.198 \cdot x_5 - 1.223 \\ -0.446 \cdot x_5 - 2.001 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 0.929 & 0.152 \\ 0.152 & 0.613 \end{pmatrix}.$$

2.4. Score-based structure learning in GBNs

In this section the process of finding the optimal graphical structure of a GBN for a given data set using score-based structure learning is described. For more information concerning structure learning in GBNs we refer to [33], [19, ch. 18] and [18].

Score-based algorithms are composed of a search strategy and score function. The score function evaluates how well the proposal graph represents independencies and conditional independencies of the data assuming that it has joint Gaussian distribution. The search space of possible graph structures is traversed according to a certain search strategy and graphs are evaluated. The one with maximum score is chosen.

There are many different score functions and search strategies, for a good overview see [18].

2.4.1. Score functions

In this thesis, only the following likelihood based score functions are applied.

Definition 2.33 (Likelihood-based score functions for GBNs)**.** Consider a data set $\mathscr{D}$ and a GBN $\mathscr{G}$. We define the following ***likelihood-based score functions***:

$$\begin{aligned} score_\ell &= \ell(\boldsymbol{\theta}^\ell; \mathscr{G}, \mathscr{D}), \\ score_{AIC} &= -AIC(\boldsymbol{\theta}^{AIC}; \mathscr{G}, \mathscr{D}), \\ score_{BIC} &= -BIC(\boldsymbol{\theta}^{BIC}; \mathscr{G}, \mathscr{D}). \end{aligned}$$

Here, $\boldsymbol{\theta}^\ell$, $\boldsymbol{\theta}^{AIC}$ and $\boldsymbol{\theta}^{BIC}$ are the parameter vectors maximizing the log-likelihood, -AIC and -BIC respectively.

Score functions that have desirable properties are favoured, such properties include decomposability and score equivalence.

Decomposability: A score function is called decomposable if it can be written as a sum over the nodes of functions depending only on the node and its parents. Observe that the score functions in Definition 2.33 are decomposable. For example, $score_\ell$ can be written as

$$\begin{aligned} score_\ell &= \ell(\boldsymbol{\theta}^\ell; \mathscr{G}, \mathscr{D}) \\ &= \sum_{v \in V} \left[\sum_{m=1}^{M} \log f_{v|pa(v)}\left(x_v^{(m)} | \mathbf{x}_{pa(v)}^{(m)}; \mu_v^\ell, \sigma_v^\ell, \{\phi_v^{w,\ell}\}_{w \in pa(v)}\right) \right]. \end{aligned}$$

Here, the functions $f_{v|pa(v)}$ do not require the entire vector $\boldsymbol{\theta}^\ell$ as input, instead they only need the relevant parameters; μ_v^ℓ, σ_v^ℓ and $\{\phi_v^{w,\ell}\}_{w \in pa(v)}$. If this were not the case, then the score function would not be decomposable.

Decomposability is a desirable trait for a score function, since it allows for fast computations. This is because the score of a node v will only be affected by arc operations concerning arcs pointing towards v. Thus, computing the score change of an arc operation becomes a local problem.

Score equivalency: Equivalent graphs induce the same probability distributions, therefore it would be logical if they are assigned the same score. A score function satisfying this property is called score equivalent. The score functions in Definition 2.33 are score equivalent. Indeed, two equivalent GBNs induce the same set of probability distributions, and therefore optimizing the log-likelihood, AIC or BIC for both graphs results in an identical score.

2.4.2. Search strategy

In this thesis, the search strategy that will be employed is the Hill climbing algorithm [31]. This algorithm is initialized with a graph, often equal to the empty graph. At each iteration, the score of the current graph

$\mathcal{G}_{max}$ is compared with all neighbouring graphs; i.e. graphs reachable from $\mathcal{G}_{max}$ by arc addition, removal or reversal.

If there is a neighbour for which the score increases, then the current graph is replaced by a neighbour with the highest score. The described steps are formalized in Algorithm 1. Hereunder, we consider an example where the Hill climbing algorithm is applied to the data set from Example 2.32.

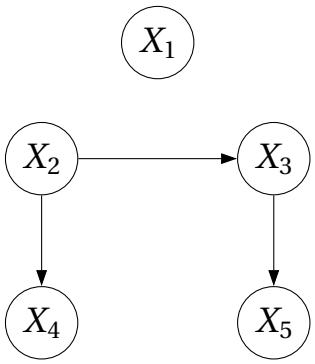
Algorithm 1 Hill climbing for GBNs

Input: DAG $\mathcal{G}$, data $\mathcal{D}$, score function $score(\mathcal{G};\mathcal{D})$
Output: the DAG $\mathcal{G}_{max}$ that locally maximizes $score(\mathcal{G};\mathcal{D})$

```
G_max ← G
S_max ← score(G; D)
while S_max increases do
    for each arc operation e on G_max resulting in a DAG G_e do
        compute the score delta Δ(e) = score(G_e; D) − S_max
    end for
    if max{Δ(e)} > 0 then
        e* = arg max_e {Δ(e)}
        G_max ← G_{e*}
        S_max ← S_max + Δ(e*)
    end if
end while
return G_max
```

Example 2.34. Let us apply the Hill climbing algorithm implemented in the bnlearn function hc() to the data set in Figure 2.11. By default, bnlearn uses the $score_{BIC}$, and therefore we do the same.

After starting with the empty graph, the algorithm adds the arcs $2 \to 4$, $2 \to 3$ and $3 \to 5$, giving us the graph displayed below.



Now, the algorithm will compute the change in score for every arc operation e resulting in a DAG $\mathcal{G}_e$. This provides us with Table 2.1. There are two operations which provide the highest score delta, i.e. the additions of the arcs $1 \to 2$ and $2 \to 1$. When confronted with two operations resulting in an equal increase in score, bnlearn consults the order of the columns in the data-frame.

For example, if the columns of the data-frame are ordered as X_1, X_2, X_3, X_4 and X_5, it will add the arc $1 \to 2$, since 1 comes before 2. However, if we choose another order such as X_2, X_4, X_1, X_5 and X_3, then bnlearn prefers the arc $2 \to 1$. Thus, depending on the order of the columns either arc $1 \to 2$ or arc $2 \to 1$ is added, hereafter its search is continued.

If we supplied ***bnlearn*** with the column order X_1, X_2, X_3, X_4 and X_5, then the Hill climbing algorithm finds

the correct graph, i.e. the graph in Figure 2.10. In this case we find the same fitted model from Example 2.32.

If we order the columns as X_2, X_4, X_1, X_5 and X_3, then bnlearn finds the graph displayed below. Observe that this graph is equivalent to the true graph. Since equivalent graphs induce the same set of probability distributions, fitting a GBN with this graph provides an identical distribution to the one found in Example 2.32.

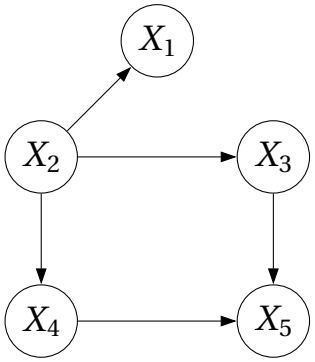


Figure 2.12: Found graph when applying the Hill climbing algorithm in bnlearn to the data in Figure 2.11 with the column order X_2, X_4, X_1, X_5 and X_3.

operation	arc e	Δe
addition	$1 \to 2$	**122.073**
addition	$1 \to 3$	-3.442
addition	$1 \to 4$	-1.767
addition	$1 \to 5$	3.342
addition	$2 \to 1$	**122.073**
addition	$2 \to 5$	51.193
addition	$3 \to 1$	60.408
addition	$3 \to 4$	-3.366
addition	$4 \to 1$	119.551
addition	$4 \to 3$	-3.366
addition	$4 \to 5$	62.569
addition	$5 \to 1$	49.182
addition	$5 \to 4$	5.431
remove	$2 \to 3$	-374.682
remove	$2 \to 4$	-1282.861
remove	$3 \to 5$	-310.571
reverse	$2 \to 3$	0
reverse	$2 \to 4$	0
reverse	$3 \to 5$	-229.583

Table 2.1: The score-delta of the BIC-based score function for all possible arc operations.

2.4.3. Performance metrics

There are many methods to test the performance of structure learning algorithms ([18]). We will discuss three approaches; graphical distance, inference measures and statistical distance. It should be noted that the first approach requires us to know the true graphical structure, and the third approach requires us to know the true distribution. In this thesis, algorithms will be tested on simulated data sets for which the true graph and distribution are known. Therefore, this will not pose a problem.

First approach: A simple metric would be to run the algorithm and check if the estimated graph matches the true graph. For instance, one could compute the Structural Hamming Distance (SHD) between the true and

the estimated graph ([36]). This metric returns the number of arc operations it would require to transform the estimated DAG into the true DAG.

A down sight of such metrics is that equivalent graphs are often assigned a distance greater than zero. Indeed, the SHD between the equivalent graphs in Figure 2.9 is greater than zero.

By Theorem 2.29, two BNs are equivalent if they have the same skeleton and uncoupled v-structures. Therefore, we propose a metric based on these two properties. Moreover, instead of measuring the difference between uncoupled v-structures we also take into account distinct coupled v-structures. This means that two equivalent graphs might still be assigned a distance greater than zero.. However, later in this thesis, this choice will be justified when we apply the metric to the pair-copula Bayesian networks. Here, we will see that two PCBNs can only be equivalent if their coupled v-structures are also the same. Hence, we propose the following distance metric.

Definition 2.35 (Distance)**.** Let $\mathcal{G}_1 = (V, E_1)$ and $\mathcal{G}_2 = (V, E_2)$ be two DAGs, and let $K \subseteq V$ be the set containing all nodes in V that correspond to a v-structure in either $\mathcal{G}_1$ or $\mathcal{G}_2$. Then, the ***distance*** between the $\mathcal{G}_1$ and $\mathcal{G}_2$ is given by

$$distance(\mathcal{G}_1, \mathcal{G}_2) = \sum_{v,v' \in V} \mathbb{1}\Big(v - v' \in S(E_1) \Delta S(E_2)\Big) + \sum_{v \in K} \Big|pa(v)(\mathcal{G}_1) \Delta pa(v)(\mathcal{G}_2)\Big|$$

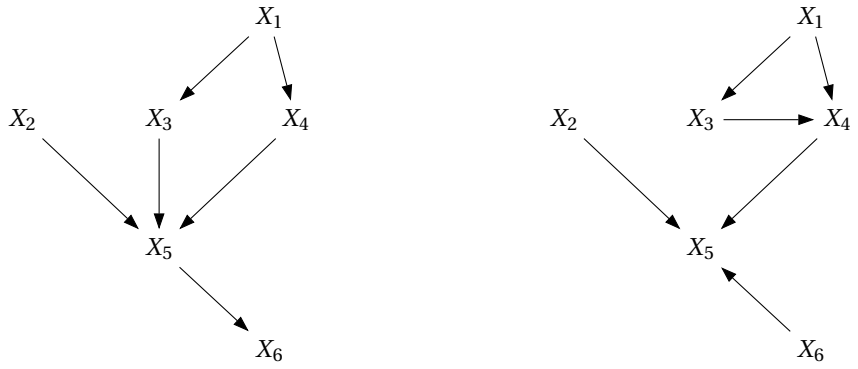
where Δ represents the symmetric difference and $S(E_1)$ and $S(E_2)$ are as in Definition 2.4.

The first sum measures the number of different edges in the two skeletons, and the second sum compares the parents of the v-structures in both graphs.

As an example, let us now compute the distance between the two graphs below. The skeletons differ in two edges; $X_3 - X_4$ and $X_3 - X_5$. Moreover, the v-structures at X_4 and X_5 are different;

$$\Big|pa(X_4)(\mathcal{G}_1) \Delta pa(X_4)(\mathcal{G}_2)\Big| = \Big|\{X_3\}\Big| = 1 \quad \text{and} \quad \Big|pa(X_5)(\mathcal{G}_1) \Delta pa(X_5)(\mathcal{G}_2)\Big| = \Big|\{X_3, X_6\}\Big| = 2.$$

Thus, the distance between the two graph is $2 + 1 + 2 = 5$.



(a) $\mathcal{G}_1$ (b) $\mathcal{G}_2$

Figure 2.13: Two graphs whose distance is equal to five.

Second approach: In the inference-based approach, the performance of an estimated GBN is measured by how well it fits the data. This is generally done by using the log-likelihood, AIC or BIC. To compare an estimated GBN to the true GBN, one can compare the inference-based metrics of the optimized parameter vector for both the true and estimated graph. For instance, in Example 2.34 we have seen that the BIC of the estimated GBN and the true GBN were relatively close; 11445.13 and 11453.3, respectively.

Third approach: Alternatively, it is possible to measure the distance between the true probability measure and the estimated measure. This can for instance be done with the Kullback-Leibler divergence ([20]).

Definition 2.36 (Kullback-Leibler divergence)**.** Let F be the true probability measure and $\hat{F}$ the estimated measure. The ***Kullback-Leibler*** divergence between F and $\hat{F}$ is defined by

$$\begin{aligned} KL(F,\hat{F}) &= \int_{-\infty}^{\infty} \log\left(\frac{f(\mathbf{x})}{\hat{f}(\mathbf{x})}\right) \mathrm{d}F(\mathbf{x}) \\ &= \mathbb{E}\left[\log\left(\frac{f(\mathbf{x})}{\hat{f}(\mathbf{x})}\right)\right] \end{aligned}$$

2.4.4. Hill climbing extensions

In this thesis, we will apply the standard version of the Hill climbing algorithm. An obvious flaw of the Hill climbing algorithm is that it could get stuck in a local maximum, which is not equal to the global maximum. We briefly discuss two extensions which aim to solve this problem.

Random restarts: One approach is to use random restarts once the algorithm reaches a local maximum ([13]). This means that the found graph is augmented with a certain amount of random arc operations, in the hope of finding a better graph.

Tabu search: Here, the algorithm keeps a "tabu" list of recently visited graphs which the algorithm will avoid ([6, p. 99]). Moreover, instead of only moving to graphs which increase the score function, it applies the optimal arc operation, even it results in a lower score. This means that when reaching a local maximum, the algorithm will continue its search. Generally, the number of allowed decreasing steps is specified, and if this number is reached, then the algorithm returns the last found local maximum.

2.5. Copulas

This section will provide the necessary background knowledge on copulas. For a more in depth explanation it is advised to read [7, 17, 26].

Definition 2.37 (Copula)**.** A copula is distribution on the d-dimensional unit hypercube, $[0,1]^d$, with uniform marginal distributions.

Copulas are able to model any real valued multivariate distribution by Sklar's theorem [35].

Theorem 2.38 (Sklar's theorem)**.** Let F be a CDF on $\mathbb{R}^d$ with univariate margins $F_1,\dots,F_d$. Then, there exists a copula $C : [0,1]^d \mapsto [0,1]$ such that for all $\mathbf{x} = (x_1,\dots,x_d) \in \mathbb{R}^d$:

$$F(\mathbf{x}) = C\big(F_1(x_1),\dots,F_d(x_d)\big).$$

If the univariate margins are continuous, then C is unique.

Sklar's theorem immediately implies a similar equation for the joint PDF. Using the same setting as in Theorem 2.38, for the joint density f and marginal densities $f_1,\dots,f_d$ corresponding to F and $F_1,\dots,F_d$ assuming that they exist, we have that

$$f(x_1,\dots,x_d) = f_1(x_1)\dots f_d(x_d)c\big(F_1(x_1),\dots,F_d(x_d)\big),$$

where c is the PDF corresponding to C.

2.5.1. Bivariate copulas

Before examining various bivariate copulas we define the concepts of Kendall's τ and tail dependence. These are dependence measures used to summarize the dependence between random variables.

Definition 2.39 (Kendall's tau)**.** Let X and Y be two real-valued random variables joined by copula C with distributions F_X and F_Y. Then, the X and Y are correlated with Kendall's τ equal to

$$\tau = 4\int_{\mathbb{R}}\int_{\mathbb{R}} C(F_X(x),F_Y(y))\,\mathrm{d}C(F_X(x),F_Y(y)) - 1.$$

Definition 2.40 (Tail dependence)**.** Let X and Y be two real-valued random variables joined by copula C with distributions F_X and F_Y. Then, the upper and lower tail index are defined as

$$\lambda_U = \lim_{u\to 1}\frac{1-2u+C(u,u)}{1-u} \quad \text{and} \quad \lambda_L = \lim_{u\to 0}\frac{C(u,u)}{u},$$

respectively. If $\lambda_U \in (0,1]$ (respectively $\lambda_L \in (0,1]$), then X and Y are upper tail dependent (respectively lower tail dependent). If $\lambda_U = 0$ (respectively $\lambda_L = 0$) then X and Y are upper tail independent (respectively lower tail independent).

There is a vast amount of parametric copula families to choose from. An in depth overview can be found in [16] and [26].

This research will focus on a subset of widely used copula families. In particular we will use the Gaussian, Clayton, Gumbel, Frank and Joe copula families, which all require the specification of one parameter.

The Gaussian copula with correlation $\rho \in [-1,1]$ is constructed by using the multivariate normal distribution and is defined by

$$\mathbf{C}_\rho(u_1,u_2) = \mathbf{\Phi}_\rho\big(\Phi^{-1}(u_1),\Phi^{-1}(u_2)\big)$$

where the functions $\mathbf{\Phi}_\rho$ and Φ correspond to the multivariate normal with correlation ρ and the univariate standard normal distribution.

The Clayton, Gumbel, Frank and Joe copulas fall into the class of Archimedean copulas, meaning that they are constructed using a generator function. A generator function is a convex and strictly decreasing function $\varphi : (0,1] \mapsto [0,\infty]$, satisfying $\varphi(0) = 1$. For such functions the expression

$$C(u_1, u_2) = \varphi^{-1}\big(\varphi(u_1), \varphi(u_2)\big)$$

will be a properly defined copula, i.e. a CDF on $[0,1]^2$ with uniform margins. The generator functions of our copulas and additional information about Archimedean copulas can be found in [26, Section 4.3].

2.5.2. Bivariate estimation

There are several methods to estimate a copula corresponding to a certain 2-dimensional data set. A clear overview can be found in [23]. One approach is to apply a two-step procedure. That is, first transform the marginal data to uniforms, by either parametric or empirical distributions. Hereafter, the copula will be estimated using the uniform data. This is the approach that will be used during this thesis.

To estimate a bivariate copula, we will select the family and parameters based on likelihood criteria, e.g. AIC, BIC and log-likelihood. These are defined in the same way as in Definition 2.31. That is, for a given dataset $\mathscr{D} = \{(u_1^{(m)}, u_2^{(m)})\}_{m\in\{1,\dots,M\}}$ containing M samples a of 2-dimensional random vector, the log-likelihood, AIC and BIC of a copula c with parameters $\boldsymbol{\theta}$ are given by

$$\begin{aligned}
\ell(\boldsymbol{\theta}; \mathscr{D}) &= \sum_{m=1}^{M} \log c\big(u_1^{(m)}, u_2^{(m)}; \boldsymbol{\theta}\big), \\
AIC(\boldsymbol{\theta}; \mathscr{D}) &= -2 \cdot \ell(\boldsymbol{\theta}; \mathscr{D}) + 2k, \\
BIC(\boldsymbol{\theta}; \mathscr{D}) &= -2 \cdot \ell(\boldsymbol{\theta}; \mathscr{D}) + \log(M)\, k,
\end{aligned}$$

where k is the number of parameters. Naturally, we will pick the copula family and parameters which optimize one of these criteria, that is minimize the AIC or BIC, or maximize the log-likelihood. Model selection among parametric copula families using such procedures is implemented by the function BiCopSelect() from the VineCopula package in R ([24]). Hence, this research will utilize said function for the model selection of bivariate copulas.

2.5.3. Conditional copulas

When decomposing high-dimensional copulas into a product of bivariate copulas we will make use of conditional copulas, which are copulas that join conditional marginal distributions. For example, if we have random variables $X_1, \dots, X_n$, $i \neq j \in \{1, \dots, n\}$ and $K \subseteq \{1, \dots, n\}$ with $K \cap \{i, j\} = \emptyset$, then

$$F_{i,j|K}(x_i, x_j|\mathbf{x}_K) = C_{i,j|K}\big(F_{i|K}(x_{i|K}|\mathbf{x}_K), F_{i|K}(x_{j|K}|\mathbf{x}_K)\,\big|\,\mathbf{x}_K\big),$$

where $C_{i,j|K}$ is a conditional copula. Formally, this copula is a 2-dimensional CDF with uniform margins which depends on the realizations of the conditioning variables $\mathbf{x}_K$. Modeling such functions is quite complex. Therefore, it is often assumed that the copula does not explicitly depend on the conditioning variables. That is, we assume that

$$F_{i,j|K}(x_i, x_j|\mathbf{x}_K) = C_{i,j|K}\big(F_{i|K}(x_{i|K}|\mathbf{x}_K), F_{i|K}(x_{j|K}|\mathbf{x}_K)\big).$$

This assumptions is referred to as the "simplifying assumption". It should be noted that the usage of the simplifying assumption has been debated in the literature. In [14] it is stated that using the simplifying assumption will still lead to adequate approximations of the true copulas. Moreover, it is argued that their usage is

necessary in the application of the pair-copula construction, as non-simplified copulas require significantly more computational power. However, [2] shows examples for which it is argued that models using the simplifying assumptions will not suffice. Throughout this thesis, the simplifying assumptions will assumed to hold.

2.5.4. Pair-copula construction

The pair-copula construction is a method which enables us to model a multivariate distributions by decomposing the PDF into a product of bivariate (conditional) copulas. Moreover, the construction is hierarchical in nature. This means that the model can be graphically represented by a collection of trees. The pair-copula construction will not explicitly be used in this research. However, it has a strong connection with the pair-copula Bayesian network. Therefore, this section will briefly explain the ideas behind the pair-copula construction. For an extensive explanation on the subject we refer to [17, 5, 7, 1].

Consider the continuous random variables $X_1, \dots, X_n$. By factorizing the joint PDF f by

$$f(x_1, \dots, x_n) = \prod_{i=1}^{n} f(x_i | x_1, \dots, x_{i-1}), \tag{2.2}$$

we are left with n conditional distributions These function can be expressed with bivariate copulas with the help of Sklar's theorem. For example, in the 3-dimensional case we can rewrite the conditional densities by

$$\begin{aligned}
f_{2|1}(x_2|x_1) &= \frac{f_{1,2}(x_1, x_2)}{f_1(x_1)} \\
&= \frac{f_1(x_1) \cdot f_2(x_2) \cdot c_{1,2}\big(F_1(x_1), F_2(x_2)\big)}{f_1(x_1)} \\
&= f_2(x_2) \cdot c_{1,2}\big(F_1(x_1), F_2(x_2)\big), \\
f_{3|2,1}(x_3|x_1, x_2) &= \frac{f_{2,3|1}(x_2, x_3)}{f_{2|1}(x_2|x_1)} \\
&= \frac{f_{2|1}(x_2|x_1) \cdot f_{3|1}(x_3|x_1) \cdot c_{2,3|1}\big(F_{2|1}(x_2|x_1), F_{3|1}(x_3|x_1)\big)}{f_{2|1}(x_2|x_1)} \\
&= f_{3|1}(x_3|x_1) \cdot c_{2,3|1}\big(F_{2|1}(x_2|x_1), F_{3|1}(x_3|x_1)\big) \\
&= f_3(x_3) \cdot c_{1,3}\big(F_1(x_1), F_3(x_3)\big) \cdot c_{2,3|1}\big(F_{2|1}(x_2|x_1), F_{3|1}(x_3|x_1)\big),
\end{aligned} \tag{2.3}$$

where the conditional CDFs are computed by

$$\begin{aligned}
F_{2|1}(x_2|x_1) &= \frac{\partial C_{1,2}\big(F_1(x_1), F_2(x_2)\big)}{\partial F_1(x_1)}, \\
F_{3|1}(x_3|x_1) &= \frac{\partial C_{1,3}\big(F_1(x_1), F_3(x_3)\big)}{\partial F_1(x_1)},
\end{aligned}$$

giving us the joint PDF

$$\begin{aligned}
f(x_1, x_2, x_3) = f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3) \cdot c_{1,2}\big(F_1(x_1), F_2(x_2)\big) \cdot c_{2,3|1}\big(F_{2|1}(x_2|x_1), F_{3|1}(x_3|x_1)\big) \\
\cdot c_{1,3}\big(F_1(x_1), F_3(x_3)\big).
\end{aligned}$$

It should be noted that this decomposition is not unique. Indeed, the ordering of the variables in Equation (2.2) is arbitrary. Furthermore, we could have chosen $\frac{f_{1,3|2}}{f_{1|2}}$ on the right-hand-side of Equation (2.3). For most parametric families, these decompositions are not equivalent whenever the simplifying assumption is assumed. Remark that the found joint density is Markovian w.r.t. the Bayesian network displayed in Figure 2.14, highlighting the connection between the two concepts.

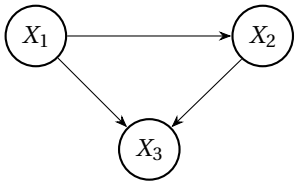


Figure 2.14: Bayesian network with 3 nodes.

2.5.5. h-functions

In Section 2.5.4 we have seen that the pair-copula construction requires the computation of conditional copulas which in turn require the computation of the conditional margins. That is, we must compute expressions of the type $F_{v|K}(x_v|\mathbf{x}_K)$ where $K \subseteq V$ and $\{v\} \cap K = \emptyset$. Using the notation $K_{-w} = K \setminus \{w\}$ we have that for all $w \in K$

$$F_{v|K}(x_v|\mathbf{x}_K) = \frac{\partial C_{v,w|K_{-w}}\left(F_{v|K_{-w}}(x_w|\mathbf{x}_{K_{-w}}), F_{w|K_{-w}}(x_v|\mathbf{x}_{K_{-w}})\right)}{\partial F_{w|K_{-w}}(x_w|\mathbf{x}_{K_{-w}})} \tag{2.4}$$

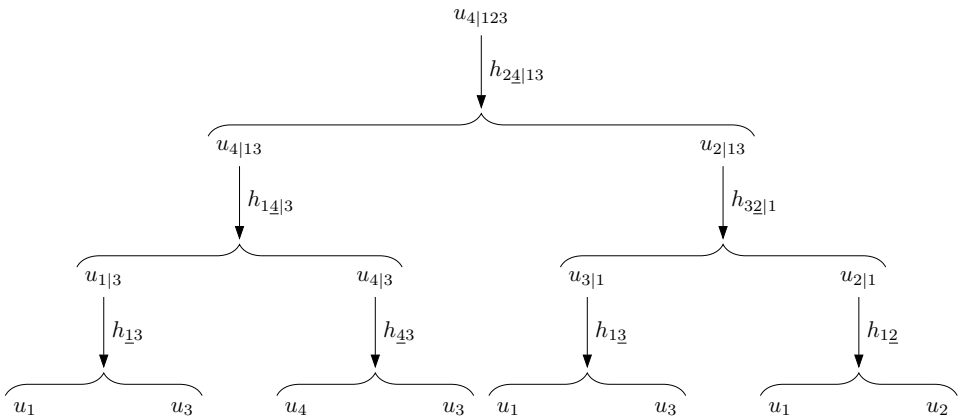
by [17]. Since this expression is quite cumbersome, a shorter notation for the conditional margins is used throughout the literature; "h-functions". Suppose that $v, w \in V$, then

$$h_{\underline{v},w}(F_v(x_v), F_w(x_w)) = \frac{\partial C_{v,w}(F_v(x_v), F_w(x_w))}{\partial F_w(x_w)} = F_{v|w}(x_v|x_w).$$

similarly $h_{v,\underline{w}} = F_{w|v}(x_w|x_v)$. The h-functions extend naturally to larger conditioning sets. For example, Equations (2.4) is written as

$$F_{v|K}(x_v|\mathbf{x}_K) = h_{\underline{v},w|K_{-w}}(F_{v|K_{-w}}(\mathbf{x}_{v|K_{-w}}), F_{w|K_{-w}}(\mathbf{x}_{w|K_{-w}})).$$

The h-functions are recursive functions, in the sense that their inputs are two lower dimensional conditional margins. These margins can be conditional margins which must again be computed with h-functions. This recursion continues until we are left with unconditional margins. For example, the conditional margin $u_{2|143}$ can be computed with the recursion displayed below. This recursion is certainly not unique.



2.5.6. Notation

For a general multivariate random vector $\mathbf{X} = (X_1, \cdots, X_n)$ with non-uniform margins we will need to use expressions of the form $f_i(x_i)$, $F_i(x_i)$ or $F_i|K(x_i|\mathbf{x}_K)$ in our equations. This notation is cumbersome and results in long equations. Moreover, we will be using a two-step process during estimation. To simplify presentation we first remove the information of the marginals leaving us with uniform data. The main focus of this thesis lies in the copula part of the estimation process. Therefore, we will only discuss uniform random variables

$\mathbf{U} = (U_1, \cdots, U_n)$. This is rather convenient since then we have that $f_i(u_i) = 1$ and $F_i(u_i) = u_i$. Note that the term $F_{i|K}(u_i|u_K)$ is not necessarily uniform and will be denoted by $u_{i|K}$.

For example, the joint density of a random vector (X_1, X_2, X_3) can be written as

$$f(x_1, x_2, x_3) = f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3) \cdot c_{123}\big(F_1(x_1), F_2(x_2), F_3(x_3)\big).$$

The estimation of the univariate PDFs, f_1, f_2 and f_3, falls into the first step of the estimation process. We will assume that this part has been completed, allowing us to apply the probability integral transform to find that

$$f(x_1, x_2, x_3) = \hat{f}_1(x_1) \cdot \hat{f}_2(x_2) \cdot \hat{f}_3(x_3) \cdot c_{123}\big(\hat{F}_1(x_1), \hat{F}_2(x_2), \hat{F}_3(x_3)\big).$$

The second step of the estimation process involves finding a decomposition for $c_{123}\big(u_1, u_2, u_3\big)$ consisting of bivariate copulas, and estimating these copulas. For example, if f is Markovian w.r.t. the Bayesian network in Figure 2.14, then c_{123} can be decomposed as

$$c_{123}\big(u_1, u_2, u_3\big) = c_{1,2}\big(u_1, u_2\big) \cdot c_{2,3|1}\big(u_{2|1}, u_{3|1}\big) \cdot c_{1,3}\big(u_1, u_3\big).$$

Hence, throughout this thesis, we will concern ourselves with finding the decomposition of PDFs with univariate marginal distributions denoted by c. Finally, we will also be dropping the commas in the subscripts, since in most cases this will not pose any problems. Thus, the equation reduces further to

$$c(u_1, u_2, u_3) = c_{12}\big(u_1, u_2\big) \cdot c_{23|1}\big(u_{2|1}, u_{3|1}\big) \cdot c_{13}\big(u_1, u_3\big).$$

3

Pair copula Bayesian network

In the previous chapter, we saw that a density f_V, which is induced by a BN, can be factorized by

$$f_V(\mathbf{x}_V) = \prod_{v \in V} f_{v|pa(v)}(x_v|\mathbf{x}_{pa(v)}).$$

Furthermore, we have discussed the pair-copula construction. Here, it was shown that we can decompose conditional densities using bivariate conditional copulas. By decomposing all conditional densities in the factorization of the BN, we can construct a pair-copula Bayesian network (PCBN). Before presenting the general definition of the PCBN, we will examine several examples. Consider the BN displayed in Figure 3.1.

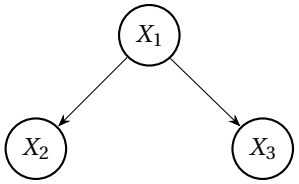


Figure 3.1: Simple Bayesian network.

The joint density can be factorized as

$$f(x_1, x_2, x_3) = f_1(x_1) \cdot f_{2|1}(x_2|x_1) \cdot f_{3|1}(x_3|x_1).$$

In Section 2.5.4 we have seen that the conditional densities $f_{2|1}$ and $f_{3|1}$ can be decomposed by

$$f_{2|1}(x_2|x_1) = f_2(x_2) \cdot c_{12}\big(F_1(x_1), F_2(x_2)\big),$$
$$f_{3|1}(x_3|x_1) = f_3(x_3) \cdot c_{13}\big(F_1(x_1), F_3(x_3)\big).$$

By substituting these expressions into the factorization we find that

$$f(x_1, x_2, x_3) = f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3) \cdot c_{12}\big(F_1(x_1), F_2(x_2)\big) \cdot c_{13}\big(F_1(x_1), F_3(x_3)\big).$$

Thus, we need copulas c_{12} and c_{13} which correspond to the arcs $X_1 \to X_2$ and $X_1 \to X_3$, respectively. So, the graph immediately tells us which copulas are required in order to decompose all conditional densities using the pair-copula construction. However, this is not always the case. Specifically, when we have nodes with multiple parents, i.e. v-structures. For such nodes, we need to define a total order on the parental set. For instance, let us examine the BN displayed in Figure 3.2.

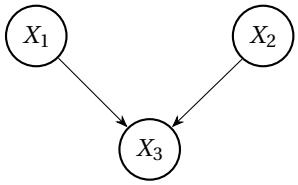


Figure 3.2: Simple Bayesian network with a v-structure.

The joint density can be factorized by

$$f(x_1, x_2, x_3) = f_1(x_1) \cdot f_2(x_2) \cdot f_{3|12}(x_3|x_1, x_2).$$

By using the same computation as in Section 2.5.4, we can decompose the conditional density $f_{3|12}$ in two (generally) distinct ways;

$$\begin{aligned} f_{3|12}(x_3|x_1, x_2) &= f_3(x_3) \cdot c_{23}\big(F_2(x_2), F(x_3)\big) \cdot c_{13|2}\big(F_{1|2}(x_1|x_2), F_{3|2}(x_3|x_2)|x_2\big), \\ &= f_3(x_3) \cdot c_{13}\big(F_1(x_1), F(x_3)\big) \cdot c_{23|1}\big(F_{2|1}(x_2|x_1), F_{3|1}(x_3|x_1)|x_1\big). \end{aligned}$$

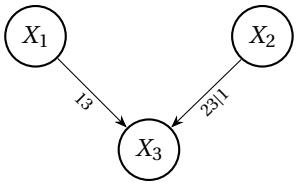
Theoretically, these decompositions are equal. However, they will not provide the same estimators during inference and might be very different when the simplifying assumption is assumed. Thus, we have two distinct factorizations of the joint density,

$$f(x_1, x_2, x_3) = f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3) \cdot c_{13}\big(F_1(x_1), F_3(x_3)\big) \cdot c_{23|1}\big(F_{2|1}(x_2|x_1), F_{3|1}(x_3|x_1)\big|x_1\big)$$

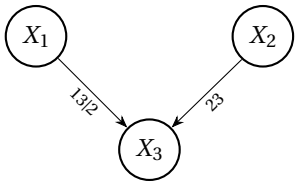
and

$$f(x_1, x_2, x_3) = f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3) \cdot c_{23}\big(F_2(x_2), F_3(x_3)\big) \cdot c_{13|2}\big(F_{1|2}(x_1|x_2), F_{3|2}(x_3|x_2)\big|x_2\big).$$

The graph does not specify which decomposition is used. Therefore, we define a total order on the parental set of each node, see Definition 2.18. The order of the parents will determine which copulas are used in the decomposition. For this example, the order $1 <_3 2$ implies that we will be using the copulas c_{13} and $c_{23|1}$. Hence, the arc connecting node 3 with the first parent (1) in the total order is assigned the unconditional copula c_{13}, and the arc connecting node 3 with the second parent (2) in the order is assigned the conditional copula $c_{23|1}$. Similarly, the order $2 <_3 1$ implies the usage of copulas c_{23} and $c_{13|2}$. When displaying a PCBN in a figure we will assign the copulas along the arcs of the graph. Here, we do not fully write out c_{13} or $c_{23|1}$, but simply put 13 and 23|1, respectively. The resulting PCBNs are displayed in Figure 3.3.



(a) $1 <_3 2$



(b) $2 <_3 1$

Figure 3.3: Simple PCBNs with a v-structure.

This concept is easily extended to the general case where we have an arbitrary DAG $\mathcal{G}$. For each node $v \in V$ and $w \in pa(v)$, the arc $w \to v$ is assigned the copula $c_{wv|pa(v\downarrow w)}$, where the set $pa(v \downarrow w) := \{z \in pa(v); z <_v w\}$ contains all parents earlier than w in the order, see Definition 2.20. For example, the graph in Figure 3.4, with $1 <_4 2 <_4 < 3$, has corresponding copulas

$$\begin{aligned} c_{14|pa(4\downarrow 1)} &= c_{14}, \\ c_{24|pa(4\downarrow 2)} &= c_{24|1}, \\ c_{34|pa(4\downarrow 3)} &= c_{34|12}. \end{aligned}$$

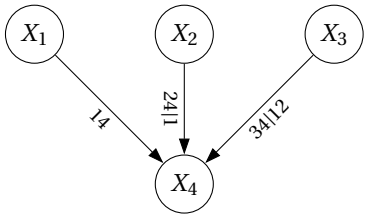


Figure 3.4: PCBN where node X_4 has corresponding parental order $1 <_4 2 <_4 3$.

The graphical structure and the parental orders will provide us with an assignment of copulas such that we can decompose the joint density using pair copula constructions. Moreover, this decomposition will be in accordance with the conditional indecencies induced by the d-separations in the graph. Thus, in the formal definition of the PCBN, we must include the parental order for each node.

Definition 3.1 (Pair-copula Bayesian network)**.** A ***pair-copula Bayesian network*** is composed of

- a pair $(\mathcal{G},\mathcal{O})$ consisting of a DAG $\mathcal{G}$ and a collection of orderings $\mathcal{O} = \{<_V;\ v \in V\}$,
- a collection marginal densities $\{f_v;\ v \in V\}$,
- a collection of (conditional) copulas $\{c_{wv|pa(v\downarrow w)};\ w \to v \in E\}$.

The set of conditional independencies allows for the decomposition of the joint density as a product of the marginal densities and copulas;

$$f_V(\mathbf{x}_V) = \prod_{v\in V} f_v(x_v) \prod_{w\in pa(v)} c_{wv|pa(v\downarrow w)}\big(F_{w|pa(v\downarrow w)}(x_w|\mathbf{x}_{pa(v\downarrow w)}), F_{v|pa(v\downarrow w)}(x_v|\mathbf{x}_{pa(v\downarrow w)})\,\big|\,\mathbf{x}_{pa(v\downarrow w)}\big). \tag{3.1}$$

An important remark is that the set of probability measures induced by a PCBN $(\mathcal{G},\mathcal{O})$ is equal to the set of measures induced by the BN $\mathcal{G}$. This is due to the fact that any Markovian density w.r.t. $\mathcal{G}$ can be decomposed according to Equation (3.1) regardless of the choice of $\mathcal{O}$. This result follows by Theorem 3.2 which was proven in [4].

Theorem 3.2. Let $(\mathcal{G},\mathcal{O})$ be a PCBN and let P be an absolutely continuous probability distribution with strictly increasing univariate marginal CDFs which is Markovian w.r.t. $\mathcal{G}$. Then, P is uniquely determined by its margins and the set of conditional pair copulas $\{c_{wv|pa(v\downarrow w)};\ w \to v \in E\}$. Moreover, the density can be factorized as in Equation (3.1).

In Section 2.3, we have seen that two GBNs with equivalent graphs induce the same set of probability measures. This is not the case for the PCBN. Indeed, two PCBNs can have an equivalent DAG but a different assignment of copulas, and hence distinct distributions. Now, consider two DAGs whose distance as defined in Definition 2.35 is equal to zero. Then, these graphs have the same v-structures and skeleton. Hence, for both graphs we can assign the copulas such that their corresponding PDFs are the same.

The PCBN $(\mathcal{G},\mathcal{O})$ provides us with a set of conditional copulas which are assigned to arcs in the graph; $\{c_{wv|pa(v\downarrow w)};\ w \to v \in E\}$. These copulas are said to be **specified** by the PCBN. That is, when applying a PCBN model they are assumed to be known. For example, for the PCBN in Figure 3.4 we have

$$\{c_{wv|pa(v\downarrow w)};\ w \to v \in E\} = \{c_{14}, c_{24|1}, c_{34|12}\}.$$

Furthermore, the graph of a PCBN induces conditional independencies between random variables by d-separation. For example, for the PCBN in Figure 3.4 we have

$$d - sep_{\mathcal{G}}\big(1,2\,\big|\,3\big).$$

Hence, the PCBN implies that X_1 and X_2 are independent given X_3, and thus we have that $c_{12|3}$ is equal to the independence copula. This copula is also regarded as specified. In general, if we have $d-sep_{\mathcal{G}}\big(w,v\,\big|\,K\big)$ with $w,v\in V$ and $K\subseteq V\setminus\{w,v\}$, then the copula $c_{wv|K}$ is equal to the independence copula. Hence, it is specified by the PCBN.

Later we will see that the computation of the joint density may require a copula which is not specified by the PCBN. In this case, we must compute this copula. Therefore, it will be convenient to have conditions in place that state whether a copula is specified by the PCBN or not.

For example, for the PCBN in Figure 3.4, the copulas $c_{14|2}$ and $c_{13|4}$ are not specified. Indeed, the former copula is not specified since the arc $1\to 4$ has been assigned the copula c_{14} which is not equal to $c_{14|2}$. The latter copula is not specified as there is no arc between nodes 1 and 3 and this copula is not the independence copula, since $\cancel{d-sep_{\mathcal{G}}}\big(1,3\,\big|\,4\big)$.

Thus, in general, a copula $c_{wv|K}$ with $w,v\in V$ and $K\subseteq V\setminus\{w,v\}$ is specified by the PCBN when it is assigned to an arc in the graph or equal to the independence copula by d-separation. Naturally, both types of arcs, $w\to v$ or $v\to w$, could be assigned with the copula above, which will be denoted as $c_{wv|pa(v\downarrow w)}$ and $c_{vw|pa(w\downarrow v)}$, respectively. For one of these two copulas to be equal to $c_{wv|K}$, we must have either $K=pa(v\downarrow w)$ or $K=pa(w\downarrow v)$. Hence, a copula $c_{wv|K}$ is assigned to an arc in the graph if one of the following is satisfied:

- $w\to v$ and $pa(v\downarrow w)=K$.
- $v\to w$ and $pa(w\downarrow v)=K$.

Thus, we have the following definition.

Definition 3.3. Let $(\mathcal{G},\mathcal{O})$ be a PCBN. A conditional copula $c_{wv|K}$ with $w,v\in V$ and $K\subseteq V\setminus\{w,v\}$ is ***specified*** by the PCBN if one of the following is satisfied:

(i) $d-sep_{\mathcal{G}}\big(w,v\,\big|\,K\big)$.

(ii) $w\to v$ and $pa(v\downarrow w)=K$.

(iii) $v\to w$ and $pa(w\downarrow v)=K$.

The expression found for the joint density in Equation (3.1) suffers from heavy notation. However, we can notice that this density is always built as a product of the marginal densities for nodes in V (the first product in Equation (3.1)) and a copula density (the second product in Equation (3.1)). This copula density is then decomposed as a product of copula densities assigned to arcs in the graph.

To simplify the notation, from this point on we will only discuss the copula density corresponding to the density f_V, which is denoted as c_V. This is equivalent to considering densities with uniform margins. Moreover, we will make use of the more straightforward notation for the arguments of the copulas introduced in Section 2.5.6 and assume that the simplifying assumption holds, which will allow us to drop the conditional terms $(\mathbf{x}_{pa(v\downarrow w)})$ from the copulas. Thus, instead of using Equation (3.1) we will be writing

$$c(\mathbf{u}_V)=\prod_{v\in V}\prod_{w\in pa(v)}c_{wv|pa(v\downarrow w)}\big(u_{w|pa(v\downarrow w)},u_{v|pa(v\downarrow w)}\big). \tag{3.2}$$

where the conditional margins $u_{w|pa(v\downarrow w)}$ and $u_{v|pa(v\downarrow w)}$ are computed with a recursion of h-functions (see Section 2.5.5).

For example, consider the PCBN in Figure 3.5. The parental orders of nodes 4 and 5 can be easily seen from the copulas assigned to the arks. The term $c_{45|123}\big(u_{4|123},u_{5|123}\big)$ appears in its joint density, which requires

the computation of the conditional margin $u_{4|pa(5\downarrow 4)} = u_{4|123}$. Remark that in Section 2.5.5, we displayed a recursion of h-functions to compute this margin. An important observation is that all h-functions in this recursion correspond to copulas which are specified by the PCBN. Indeed, the h-functions in this recursion are $h_{1\underline{2}}$, $h_{\underline{1}3}$, $h_{1\underline{3}}$, $h_{3\underline{4}}$, $h_{14|\underline{3}}$, $h_{23|1}$ and $h_{\underline{2}4|13}$ and they correspond to the copulas c_{12}, c_{13}, c_{34}, $c_{14|3}$, $c_{23|1}$ and $c_{24|13}$, which are all specified by the PCBN (c_{13}, $c_{23|1}$ are independent copulas).

If an h-function corresponds to a specified copula, then the h-function is also said to be **specified** and the recursion composed of only specified h-functions is called a **proper** recursion. There exist recursions of h-functions where some copulas are not specified. For example, to compute $u_{4|123}$ we could have chosen the recursion shown below. We can see that the h-function shown in red ($h_{1\underline{2}|3}$) corresponds to a copula ($c_{12|3}$) that is not specified by the PCBN. In such a case, we say that the recursion is **not proper**.

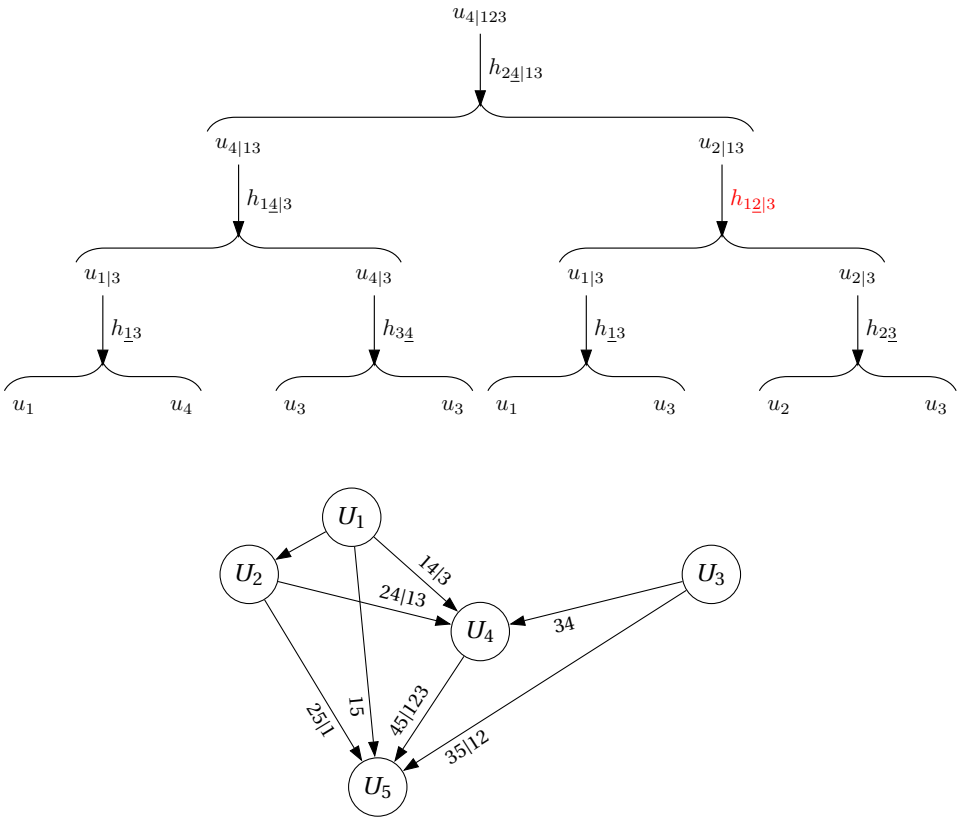


Figure 3.5: A PCBN consisting of five nodes.

Naturally, it would be convenient if all conditional margins in the joint density can be computed with a proper recursion. If an h-function is not specified, we must compute its corresponding copula. In the example above, we could simply take the recursion from Section 2.5.5. However, in the next section, we will see that in some cases a conditional margin cannot be computed with a proper recursion. That is, we will encounter conditional margins for which any possible recursion of h-functions contains a non-specified h-function.

3.1. Problematic conditional margins

We have seen that each conditional margin appearing in the joint density of a PCBN is computed with a recursion of h-functions. Problems occur if there exists no proper recursion to compute a conditional margin. In this case, the computation of the conditional margin will require integration. Let us investigate an example of a PCBN with such problematic conditional margins.

Example 3.4. Consider the PCBN displayed in Figure 3.6.

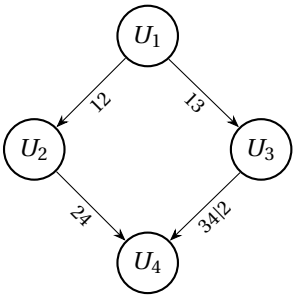


Figure 3.6: PCBN where conditional margin $u_{3|2}$ requires integration.

The density corresponding to this network can be factorized by

$$c(\mathbf{u}_V) = c_{12}\big(u_1, u_2\big) \cdot c_{13}\big(u_1, u_3\big) \cdot c_{24}\big(u_2, u_4\big) \cdot c_{34|2}\big(u_{3|2}, u_{4|2}\big).$$

Here, we need to compute the conditional margin $u_{3|2}$ by

$$u_{3|2} = h_{2\underline{3}}(u_2, u_3).$$

Thus, we need the copula c_{23}. Remark that $\cancel{d\text{-}sep_{\mathcal{G}}}\big(2,3 \,\big|\, \emptyset\big)$, and no arc in the network is assigned the copula c_{23}. Therefore, c_{23} is not specified by the PCBN, and thus we must compute the conditional margin by

$$\begin{aligned}
u_{3|2} &= h_{2\underline{3}}(u_2, u_3) \\
&= \int_0^{u_3} c_{23}(u_2, w_3)\, \mathrm{d}w_3 \\
&= \int_0^{u_3} \int_0^1 c_{123}(w_1, u_2, w_3)\, \mathrm{d}w_1\, \mathrm{d}w_3 \\
&= \int_0^{u_3} \int_0^1 c_{12}(w_1, u_2) \cdot c_{13}(w_1, w_3)\, \mathrm{d}w_1\, \mathrm{d}w_3 \\
&= \int_0^1 c_{12}(w_1, u_2) \cdot \Big(\int_0^{u_3} c_{13}(w_1, w_3)\, \mathrm{d}w_3 \Big)\, \mathrm{d}w_1 \\
&= \int_0^1 c_{12}(w_1, u_2) \cdot h_{1\underline{3}}(w_1, u_3)\, \mathrm{d}w_1.
\end{aligned} \tag{3.3}$$

So, we require integration in order to compute the conditional margin $u_{3|2}$. It should be noted that the conditional margin $u_{4|2}$ does not pose a problem since it can be computed with the h-function $h_{2\underline{4}}(u_2, u_4)$, whose corresponding copula c_{24} is specified by the PCBN.

The integrals appearing in the computation of the conditional margins do not pose a problem theoretically. In practice however, these integrals cannot in general be computed analytically. Hence, we have to recourse to numerical integration, which is not optimal as the estimation of the joint density compels us to integrate many times. In this example, integrating might still be doable, since it is merely a one-dimensional integral. However, we can construct graphs requiring integration in an arbitrary amount of dimensions. For example, for the graph in Figure 3.7, we must compute $u_{z|w}$ by

$$\begin{aligned}
u_{z|w} &= h_{w\underline{z}}(u_w, u_z) \\
&= \int_0^{u_z} c_{wz}(u_w, w_z)\, \mathrm{d}w_z \\
&= \int_0^{u_z} \int_0^1 \cdots \int_0^1 c_{1w}(w_1, u_w) c_{nz}(w_n, w_z) \prod_{i=1}^{n-1} c_{i,i+1}(w_i, w_{i+1})\, \mathrm{d}w_1 \ldots \mathrm{d}w_n\, \mathrm{d}w_z \\
&= \int_0^1 \cdots \int_0^1 c_{1w}(w_1, u_w) h_{n\underline{z}}(w_n, w_z) \prod_{i=1}^{n-1} c_{i,i+1}(w_i, w_{i+1})\, \mathrm{d}w_1 \ldots \mathrm{d}w_n.
\end{aligned}$$

Thus, it is interesting to find out under which conditions the computation of the joint density does not require integration. We remark that the graphical structure in Figure 3.7 is an example of an active cycle, which will be defined in Section 3.3. It will be seen that active cycles always require integration.

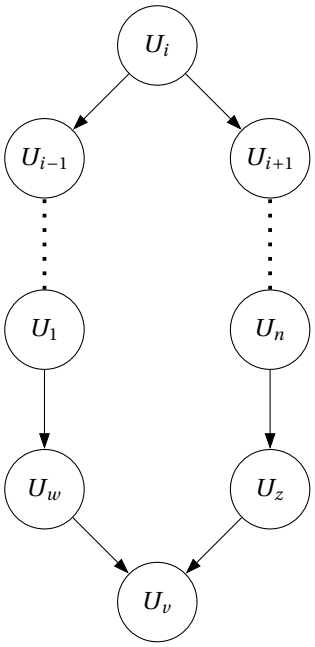


Figure 3.7: Graph where the integration of $u_{z|w}$ requires computation in n dimensions.

We do not need to integrate if all conditional margins appearing in the joint density can be computed with copulas that are specified by the PCBN. For example, computing the conditional margin

$$u_{3|2} = h_{2\underline{3}}(u_2, u_3)$$

will not be a problem for the PCBNs displayed in Figure 3.8. In the PCBN (a), the copula c_{23} is specified by the arc $2 \to 3$, and in the PCBN (b), c_{23} is specified because it is the independent copula due to the d-separation $d-sep_{\mathcal{G}}(2,3\,|\,\emptyset)$.

Figure 3.8: Two PCBN where the conditional margin $u_{3|2}$ does not require integration.

An important remark is that in many cases, certain orderings $\mathcal{O}$ necessitate integration, while others do not. For example, in Figure 3.8a, if we change the order of $<_3$ to $1 <_3 2$, then the copula c_{23} is not specified by the PCBN. Hence, we can no longer compute the conditional margin $u_{3|2}$ without integration.

Thus, to prevent the need for integration in the evaluation of the joint density, it will not suffice to only provide conditions on the allowed graphical structure of the graph. We also need to choose the order of parents $\mathcal{O}$ in a particular manner.

In the example above, the cardinality of the conditioning set of the conditional margin ($u_{3|2}$) was equal to one. Hence, one h-function is needed. In such a case it is easy to determine whether this h-function is specified or not.

If we need to compute a conditional margin of the form $u_{v|K}$, with $|K| > 1$, then all conditional margins in the recursion presented already have to be considered.

For example, for a conditional margin $u_{v|K}$, we can pick any $k \in K$ and compute the margin by

$$u_{v|K} = h_{k\underline{v}|K\setminus\{k\}}\left(u_{k|K\setminus\{k\}}, u_{v|K\setminus\{k\}}\right).$$

Naturally, different choices of k will lead to different conditional margins, i.e. $u_{k|K\setminus\{k\}}$, $u_{v|K\setminus\{k\}}$ as arguments of $h_{k\underline{v}|K\setminus\{k\}}$. To be able to compute $u_{v|K}$ without integration we will need to choose k such that both $u_{k|K\setminus\{k\}}$ and $u_{v|K\setminus\{k\}}$ can be computed with h-functions specified by the PCBN.

Such a recursion may not exist, as was seen for the conditional margin $u_{3|2}$ in the PCBN of Figure 3.6. In this case, the conditional margin is said to require integration. Hence, the joint density requires integration if its copula decomposition contains a conditional margin for which there does not exist a proper recursion.

Let us examine a more complicated example where the computation of the joint density of a PCBN requires integration.

Example 3.5. Consider the PCBN in Figure 3.9.

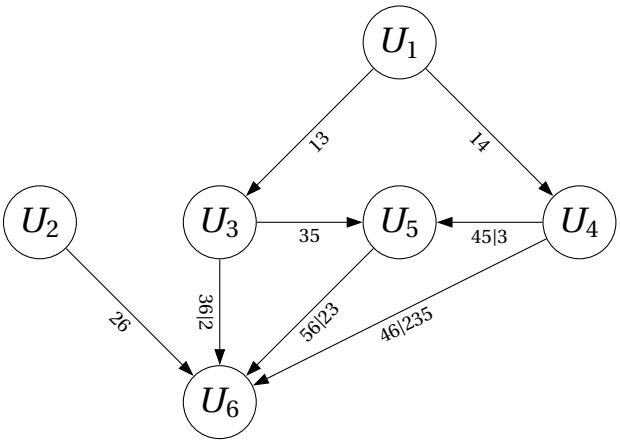
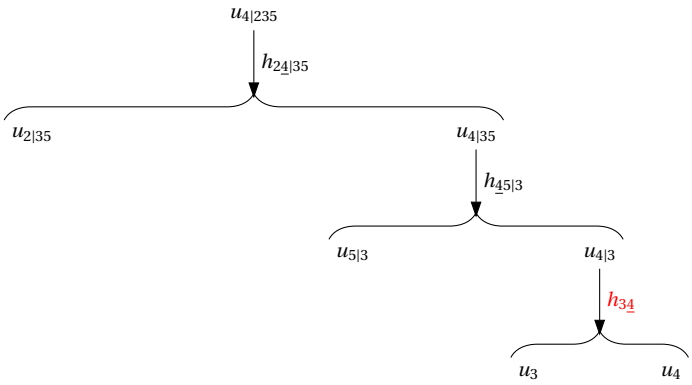


Figure 3.9: PCBN where the computation of $u_{4|235}$ requires integration.

To compute the joint density, we need to compute the conditional margin $u_{4|235}$ which is an argument of copula $c_{46|235}$. We will demonstrate that there exists no recursion of specified h-functions to compute this conditional margin.

To start the recursive process we may consider the following h-functions: $h_{2\underline{4}|35}$, $h_{3\underline{4}|25}$ or $h_{\underline{4}5|23}$. Only the first one is specified by the PCBN as the corresponding copula $c_{24|35}$ is the independent copula. This is because the $d - sep_{\mathcal{G}}\left(2,4 \middle| \{3,5\}\right)$ holds. Hence, to avoid integration, the recursion must start with $h_{2\underline{4}|35}$. Now, we have to compute the conditional margins $u_{2|35}$ and $u_{4|35}$. The former can be computed with no issues as U_2 is

independent of both U_3 and U_5. Hence, we focus on the latter margin, which can be computed with $h_{3\underline{4}|5}$, which is not specified, or $h_{\underline{4}5|3}$, which is specified. Thus, $u_{4|35}$ is computed with $h_{\underline{4}5|3}$, whose arguments are $u_{4|3}$ and $u_{5|3}$. The margin $u_{5|3}$ is computed from the specified copula c_{35}, but $u_{4|3}$ has to be computed with $h_{3\underline{4}}$, whose corresponding copula is not specified by the PCBN. Hence, we cannot construct a recursion of specified h-functions to compute the conditional margin $u_{4|235}$. The process is shown graphically below.



Note that the reason that we cannot compute the margin $u_{4|235}$ without integration is that nodes U_1, U_3, U_4, and U_6 form an active cycle, which will be defined and discussed in Section 3.3.

To show that a density cannot be computed without integration (a recursion of specified copulas does not exist) it suffices to investigate a specific part of the recursion. In the example above, the computations of conditional margins $u_{4|K}$ with $K \subseteq pa(6 \downarrow 4) := \{2,3,5\}$ were investigated. The same approach will be used later, in Sections 3.3, 3.4 and 3.5, to prove that certain choices of parental orderings or certain graphical structures for all choices of orderings require integration.

In proving the results in this thesis, we will often consider arguments similar to the ones described in the previous example. We will investigate a particular part of the recursion. That is, for a conditional margin of the form $u_{w|pa(v\downarrow w)}$ we will show that we cannot construct the part of the recursion involving margins of the form $u_{w|K}$ with $K \subseteq pa(v \downarrow w)$.

The conditional margins which appear in the factorization of the joint density in Equation (3.1) are of the form $u_{w|pa(v\downarrow w)}$ and $u_{v|pa(v\downarrow w)}$. It should be noted that we can always find a specified h-function to start the recursion for the computation of $u_{v|pa(v\downarrow w)}$. Since, for z the largest element in $pa(v \downarrow w)$ with respect to order $<_v$, we have that the copula $c_{zv|pa(v\downarrow z)}$ is specified by the definition of the PCBN. Moreover, we have $pa(v \downarrow z) = pa(v \downarrow w) \setminus \{z\}$. Hence, we can compute the margin by

$$u_{v|pa(v\downarrow w)} = h_{z\underline{v}|pa(v\downarrow z)}\big(u_{z|pa(v\downarrow z)}, u_{v|pa(v\downarrow z)}\big),$$

where the margins $u_{z|pa(v\downarrow z)}$ and $u_{v|pa(v\downarrow z)}$ also appear in the factorization of the joint density. For example, for the PCBN in Figure 3.9, the conditional margin $u_{6|pa(6\downarrow 4)} = u_{6|235}$ can be computed with $c_{56|pa(6\downarrow 5)}$ by

$$u_{6|235} = h_{5\underline{6}|23}\big(u_{5|23}, u_{6|23}\big),$$

where $u_{5|23} = u_{5|pa(6\downarrow 5)}$ and $u_{6|23} = u_{6|pa(6\downarrow 5)}$ appear in the joint density.

If we assume that all conditional margins of the form $u_{w|pa(v\downarrow w)}$ are computable without integration, then finding a recursion of specified h-functions for a margin $u_{v|pa(v\downarrow w)}$ becomes quite trivial. Indeed, we simply start the recursion with a conditional copula $c_{zv|pa(v\downarrow z)}$, which requires the computation of the margin $u_{v|pa(v\downarrow z)}$. Now, we repeat the same process for $u_{v|pa(v\downarrow z)}$.

Thus, to show that the joint density can be computed without integrating, it is sufficient to prove that any conditional margin of the form $u_{w|pa(v\downarrow w)}$ can be computed without integration. This result is proven formally in the lemma below.

Lemma 3.6. Let $(\mathcal{G},\mathcal{O})$ be a PCBN. If any conditional margin of the form $u_{w|pa(v\downarrow w)}$ does not require integration, then the joint density can be computed without integration

Proof. It suffices to prove that all terms of the form $u_{v|pa(v\downarrow w)}$ can be computed without integration.

Consider an arbitrary node $v \in V$. If, $\left|pa(v)\right| = 0$, then $pa(v) = \emptyset$, hence no terms of the form $u_{v|pa(v\downarrow w)}$ appear in the factorization of f_V. If, $\left|pa(v)\right| = 1$, then for w the single node in $pa(v)$, we have that c_{wv} is specified. Thus, we can compute $u_{v|w}$ by

$$u_{v|w} = h_{w\underline{v}}\big(u_w, u_v\big).$$

Assume that $\left|pa(v)\right| > 1$ and let $pa(v) := \{w_1,\dots,w_n\}$ be an ordered set according to $<_v$. We must show that for all $i \in \{1,\dots,n\}$ the conditional margins $u_{v|pa(v\downarrow w_i)}$ do not require integration.

We use induction. As above, for $i = 1$, the statement clearly holds. Now, suppose that $u_{v|pa(v\downarrow w_{i-1})}$ can be computed without integration. The conditional margin $u_{v|pa(v\downarrow w_i)}$ can be computed by

$$u_{v|pa(v\downarrow w_i)} = \frac{\partial C_{w_{i-1},v|pa(v\downarrow w_{i-1})}\big(u_{w_{i-1}|pa(v\downarrow w_{i-1})}, u_{v|pa(v\downarrow w_{i-1})}\big)}{\partial u_{w_{i-1}|pa(v\downarrow w_{i-1})}},$$

where the copula $C_{w_{i-1},v|pa(v\downarrow w_{i-1})}$ is specified by the PCBN. Moreover, the conditional margin $u_{v|pa(v\downarrow w_{i-1})}$ does not require integration by the induction hypothesis, and by assumption, $u_{w_{i-1}|pa(v\downarrow w_{i-1})}$ does not require integration by the assumptions of the lemma. Hence, the conditional margin $u_{v|pa(v\downarrow w_i)}$ does not require integration. □

To summarize, we have seen that the computation of the joint density of a PCBN may require integration. This can be caused by the structure of the graph or choice of $\mathcal{O}$. Therefore, it is of interest to determine conditions on the graph structure that guarantee the existence of an order $\mathcal{O}$ such that the joint density does not require integration. To achieve this, first in Section 3.2 we discuss a subclass of graphs for which the integration is not needed irrespective of the choice of order $\mathcal{O}$. In Sections 3.3 and 3.4, two types of graphical structures, namely actives cycles and interfering v-structures, are presented which will always require integration. Moreover, we will prove that for any graph not containing these structures, we can choose $\mathcal{O}$ such that the joint density does not require integration. This result is proven in Theorem 3.15 which is the main result of this thesis.

3.2. Multitrees

The simple graphical structure which does not lead to integration irrespective of the copula assignments, occurs when all parents are mutually independent, i.e. d-separated given the empty set. PCBNs with graphs satisfying this restriction will not require integration. To see this, we investigate an example of such a PCBN.

Example 3.7. Consider the PCBN displayed in Figure 3.10 which has corresponding factorization

$$f_V(\mathbf{u}_V) = c_{14}(u_1, u_4) \cdot c_{25}(u_2, u_5) \cdot c_{35|2}(u_{3|2}, u_{5|2}) \cdot c_{45|23}(u_{4|23}, u_{5|23}).$$

By Lemma 3.6, we only have to show that conditional margins of the form $u_{w|pa(v)\setminus w)}$ do not require integration. Hence, the conditional margins of interest are $u_{3|2}$ and $u_{4|23}$. Remark that U_2, U_3, and U_4 are mutually independent due to the d-separations represented by the graph. Thus, we have $u_{3|2} = u_3$ and $u_{4|23} = u_4$.

So, we have found expressions for all conditional margins without any integrals. Moreover, the same principle can be applied regardless of the choice of parental order $<_5$, since any conditional margin is directly computable due to mutual independence of the parents.

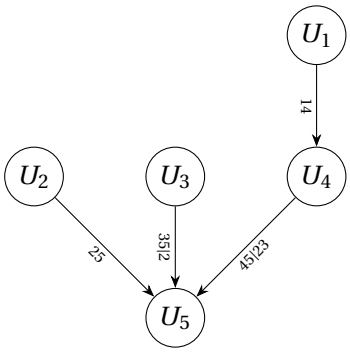


Figure 3.10: PCBN with mutually independent parents.

From Example 3.7, it is clear that we do not require integration if all parents are mutually independent, i.e. for all $v \in V$ and $w, z \in pa(v)$ with $w \neq z$ we have that $d - sep_{\mathcal{G}}(w, z \,|\, \emptyset)$. This is exactly the case for the subclass of graphs called multitrees, see Definition 2.24 and Theorem 2.25. Thus, a PCBN $(\mathcal{G}, \mathcal{C})$ where $\mathcal{G}$ is a multitree does not require integration. We will now prove this in full generality.

Theorem 3.8. Let $(\mathcal{G}, \mathcal{C})$ be a PCBN where $\mathcal{G}$ is a multitree. Then, the corresponding joint density can be computed without integration.

Proof. By Theorem 2.25, for all $v \in V$ and $w, z \in pa(v)$ with $w \neq z$, we have that $d - sep_{\mathcal{G}}(w, z \,|\, \emptyset)$. Hence, for any recursion used to compute a conditional margin of the form $u_{w|pa(v)\setminus w)}$, the copulas corresponding to the h-functions will be specified by the PCBN. That is, they are all equal to the independence copula. Hence, the computation of $u_{w|pa(v)\setminus w)} = u_w$ does not require integration. Consequently, Lemma 3.6 implies that the joint density does not require integration, completing the proof. □

It should be remarked that for any multitree $\mathcal{G}$, the pair $(\mathcal{G}, \mathcal{C})$ will not require integration for any choice of $\mathcal{C}$. Furthermore, it is not the case that multitrees are the only graphs for which integration is not needed (e.g. the PCBN in Figure 3.8a).

Before we state the main results of this thesis several graphical structures that necessitate integration are investigated.

3.3. Active cycles

We have seen that for the PCBN in Figure 3.6, we will need to determine the copula c_{23} through integration in order to compute the conditional margin $u_{3|2}$. This is because the copula c_{23} is not specified. Indeed, there is no arc between nodes 2 and 3, and we have $\not{d-sep}_{\mathcal{G}}(2,3\,|\,\emptyset)$ by the trail $2 \leftarrow 1 \rightarrow 3$. Note that this trail combined with the v-structure $2 \rightarrow 4 \leftarrow 3$ forms the cycle $4-2-1-3-4$ in the corresponding undirected graph. Such undirected cycles will always lead to a problematic conditional margin. We will refer to these structures as active cycles.

Definition 3.9. Let $\mathcal{G}$ be a DAG. Consider a node $v \in V$ with distinct parents $w, z \in pa(v)$ which are connected by a trail $w \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons z$ satisfying the following conditions:

(i) $n > 1$.

(ii) $w \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons z$ consists of only diverging or serial connections, see Definition 2.7.

(iii) $v \leftarrow w \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons z \rightarrow v$ contains no chords, see Definition 2.8.

Then, the trail $v \leftarrow w \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons z \rightarrow v$ is called an ***active cycle*** in $\mathcal{G}$. Furthermore, $\mathcal{G}$ is said to contain an active cycle.

The presence of an active cycle in the graph necessitates integration. This statement is proven in Theorem 3.11. Before proving the theorem we will first establish a convenient lemma. This lemma will be used in the proofs of Theorem 3.11, Theorem 3.14, and Lemma 3.20.

In each of these proofs, we will show that there is a conditional margin of the form $u_{w|pa(v\downarrow w)}$ for which there is no recursion consisting of specified copulas. In Example 3.5, we have seen that it suffices to only examine a specific part of the recursion. That is, we are only concerned with the computation of conditional margins of the form $u_{w|K}$ with $K \subseteq pa(v \downarrow w)$. In this part of the recursion, we repeatedly apply the following operations starting with $K = pa(v \downarrow w)$:

1. We pick a $k \in K$ such that the copula $c_{kw|K_{-k}}$ is specified by the PCBN, where we use the notation $K_{-k} := K \setminus \{k\}$. If there is no such k, then $u_{w|K_{-k}}$ must be computed using integration.

2. We compute the conditional margin by

$$u_{w|K} = h_{\underline{w}k|K_{-k}}(u_{w|K_{-k}}, u_{k|K_{-k}})$$

   which requires the computation of $u_{w|K_{-k}}$.

3. Return to Step 1, with $K = K_{-k}$.

In Step 1, there can be multiple choices for k, each resulting in a distinct recursion. Remark that for any recursion, we can define an ordered set $O := (o_1, \ldots, o_n) = pa(v \downarrow w)$ such that $i < j$ implies that o_j is picked before o_i in Step 1. For example, the recursion to compute $u_{4|234}$ in Example 3.5 has $O = (3,5,2)$. Consequently, any conditional margin $u_{w|o_1,\ldots,o_i}$ in the recursion is computed with $h_{o_i\underline{w}|o_1,\ldots,o_{i-1}}$. Indeed, the recursion in Example 3.5 contains the h-functions $h_{2\underline{4}|35}$, $h_{\underline{4}5|3}$ and $h_{3\underline{4}}$. Naturally, to prevent integration, O must be ordered such that any copula $c_{o_i w|o_1,\ldots,o_{i-1}}$ is specified by the PCBN. If there is no such O, then there is no recursion of specified h-functions [1] .

For an ordered set O to exist, there cannot be a node $z \in O$ for which no copula of the form $u_{zw|K}$ with

[1] This ordered set O bears great resemblance to the partial order we will later define in Section 4.1. Indeed, they are based on the same intuition; the parents must be ordered in a particular way. One of the differences being that this O is a subset of $pa(v \downarrow w)$ while the partial order is a subset of $pa(v)$.

$K \subseteq pa(v \downarrow w) \setminus \{z\}$ is specified. Indeed, suppose that $z := o_i$, then $c_{zw|o_1,\dots,o_{i-1}}$ would not be specified. In Example 3.5, node 3 is exactly such a node z, since none of the copulas c_{34}, $c_{34|2}$, $c_{34|5}$ and $c_{34|25}$ are specified. Thus, we have the following lemma.

Lemma 3.10. Let $(\mathcal{G},\mathcal{O})$ be a PCBN. Consider the nodes $v \in V$ and $w \in pa(v)$. The computation of the conditional margin $u_{w|pa(v\downarrow w)}$ requires integration if there exists a node $z \in pa(v \downarrow w)$ such that for all $K \subseteq pa(v \downarrow w) \setminus \{z\}$ the copula $c_{zw|K}$ is not specified by the PCBN.

Proof. Pick a recursion to compute $u_{w|pa(v\downarrow w)}$. Let $O := (o_1,\dots,o_n) = pa(v \downarrow w)$ such that for all $i \in \{1,\dots,n\}$ the h-function $h_{o_i w|o_1,\dots,o_{i-1}}$ appears in the recursion. Let $j \in \{1,\dots,n\}$ such that $o_j = z$. Subsequently, the conditional margin $u_{w|o_1,\dots,o_{j-1},z}$ is computed with $h_{z\underline{w}|o_1,\dots,o_{j-1}}$. Since $\{o_1,\dots,o_{j-1}\} \subseteq pa(v \downarrow w) \setminus \{z\}$, we have that $c_{zw|o_1,\dots,o_{j-1}}$ is not specified by the PCBN. Hence, the computation of $u_{w|pa(v\downarrow w)}$ requires integration □

Now we are ready to prove that a PCBN containing an active cycle will require integration.

Theorem 3.11. Let $(\mathcal{G},\mathcal{O})$ be a PCBN. If $\mathcal{G}$ contains an active cycle, then the computation of the joint density requires integration.

Proof. Consider an active cycle in $\mathcal{G}$ of the form

$$v \leftarrow w \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons z \rightarrow v.$$

Since w and z are both parents of v, we have either $w <_v z$ or $z <_v w$. Without loss of generality, we assume that $z <_v w$. We need to prove that the margin $u_{w|pa(v\downarrow w)}$ requires integration. Due to Lemma 3.10, we must show that for any $K \subseteq pa(v \downarrow w) \setminus \{z\}$ the copula $c_{zw|K}$ is not specified by the PCBN. Consider an arbitrary $K \subseteq pa(v \downarrow w) \setminus \{z\}$.

Note that w and z are not adjacent and the trail between w and z is not blocked by any subset of nodes in $pa(v)$, because of the existence of trail $w \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons z$ without a cord. Because $K \subseteq pa(v \downarrow w) \subseteq pa(v)$, we have that $\not{d}\text{-}\not{sep}_{\mathcal{G}}(w,z \mid K)$. Thus, the copula $c_{zw|K}$ is not specified by the PCBN. □

3.4. Interfering v-structures

In Figure 3.8a, we saw that the v-structure around node 4 required the copula c_{23} to compute the conditional margin $u_{3|2}$. Hence, the order $1 <_3 3$ will lead to integration, but if we pick the ordering of 3 required by v-structure at node 4, i.e. $2 <_3 1$, the integration is not needed. In the next example, we show a situation where a v-structure is influenced by two distinct v-structures which require different orders.

Example 3.12. Consider the PCBN in Figure 3.11. Here, we have three v-structures where the ordering of node 3 has yet to be decided. Naturally, we would like to pick $<_3$ such that we will not require integration. The v-structures around nodes 4 and 5 require us to compute the conditional margins $u_{3|1}$ and $u_{2|3}$ respectively. The former implies that c_{13} must be specified, and thus we must have $1 <_3 2$. But, the latter requires c_{23} to be specified, implying that $2 <_3 1$. Since we cannot have both, there does not exist a suitable ordering for node 3. This issue will occur for every possible pair of orders $<_4$ and $<_5$.

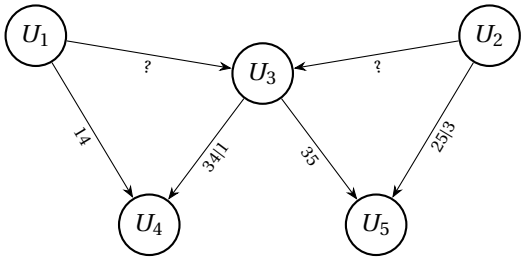


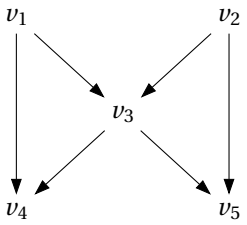
Figure 3.11: PCBN with interfering v-structures.

If $\mathcal{G}$ contains three v-structures (or more) that interact in a similar fashion as in Example 3.12, then the joint density will require integration for any choice of $\mathcal{O}$. Such v-structures will be referred to as **interfering v-structures**.

Definition 3.13. Let $(\mathcal{G},\mathcal{O})$ be a PCBN. Consider the nodes $v_1, v_2, v_3, v_4, v_5 \in V$, satisfying the following conditions:

- $v_3 \in pa(v_4) \cap pa(v_5)$.
- $v_1 \in pa(v_3) \cap pa(v_4)$ and $v_1 \notin pa(v_5)$.
- $v_2 \in pa(v_3) \cap pa(v_5)$ and $v_2 \notin pa(v_4)$.

That is, $\mathcal{G}$ contains the subgraph below.



Then, the nodes v_1, v_2, v_3, v_4 and v_5 are said to be ***interfering v-structures***. Moreover, $\mathcal{G}$ is said to contain interfering v-structures.

We will now prove in full generality that for any graph containing interfering v-structures, the computation of the joint density will require integration for any choice of $\mathcal{O}$.

Theorem 3.14. Let $(\mathcal{G},\mathcal{O})$ be a PCBN. If $\mathcal{G}$ contains interfering v-structures, then the computation of the joint density requires integration.

Proof. Let $v_1, v_2, v_3, v_4, v_5 \in {}^{/}V$ be nodes corresponding to interfering v-structures in $\mathcal{G}$. We have 8 distinct cases concerning constraints of parental orderings of nodes 3, 4, and 5, these are:

$$\begin{aligned} &1 <_3 2,\ 1 <_4 3 \text{ and } 2 <_5 3, && 2 <_3 1,\ 1 <_4 3 \text{ and } 2 <_5 3\\ &1 <_3 2,\ 1 <_4 3 \text{ and } 3 <_5 2, && 2 <_3 1,\ 1 <_4 3 \text{ and } 3 <_5 2\\ &1 <_3 2,\ 3 <_4 1 \text{ and } 2 <_5 3, && 2 <_3 1,\ 3 <_4 1 \text{ and } 2 <_5 3\\ &1 <_3 2,\ 3 <_4 1 \text{ and } 3 <_5 2, && 2 <_3 1,\ 3 <_4 1 \text{ and } 3 <_5 2. \end{aligned}$$

Since all cases are analogous, we will only consider the case when: $1 <_3 2$, $1 <_4 3$ and $2 <_5 3$. Remark that we have

$$\begin{aligned} 1 \in pa(4 \downarrow 3) \text{ and } 1 \notin pa(5 \downarrow 3) \subseteq pa(5),\\ 2 \in pa(5 \downarrow 3) \text{ and } 2 \notin pa(4 \downarrow 3) \subseteq pa(4). \end{aligned}$$

The factorization in Equation (3.2) requires us to compute the conditional margin $u_{3|pa(5\downarrow 3)}$ where $2 \in pa(5 \downarrow 3)$.

Because of the arc $2 \to 3$, the nodes 2 and 3 are not d-separated given any subset of $V \setminus \{2,3\}$. Hence, we have that $\not{d\text{-}sep}_{\mathcal{G}}\big(2,3 \,\big|\, K\big)$ for any $K \subseteq pa(5 \downarrow 3) \setminus \{2\} \subseteq V \setminus \{2,3\}$. Thus, the copula $c_{23|K}$ is not specified to be the independence copula for any $K \subseteq pa(5 \downarrow 3) \setminus \{2\}$.

The arc $2 \to 3$ has the assigned copula $c_{23|pa(3\downarrow 2)}$ with $1 \in pa(3 \downarrow 2)$. Since $1 \notin pa(5)$ we have that $1 \notin K$ for any $K \subseteq pa(5 \downarrow 3) \setminus \{2\}$. Hence, the copula $c_{23|K}$ is not specified by an arc for any $K \subseteq pa(5 \downarrow 3) \setminus \{2\}$.

Thus, for any $K \subseteq pa(5 \downarrow 3) \setminus \{2\}$ the copula $c_{23|K}$ is not specified. Consequently, we can apply Lemma 3.10, with $z = 2$, $w = 3$ and $v = 5$ in the notation of the lemma, to find that the computation of $u_{3|pa(5\downarrow 3)}$ requires integration. □

The active cycles and interfering v-structures are the only graphical structures which will necessitate integration. Thus, we propose the following theorem.

Theorem 3.15. Let $(\mathcal{G},\mathcal{O})$ be a PCBN. The computation of the joint density does not require integration if and only if $\mathcal{G}$ contains no active cycles or interfering v-structures.

Proof. The sufficiency is proven using contraposition and applying Theorems 3.11 and 3.14. The necessity is proven by Theorem 4.5. □

To prove the necessity in Theorem 3.15, we must demonstrate that for any graph $\mathcal{G}$ that does not contain active cycles or interfering v-structures, we can find an ordering $\mathcal{O}$ such that we do not need to integrate. Therefore, we will construct an algorithm which is able to find a suitable $\mathcal{O}$ for any restricted DAG $\mathcal{G}$. First, in Section 3.5, we will establish certain restrictions that a parental order $<_v$ with $v \in V$ must abide to. Hereafter, we will construct the algorithm in Section 4.1 and prove Theorem 4.5 which states that the algorithm is always able to find a suitable order.

3.5. B-sets

Consider the graph in Figure 3.12. To compute the joint density we will need to compute the conditional margin $u_{3|1}$ or $u_{1|3}$, depending on the order $<_4$. Thus, to prevent integration, the copula c_{13} must be assigned to the arc $1 \to 3$. Therefore, the arc $1 \to 3$ is said to **blocked** by the v-structure at node 4. That is, we know that we must have $1 <_3 2$, otherwise, we must integrate. Moreover, we say that the order $<_3$ must **abide** to the blocked arc $1 \to 3$.

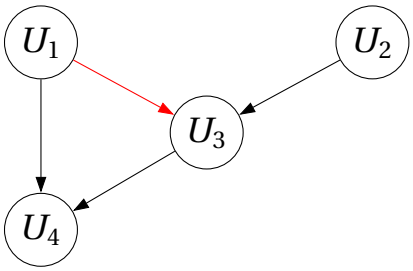


Figure 3.12: Graph with one blocked arc coloured in red.

A v-structure can block multiple arcs at the same time, see Figure 3.13. Here, the v-structure at node 5 requires us to compute one of the margins $u_{4|12}$, $u_{2|14}$ or $u_{1|24}$ which means that we will need copula $c_{14|2}$ or $c_{24|1}$. Hence, the arcs $1 \to 4$ and $2 \to 4$ are blocked by 5. That is, we must have $1 <_4 2 <_4 3$ or $2 <_4 1 <_4 3$.

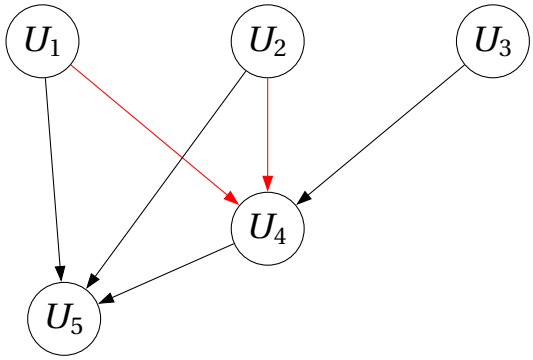


Figure 3.13: Graph with two blocked arcs coloured in red.

To find the blocked arcs in the example above, we must examine if there are any children with overlapping parental sets. For example, in Figure 3.13, we have that 5 is a child of 4, and 4 and 5 both have 1 and 2 as parents. Such graphical structures will always produce blocked arcs. To formalize this concept we introduce the **B-sets**.

Definition 3.16 (B-set)**.** Let $\mathcal{G} = (V, E)$ be a DAG. For $v_1, v_2 \in V$, we denote by

$$B(v_1, v_2) := \begin{cases} pa(v_1) \cap pa(v_2), & \text{if } v_1 \to v_2, \\ \emptyset, & \text{else.} \end{cases}$$

We say that $B(v_1, v_2)$ is a ***B-set*** of v_1.

The B-sets will provide us with clear restrictions an order must abide to, so that the joint density does not require integration. That is, if we have two nodes $v_1, v_2 \in pa(v)$ and v_1 is in a certain B-set of v which does not contain v_2, then $v_1 <_v v_2$. It should be noted that a node has as many B-sets as it has children. Some of them can be empty and not all of them have to be distinct. For example, in Figure 3.14 we have three distinct B-sets for node 6, i.e.

- $B(6,7) = \{1,3,4\}$,
- $B(6,8) = \{3,4\}$,
- $B(6,9) = \{4\}$.

Thus, the allowed orderings are $4 <_6 3 <_6 1 <_6 2 <_6 5$ and $4 <_6 3 <_6 1 <_6 5 <_6 2$. Remark that for each pair of B-sets we have that one is contained in the other, i.e. $B(6,9) \subseteq B(6,8) \subseteq B(6,7)$ in this example. This property is true in general if the graph does not contain interfering v-structures, as proven in the lemma below.

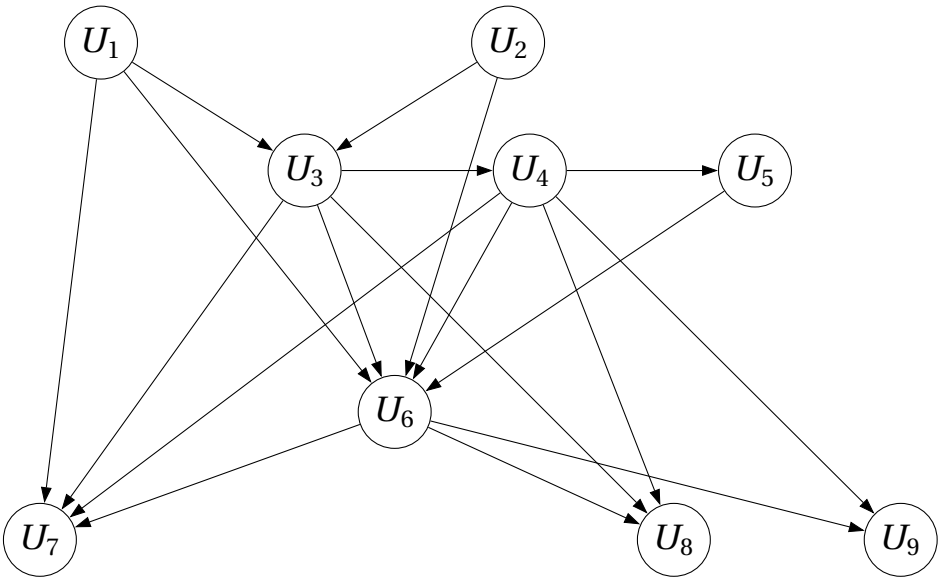


Figure 3.14: Graph where node U_6 has three distinct B-sets.

Lemma 3.17. Let $\mathcal{G} = (V, E)$ be a DAG. The following two statements are equivalent:

(i) $\mathcal{G}$ does not contain interfering v-structures.

(ii) For all $v_1 \in V$ and $v_2, v_3 \in ch(v_1)$ we have $B(v_1, v_2) \subseteq B(v_1, v_3)$ or $B(v_1, v_3) \subseteq B(v_1, v_2)$.

Proof. If $\mathcal{G}$ contains interfering v-structures, then (ii) is violated. For example, in Figure 3.11 we have $\{v_1\} = B(v_3, v_4) \not\subseteq B(v_3, v_5) = \{v_2\}$ and $B(v_3, v_5) \not\subseteq B(v_3, v_4)$.

If (ii) is violated, then we can find $v_4 \in B(v_1, v_2) \setminus B(v_1, v_3)$ and $v_5 \in B(v_1, v_3) \setminus B(v_1, v_2)$. In this case the nodes $v_1, \ldots, v_5$ are exactly interfering v-structures. □

Thus, if our graph does not contain interfering v-structures, we can order the B-sets corresponding to a node according to the inclusion relation $\subseteq$. It will be convenient to order the B-sets based on this order. For example, for the graph in Figure 3.14, we will denote $B(6,9)$, $B(6,8)$ and $B(6,7)$ by $B_1(6)$, $B_2(6)$ and $B_3(6)$, respectively. Moreover, we define $B_0(6) := \emptyset$ and $B_3(6) := pa(6)$. Note that distinct children of the same node can lead to identical B-sets. For example, in Figure 3.15 we have that $B(3,4) = B(3,5) = \{1,2\}$. Naturally, identical B-sets will give the same information regarding potential restrictions on the parental order. Remark that sets of the form $\{B(v_1, v_2);\ v_2 \in ch(v_2)\}$ will not contain any duplicates[2]. Thus, in Figure 3.14, we have

[2]By the standard definition of a set. A set containing duplicates would be a so-called multiset.

$\{B(6,v_2);\ v_2 \in ch(6)\} = \{\emptyset,\{4\},\{3,4\},\{1,3,4\},\{1,2,3,4,5\}\}$. The sorted sequence of these subsets with respect to the inclusion relation is referred to as the **B-sets** of node U_6.

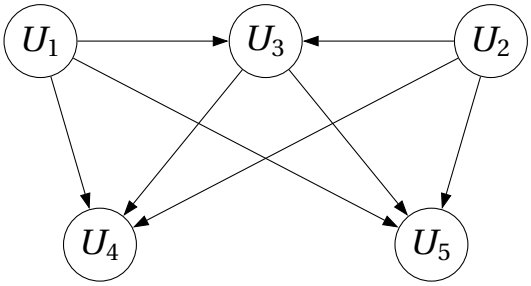


Figure 3.15: Graph where node U_3 has two identical B-sets.

Definition 3.18 (B-sets)**.** Let $\mathcal{G}$ be a DAG with no interfering v-structures and let $v \in V$ with

$$Q = Q(v) := \big|\{B(v,w);\ w \in ch(v)\}\big|.$$

the number of distinct B-sets corresponding to v. We denote by $B_1(v),\ldots,B_Q(v)$ the sorted sequence of $\{B(v,v_2);\ v_2 \in ch(v)\}$ in increasing order with respect to $\subseteq$. We also define $B_0(v) := \emptyset$ and $B_{Q+1} := pa(v)$. The sequence $\big(B_q(v)\big)_{q=0,\ldots,Q+1}$ is referred to as the ***B-sets*** of v.

Furthermore, for each B-set B_q with $q < Q(v)+1$, we denote by b_q an arbitrary node such that $B_q = B(v,b_q)$. This node b_q may not be unique.

The B-sets introduce the restriction that all nodes in B_q must be earlier than nodes in $B_{q+1} \setminus B_q$ in order $<_v$. We denote this by $B_q <_v B_{q+1} \setminus B_q$. Hence, we must have

$$B_1 <_v B_2 \setminus B_1 <_v B_3 \setminus B_2 <_v \cdots\cdots\cdots <_v B_{Q+1} \setminus B_Q.$$

For example, for node 6 in Figure 3.14, we have

$$\{4\} <_6 \{3\} <_6 \{1\} <_6 \{2,5\}.$$

Indeed, we have seen that the only orders not resulting in integration are $4 <_6 3 <_6 1 <_6 2 <_6 5$ and $4 <_6 3 <_6 1 <_6 5 <_6 2$.

We now state the following definition.

Definition 3.19 (Abiding by the B-sets)**.** Let $(\mathcal{G},\mathcal{O})$ be a PCBN where $\mathcal{G}$ contains no interfering v-structures. A parental order $<_v$ is said to ***abide*** by the B-sets if

$$B_1 <_v B_2 \setminus B_1 <_v B_3 \setminus B_2 <_v \cdots\cdots\cdots <_v B_{Q+1} \setminus B_Q.$$

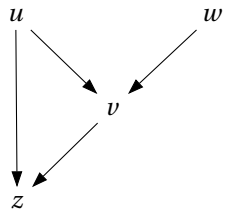
Similarly, a set of orders $\mathcal{O} := \{<_v;\ v \in V\}$ abides by the B-sets if all its parental orders $<_v$ abide by the B-sets.

Any PCBN whose set of orders does not abide by the B-sets will require integration.

Lemma 3.20. Let $(\mathcal{G},\mathcal{O})$ be a PCBN where $\mathcal{G}$ contains no active cycles or interfering v-structures. If $\mathcal{O}$ does not abide by the B-sets, then the computation of the joint density requires integration.

Proof. By assumption, we can pick a node v in V such that $<_v$ does not abide by the B-sets. Hence, there

exist $u, w \in pa(v)$ and a node $z \in V$ such that $\mathcal{G}$ contains the subgraph below with $w \notin pa(z)$ and $w <_v u$.



The factorization of the joint density requires the computation of the conditional margins $u_{v|pa(z\downarrow v)}$ and $u_{u|pa(z\downarrow u)}$. Remark that we can have $u <_z v$ or $v <_z u$. Since both cases are analogous, we will only consider the case when $u <_z v$, which implies that $u \in pa(z \downarrow v)$. To compute the joint density, we need the conditional margin $u_{v|pa(z\downarrow v)}$, and we will show that this margin requires integration. Using Lemma 3.10, it remains to prove that for any $K \subseteq pa(z \downarrow v) \setminus \{u\}$ the copula $c_{uv|K}$ is not specified by the PCBN. Consider an arbitrary $K \subseteq pa(z \downarrow v) \setminus \{u\}$.

Because of the arc $u \to v$, we have that $\not{d}\text{-}\not{sep}_{\mathcal{G}}(u, v \,|\, K)$. Hence, the copula $c_{uv|K}$ is not the independence copula. Moreover, copula $c_{uv|pa(v\downarrow u)}$, where $w \in pa(v \downarrow u)$, is assigned to the arc $u \to v$. Since $w \notin pa(z \downarrow v) \subseteq pa(z)$, we have that $w \notin K$. Therefore, the copula $c_{uv|pa(v\downarrow u)}$ is not equal to the copula $c_{uv|K}$. Hence, the copula $c_{uv|K}$ is not specified by the PCBN.

□

4

Determining assignment of copulas

In this chapter, we illustrate the process of finding a suitable set of orders $\mathcal{O}$ for an arbitrary DAG $\mathcal{G}$ without active cycles and interfering v-structures. First, we will provide an intuitive motivation for the algorithm together with the necessary terminology followed by the algorithm itself. Hereafter, we prove that the algorithm is able to identify all possible orderings, which do not necessitate integration.

4.1. Algorithm

The idea behind Algorithm 2 is that for any node v in V we find a suitable ordering $<_v$. We determine these parental orders sequentially, following an arbitrary well-ordering of the graph. That is, we pick a well-ordering $<$, such that $V = (v_1, \dots, v_n)$ where $i < j$ implies $v_i < v_j$ and determine all $<_{v_i}$'s in this order. As a result, when we arrive at a node v we will have already chosen the order $<_z$ for all nodes in $z \in pa(v)$. The process of finding a suitable order $<_v$ involves growing an ordered set O_v^k, referred to as a partial order.

Definition 4.1 (Partial order)**.** For a node $v \in V$, an ordered subset of k parents will be referred to as a ***partial order*** denoted by O_v^k. Thus, we have

$$O_v^k = (o_1, \dots, o_k)$$

with $k \leq \left|pa(v)\right|$ and $o_i \in pa(v)$ for all $i \in \{1, \dots, k\}$. Note that we will omit the subscript (v) whenever it is evident from the context to which node v we refer, and we drop the superscript (k) if the partial order is of arbitrary size.

An initial state is $O_v^0 = \emptyset$ to which at each iteration a node from the set $pa(v)$ is added until we have found $O_v^{|pa(v)|}$. A node can be added to a partial order O_v^k if it satisfies certain constraints. Specifically, we can add a $w \in$
$pa v$ such that the conditions on the B-sets are satisfied and for which we can compute the conditional margin $u_{w|O_v^k}$ with a proper recursion. Remember that by Lemma 3.6 this implies that all conditional margins of the form $u_{v|O_v^k}$ are computable without integration as well. The first constraint dictates that we must add a node from the smallest possible B-set. That is, we only incorporate a node from $B_i(v)$ if all nodes from $B_{i-1}(v)$ are already included in O_v^k. Then, the elements of the smallest B-set larger than O_v^k (denoted as $B(O_v^k)$) are added.

Definition 4.2. The smallest B-set strictly larger than a partial order O_v^k with $k < \left|pa(v)\right|$ is denoted by $B(O_v^k)$. Thus, $B(O_v^k) := B_{\tilde{q}}(v)$ with

$$\tilde{q} := \min\{q \in \{1, \dots, Q(v)+1\};\, O_v^k \subsetneq B_q(v))\}.$$

Remark that such a $\tilde{q}$ always exists, since by definition $B_{Q(v)+1}(v) = pa(v)$. Furthermore, we remark that that $B(O_v^{|pa(v)|})$ is not defined.

Remark 4.3. In Definition 4.2, the set $B(O_v^{|pa(v)|})$ is not defined because it is not possible to find a strictly larger B-set than $O_v^{|pa(v)|}$. Furthermore, the largest B-set $B_{Q(v)+1}$ is equal to $pa(v)$, by definition. Consequently, we always have $O_v^k \subsetneq B(O_v^k)$.

It is important to note that including a node $w \in B(O_v^k) \setminus O_v^k$ ensures that our order abides by the B-sets. Therefore, the only allowed additions to a partial order O_v^k are nodes in $B(O_v^k) \setminus O_v^k$ for which we can compute $u_{w|O_v^k}$ without integration. These nodes will be referred to as possible candidates. The set of possible candidates can be divided into three subsets. The most elementary case is whenever $d - sep_{\mathcal{G}}\big(w, O_v^k \,|\, \emptyset\big)$. Here, we have $u_{w|O_v^k} = u_w$ which obviously does not require any integration. We will refer to these nodes as possible candidates by independence.

If this d-separation does not hold, the computation of the conditional margin $u_{w|O_v^k}$ will require integration, unless we can compute it with a proper recursion. Naturally, this recursion must start with an h-function corresponding to copulas of the form

$$c_{wo|pa(o)\setminus w)} \quad \text{or} \quad c_{ow|pa(w)\setminus o)}$$

where o is a node in O_v^k. Such copulas are already specified in the PCBN as they concern parents of node v, hence they appear earlier in the total order of nodes. A potential node w that could be added to O_v^k can be connected to a node already in O_v^k by incoming or outgoing arc, but this node will be eligible to be added to O_v^k if the copula $c_{wo|O_v^k\setminus\{o\}}$ corresponds to a copula already assigned to such arc. This gives rise to the other two subsets:

- $\exists o \in O_v^k : w \to o$ and $c_{wo|O_v^k\setminus\{o\}} = c_{wo|pa(o)\setminus w)}$,
- $\exists o \in O_v^k : o \to w$ and $c_{wo|O_v^k\setminus\{o\}} = c_{ow|pa(w)\setminus o)}$

and the margins of these copulas $u_{w|pa(o)\setminus w)}$, $u_{o|pa(o)\setminus w)}$, $u_{w|pa(w)\setminus o)}$, $u_{o|pa(w)\setminus o)}$ can be computed without integration. We will refer to the nodes as belonging to these two subsets as possible candidates by incoming arcs (outgoing arc), respectively.

It is clear how the set of possible candidates by independence can be found. Below three examples are presented in which the subsets of possible candidates are illustrated. We show the conditions that have to be satisfied for a node to belong to sets of possible candidates by incoming and outgoing arcs.

1. By independence: We will examine the graph in Figure 4.1 and determine the parental order for node 4. Naturally, we initialize our algorithm with the ordered set $O_4^0 = \emptyset$. Note that we have the following B-sets of node 4:

   $$B_1(4) = \{1\} \quad \text{and} \quad B_2(4) = \{1,2,3\}.$$

   At each step of the algorithm, we will display the graph, highlighting the nodes belonging to each subset. Nodes currently present in O_4^k will be colored in blue, while the potential candidates by independence will be shown in red.

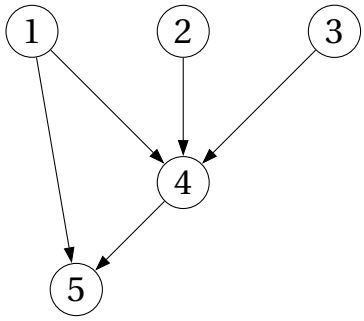
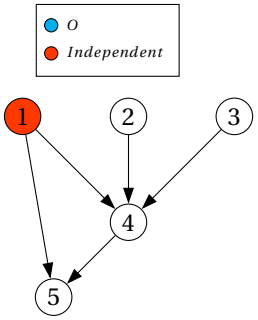


Figure 4.1: Graph for which we will determine the order $<_4$.

Iteration 1

Add a node from the set

$$B(O_4^0) \setminus O_4^0 = \{1\} \setminus \emptyset = \{1\}.$$



Since $u_{1|O^0} = u_1$ is an unconditional margin, node 1 is considered a possible candidate by independence since we have that

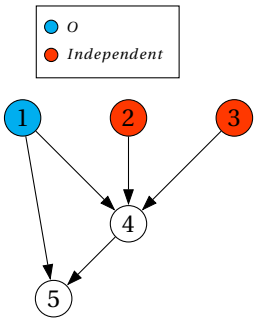
$$d - sep_{\mathcal{G}}(1, O_4^0 \,|\, \emptyset) = d - sep_{\mathcal{G}}(1, \emptyset \,|\, \emptyset)$$

which is assumed to hold by convention, see Remark 2.23. Consequently, we color it red in the graph on the right. Thus, $O_4^1 = (1)$.

Iteration 2

To determine the possible candidates to add to O_4^1, we consider the set

$$B(O_4^1) \setminus O_4^1 = \{1,2,3\} \setminus \{1\} = \{2,3\}.$$



We need to select $w \in \{2,3\}$ such that we can compute $u_{w|O_4^1}$ without integration. Since we have

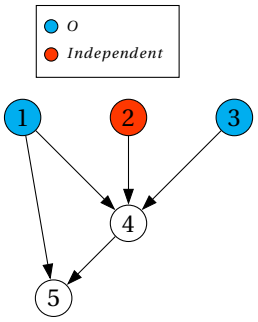
$$d - sep_{\mathcal{G}}(3, 1 \,|\, \emptyset) \Rightarrow u_{3|1} = u_3,$$
$$d - sep_{\mathcal{G}}(2, 1 \,|\, \emptyset) \Rightarrow u_{2|1} = u_2,$$

both nodes 2 and 3 are possible candidates by independence. Suppose we add node 3 and update the ordered set to $O_4^2 = (1,3)$.

Iteration 3

Finally, we have

$$B(O_4^2) \setminus O_4^2 = \{1,2,3\} \setminus \{1,3\} = \{2\},$$



leaving us with one choice. The conditional margin $u_{2|13}$ can be computed without integration since

$$d - sep_{\mathcal{G}}(2, \{1,3\} \,|\, \emptyset) \Rightarrow u_{2|13} = u_2.$$

Thus, the partial order is $O_4^3 = (1,3,2)$, giving us the order $1 <_4 3 <_4 2$.

The pattern in the example is clear, at each iteration we add a node $w \in B(O_v^k) \setminus O_v^k$ such that $d - sep_{\mathcal{G}}(w, O_v^k \,|\, \emptyset)$

holds.

We can observe that the algorithm requires the subgraph induced by the parents, children and the node v to find the ordered sequence of B-sets and choices of candidates. In the example above, it would have sufficed to only display a subgraph containing nodes in $pa(4)$. Therefore, from this point on, we will display the subgraph of parents while running the algorithm.

2. Incoming arcs: Consider the graph in Figure 4.2 for which the orders $<_4$ and $<_5$ have already been chosen: $1 <_4 2 <_4 3$ and $1 <_5 2 <_5 4 <_5 3$. Our objective is to choose a suitable ordering $<_6$ by growing a partial order O_6^k. Note that node 6 does not have any corresponding B-sets, as it has no children.

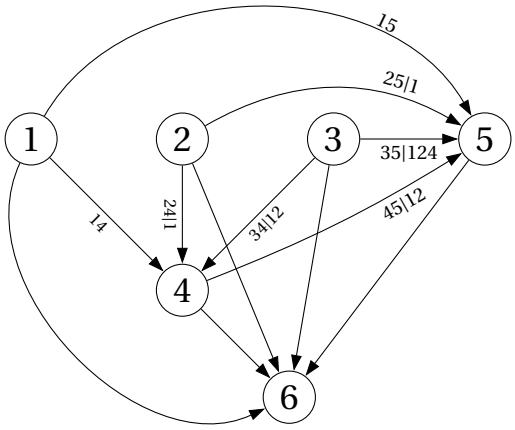
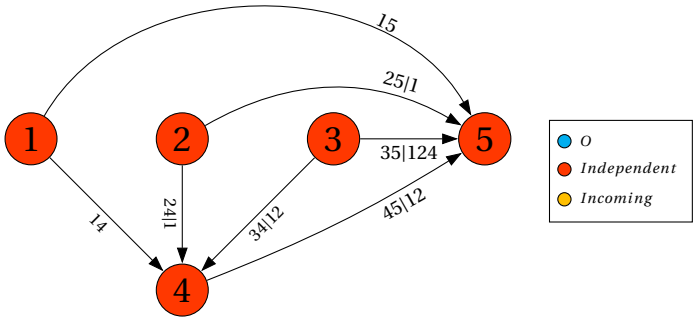


Figure 4.2: Graph for which we will determine $<_6$.

Iteration 1

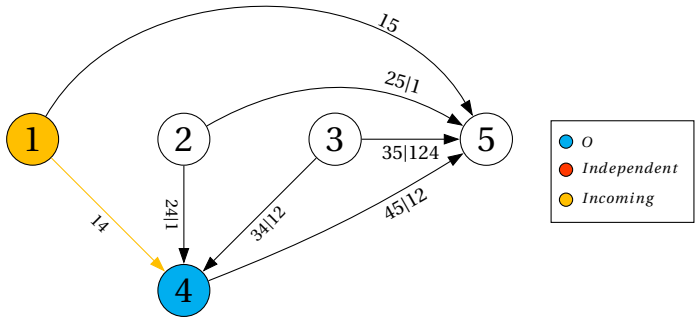
Any node in $pa(6)$ can be added to $O_6^0 = \emptyset$ by independence. Suppose that we choose node 4 and get $O_6^1 = (4)$.



Iteration 2

None of the nodes in $B(O_6^1) \setminus O_6^1$={1,2,3,5} are d-separated from 4 by the empty set. Therefore, we must use one of the copulas corresponding to an arc connected to 4. The only suitable arc is the incoming arc $1 \to 4$, since its corresponding copula c_{14} allows computation of the margin $u_{1|O^1} = u_{1|4}$. To represent this, we color the incoming arc and corresponding node yellow. Thus, we add 1 to the ordered set and

get $O_6^2 = (4,1)$.



Iteration 3

Next, we consider as possible candidates nodes in the set

$$B(O_6^2) \setminus O_6^2 = \{1,2,3,4,5\} \setminus \{1,4\} = \{2,3,5\}.$$

There are two incoming arcs to node 4, i.e. $2 \to 4$ and $3 \to 4$. Node 2 is a possible candidate since the margin $u_{2|O_6^2} = u_{2|14}$ can be computed with the copula $c_{24|pa(4\downarrow 2)} = c_{24|1}$. Indeed, we remark that $pa(4 \downarrow 2) = \{z \in pa(4);\, z <_4 2\} = \{1\}$.

Node 3 is not a possible candidate by incoming arc $3 \to 4$. The copula corresponding to this arc, $c_{34|pa(4\downarrow 3)} = c_{34|12}$, contains node 2 in its conditioning set whereas $2 \notin O_6^2 = \{1,4\}$. Hence, computing $u_{3|O_6^2} = u_{3|14}$ from the copula $c_{34|12}$ requires integration with respect to node 2:

$$u_{3|14} = \int_0^1 \frac{\partial C_{34|12}(u_3, u_4 \,|\, u_1, w_2)}{\partial u_4} \,\mathrm{d}w_2.$$

Note that

$$pa(4 \downarrow 3) = \{z \in pa(4);\, z <_4 3\} = \{1,2\} \not\subseteq \{1,4\} = O_6^2.$$

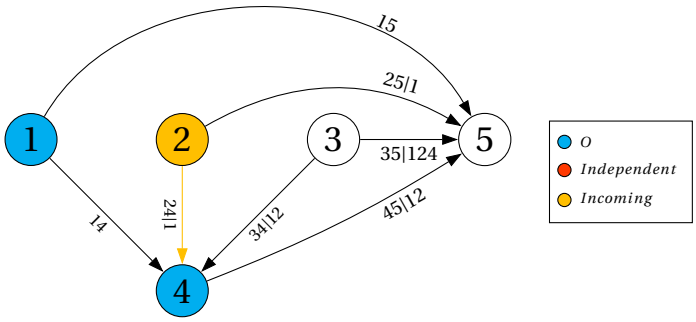
This condition is necessary: for a node w to be considered as a possible candidate for O_v^k by incoming arc $w \to o$, it must be that $pa(o \downarrow w) \subseteq O_v^k$. Thus, it must be that parents of node o earlier in the ordering than w must already be included in O_v^k.

This is condition not sufficient (as shown in Iteration 5 below). Indeed, if O_v^k contains nodes that are not in $pa(o \downarrow w)$, then these elements should be "removable" from the conditioning set of a copula by d-separation. Hence, another required condition is $d\text{-}sep_{\mathcal{G}}\big(w, O_v^k \setminus \underline{pa}(o \downarrow w) \,\big|\, \underline{pa}(o \downarrow w)\big)$. In Iteration 5, we examine this condition more closely.

In this example, node 2 is the only node satisfying both of these conditions. Indeed, for $w = 2$ and $o = 4$, we have

$$d\text{-}sep_{\mathcal{G}}\big(w, O_v^k \setminus \underline{pa}(o \downarrow w) \,\big|\, \underline{pa}(o \downarrow w)\big) = d\text{-}sep_{\mathcal{G}}\big(2, \{1,4\} \setminus \{1,4\} \,\big|\, \{1,4\}\big) = d\text{-}sep_{\mathcal{G}}\big(2, \emptyset \,\big|\, \{1,4\}\big)$$

which is satisfied by convention, see Remark 2.23. Therefore, we add it to the partial order O_6^2, and obtain $O_6^3 = (4,1,2)$.



Iteration 4

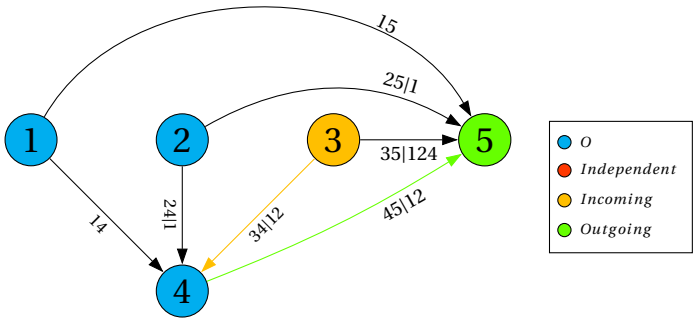
First, we remark that node 5 is a possible candidate by the outgoing arc $4 \to 5$. To highlight this, the node and its corresponding arc will be colored green. For the sake of this example, we will focus on incoming arcs.

Now we are able to use the incoming arc $3 \to 4$ since the condition $pa(4 \downarrow 3) = \{1,2\} \subseteq O_6^3 = \{4,1,2\}$ is satisfied. Indeed, we can compute the conditional margin $u_{3|O_6^3} = u_{3|124}$ with the copula $c_{34|\underline{pa}(4\downarrow 3)} = c_{34|12}$. Remark that the second condition is also satisfied since

$$d - sep_{\mathcal{G}}\left(w, O_6^3 \setminus \underline{pa}(4 \downarrow 3) \,\middle|\, \underline{pa}(4 \downarrow 3)\right) = d - sep_{\mathcal{G}}\left(3, \{1,2,4\} \setminus \{1,2,4\} \,\middle|\, \{1,2,4\}\right) = d - sep_{\mathcal{G}}\left(3, \emptyset \,\middle|\, \{1,2,4\}\right)$$

holds by convention. Hence, node 3 is a possible candidate by incoming arc $3 \to 4$ at this iteration.

However, rather than adding node 3, we add node 5 by the outgoing arc $4 \to 5$. Subsequently, in the next iteration, we will encounter a situation where the second condition proves to be necessary. So, we extend the partial order to $O_6^4 = (4,1,2,5)$.



Iteration 5

After the addition of node 5, the only remaining node is node 3. There are two incoming arcs $3 \to 5$ and $3 \to 4$. Note that in the previous iteration, it was possible to use the incoming arc $3 \to 4$. We now explain why we cannot use the incoming arc $3 \to 4$ anymore, even though the condition $pa(4 \downarrow 3) = \{1,2\} \subseteq \{1,2,4,5\} = O_6^4$ is satisfied.

The reason for this is the following: we need the conditional margin $u_{3|O_6^4} = u_{3|1245}$ but the arc $3 \to 4$ corresponds to the copula $c_{34|pa(4\downarrow 3)} = c_{34|12}$. From this copula $u_{3|124}$ can be computed, and

$$u_{3|O_6^4} = u_{3|1245} \neq u_{3|124} = u_{3|pa(4\downarrow 3)\sqcup\{4\}} = u_{3|\underline{pa}(4\downarrow 3)}. \tag{4.1}$$

The margin that can be computed is not the same as the one we require. There is no equality in Equation 4.1. Removing node 5 from the conditioning set in $u_{3|1245}$ requires the d-separation

$$d-sep_{\mathcal{G}}\big(3,5\,\big|\,\{1,2,4\}\big) = d-sep_{\mathcal{G}}\Big(3, O_6^4 \setminus \underline{pa}(4\downarrow 3)\,\Big|\,\underline{pa}(4\downarrow 3)\Big)$$

which does not hold due to the arc $3 \to 5$.

The general condition is that a node w is a possible candidate for O_v^k by an incoming arc $w \to o$, if

$$d-sep_{\mathcal{G}}\Big(w, O_v^k \setminus \underline{pa}(o\downarrow w)\,\Big|\,\underline{pa}(o\downarrow w)\Big)$$

and

$$pa(o\downarrow w) \subseteq O_v^k$$

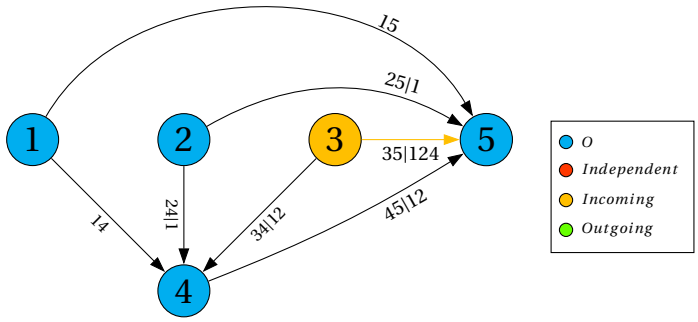
hold. In this example, the incoming arc $3 \to 5$ satisfies these restrictions with $o=5$, $w=3$ because

$$d-sep_{\mathcal{G}}\Big(3, \{1,2,4,5\}\setminus\{1,2,4,5\}\,\Big|\,\{1,2,4,5\}\Big) = d-sep_{\mathcal{G}}\Big(3,\emptyset\,\Big|\,\{1,2,4,5\}\Big).$$

and

$$pa(5\downarrow 3) = \{1,2,4\} \subseteq \{1,2,4,5\} = O_6^4$$

hold. Therefore, we can add node 3 by incoming arc $3 \to 5$ to obtain $O_6^5 = (4,1,2,5,3)$, giving us the order; $4 <_6 1 <_6 2 <_6 5 <_6 3$.



To summarize, w is a possible candidate that can be added to O_v^k by an incoming arc $w \to o$ with $o \in O_v^k$, if we can compute the conditional margin $u_{w|O_v^k}$ with the conditional copula $c_{wo|pa(o\downarrow w)}$. This requires satisfying the following restrictions:

- $pa(o\downarrow w) \subseteq O_v^k$.
- $d-sep_{\mathcal{G}}\big(w, O_v^k \setminus \underline{pa}(o\downarrow w)\,\big|\,\underline{pa}(o\downarrow w)\big)$.

Remark that the set $\underline{pa}(o\downarrow w) := pa(o\downarrow w) \sqcup \{o\}$ contains all parents of o ordered lower than w according to $<_o$ and o itself, see Definition 2.20

3. Outgoing arcs: Consider the graph in in Figure 4.3 for which we will determine a suitable order for node 7. Note that the order $<_5$ has already been determined, i.e. $1 <_5 2 <_5 3 <_5 4$. Furthermore, node 7 has no corresponding B-sets, as it has no children.

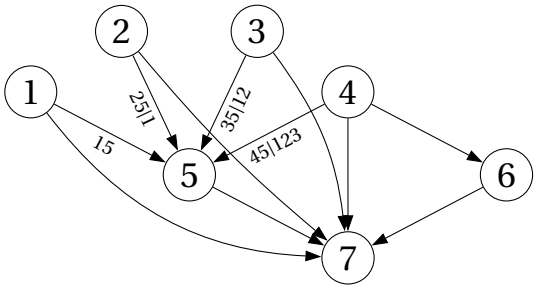
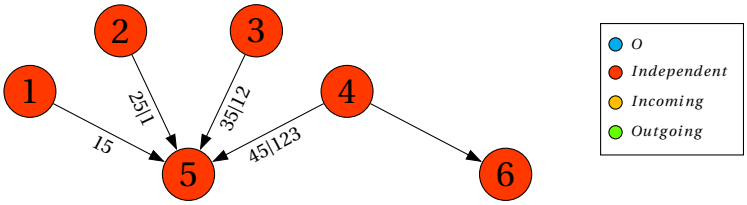


Figure 4.3: Graph for which we will determine the order $<_7$.

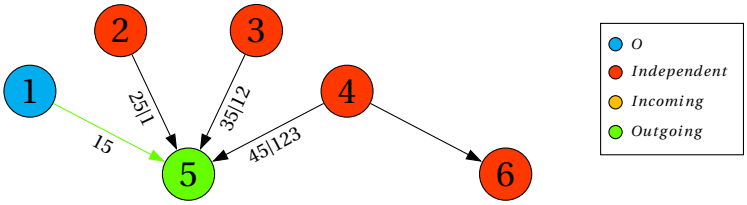
Iteration 1

We commence with $O_7^0 = \emptyset$, to which any node in the set $B(O_7^0) \setminus O_7^0 = \{1,2,3,4,5,6\}$ can be added by independence. We start with 1, so $O_7^1 = (1)$.



Iteration 2

Nodes 2, 3, 4 and 6 can be added by independence. Moreover, node 5 is a possible candidate since we can compute the margin $u_{1|5}$ using the copula c_{15} which corresponds to outgoing arc $1 \to 5$. Let us add node 3 instead and get $O_7^2 = (1,3)$.



Iteration 3

Node 5 is not a possible candidate by the outgoing arc $3 \to 5$, since its corresponding copula $c_{35|pa(5\downarrow 3)} = c_{35|12}$ cannot be used to compute $u_{5|O_7^2} = u_{5|13}$ without integration. That is, we have

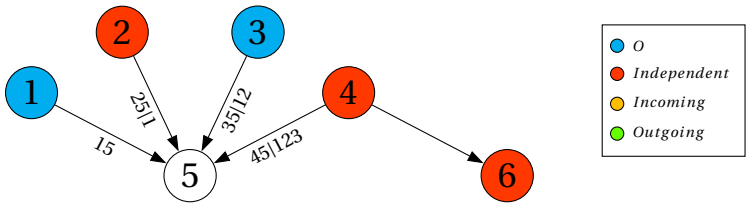
$$u_{5|13} = \int_0^1 \frac{\partial C_{35|12}(u_3, u_5 \,|\, u_1, u_w)}{\partial u_3} \,\mathrm{d}w_2.$$

Note that

$$pa(5 \downarrow 3) = \{1,2\} \not\subseteq \{1,3\} = O_7^2. \tag{4.2}$$

If the inclusion in Equation 4.2 were to hold, then the margin $u_{3|O_7^2}$ could be computed using $c_{35|pa(5\downarrow 3)}$ without integration. This gives rise to the first necessary condition for w to be a possible candidate to be added to O_v^k by an outgoing arc $o \to w$: $pa(w \downarrow o) \subseteq O_v^k$ must hold.

We add to $O_7^2 = (1,3)$ node 2 by independence, resulting in O_7^3=(1,3,2).



Iteration 4

With the addition of node 2, we observe that

$$pa(5 \downarrow 3) = \{1,2\} \subseteq \{1,2,3\} = O_7^3.$$

Thus, node 5 satisfies the first condition. The second condition that is needed is as follows:

$$d - sep_{\mathcal{G}}\left(w, O_v^k \setminus \overline{pa}(w \downarrow o) \,\middle|\, \overline{pa}(w \downarrow o)\right).$$

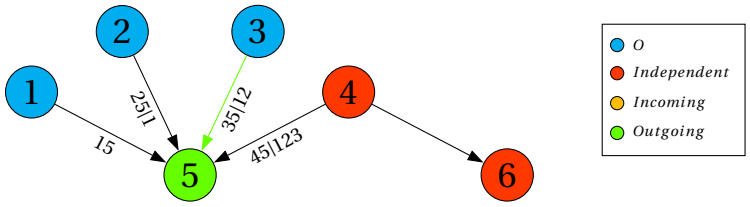
This condition is analogous to the one in the case of candidates by an incoming arc. A motivation for this condition is given in Iteration 5.

Remark that arc $3 \rightarrow 5$ satisfies the second condition, since

$$d - sep_{\mathcal{G}}\left(5, O_7^3 \setminus \overline{pa}(5 \downarrow 3) \,\middle|\, \overline{pa}(5 \downarrow 3)\right) = d - sep_{\mathcal{G}}\left(5, \{1,2,3\} \setminus \{1,2,3\} \,\middle|\, \{1,2,3\}\right) = d - sep_{\mathcal{G}}\left(5, \emptyset \,\middle|\, \{1,2,3\}\right).$$

Hence, node 5 is a possible candidate. Indeed, the margin $u_{5|O_7^3} = u_{5|123}$ can be computed with the copula $c_{35|pa(5\downarrow 3)} = c_{35|12}$.

To illustrate the necessity of the second condition, we proceed by adding node 6 to the ordered set by independence, providing us with $O_7^4 = (1,3,2,6)$.



Iteration 5

With node 6 added, we can no longer add node 5 by the outgoing arc $3 \rightarrow 5$. This is because we have

$$u_{5|O_7^4} = u_{5|1236} \neq u_{5|123} = u_{5|\overline{pa}(5\downarrow 3))}.$$

This inequality prevents us from using the copula $c_{35|pa(5\downarrow 3)} = c_{35|12}$. The conditional margin could be obtained if we could remove 6 from the conditioning set of $u_{5|O_7^4} = u_{5|1236}$. However, this would require

$$d - sep_{\mathcal{G}}\left(5, O_7^4 \setminus \overline{pa}(5 \downarrow 3) \,\middle|\, \overline{pa}(5 \downarrow 3)\right) = d - sep_{\mathcal{G}}\left(5, 6 \,\middle|\, \{1,2,3\}\right)$$

which is not satisfied since the trail $5 \leftarrow 4 \rightarrow 6$ is activate given $\{1,2,3\}$.

Thus, we have that a node w is a possible candidate for O_v^k by an outgoing arc $o \to w$, if

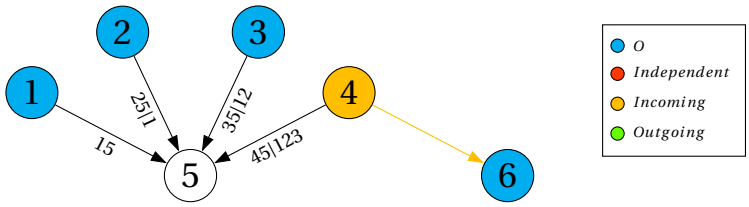
$$d - sep_{\mathcal{G}}\left(w, O_v^k \setminus \overline{pa}(w \downarrow o) \,\middle|\, \overline{pa}(w \downarrow o)\right)$$

and

$$pa(w \downarrow o) \subseteq O_v^k$$

hold.

We cannot add node 5 but we can include node 4 by the incoming arc $4 \to 6$, giving us $O_7^5 = (1,3,2,6,4)$.



Iteration 6

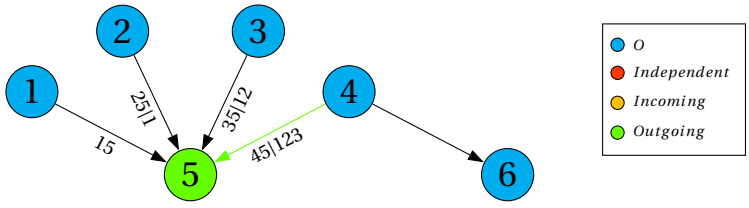
Now, node 5 is a possible candidate for O_7^5 by the outgoing arc $4 \to 5$ since both restrictions are satisfied. Indeed,

$$d - sep_{\mathcal{G}}\left(5, O_7^5 \setminus \overline{pa}(5 \downarrow 4) \,\middle|\, \overline{pa}(5 \downarrow 4)\right) = d - sep_{\mathcal{G}}\left(5, \emptyset \,\middle|\, \{1,2,3,4\}\right)$$

and

$$pa(5 \downarrow 4) = \{1,2,3\} \subseteq \{1,2,3,4,6\} = O_7^5$$

hold. Thus, we can expand the partial order with node 5 to obtain $O_7^6 = (1,3,2,6,4,5)$, giving us the order; $1 <_7 3 <_7 2 <_7 6 <_7 4 <_7 5$.



To summarize, w is a possible candidate to be added to O_v^k by an outgoing arc $o \to w$ with $o \in O_v^k$, if we can compute the conditional margin $u_{w|O_v^k}$ with the conditional copula $c_{wo|pa(w\downarrow o)}$. The following restrictions must be satisfied:

- $pa(w \downarrow o) \subseteq O_v^k$.
- $d - sep_{\mathcal{G}}\left(w, O_v^k \setminus \overline{pa}(w \downarrow o) \,\middle|\, \overline{pa}(w \downarrow o)\right)$.

Remark that the set $\overline{pa}(w \downarrow o) := pa(w \downarrow o) \sqcup \{o\}$ is defined in Definition 2.20. This set contains o and all parents of w which are lower than o according to $<_w$.

Now we are ready to define in full generality the set of possible candidates for a partial order of node v in the k-th iteration, O_v^k. This set will be denoted as $Poss.Cand(O_v^k)$.

Definition 4.4 (Possible candidates)**.** The set of ***possible candidates*** for a partial order O_v^k is defined by

$$Poss.Cand(O_v^k) := Poss.Cand_{Ind}(O_v^k) \sqcup Poss.Cand_{In}(O_v^k) \sqcup Poss.Cand_{Out}(O_v^k),$$

where the three disjoint sets are defined as

$$\begin{aligned}
Poss.Cand_{Ind}(O_v^k) &:= \{w \in B(O_v^k) \setminus O_v^k;\ d-sep_{\mathcal{G}}(w, O_v^k \,|\, \emptyset)\},\\
Poss.Cand_{In}(O_v^k) &:= \{w \in B(O_v^k) \setminus O_v^k;\ \exists o \in O_v^k,\ w \to o,\ pa(o \downarrow w) \subseteq O_v^k \text{ and } d-sep_{\mathcal{G}}(w, O_v^k \setminus \underline{pa}(o \downarrow w) \,|\, \underline{pa}(o \downarrow w))\},\\
Poss.Cand_{Out}(O_v^k) &:= \{w \in B(O_v^k) \setminus O_v^k;\ \exists o \in O_v^k,\ o \to w,\ pa(w \downarrow o) \subseteq O_v^k \text{ and } d-sep_{\mathcal{G}}(w, O_v^k \setminus \overline{pa}(w \downarrow o) \,|\, \overline{pa}(w \downarrow o))\}.
\end{aligned}$$

The proof that the three subsets, $Poss.Cand_{Ind}(O_v^k)$, $Poss.Cand_{In}(O_v^k)$ and $Poss.Cand_{Out}(O_v^k)$, are disjoint can be found in the Appendix in Lemma A.1.

In each iteration, an arbitrary node is selected from $Poss.Cand(O_v^k)$, and added to the current partial order O_v^k. This process continues until we have exhausted the parental set, and found a partial order $O_v^{|pa(v)|}$. Subsequently, the parental order $<_v$ is established based on the ordered set. That is, $<_v$ is chosen such that for $w_1, w_2 \in pa(v)$, if w_1 appears before w_2 in the ordered set $O_v^{|pa(v)|}$, then $w_1 <_v w_2$.

All steps described above provide us with Algorithm 2. Given a DAG $\mathcal{G}$ without active cycles and interfering v-structures, it is always able to find a set of orders $\mathcal{O}$ not necessitating integration. Moreover, any suitable set orders $\mathcal{O}$ can be found by the algorithm. These statements are formalized in the theorem below which is proven in Section 4.2.

Theorem 4.5. Let $\mathcal{G}$ be a DAG containing no active cycles nor interfering v-structures. Then, the following statements concerning Algorithm 2 hold:

- The joint density of a PCBN $(\mathcal{G}, \mathcal{O})$ does not requires integration if and only if the set of orders $\mathcal{O}$ is determined by Algorithm 2.
- Given $\mathcal{G}$, the algorithm can find at least one set of orders $\mathcal{O}$ not resulting in integration.

Algorithm 2 Finding a suitable $\mathcal{O}$.

Input: restricted DAG $\mathcal{G}$
Output: set of orderings $\mathcal{O}$ for which we will not require integration

for each node v in V according to a well-ordering **do**
  $O \leftarrow \emptyset$
  while $\left|O\right| < \left|pa(v)\right|$ **do**
    $B(O) \leftarrow$ smallest B-set strictly larger than O
    $Poss.Cand_{Ind}, Poss.Cand_{In}, Poss.Cand_{Out} \leftarrow \emptyset$
    for each w in $B(O) \setminus O$ **do**
      if $d - sep_{\mathcal{G}}\left(w, O \middle| \emptyset\right)$ **then**
        $Poss.Cand_{Ind} \leftarrow Poss.Cand_{Ind} \sqcup \{w\}$
      end if
      if $\exists o \in O$ s.t. $pa(o \downarrow w) \subseteq O$ and $d - sep_{\mathcal{G}}\left(w, O \setminus \underline{pa}(o \downarrow w) \middle| \underline{pa}(o \downarrow w)\right)$ **then**
        $Poss.Cand_{In} \leftarrow Poss.Cand_{In} \sqcup \{w\}$
      end if
      if $\exists o \in O$ s.t. $pa(w \downarrow o) \subseteq O$ and $d - sep_{\mathcal{G}}\left(w, O \setminus \overline{pa}(w \downarrow o) \middle| \overline{pa}(w \downarrow o)\right)$ **then**
        $Poss.Cand_{Out} \leftarrow Poss.Cand_{Out} \sqcup \{w\}$
      end if
    end for
    $Poss.Cand \leftarrow Poss.Cand_{Ind} \sqcup Poss.Cand_{In} \sqcup Poss.Cand_{Out}$
    Append one element of $Poss.Cand$ to the ordered set O
  end while
  Set $<_v$ according to O
end for
$\mathcal{O} \leftarrow \{<_v; v \in V\}$
return $\mathcal{O}$

4.2. Proof of Theorem 4.5.

In this section, we prove Theorem 4.5 by demonstrating that for a DAG $\mathcal{G}$ without active cycles and interfering v-structures, the algorithm finds a set of orders $\mathcal{O}$ for which the computation of the density does not require integration. Furthermore, we show that the algorithm is capable of identifying all suitable orderings. To accomplish this, it suffices to prove that for every node v, at each step of our algorithm with current partial order O_v^k with $k < |pa(v)|$, the following properties hold:

P1. For any node w in $Poss.Cand(O_v^k)$, we can compute $u_{w|O_v^k}$ and $u_{v|O_v^k}$ without integration.

P2. For any node w not in $Poss.Cand(O_v^k)$, integration is required to compute $u_{w|O_v^k}$.

P3. The set $Poss.Cand(O_v^k)$ is not empty.

These results are proved in separate subsections. P1 will be proven by induction. We will assume that the arguments of copulas assigned to arcs of the BN up to the current point of the algorithm (copulas assigned to arcs pointing to a node earlier in the well-ordering than node v and copulas assigned to arcs from nodes in O_v^k to v) do not require integration. This means that the following copulas have been assigned by our algorithm upon the arrival at O_v^k:

- $c_{xy|pa(y\downarrow x)}$ with $x \to y \in E$ and $y < v$.
- $c_{o_j v|O_v^{j-1}}$ with $j \leq k$.

By proving P1 and P2 we show that the restrictions placed on the set of possible candidates (see Definition 4.4) are necessary and sufficient to prevent integration in the density corresponding to a PCBN.

Moreover, we can conclude that the algorithm is able to find all possible orderings of parents that do not lead to integration.

P3 implies that we will never encounter a case where there is no possible candidate to be added. Hence, the algorithm will never terminate prematurely and will return a suitable set of orders $\mathcal{O}$.

We remark that the proof of P3 requires many additional results regarding trails, B-sets, partial orders and possible candidates. These results can be found in Chapter 5. Most of the lemmas are interesting in their own right, and do not only serve as a tool to prove P3. For instance, Lemma 5.15 provides us with a clear intuition on how possible candidates are selected by the algorithm. Furthermore, some lemmas can be applied outside of our specific framework, e.g. Lemma 5.6 which can be applied to a general DAG and not only a restricted DAG.

4.2.1. Set of possible candidates (P1)

We must prove that for any $w \in Poss.Cand(O_v^k)$, we can compute the conditional margins $u_{w|O_v^k}$ and $u_{v|O_v^k}$ without the need for integration. This means that both margins can be computed with a proper recursion of h-functions, see Chapter 3. By induction, the conditional margins of copulas assigned to arcs of the BN up to the current point of the algorithm (copulas assigned to arcs pointing to a node earlier in the well-ordering than node v and copulas assigned to arcs from nodes in O_v^k to v) do not require integration.

First, we consider the conditional margin $u_{v|O_v^k}$. The copula $c_{o_k v|O_v^{k-1}}$ has been assigned by our algorithm, and hence by the induction hypothesis, the conditional margins $u_{o_k|O_v^{k-1}}$ and $u_{v|O_v^{k-1}}$ are computable without integration. Consequently, $u_{v|O_v^k}$ can be computed with a specified h-function (see Chapter 3) by

$$u_{v|O_v^k} = h_{o_k \underline{v}|O_v^{k-1}}.$$

It remains to prove that $u_{w|O_v^k}$ does not require integration.

Since $w \in Poss.Cand(O_v^k)$, w must be in one of the three subsets of $Poss.Cand(O_v^k)$ defined in Definition 4.4. Hence, three cases are considered:

1. $\underline{w \in Poss.Cand_{Ind}(O_v^k)}$: By definition of $Poss.Cand_{Ind}(O_v^k)$, we have $d-sep_{\mathcal{G}}\big(w, O_v^k \,\big|\, \emptyset\big)$. Thus, the conditional margin is
$$u_{w|O_v^k} = u_w,$$
which does not require any integration.

2. $\underline{w \in Poss.Cand_{In}(O_v^k)}$: There exists an $o \in O_v^k$ such that

   (i) $w \to o \in E$,

   (ii) $pa(o \downarrow w) \subseteq O_v^k$,

   (iii) $d-sep_{\mathcal{G}}\big(w, O_v^k \setminus \underline{pa}(o \downarrow w) \,\big|\, \underline{pa}(o \downarrow w)\big)$.

   Note that due to (ii), O_v^k can be split into two parts as
$$O_v^k = \big(O_v^k \setminus \underline{pa}(o \downarrow w)\big) \sqcup \underline{pa}(o \downarrow w).$$
   Then, by (iii) we have that elements of $O_v^k \setminus \underline{pa}(o \downarrow w)$ can be removed from the conditioning set O_v^k, so
$$u_{w|O_v^k} = u_{w|\left(O_v^k \setminus \underline{pa}(o\downarrow w)\right) \sqcup \underline{pa}(o\downarrow w)} = u_{w|\underline{pa}(o\downarrow w)}.$$
   Remark that the arc $w \to o$ exists by (i), and is assigned the copula $c_{wo|pa(o\downarrow w)}$ which has been specified by the algorithm. Moreover, by the induction hypothesis the conditional margins $u_{w|pa(o\downarrow w)}$ and $u_{o|pa(o\downarrow w)}$ are computable without integration. Hence, the conditional margin $u_{w|\underline{pa}(o\downarrow w)}$ can be computed without integration as follows:
$$u_{w|O_v^k} = u_{w|\underline{pa}(o\downarrow w)} = h_{\underline{w}o|pa(o\downarrow w)}(u_{w|pa(o\downarrow w)}, u_{o|pa(o\downarrow w)}).$$

3. $\underline{w \in Poss.Cand_{Out}(O_v^k)}$: There exists an $o \in O_v^k$ such that

   (i) $o \to w \in E$,

   (ii) $pa(w \downarrow o) \subseteq O_v^k$,

   (iii) $d-sep_{\mathcal{G}}\big(w, O_v^k \setminus \overline{pa}(w \downarrow o) \,\big|\, \overline{pa}(w \downarrow o)\big)$.

   As in the previous case, by (ii) we have that
$$O_v^k = \big(O_v^k \setminus \overline{pa}(w \downarrow o)\big) \sqcup \overline{pa}(w \downarrow o).$$
   Then, by (iii) the elements of $O_v^k \setminus \overline{pa}(w \downarrow o)$ can be removed from the conditioning set;
$$u_{w|O_v^k} = u_{w|\left(O_v^k \setminus \overline{pa}(w\downarrow o)\right) \sqcup \overline{pa}(w\downarrow o)} = u_{w|\overline{pa}(w\downarrow o)}.$$
   The arc $o \to w$ exists by (i), and is assigned the copula $c_{ow|pa(w\downarrow o)}$ which has been specified by the algorithm. Furthermore, by induction the conditional margins $u_{o|pa(w\downarrow o)}$ and $u_{w|pa(w\downarrow o)}$ are computable without integration. Hence, $u_{w|O_v^k}$ can be computed without integration by
$$u_{w|O_v^k} = u_{w|\overline{pa}(w\downarrow o)} = h_{o\underline{w}|pa(w\downarrow o)}(u_{o|pa(w\downarrow o)}, u_{w|pa(w\downarrow o)}).$$

This concludes the proof of P1.

4.2.2. Outside the set of possible candidates (P2)

We want to show that if at any point in the algorithm, O_v^k is extended by w outside $Poss.Cand(O_v^k)$, then in the computation of $u_{w|O_v^k}$, integration will be needed. Hence, we must show that there is no proper recursion of h-functions to compute this conditional margin, see Chapter 3.

First, we remove elements from the conditioning set of $u_{w|O_v^k}$ by d-separation. In particular, we pick the smallest possible set $A \subseteq O_v^k$ such that $d-sep_{\mathcal{G}}\big(w, O_v^k \setminus A \,\big|\, A\big)$, giving us the equality $u_{w|O_v^k} = u_{w|A}$. Now, it remains to prove that no proper recursion exists to compute $u_{w|A}$.

We prove this by contradiction. Assume that a proper recursion exists. This recursion must start with an h-function corresponding to a specified copula $c_{aw|A\setminus\{a\}}$ with $a \in A$.

First, we show that this copula cannot be equal to the independence copula. If this was the case, then we would have $d-sep_{\mathcal{G}}\big(w, a \,\big|\, A\setminus\{a\}\big)$, implying $d-sep_{\mathcal{G}}\big(w, O_v^k \setminus A \,\big|\, A\big)$ and $d-sep_{\mathcal{G}}\big(w, a \,\big|\, A\setminus\{a\}\big)$. By the contraction property of d-separation, see [27, p. 128], this implies that $d-sep_{\mathcal{G}}\big(w, O_v^k \setminus \big(A\setminus\{a\}\big) \,\big|\, A\setminus\{a\}\big)$. This is a contradiction with the definition of the set A. Indeed, we picked A to be the smallest set for which this d-separation holds. Hence, the copula $c_{aw|A\setminus\{a\}}$ is not equal to the independence copula, and therefore must be specified by an arc $w \to a$ or $a \to w$. We consider both cases.

- $\underline{w \to a}$: In this case we have that $c_{aw|A\setminus\{a\}} = c_{wa|pa(a\downarrow w)}$. Hence, $A\setminus\{a\} = pa(a\downarrow w)$, and thus $A = \underline{pa}(a\downarrow w) := pa(a\downarrow w)\sqcup\{a\}$. This gives

$$d-sep_{\mathcal{G}}\big(w, O_v^k\setminus A \,\big|\, A\big) = d-sep_{\mathcal{G}}\big(w, O_v^k \setminus \underline{pa}(a\downarrow w) \,\big|\, \underline{pa}(a\downarrow w)\big),$$

  Moreover, it is the case that $\underline{pa}(a\downarrow w) = A \subseteq O_v^k$, which means that $w \in Poss.Cand_{In}(O_v^k)$ by the incoming arc $w \to a$. This of course contradicts the fact that $w \notin Poss.Cand(O_v^k)$.

- $\underline{a \to w}$: In this case $c_{wa|A\setminus\{w\}} = c_{aw|pa(w\downarrow a)}$. Hence, we have that $A\setminus\{a\} = pa(w\downarrow a)$, and thus $A = \overline{pa}(w\downarrow a) := pa(w\downarrow a)\sqcup\{a\}$. Similarly to the earlier case

$$d-sep_{\mathcal{G}}\big(w, O_v^k\setminus A \,\big|\, A\big) = d-sep_{\mathcal{G}}\big(w, O_v^k \setminus \overline{pa}(w\downarrow a) \,\big|\, \overline{pa}(w\downarrow a)\big).$$

  Moreover, $\overline{pa}(w\downarrow a) = A \subseteq O_v^k$ which means that $w \in Poss.Cand_{Out}(O_v^k)$ by the outgoing arc $a \to w$. This contradicts the fact that $w \notin Poss.Cand(O_v^k)$.

Thus, there is no proper recursion to compute $u_{w|O_v^k}$, and the proof of P2 is concluded.

4.2.3. The set of possible candidates is not empty (P3)

We will show that at any point of the algorithm, we are able to extend the current order O_v^k with a possible candidate $w \in Poss.Cand(O_v^k)$. Thus, we must prove that there exists a node $w \in Poss.Cand(O_v^k)$.

By definition, we have that

$$Poss.Cand(O_v^k) = Poss.Cand_{Ind}(O_v^k) \sqcup Poss.Cand_{In}(O_v^k) \sqcup Poss.Cand_{Out}(O_v^k).$$

If $Poss.Cand_{Ind}(O_v^k)$ is not empty, then the proof is complete. Therefore, we assume that $Poss.Cand_{Ind}(O_v^k) = \emptyset$. Consequently, we can apply Lemma 5.14 to find that $ad(O_v^k) \cap B(O_v^k) \neq \emptyset$. Thus, there must exist a $w_1 \in B(O_v^k)\setminus O_v^k$ and $o_1 \in O_v^k$ such that $w_1 \to o_1 \in E$ or $o_1 \to w_1 \in E$.

We will now show that the existence of an arc $w_1 \to o_1$ implies that $Poss.Cand_{In}(O_v^k)$ is not empty. Thus, after establishing this claim we will assume that no arc of the form $w_1 \to o_1$ exists. Hereafter, we prove that this

statement paired with the existence of an arc $o_1 \to w_1$ implies that $Poss.Cand_{Out}(O_v^k)$ is not empty, concluding the proof. The cases of the existence of the arcs $w_1 \to o_1$ and $o_1 \to w_1$ are considered separately.

First case: $w_1 \to o_1 \in E$.

If w_1 can be added to O_v^k by the incoming arc $w_1 \to o_1$, then $w_1 \in Poss.Cand_{In}(O_v^k)$, completing the proof. Thus, we assume that $w_1 \notin Poss.Cand_{In}(O_v^k)$. Since $w_1 \in pa(o_1) \cap \left(B(O_v^k) \setminus O_v^k\right)$, then we must show that w_1 does not belong to the set $pa(o_1) \cap \left(B(O_v^k) \setminus O_v^k\right) \cap Poss.Cand_{In}(O_v^k)$. Assume now that for all $w \in pa(o_1) \cap \left(B(O_v^k) \setminus O_v^k\right)$, we have $w \notin Poss.Cand_{In}(O_v^k)$. To find a contradiction with the statement above, the lemma below will be used. This lemma states that under the assumptions above, the arc $w_1 \to o_1$ implies the existence of another pair of nodes $w_2 \in B(O_v^k) \setminus O_v^k$ and $o_2 \in O_v^k$ such that $w_2 \to o_2$ and $o_1 \to o_2$. It is the case that $o_2 \neq o_1$, but w_1 and w_2 may be the same node. We will again apply this lemma with arc $w_2 \to o_2$ and obtain a sequence of arcs.

Lemma 4.6. Let $w_1 \in B(O_v^k) \setminus O_v^k$ and $o_1 \in O_v^k$ such that $w_1 \to o_1$. Assume

$$pa(o_1) \cap \left(B(O_v^k) \setminus O_v^k\right) \cap Poss.Cand_{In}(O_v^k) = \emptyset.$$

Then, there exist $w_2 \in B(O_v^k) \setminus O_v^k$ and $o_2 \in O_v^k$ such that $w_2 \to o_2$ and $o_1 \to o_2$.

By applying Lemma 4.6 iteratively, we obtain a sequence of connected nodes of $\mathcal{G}$

$$o_1 \to o_2 \to o_3 \to \cdots$$

Since the graph $\mathcal{G}$ is acyclic and has a finite number of nodes, this sequence must therefore be finite. Let o^* be the last element of the longest sequence that can be constructed starting from o_1. Then some parent of o^* must belong to $Poss.Cand_{In}(O_v^k)$, otherwise, o^* would not be the last node in the sequence. This means that $Poss.Cand_{In}(O_v^k) \neq \emptyset$, completing the proof. It remains to prove Lemma 4.6.

Proof of Lemma 4.6. Without loss of generality, we can assume that w_1 is the smallest element with respect to $<_{o_1}$ in $pa(o_1) \cap \left(B(O_v^k) \setminus O_v^k\right)$. By assumption, $w_1 \notin Poss.Cand_{In}(O_v^k)$, hence one of the conditions below must be violated:

1. $pa(o_1 \downarrow w_1) \subseteq O_v^k$.
2. $d\text{-}sep_{\mathcal{G}}\left(w_1, O_v^k \setminus \underline{pa}(o_1 \downarrow w_1) \,\middle|\, \underline{pa}(o_1 \downarrow w_1)\right)$.

The first restriction is satisfied by the lemma below.

Lemma 4.7. Let w_1 be the smallest element in $B(O_v^k) \cap pa(o_1)$ with respect to $<_{o_1}$. Then, $pa(o_1 \downarrow w_1) \subseteq O_v^k$.

Proof of Lemma 4.7. Suppose that there exists an $x \in pa(o_1 \downarrow w_1) \setminus O_v^k$. That is, $x \in pa(o_1) \setminus O_v^k$ and $x <_{o_1} w_1$. Since w_1 is the smallest node in $B(O_v^k) \cap pa(o_1)$, we have $x \notin B(O_v^k)$. Remark that o_1 has corresponding B-set $B(O_v^k) \cap pa(o_1)$ which contains w_1. Since $x <_{o_1} w$, x must also be included in this B-set. Otherwise, our algorithm would have never chosen the order $x <_{o_1} w$. Indeed, by Lemma 5.17 all previously determined parental orders chosen by the algorithm abide by the B-sets in the sense of Definition 3.19. But then, $x \in B(O_v^k)$ as well, which is a contradiction with the definition of w_1. □

Therefore, the second condition must be violated. We consider two possible cases.

- $\underline{pa}(o_1 \uparrow w_1) \cap O_v^k \neq \emptyset$: Lemma 5.19 immediately implies that there exists an $o_2 \in O_v^k$ as desired (where w_1 is w, o_1 is o_i and o_2 is $\tilde{o}$ in the notation of Lemma 5.19) and we set $w_2 := w_1$.

- $\underline{pa}(o_1 \uparrow w_1) \cap O_v^k = \emptyset$: We can apply Lemma 5.28 (with $w = w_1$ and $o = o_1$ in the notation of Lemma 5.28) to find that there exists a trail between w_1 and a node $\tilde{o} \in O_v^k \setminus \underline{pa}(o_1 \downarrow w_1)$ which is activated by $\underline{pa}(o_1 \downarrow w_1)$ containing no converging connections. Let us pick a shortest such trail.

$$w_1 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons \tilde{o}.$$

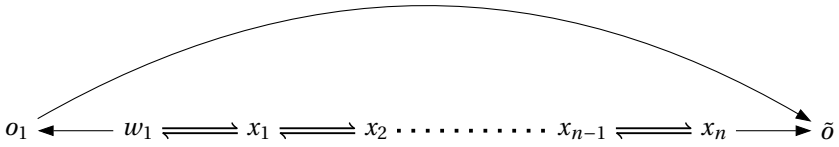
Remark that this trail is a shortest trail activated by the empty set between $w_1 \in B(O_v^k)$ and $\tilde{o} \in O_v^k \subseteq B(O_v^k)$ consisting of nodes in $V \setminus \underline{pa}(o_1 \downarrow w_1)$. Therefore, we can combine Lemma 5.5 (with $K = V \setminus \underline{pa}(o_1 \downarrow w_1)$) and Lemma 5.12 to find that $\{x_i\}_{i=1,\dots,n} \subseteq B(O_v^k)$. In particular we obtain $x_n \in B(O_v^k)$.

Furthermore, x_n cannot be contained in O_v^k. Otherwise, the trail from w to x_n would be an even shorter active trail from w to a node in O_v^k. Hence, $x_n \in B(O_v^k) \setminus O_v^k$.

Now, the trail

$$o_1 \leftarrow w \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons \tilde{o}$$

is an active trail between two nodes in O_v^k given the empty set consisting of nodes not in O_v^k. Thus, by Lemma 5.15, o_1 and $\tilde{o}$ must be adjacent. By the assumption that $pa(o_1 \uparrow w_1) \cap O_v^k = \emptyset$, we have $\tilde{o} \notin pa(o_1 \uparrow w_1)$. Remark that $\tilde{o} \notin \underline{pa}(o_1 \downarrow w_1)$ (by definition of $\tilde{o}$) and that $pa(o_1) = pa(o_1 \downarrow w_1) \sqcup pa(o_1 \uparrow w_1) \sqcup \{w_1\}$. This shows that $\tilde{o} \notin pa(o_1)$. Therefore, we must have $o_1 \in pa(\tilde{o})$; this means that we have the subgraph below.



Clearly, $\tilde{o}$ is our desired node o_2 and x_n is our desired node w_2. Indeed, we have $o_1 \to \tilde{o}$ and $x_n \to \tilde{o}$, with $\tilde{o} \in O_v^k$ and $x_n \in B(O_v^k) \setminus O_v^k$.

□

Second case: $o_1 \to w_1 \in E$.

First, we remark that if E contains arcs of the form $w \to o$ with $w \in B(O_v^k) \setminus O_v^k$ and $o \in O_v^k$, then by the previous case we have that $Poss.Cand_{In}(O_v^k) \neq \emptyset$. Therefore, we can assume that E contains no such arcs.

If $w_1 \in Poss.Cand_{Out}(O_v^k)$, then the proof is complete. Assume now that $w_1 \notin Poss.Cand_{Out}(O_v^k)$. We will show that $Poss.Cand_{Out}(O_v^k) \neq \emptyset$ with the lemma below. The lemma states that under the assumptions above, the arc $o_1 \to w_1$ implies the existence of another pair of nodes $w_2 \in B(O_v^k) \setminus O_v^k$ and $o_2 \in O_v^k$ such that w_1 and w_2 are connected by a trail where all arcs point in direction of w_1. We can repeat this argument and construct a sequence of nodes.

Lemma 4.8. Let $w_1 \in B(O_v^k) \setminus O_v^k$ and $o_1 \in O_v^k$ such that $o_1 \to w_1 \in E$. Assume that $w_1 \notin Poss.Cand_{Out}(O_v^k)$, and E contains no arcs of the form $w \to o$ with $w \in B(O_v^k) \setminus O_v^k$ and $o \in O_v^k$. Then, there exist $w_2 \in B(O_v^k) \setminus O_v^k$ and $o_2 \in O_v^k$ such that $o_2 \to w_2$, and w_1 and w_2 are connected by a trail of the form

$$w_1 \leftarrow x_1 \leftarrow \cdots \leftarrow x_n \leftarrow w_2.$$

By applying Lemma 4.8 iteratively, we obtain a sequence

$$w_1 \leftarrow \cdots \leftarrow w_2 \leftarrow \cdots \leftarrow w_3 \leftarrow \cdots .$$

Since the graph $\mathcal{G}$ is acyclic and has a finite set of nodes, this sequence must be finite. Let w^* be the last element of the longest sequence that can be constructed starting from w_1. Then, w^* must belong to $Poss.Cand_{Out}(O_v^k)$, otherwise, it would not be the last. Hence, we have $Poss.Cand_{Out}(O_v^k) \neq \emptyset$, completing the proof. It remains to prove Lemma 4.8.

Proof of Lemma 4.8. Without loss of generality, we can assume that o_1 is the highest element in $O_v^k \cap pa(w_1)$ with respect to $<_{w_1}$.

We consider two cases.

- $pa(w_1 \downarrow o_1) \setminus O_v^k \neq \emptyset$: Let $x_1 \in pa(w_1 \downarrow o_1) \setminus O_v^k$. That is, $x_1 \in pa(w_1) \setminus O_v^k$ and $x_1 <_{w_1} o_1$. Remark that $B(O_v^k) \cap pa(w_1)$ is a B-set corresponding to node w_1 which contains the node o_1. By Lemma 5.17, all parental orders determined by the algorithm abide by the B-sets. Consequently, the order $x_1 <_{w_1} o_1$ implies that $x_1 \in B(O_v^k) \cap pa(w_1)$, and thus $x_1 \in B(O_v^k)$. Hence, $x_1 \in B(O_v^k) \setminus O_v^k$.

  By the assumption that $Poss.Cand_{Ind}(O_v^k) = \emptyset$, we have $\not{d\text{-}sep}_{\mathcal{G}}\left(x_1, O_v^k \,|\, \emptyset\right)$. Let us pick a shortest trail from x_1 to O_v^k

  $$x_1 \rightleftharpoons x_2 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons \tilde{o} \tag{4.3}$$

  activated by the empty set with $\tilde{o} \in O_v^k$. Note that x_1 and $\tilde{o}$ are both included in the B-set $B(O_v^k)$. Therefore, by Lemma 5.12, we have that $x_i \in B(O_v^k)$ for all $i = 1, \ldots, n$. In particular, $x_n \in B(O_v^k)$. Furthermore, x_n is not contained in the set O_v^k. Otherwise, the trail

  $$x_1 \rightleftharpoons x_2 \rightleftharpoons \cdots \rightleftharpoons x_n$$

  would be a shorter trail from x_1 to O_v^k activated by the empty set than (4.3), which is a contradiction. Therefore, x_n must be in $B(O_v^k) \setminus O_v^k$.

  We assumed that there is no arc pointing from a node in $B(O_v^k) \setminus O_v^k$ to a node in O_v^k. This means that the arc $x_n \rightarrow \tilde{o}$ is not possible. Consequently, the trail (4.3) must contain the arc $x_n \leftarrow \tilde{o}$.

  Since, the trail is activated by the empty set it contains no converging connections by Lemma 5.1. Therefore, (4.3) must be of the form

  $$w_1 \leftarrow x_1 \leftarrow \cdots \leftarrow x_n \leftarrow \tilde{o}.$$

  with $x_n \in B(O_v^k) \setminus O_v^k$ and $\tilde{o} \in O_v^k$. Hence, x_n is our desired node w_2 and $\tilde{o}$ is our desired node o_2.

- $pa(w_1 \downarrow o_1) \setminus O_v^k = \emptyset$: By assumption, we have that $w_1 \notin Poss.Cand_{Out}(O_v^k)$. Hence, one of the following conditions is violated:

  1. $pa(w_1 \downarrow o_1) \subseteq O_v^k$.
  2. $\not{d\text{-}sep}_{\mathcal{G}}\left(w_1, O_v^k \setminus \overline{pa}(w_1 \downarrow o_1) \,|\, \overline{pa}(w_1 \downarrow o_1)\right)$.

  Remark that the assumption that $pa(w_1 \downarrow o_1) \setminus O_v^k = \emptyset$ implies that the first condition is satisfied. Hence, the second condition must be violated. Therefore, there exists a trail between w_1 and a node in $O_v^k \setminus \overline{pa}(w_1 \downarrow o_1)$ activated by $\overline{pa}(w_1 \downarrow o_1)$. By Lemma 5.29, there exists such a trail containing no converging

connections. Thus, we can pick a shortest trail from w to $O_v^k \setminus \overline{pa}(w_1 \downarrow o_1)$ activated by $\overline{pa}(w_1 \downarrow o_1)$ containing no converging connections:

$$w_1 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons \tilde{o} \tag{4.4}$$

with $\tilde{o} \in B(O_v^k) \setminus O_v^k$.

First, we show that (4.4) must be of length $n > 1$. If $n = 0$, then we would have that $w_1 \rightleftharpoons \tilde{o}$. This arc must point to the left, since the arc $w_1 \rightarrow \tilde{o}$ is an arc from a node in $B(O_v^k) \setminus O_v^k$ to a node in O_v^k which cannot be present by the assumptions of the lemma. By the arc $w_1 \leftarrow \tilde{o}$, we know that $\tilde{o} \in pa(w_1)$. Moreover, by definition the node $\tilde{o}$ is included in $O_v^k \setminus \overline{pa}(w_1 \downarrow o_1)$, and thus $\tilde{o} \in O_v^k \setminus \overline{pa}(w_1 \downarrow o_1) \cap pa(w_1) = O_v^k \cap pa(w_1 \uparrow o_1)$. This means that $o_1 <_{w_1} \tilde{o}$. But, we picked o_1 to be largest element in $O_v^k \cap pa(w_1)$ according to $<_{w_1}$. Therefore, $o_1 <_{w_1} \tilde{o}$ is not possible, proving that $n > 1$.

Now, we show that for all $i = 1, \ldots, n$, $x_i \notin O_v^k$. Suppose that for some i, we have that $x_i \in O_v^k$. The set O_v^k can be rewritten as $O_v^k = \left(O_v^k \setminus \overline{pa}(w_1 \downarrow o_1)\right) \sqcup \overline{pa}(w_1 \downarrow o_1)$. Since $x_i \in O_v^k$, it must be in $O_v^k \setminus \overline{pa}(w_1 \downarrow o_1)$ or in $\overline{pa}(w_1 \downarrow o_1)$. If $x_i \in O_v^k \setminus \overline{pa}(w_1 \downarrow o_1)$, then the $w_1 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_i$ would be a shorter trail between w_1 and $O_v^k \setminus \overline{pa}(w_1 \downarrow o_1)$ activated by $\overline{pa}(w_1 \downarrow o_1)$ than (4.4). This is a contradiction, because we picked a shortest such trail. Hence, x_i must be in $\overline{pa}(w_1 \downarrow o_1)$. However, in this case the trail (4.4) would be blocked by $\overline{pa}(w_1 \downarrow o_1)$ which is also a contradiction. Therefore, x_i cannot be in O_v^k proving the claim.

The trail (4.4) is a shortest trail activated by the empty set between two nodes in $B(O_v^k)$ (w and $\tilde{o}$), and thus by Lemma 5.12, $x_i \in B(O_v^k)$ for all $i = 1, \ldots, n$.

This means that for all $i = 1, \ldots, n$, $x_i \in B(O_v^k) \setminus O_v^k$, in particular $x_n \in B(O_v^k) \setminus O_v^k$. Similarly to the previous case, this implies that we must have $x_n \leftarrow \tilde{o}$, and therefore (4.4) takes the form

$$w_1 \leftarrow x_1 \leftarrow \cdots \leftarrow x_n \leftarrow \tilde{o}$$

with $x_n \in B(O_v^k) \setminus O_v^k$ and $\tilde{o} \in O_v^k$ and $n > 0$. So, x_n is our desired node w_2 and $\tilde{o}$ is our desired node o_2.

The proof of Lemma 4.6 is completed. □

5

On the properties of restricted DAGs

In this chapter we prove many useful results concerning restricted DAGs. These include properties concerning trails, B-sets, partial orders, possible candidates and more. Although, the ultimate purpose of the lemmas presented in this chapter is to solve the proof in Section 4.2.3, each lemma provides an interesting insight on their own.

5.1. About trails with no converging connection

First, a simple but interesting result which states that trails with no converging connections are equivalent to trails activated by the empty set is presented.

Lemma 5.1. A trail is activated by the empty set if and only if it does not contain a converging connection.

Proof. This statement follows directly from the definition of d-separation, see Definition 2.22. □

If a trail contains no converging connections, then it must have at most one diverging connection. An intuitive property of such a diverging node is that it is an ancestor of both end-points. Therefore, we refer to it as a **common ancestor**. Whenever a trail contains only serial connections, the common ancestor is defined to be the end-point to which the arrows point away from.

Definition 5.2 (Common ancestor)**.** Let $\mathcal{G}$ be a DAG and let x_0 and x_{n+1} be two nodes joined by a trail

$$x_0 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons x_{n+1}$$

with no converging connections. The ***common ancestor*** among this trail is defined as follows.

- If x_0 is an ancestor of x_{n+1}, then $x_m = x_0$.
- If x_{n+1} is an ancestor of x_0, then $x_m = x_{n+1}$.
- If x_0 is not an ancestor of x_{n+1} and vice versa, then x_m with $m \in \{1, \dots, n\}$ is an ancestor of both x_0 and x_{n+1}.

Remark that x_m is well-defined, since strictly one of the cases above holds.

We remark that in every figure from now on the common ancestors will be displayed in the middle of a trail. Therefore, the common ancestor will always be denoted with a subscript "m" which is an abbreviation for

"middle".

In many proofs we will repeatedly pick trails between nodes, e.g. x_0 and x_{n+1}, for which all nodes on the trail are included in a certain subset $K \subseteq V$. In this case we say that the trail consists only of elements of K. This does not include the end-points (x_0 and x_{n+1}), i.e. these end-points may or may not be in K.

Definition 5.3. Let $\mathcal{G} = (V, E)$ be a DAG, let $K \subseteq V$, and let $x_0 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_{n+1}$ be a trail. We say that the trail consists only of elements of K if $\forall\, i = 1, \ldots, n$, $x_i \in K$.

In few lemmas below we prove that a shortest trail satisfying a certain property also satisfies a second property. Let us first formalize what is meant by a property of a trail.

Definition 5.4 (Trail property)**.** Let $\mathcal{G}$ be a DAG containing a trail $x_0 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_{n+1}$. A ***property*** $\mathfrak{P} := \mathfrak{P}(x_0, \ldots, x_{n+1})$ specifies the existence of certain arcs between the nodes on the trail. Here, we mean that $\mathfrak{P}$ states that E contains a certain set of arcs $\{x_i \to x_j;\, i \in I, j \in J\}$ with $I, J \subseteq \{0, 1, \ldots, n+1\}$.

For instance, the following are regarded as trail properties:

- The first arc of the trail points to the left; $x_0 \leftarrow x_1$.
- The i-th and j-th node on the trail are adjacent; $x_i \rightleftharpoons x_j$.
- The trail is of the form $x_0 \leftarrow x_1 \to x_2 \to \cdots \to x_{n-1} \to x_n$, and we have that $x_0 \to x_{n-1}$.

Proofs where one property of a trail implies another will not only hold for shortest trails but also for shortest trails consisting of nodes in a subset $K \subseteq V$. For instance, Lemma 5.6 also holds for shortest trails activated by the empty set consisting of nodes in K. Instead of repeatedly saying that a statement holds for both a shortest trail and a shortest trail consisting of nodes in a subset K and proving both cases, we establish the following lemma.

Lemma 5.5. For a DAG $\mathcal{G}$ with no active cycles and interfering v-structures, for a trail

$$x_0 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_{n+1}, \tag{5.1}$$

let $\mathfrak{P}_1(x_0, x_1, \ldots, x_{n+1})$ and $\mathfrak{P}_2(x_0, x_1, \ldots, x_{n+1})$ be two statements. Assume that for any DAG $\mathcal{G} = (V, E)$ with no active cycles, nor interfering v-structures, for any $x_0, x_{n+1} \in V$, and for any shortest trail (5.1) between x_0 and x_{n+1} that satisfies $\mathfrak{P}_1$, the property $\mathfrak{P}_2$ holds.

Let $\mathcal{G} = (V, E)$ be a DAG with no active cycles, nor interfering v-structures, let $K \subseteq V$. Then for any shortest trail between x_0 and x_{n+1} that satisfies $\mathfrak{P}_1$ and that consists only of elements of K, the property $\mathfrak{P}_2$ still holds.

Proof. Let $x_0 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_{n+1}$ be a shortest trail between x_0 and x_{n+1} that satisfies $\mathfrak{P}_1$ and consists only of elements of K. Consider the subgraph $\mathcal{G}^*$ induced by $\{x_0, x_{n+1}\} \cup K$. Note that this trail is a shortest trail satisfying $\mathfrak{P}_1$ between x_0 and x_{n+1} in $\mathcal{G}^*$. Therefore, by assumption, it must satisfy $\mathfrak{P}_2$. □

During later proofs we will often pick a shortest trail. In many occasions, the property of being a shortest trail allows us to exclude the presence of certain chords. For instance, a shortest trail activated by the empty set does not contain a chord.

Lemma 5.6. Let $\mathcal{G}$ be a DAG with no active cycles and let

$$x_0 \rightleftharpoons x_1 \rightleftharpoons \ldots \rightleftharpoons x_n \rightleftharpoons x_{n+1} \tag{5.2}$$

be a trail in $\mathcal{G}$ for some $n \geq 0$. If this is the shortest trail between x_0 and x_{n+1} activated by the empty set, then (5.2) has no chords.

Proof. Let x_m be the common ancestor among (5.2), see Definition 5.2.

The proof is completed by remarking that:

- $x_i \to x_j$ with $i < j \leq m$ results in a cycle.
- $x_i \leftarrow x_j$ with $i < j \leq m$ results in a shorter trail.
- $x_i \to x_j$ with $m \leq i < j$ results in a shorter trail.
- $x_i \leftarrow x_j$ with $m \leq i < j$ results in a cycle.
- $x_i \to x_j$ with $i < m < j$ results in a shorter trail.
- $x_i \leftarrow x_j$ with $i < m < j$ results in a shorter trail.

□

The lemma below states that if $\mathcal{G}$ contains a shortest trail $x_0 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons x_{n+1}$ activated by the empty set for which $x_0 \to v$ and $x_{n+1} \to v$ for some node $v \in V$, then for all $i = 1, \ldots, n$, $x_i \to v$.

Lemma 5.7. Let $\mathcal{G}$ be a DAG with no active cycles and let

$$x_0 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons x_{n+1} \tag{5.3}$$

be a trail in $\mathcal{G}$ for some $n \geq 0$. If this is a shortest trail between x_0 and x_{n+1} activated by the empty set, then

(i) $ch(x_0) \cap ch(x_{n+1}) \subseteq \bigcap_{i=1}^{n} ch(x_i)$,

(ii) $\forall i = 1, \ldots, n,\ x_i \notin ch(x_0) \cap ch(x_{n+1})$.

Proof. (ii) is a straightforward consequence of (i). We now prove (i). Let $v \in ch(x_0) \cap ch(x_{n+1})$. To prove this, suppose that there exists an i such that $v \notin ch(x_i)$. We define the nodes x_l and x_r using the integers

$$l := \max\{j \in \{0, \ldots i-1\};\ v \in ch(x_j)\},$$
$$r := \min\{j \in \{i+1, \ldots, n+1\};\ v \in ch(x_j)\}.$$

With this notation, x_l (respectively x_r) is the first node to the left (resp. right) of x_i that is a parent of v. l and r are well-defined since $v \in ch(x_0) \cap ch(x_{n+1})$. Now, $\mathcal{G}$ contains the graph displayed in Figure 5.1. Let us consider the trail

$$v \leftarrow x_l \rightleftharpoons \cdots \rightleftharpoons x_i \rightleftharpoons \cdots \rightleftharpoons x_r \to v. \tag{5.4}$$

Any chord of this trail must be either a chord of (5.3), an arc $v \to x_j$ or an arc $x_j \to v$ with $j \in \{l+1, \ldots, r-1\}$.

The first case is not possible by Lemma 5.6.

The second case is not possible because by Lemma 5.1 trail (5.3) contains at most one diverging connection, and therefore trail (5.4) contains exactly one diverging connection. Consequently, any arc $v \to x_j$ would result in a cycle.

The third case is not possible by definition of l and r. Therefore, we have shown that (5.4) does not contain any chord. Thus, $\mathcal{G}$ contains the active cycle (5.4), which is a contradiction.

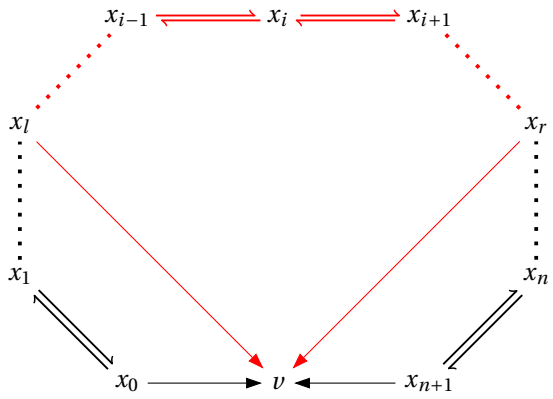


Figure 5.1: Subgraph in $\mathcal{G}$ with the active cycle coloured in red.

□

It should be noted that we cannot use Lemma 5.5 to generalize the lemma above. Because the properties 5.7(i,ii) do not only concern the nodes $x_0, x_1, \ldots, x_{n+1}$ but also their children. Therefore, we prove the generalization in the corollary below.

Corollary 5.8. Let $\mathcal{G}$ be a DAG with no active cycles and let

$$x_0 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons x_{n+1} \tag{5.5}$$

be a trail in $\mathcal{G}$ for some $n \geq 0$. If this is the shortest trail between x_0 and x_{n+1} activated by the empty set consisting of nodes in $K \subseteq V$, then

(i) $ch(x_0) \cap ch(x_{n+1}) \subseteq \bigcap_{i=1}^{n} ch(x_i)$;

(ii) $\forall i = 1, \ldots, n,\ x_i \notin ch(x_0) \cap ch(x_{n+1})$.

Proof. (ii) is a straightforward consequence of (i). We now prove (i). Trail 5.5 is a shortest trail activated by the empty set consisting of nodes in K, therefore by combining Lemmas 5.5 and 5.6 it contains no chords.

Let $\mathcal{G}^* = (V^*, E^*)$ be the subgraph induced by

$$V^* = K \cup \{x_0, x_{n+1}\} \cup \big(ch(x_0) \cap ch(x_{n+1})\big).$$

By Lemma 5.7(ii), any shortest trail between x_0 and x_{n+1} in $\mathcal{G}^*$ activated by the empty set must not contain a node in $ch(x_0) \cap ch(x_{n+1}) \cap K = ch(x_0) \cap ch(x_{n+1})$. Therefore, any shortest trail between x_0 and x_{n+1} in $\mathcal{G}^*$ activated by the empty set consists of nodes in K.

Thus, trail (5.5) is a shortest trail in $\mathcal{G}^*$ activated by the empty set. Now, we can apply Lemma 5.7 to trail (5.5) in $\mathcal{G}^*$ to find that indeed $ch(x_0) \cap ch(x_{n+1}) \subseteq \big(\cap_{i=1}^{n} ch(x_i) \cap K\big) \subseteq \cap_{i=1}^{n} ch(x_i)$. □

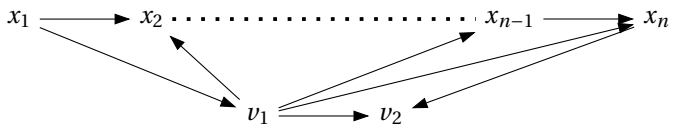
The lemma below states if $v_1 \to v_2$ for some $v_1, v_2 \in V$, the existence of a trail between v_1 and v_2 activated by the empty set and starting with an arc pointing to v_1 implies the existence of a particular subgraph. This lemma will be very useful in Section 5.3.

Lemma 5.9. Let $\mathcal{G}$ be a DAG with no active cycles, nor interfering v-structures and let $v_1, v_2 \in V$ such that $v_1 \to v_2$. Suppose that

$$v_1 \leftarrow x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons v_2$$

is a shortest trail trail activated by the empty set starting with an arc $v_1 \leftarrow x_1$. Assume that $n \geq 1$. Then, for all $i \in \{1, \ldots, n\}$, $x_i \to x_{i+1}$ with the convention that $x_{n+1} := v_2$ and for all $i \in \{2, \ldots, n\}$ $v_1 \to x_i$.

This means that $\mathcal{G}$ contains the subgraph below.



Furthermore, the lemma also holds for shortest trails activated by the empty set and of the form

$$v_1 \leftarrow x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \to v_2 \tag{5.6}$$

with $n \geq 1$.

Proof. Consider a shortest trail

$$v_1 \leftarrow x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons v_2 \tag{5.7}$$

activated by the empty set with $n \geq 1$.

Consider the case when $n = 1$. Here, the trail takes the form $v_1 \leftarrow x_1 \rightleftharpoons v_2$ with $v_1 \to v_2$. If $x_1 \leftarrow v_2$, then we obtain the cycle $v_1 \leftarrow x_1 \leftarrow v_2 \leftarrow v_1$, and therefore a contradiction. Therefore, the arc $x_1 \to v_2$ must be present, giving us exactly the claimed subgraph, completing the proof.

Now, let us assume that $n > 1$. We first show that $x_n \to v_2$. Suppose that $v_2 \to x_n$, then the trail takes the form

$$v_1 \leftarrow x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \leftarrow v_2.$$

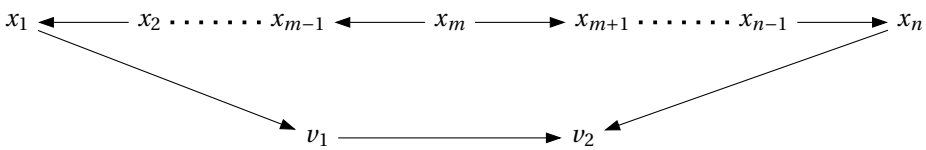
Since this trail is activated by the empty set it contains no converging connections (Lemma 5.1). Hence, the trail must take the form

$$v_1 \leftarrow x_1 \leftarrow \cdots \leftarrow x_n \leftarrow v_2.$$

However, since $v_1 \to v_2$, this results in a cycle, and therefore a contradiction. So, we get that $x_n \to v_2$.

Because we must have $x_n \to v_2$, the trails (5.6) and (5.7) coincide.

Let x_m be the common ancestor among (5.7), see Definition 5.2. Now, $\mathcal{G}$ contains the subgraph below.

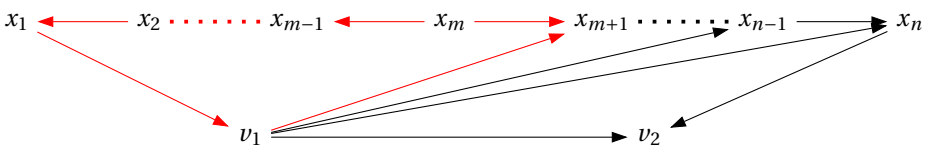


In this case, the subgraph above contains an undirected cycle with one converging connection (at v_2). Since $\mathcal{G}$ does not contain an active cycle, E must contain the appropriate chords.

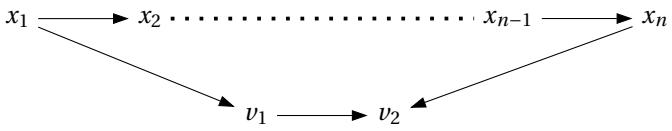
Several group of chords can be excluded:

- The trail $x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \to v_2$ is a shortest trail activated by the empty set, and therefore by Lemma 5.6 it has no chords.
- $v_1 \to x_j$ with $j \leq m$ results in a cycle.
- $x_j \to v_1$ with $j \in \{2,\ldots,m\}$ results in a trail $v_1 \leftarrow x_j \rightleftharpoons \cdots \rightleftharpoons x_n \to v_2$ which would be shorter than the shortest trail (5.7) (while still being activated by the empty set). This is a contradiction.

The only remaining chords are of the form $v_1 \to x_i$ with $i \in \{m+1,\ldots,n\}$. First, we show that the diverging node x_m must be the first node on trail (5.7), i.e. $x_m = x_1$. To see this, we consider the case where all possible chords are present in E, giving us the subgraph below. This graph contains an undirected cycle, coloured in red. Since there are no more chords which could be present, this undirected cycle is an active cycle, unless it is of length strictly smaller than 4. The undirected cycle is made up of the nodes $x_1,\ldots,x_{m+1}$ and v_1; it is therefore of length $m+2$. This means that $m+2 \leq 3$; the only possibility is then that $m = 1$, and therefore $x_m = x_1$.



Now, $\mathcal{G}$ contains the undirected cycle below. It is evident that all chords $v_1 \to x_i$ must be present for this undirected cycle not to be an active cycle. Therefore, $\mathcal{G}$ must contain the subgraph as given by the lemma, completing the proof.



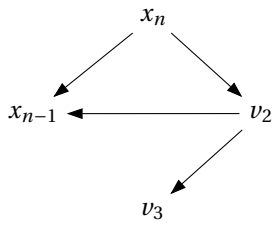
□

Similarly to the previous lemma, the lemma below states that under certain conditions the existence of a trail between two nodes v_1 and v_2 activated by the empty set implies the existence of a specific subgraph. In this case, the conditions state that v_1 and v_2 are both parents of another node v_3 and the last arc along the trail between v_1 and v_2 points towards v_2. Moreover, no node on the trail can be a parent of v_3. The lemma will be particularly useful in Section 5.4.

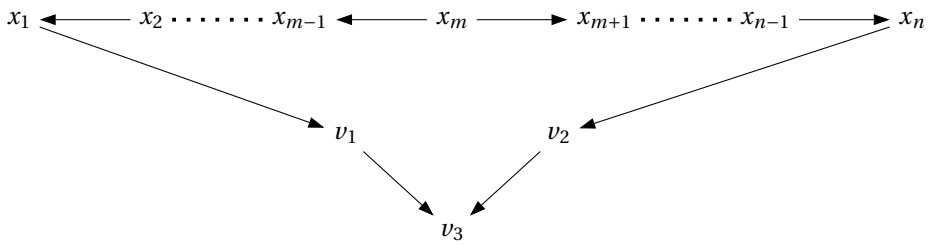
Lemma 5.10. Let $\mathcal{G}$ be a DAG with no active cycles, nor interfering v-structures and let $v_1, v_2, v_3 \in V$ such that $v_1, v_2 \in pa(v_3)$. Suppose that v_1 and v_2 are connected by a trail

$$v_1 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \to v_2 \tag{5.8}$$

activated by the empty set with $\{x_i\}_{i=1}^n \cap pa(v_3) = \emptyset$ and $n \geq 1$. If this is a shortest such trail, then $\mathcal{G}$ contains the subgraph below, with the convention $x_0 := v_1$.



Proof. Let us use the convention $x_{n+1} = v_2$. Let x_m be the common ancestor among 5.8, see Definition 5.2, Then, $\mathcal{G}$ contains the subgraph below.

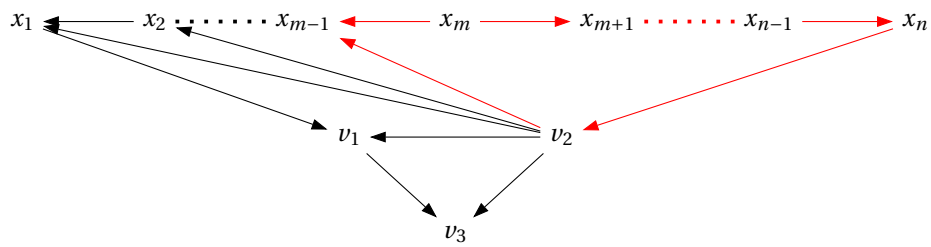


The graph above is an undirected cycle with one converging connection (at v_3). Since $\mathcal{G}$ does not contain an active cycle, E must contain the appropriate chords. Several chords can be excluded:

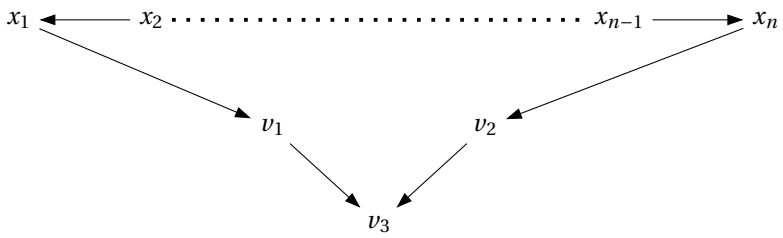
- The trail $v_1 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n$ is a shortest trail activated by the empty set, and therefore it has no chords by Lemma 5.6.
- $v_1 \to v_2$ results in a shorter trail of the form (5.8).
- $x_i \to v_2$ results in a shorter trail of the form (5.8).
- $v_2 \to x_i$ with $i \geq m$ results in a cycle.
- $v_3 \to x_i$ with $i = 1, \ldots, n$ results in a cycle.
- $x_i \to v_3$ with $i = 1, \ldots, n$ cannot be present by the assumptions of the lemma.

Hence, the only possible chords are arcs of the form $v_2 \to v_1$ and $v_2 \to x_i$ with $i \in \{1, \ldots, m-1\}$.

First, we show that the common ancestor must be the last node along the trail, i.e. $x_m = x_n$. Consider the case where all possible chords are present in E, giving us the subgraph below. This subgraph contains an undirected cycle with one converging connection (at x_{m-1}), coloured in red. Since $\mathcal{G}$ does not contain an active cycle, and there are no more arcs which could act as a chord, it must be of length strictly smaller than than 4. This undirected cycle is made up of the nodes $x_{m-1}, x_m \ldots, x_n$ and v_2. It is therefore of length $n - (m-2) + 1 = n - m + 3$. Therefore, $n - m + 3 \leq 3$ so $m = n$, and therefore $x_m := x_n$.



Now, $\mathcal{G}$ contains the undirected cycle below. It is evident that all remaining chords discussed above must be included in E. Otherwise, this undirected cycle would be an active cycle which is a contradiction. In particular, $\mathcal{G}$ contains the subgraph given by the lemma, completing the proof.



□

5.2. Properties of B-sets and possible candidates

Throughout this section we will repeatedly need the same assumptions, and therefore we formalize them below.

Assumption 5.11. Let $\mathcal{G} = (V, E)$ be a DAG. The following conditions are assumed to be satisfied:

1. $\mathcal{G}$ does not contain any active cycles, nor interfering v-structures.
2. $<$ is a well-ordering corresponding to $\mathcal{G}$
3. v is a node in V with $|\, pa(v) \,| > 0$.
4. All previous orders, i.e. $<_w$ with $w < v$, have already been determined by our algorithm.
5. O_v^k is a partial order determined by our algorithm with $k < \left|pa(v)\right|$.

Informally, the lemma below states that two nodes in a B-set B_q are either d-separated given the empty set or any shortest trail activated by the empty set between them must be contained in B_q.

Lemma 5.12. Let $\mathcal{G}$ be a DAG with no active cycles nor interfering v-structures, and let $v \in V$. Let $q \in \{1, \ldots, Q(v)+1\}$ and let $w_1, w_2 \in B_q(v)$. Then, $d - sep_{\mathcal{G}}\big(w_1, w_2 \,\big|\, \emptyset\big)$ or any shortest trail activated by the empty set joining w_1 and w_2 must consist entirely of nodes contained in B_q.

Proof. If $d - sep_{\mathcal{G}}\big(w_1, w_2 \,\big|\, \emptyset\big)$, or if $w_1 = w_2$, then the proof of this lemma is completed. Therefore we can assume that they are not independent and different from each other. Thus ~~$d - sep_{\mathcal{G}}\big(w_1, w_2 \,\big|\, \emptyset\big)$~~; let

$$w_1 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons w_2 \tag{5.9}$$

be a shortest trail between w_1 and w_2 which is activated by the empty set. First, we assume that $q \leq Q(v)$. Let $x_0 := w_1$, $x_{n+1} := w_2$, and b_q be a node corresponding to $B_q(v)$, see Definition 3.18. Because $w_1, w_2 \in B_q(v)$, we know that $v, b_q \in ch(w_1) \cap ch(w_2)$. By Lemma 5.7, for all $i = 1, \ldots, n$, we have $v, b_q \in ch(x_i)$ and $x_i \in B_q(v) = pa(v) \cap pa(b_q)$.

If $q = Q(v)+1$, we are at the last stage of the algorithm and there is no b_q, but the same reasoning shows that for $i = 1, \ldots, n$, $x_i \in B_q(v) = pa(v)$. This concludes the proof. □

We cannot apply Lemma 5.5 to Lemma 5.12, because the property that for all $i = 1, \ldots, n$, $x_i \in B_q(v)$ concerns a node v which is not on the trail. Therefore, we prove the generalization to a subset $K \subseteq V$ in the corollary hereunder.

Corollary 5.13. Let $\mathcal{G}$ be a DAG with no active cycles nor interfering v-structures, and let $v \in V$. Let $q \in \{1, \ldots, Q(v)+1\}$ and let $w_1, w_2 \in B_q(v)$. Let K be a set included in V. Then, w_1 and w_2 are either independent or for any shortest trail activated by the empty set joining w_1 and w_2 consisting of nodes in K must consist entirely of nodes contained in B_q.

Proof. First, we assume that $q \leq B_q(v)$. Let b_q be a node corresponding to B_q. Let $\mathcal{G}^* = (V^*, E^*)$ be the subgraph induced by the nodes in $V^* := \{v, w_1, w_2, b_q\} \cup K$. Note that v and b_q are children of both w_1 and w_2 in $\mathcal{G}^*$. Therefore, by Lemma 5.7(ii), any shortest trail between w_1 and w_2 in $\mathcal{G}^*$ activated by the empty set must not contain v nor b_q. This means that any shortest trail between w_1 and w_2 in $\mathcal{G}^*$ activated by the empty set must consist only of elements of K.

Consider a shortest trail in $\mathcal{G}$

$$w_1 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons w_2 \tag{5.10}$$

consisting of nodes in K, i.e. $\{x_i\}_{i=1}^n \subseteq K$. Therefore, it is a shortest trail activated by the empty set between w_1 and w_2 in $\mathcal{G}^*$. We now apply Lemma 5.12, since w_1 and w_2 belong to the B-set $B_q \cap V^*$ corresponding to v in the graph $\mathcal{G}^*$. Therefore, for all $i = 1, \ldots, n$, $x_i \in B_q$, completing the proof.

If $q = Q(v) + 1$, then the proof is analogous to the previous case, but then with $V^* := (v, w_1, w_2) \cup K$. □

Informally, the lemma below states that if the set $Poss.Cand_{Ind}(O_v^k)$ is empty, then there is a node in $B(O_v^k) \setminus O_v^k$ which is adjacent to a node in the set O_v^k.

Lemma 5.14. Under Assumption 5.11, let $w \in B(O_v^k) \setminus O_v^k$ such that $\not{d\text{-}sep}_{\mathcal{G}}\left(w, O_v^k \middle| \emptyset\right)$, i.e. $w \notin Poss.Cand_{Ind}(O_v^k)$. Then, $ad(O_v^k) \cap B(O_v^k) \neq \emptyset$, where $ad(O_v^k)$ is the adjacency set of O_v^k defined in Definition 2.14.

Proof. By assumption, we have $\not{d\text{-}sep}_{\mathcal{G}}\left(w, O_v^k \middle| \emptyset\right)$. Therefore w must be connected to O_v^k by some trail activated by the empty set. We pick a shortest trail from w to O_v^k given the empty set, as

$$w \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons o \tag{5.11}$$

where $o \in O_v^k$.

If $n = 0$, then $w \in B(O_v^k) \setminus O_v^k$ is adjacent to o and thus $w \in ad(O_v^k) \cap B(O_v^k)$.

Now, assume that $n > 0$. We will prove that $x_n \in ad(O_v^k) \cap B(O_v^k)$. Since we have a shortest trail between two nodes (w and o) in $B(O_v^k)$ with no chords, Lemma 5.12 implies that all $x_i \in B(O_v^k)$. As a particular case, we have $x_n \in B(O_v^k)$.

If $x_n \in O_v^k$, then the trail $w \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n$ would be a shorter trail from w to O_v^k than the trail in (5.11). This is a contradiction, proving that $x_n \notin O_v^k$. Therefore $x_n \in B(O_v^k) \setminus O_v^k$. Note that x_n is adjacent to o, and thus $x_n \in ad(O_v^k) \cap B(O_v^k)$. This concludes the proof. □

By Definition 4.4, we know that a node w is not a possible candidate for partial order O_v^k, if $\not{d\text{-}sep}_{\mathcal{G}}\left(w, O_v^k \middle| \emptyset\right)$ ($w \notin Poss.Cand_{Ind}(O_v^k)$) and w and O_v^k are not adjacent ($w \notin Poss.Cand_{In}(O_v^k) \sqcup Poss.Cand_{Out}(O_v^k)$). We will now prove an even stronger claim. That is, a node w is not a possible candidate to be added to a partial order O_v^k, if there exists an o in O_v^k such that:

- w and o are not adjacent.
- There exists a trail between w and o activated by the empty set which does not contain any nodes in O_v^k.

The lemma below provides a clear intuition into how the algorithm grows a partial order. For example, consider the a trail

$$o \rightleftharpoons w_1 \rightleftharpoons w_2 \rightleftharpoons \cdots \rightleftharpoons w_n,$$

with no converging connections where $o \in O_v^k$ and $\{w_i\}_{i=1}^n \subseteq pa(v) \setminus O_v^k$. In this case, we cannot add the node w_i to O_v^k for any $i \in \{2, \ldots, n\}$ since it is connected to o by an active trail $o \rightleftharpoons w_1 \rightleftharpoons \cdots \rightleftharpoons w_i$ consisting of nodes in $V \setminus O_v^k$. Consequently, we must add node w_1 before adding node w_i. If w_1 is added to O_v^k, then the same argument applies to the trail $w_1 \rightleftharpoons w_2 \rightleftharpoons \cdots \rightleftharpoons w_n$, i.e. we must add w_2 next. The recursion is clear; any node w_i can only be added after $w_1, \ldots, w_{i-1}$ have been added. So, the algorithm "walks" over trails with no converging connections, adding them one node at a time, and it is only allowed to make "jumps" whenever a node is independent from the current partial order.

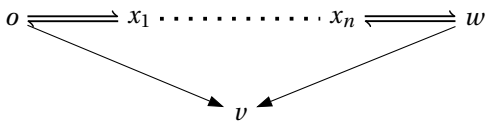
Lemma 5.15. Under Assumption 5.11, let $o \in O_v^k$ and $w \in Poss.Cand(O_v^k)$. If there exists a trail

$$o \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons w, \tag{5.12}$$

with no converging connection such that $\forall i = 1,\dots,n$, $x_i \in V \setminus O_v^k$, then w and o are adjacent.

Proof. We will employ an inductive argument, assuming that the lemma holds for all previous partial orders determined by the algorithm. By "previous partial orders" we mean all partial orders O_w^p with $w < v$ and $p \in \{1,\dots,|pa(w)|-1\}$, and O_v^p with $p \in \{1,\dots,k-1\}$.

Without loss of generality we can assume that the trail (5.12) is a shortest trail between o and w with no converging connection and satisfying $\forall i = 1,\dots,n$, $x_i \in V \setminus O_v^k$. Because o and w are parents of v by construction, $\mathcal{G}$ contains the subgraph below.



Because (5.12) has no converging connection, we know that $\not{d-sep}_{\mathcal{G}}\big(w, O_v^k \,\big|\, \emptyset\big)$. Thus, if $w \in Poss.Cand(O_v^k)$, then we must have $w \in Poss.Cand_{In}(O_v^k)$ or $w \in Poss.Cand_{Out}(O_v^k)$.

In the base case where $k = 1$ and $\big|O_v^k\big| = 1$, we know that $O_v^k = \{o\}$. Therefore we directly know that w and o are adjacent (because $w \in Poss.Cand_{In}(O_v^k)$ or $w \in Poss.Cand_{Out}(O_v^k)$, so w must be connected to some node in O_v^k, and this must be o).

We now prove the induction step. If $n = 0$, then w and o are adjacent, which concludes the proof. We now assume $n > 0$. For this, we consider both cases depending on whether $w \in Poss.Cand_{In}(O_v^k)$ or $w \in Poss.Cand_{Out}(O_v^k)$.

Case 1: $w \in Poss.Cand_{In}(O_v^k)$. By definition of $Poss.Cand_{In}(O_v^k)$ (Definition 4.4), there exists an $\tilde{o} \in O_v^k$ such that $w \to \tilde{o}$ satisfying the following restrictions:

1. $pa(\tilde{o} \downarrow w) \subseteq O_v^k$.
2. $d-sep_{\mathcal{G}}\big(w, O_v^k \setminus \underline{pa}(\tilde{o} \downarrow w) \,\big|\, \underline{pa}(\tilde{o} \downarrow w)\big)$.

First, note that if $\tilde{o} = o$, then o and w are adjacent, completing the proof. Thus, we assume that $\tilde{o} \neq o$.

To satisfy the second restriction above, any trail between w and $O_v^k \setminus \underline{pa}(\tilde{o} \downarrow w)$ must be blocked by $\underline{pa}(\tilde{o} \downarrow w)$. If we assume that $o \in O_v^k \setminus \underline{pa}(\tilde{o} \downarrow w)$, then the trail (5.12) must be blocked by $\underline{pa}(\tilde{o} \downarrow w)$. Since this trail (5.12) contains no converging connections, there must be an $x_k \in \underline{pa}(\tilde{o} \downarrow w)$ for some $k \in \{1,\dots,n\}$. The first restriction combined with the definition of $\tilde{o}$ implies that $\underline{pa}(\tilde{o} \downarrow w) := pa(\tilde{o} \downarrow w) \sqcup \{\tilde{o}\} \subseteq O_v^k$, and thus $x_k \in \underline{pa}(\tilde{o} \downarrow w) \subseteq O_v^k$ which contradicts the assumption of Lemma 5.15 that x_i belongs to $V \setminus O_v^k$ for every $i = 1,\dots,n$.

In the previous paragraph, we have proved that $o \notin O_v^k \setminus \underline{pa}(\tilde{o} \downarrow w)$. Since $o \in O_v^k$, this implies that $o \in \underline{pa}(\tilde{o} \downarrow w)$. Moreover, we assumed that $o \neq \tilde{o}$, and therefore $o \in \underline{pa}(\tilde{o} \downarrow w)$ which means that $o \to \tilde{o}$ and $o <_{\tilde{o}} w$.

We have $\tilde{o} \in ch(o)$ and $\tilde{o} \in ch(w)$. Therefore, by Lemma 5.7(i), $\tilde{o} \in ch(x_i)$ for all $i \in \{1,\dots,n\}$. Therefore, all x_i must belong to $pa(\tilde{o}) = pa(\tilde{o} \downarrow w) \sqcup \{w\} \sqcup pa(\tilde{o} \uparrow w)$. None of them is equal to w since (5.12) is a shortest trail.

By assumption, none of the x_i belong to O_v^k; the first restriction states that $pa(\tilde{o} \downarrow w) \subseteq O_v^k$; therefore all x_i belong to $pa(\tilde{o} \uparrow w)$. This means that $w <_{\tilde{o}} x_i$ for all $i = 1, \dots, n$.

Now, we have $o <_{\tilde{o}} w <_{\tilde{o}} x_i$ for all $i \in \{1, \dots, n\}$. This means that during the construction of $<_{\tilde{o}}$ in the algorithm we had $w \in Poss.Cand(O_{\tilde{o}})$ for a partial order $O_{\tilde{o}}$ which contains o but not $\{x_i\}_{i=1}^n$. Therefore, by the induction hypothesis we obtain that w and o are adjacent, which finishes the proof for this case.

Case 2: $w \in Poss.Cand_{Out}(O_v^k)$. By Definition of $Poss.Cand_{Out}(O_v^k)$ (Definition 4.4), there exists an $\tilde{o} \in O_v^k$ such that $\tilde{o} \to w$ satisfying the following conditions:

1. $pa(w \downarrow \tilde{o}) \subseteq O_v^k$.
2. $d\text{-}sep_{\mathcal{G}}\big(w, O_v^k \setminus \overline{pa}(w \downarrow \tilde{o}) \,\big|\, \overline{pa}(w \downarrow \tilde{o})\big)$.

If $\tilde{o} = o$ the proof is complete. We now assume $\tilde{o} \neq o$.

If $o \notin \overline{pa}(w \downarrow \tilde{o})$, then $o \in O_v^k \setminus \overline{pa}(w \downarrow \tilde{o})$ and therefore (5.12) is a trail from w to $O_v^k \setminus \overline{pa}(w \downarrow \tilde{o})$. By the second restriction above, this trail must be blocked by $\overline{pa}(w \downarrow \tilde{o})$. Because this trail has no converging connection there must be an $i \in \{1, \dots, n\}$ such that $x_i \in \overline{pa}(w \downarrow \tilde{o}) = pa(w \downarrow \tilde{o}) \sqcup \{\tilde{o}\} \subseteq O_v^k$ by the first restriction and the definition of $\tilde{o}$. This is a contradiction since by the assumption of the lemma x_i is in $V \setminus O_v^k$.

Therefore we have shown that $o \in \overline{pa}(w \downarrow \tilde{o})$, which implies (by definition of this set) that o and w are adjacent, proving the lemma. □

Lemma 5.15 immediately implies a very useful property of partial orders generated by our algorithm, which is proven in the corollary below. This corollary will play an important role in the proofs in Section 5.3.

Corollary 5.16. Under Assumption 5.11, let $o_i, o_j \in O_v^k$, such that o_i (respectively o_j) is the i-th node (respectively j-th node) in the partial order O_v^k and $i \neq j$. If there exists a trail

$$o_i \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons o_j, \tag{5.13}$$

with no converging connection such that $\forall m = 1, \dots, n$, $x_m \in V \setminus O_v^k$, then o_i and o_j are adjacent.

Proof. Without loss of generality, we can assume that $i < j$. This means that $o_j \in Poss.Cand(O_v^{j-1})$, with $o_i \in O_v^{j-1}$. Remark that $O_v^{j-1} \subseteq O_v^k$. Therefore, for all $m = 1, \dots, n$, $x_m \in V \setminus O_v^{j-1}$. Hence, by Lemma 5.15, o_i and o_j are adjacent. □

By Lemma 3.20 a parental order $<_v$ must abide to the B-sets in the sense of Definition 3.19 to prevent integration. All orders generated by our algorithm abide to the B-sets, as proven by the lemma below.

Lemma 5.17. All parental orders determined by the algorithm abide by the B-sets, in the sense of Definition 3.19.

Proof. When constructing the parental order for an arbitrary node v in V, the algorithm expands a partial order starting from the empty set. Each node added to the partial order must have been a possible candidate to a partial order O_v^k. By the definition of the set of possible candidates (Definition 4.4), a possible candidate of O_v^k must be in the smallest B-set $B(O_v^k)$ which is strictly larger than O_v^k (Definition 4.2). Since the construction of $<_v$ starts with $O_v^0 = \emptyset$, this means that we can never add a node in B_q before we have added all nodes from B_{q-1}.

Indeed, the smallest B-set which is strictly larger than the empty set is B_1. Hence, to start, we must add a node from B_1 to O_v^0. We will then continue adding nodes from B_1 until we reach a partial order O_v^k which is not strictly smaller than B_1. In this case we have exhausted the set B_1, i.e. $O_v^k = B_1$. This means that $B(O_v^k) = B_2$ (if it exists), and we add nodes from this set until it is exhausted as well.

Therefore, the proof is completed by a straightforward recursion, showing that $<_v$ abides by the B-sets. □

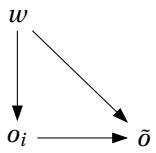
We now prove a lemma which states that sets which are d-separated cannot be adjacent. It is quite trivial but it will be useful in Lemma 5.19.

Lemma 5.18. Let $\mathcal{G} = (V, E)$ be a DAG and X, Y, Z subsets of V such that $d - sep_{\mathcal{G}}(X, Y \,|\, Z)$. Then X and Y cannot be adjacent, in the sense that $\forall x, y \in X \times Y$, $x \not\rightleftharpoons y$.

Proof. Let $x \in X$ and $y \in V$ such that $x \rightleftharpoons y$. y is adjacent to $x \in X$ so the trail $x \rightleftharpoons y$ is active given Z. This shows that $\cancel{d - sep}_{\mathcal{G}}(X, \{y\} \,|\, Z)$. Hence, y cannot be in Y. □

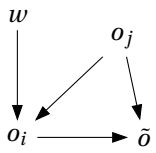
The lemma below states that under certain conditions an arc between a node $w \in B(O_v^k) \setminus O_v^k$ and a node $o_i \in O_v^k$ implies the existence of another node $\tilde{o} \in O_v^k$ such that $w \to \tilde{o} \in E$ and $o_i \to \tilde{o} \in E$.

Lemma 5.19. Following Definition 4.1, let us write the partial order O_v^k as $O_v^k = (o_1, \ldots, o_k)$. Under Assumption 5.11, let $i \in \{1, \ldots, k\}$ and $w \in B(O_v^k) \setminus O_v^k$. If $w \to o_i$ and $pa(o_i \uparrow w) \cap O_v^k \neq \emptyset$, then there exists an $\tilde{o} \in O_v^k$ such that $\tilde{o} \neq o_i$ and $\mathcal{G}$ contains the subgraph below.



Proof. Note that $w \in pa(o_i) = pa(o_i \downarrow w) \sqcup \{w\} \sqcup pa(o_i \uparrow w)$. Since $pa(o_i \uparrow w) \cap O_v^k \neq \emptyset$, let o_j be its maximum element according to $<_{o_i}$. Consequently, $o_j \to o_i$ and $w <_{o_i} o_j$.

Assume that there exists an $\tilde{o} \in O_v^k$ such that $\mathcal{G}$ contains the v-structure $o_i \to \tilde{o} \leftarrow o_j$. Thus, $\mathcal{G}$ contains the subgraph below.



Observe that $o_j \in pa(o_i)$ and $o_j \in pa(\tilde{o})$; by definition of the B-sets, $o_j \in B(o_i, \tilde{o}) = pa(o_i) \cap pa(\tilde{o})$. Because the parental order $<_{o_i}$ abides (see Definition 3.19) by the B-sets by Lemma 5.17, we can combine the facts that $w <_{o_i} o_j$ and $o_j \in B(o_i, \tilde{o})$ to obtain $w \in B(o_i, \tilde{o})$. Therefore $w \to \tilde{o} \in E$, giving us the desired subgraph and finishing the proof under the assumption of existence of $\tilde{o}$.

There remains to prove the existence of such an $\tilde{o}$. We consider two cases; when $i < j$ and $j < i$.

Case 1: $i < j$. In this case, since $o_j \to v$, o_j was added at the step $j - 1$, and by the induction hypothesis (Assumption 5.11), we must have $o_j \in Poss.Cand(O_v^{j-1})$. Because $i < j$, we obtain $o_i \in O_v^{j-1}$. Therefore,

$\not{d-sep}_{\mathcal{G}}\left(o_j, O_v^{j-1} \middle| \emptyset\right)$ because of the arc $o_j \to o_i$. Hence, $o_j \notin Poss.Cand_{Ind}(O_v^{j-1})$, and therefore o_j must be in $Poss.Cand_{In}(O_v^{j-1})$ or $Poss.Cand_{Out}(O_v^{j-1})$. We consider both cases.

- $o_j \in Poss.Cand_{In}(O_v^{j-1})$: By Definition 4.4 there must be a node $\tilde{o} \in O_v^{j-1}$ such that $o_j \to \tilde{o}$ satisfying following restrictions:

  1. $pa(\tilde{o} \downarrow o_j) \subseteq O_v^{j-1}$.
  2. $d-sep_{\mathcal{G}}\left(o_j, O_v^{j-1} \setminus \underline{pa}(\tilde{o} \downarrow o_j) \middle| \underline{pa}(\tilde{o} \downarrow o_j)\right)$.

  Remark that $w \in pa(o_i \downarrow o_j)$ and $w \notin O_v^k \supseteq O_v^{j-1}$, and thus $pa(o_i \downarrow o_j) \not\subseteq O_v^{j-1}$. This shows that $\tilde{o} \neq o_i$ otherwise the first restriction could not be satisfied.

  By combining Lemma 5.18 and the second restriction, no point in $O_v^{j-1} \setminus \underline{pa}(\tilde{o} \downarrow o_j)$ can be adjacent to o_j. Because we have the arc $o_j \to o_i$ we can deduce that $o_i \notin O_v^{j-1} \setminus \underline{pa}(\tilde{o} \downarrow o_j)$.

  Since $o_i \in O_v^{j-1}$, this means that $o_i \in \underline{pa}(\tilde{o} \downarrow o_j)$, and therefore $o_i \to \tilde{o}$. Now, we have $o_j \to \tilde{o}$ and $o_i \to \tilde{o}$ which is the desired v-structure.

- $o_j \in Poss.Cand_{Out}(O_v^{j-1})$: By Definition 4.4 there must be a node $\tilde{o} \in O_v^{j-1}$ such that $\tilde{o} \to o_j$ and the following restrictions are satisfied:

  1. $pa(o_j \downarrow \tilde{o}) \subseteq O_v^{j-1}$.
  2. $d-sep_{\mathcal{G}}\left(o_j, O_v^{j-1} \setminus \overline{pa}(o_j \downarrow \tilde{o}) \middle| \overline{pa}(o_j \downarrow \tilde{o})\right)$.

  If $\tilde{o} = o_i$, then $\mathcal{G}$ contains the cycle $o_i \to o_j \to o_i$, which is a contradiction, and thus $\tilde{o} \neq o_i$.

  By combining Lemma 5.18 and the second restriction, no point in $O_v^{j-1} \setminus \overline{pa}(o_j \downarrow \tilde{o})$ can be adjacent to o_j. Because we have the arc $o_j \to o_i$ we can deduce that $o_i \notin O_v^{j-1} \setminus \overline{pa}(o_j \downarrow \tilde{o})$.

  Since $o_i \in O_v^{j-1}$, this means that $o_i \in \overline{pa}(o_j \downarrow \tilde{o})$, and therefore $o_i \to o_j$. This provides the cycle $o_i \to o_j \to o_i$. This is a contradiction, showing that this case cannot happen: o_j cannot be in $Poss.Cand_{Out}(O_v^{j-1})$.

Case 2: $j < i$. In this case, since $o_i \to v$, o_i was added at the step $i-1$, and by the induction hypothesis (Assumption 5.11), we must have $o_i \in Poss.Cand(O_v^{i-1})$. Because $j < i$, we obtain $o_j \in O_v^{i-1}$. Therefore, $\not{d-sep}_{\mathcal{G}}\left(o_i, O_v^{i-1} \middle| \emptyset\right)$ because of the arc $o_j \to o_i$. Hence, $o_i \notin Poss.Cand_{Ind}(O_v^{i-1})$. Thus, o_i must be in $Poss.Cand_{In}(O_v^{i-1})$ or $Poss.Cand_{Out}(O_v^{i-1})$. We consider both cases.

- $o_i \in Poss.Cand_{In}(O_v^{i-1})$: By Definition 4.4 there must be a node $\tilde{o} \in O_v^{i-1}$ such that $o_i \to \tilde{o}$ satisfying:

  1. $pa(\tilde{o} \downarrow o_i) \subseteq O_v^{i-1}$.
  2. $d-sep_{\mathcal{G}}\left(o_i, O_v^{i-1} \setminus \underline{pa}(\tilde{o} \downarrow o_i) \middle| \underline{pa}(\tilde{o} \downarrow o_i)\right)$.

  Note that $\tilde{o} = o_j$ provides the cycle $o_j \to o_i \to o_j$ which is a contradiction, and thus $\tilde{o} \neq o_j$.

  As before, combining Lemma 5.18, the second restriction, and the arc $o_j \to o_i$ implies that $o_j \notin O_v^{i-1} \setminus \underline{pa}(\tilde{o} \downarrow o_i)$, and therefore $o_j \in \underline{pa}(\tilde{o} \downarrow o_i)$ which means that $o_j \to \tilde{o}$, giving us the desired v-structure.

- $o_i \in Poss.Cand_{Out}(O_v^{i-1})$: By Definition 4.4 there must be a node $\tilde{o} \in O_v^{i-1}$ such that $\tilde{o} \to o_i$ satisfying:

  1. $pa(o_i \downarrow \tilde{o}) \subseteq O_v^{i-1}$.

2. $d-sep_{\mathcal{G}}\big(o_i, O_v^{i-1} \setminus \overline{pa}(o_i \downarrow \tilde{o}) \,\big|\, \overline{pa}(o_i \downarrow \tilde{o})\big)$.

Remark that $w \in pa(o_i \downarrow o_j)$ and $w \notin O_v^k \supseteq O_v^{j-1}$, and thus $pa(o_i \downarrow o_j) \not\subseteq O_v^{j-1}$. This shows that $\tilde{o} \neq o_j$ otherwise the first restriction could not be satisfied.

Combining Lemma 5.18, the second restriction, and the existence of the arc $o_j \to o_i$ implies that $o_j \notin O_v^{i-1} \setminus \overline{pa}(o_i \downarrow \tilde{o})$. Since $o_j \in O_v^{i-1}$ this means that $o_j \in \overline{pa}(o_i \downarrow \tilde{o})$, and thus $o_j <_{o_i} \tilde{o}$. Therefore, $\tilde{o} \in pa(o_i \uparrow w)$ since $w <_{o_i} o_j$. However, this is a contradiction since $o_j \neq \tilde{o}$ was chosen to be the maximum element in $pa(o_i \uparrow w) \cap O_v^k$. This shows that it cannot happen that o_i is in $Poss.Cand_{Out}(O_v^{i-1})$.

□

5.3. Order and properties of trails that may have converging nodes

We start this section with introducing notation of general trails Hereafter, we prove Lemma 5.26. This lemma will be especially useful in Section 5.4 where we establish that under certain conditions there must be a trail between a node $w \in B(O_v^k) \setminus O_v^k$ and a subset of O_v^k without converging connections, see Lemmas 5.28 and 5.29. These results play a crucial part in the proof in Section 4.2.3. It should be noted that Lemmas 5.28 and 5.29 convey the same intuition despite a different formulation. Indeed, they both state that there must be a trail with no converging connections, however under different conditions. It will be beneficial to first establish a general framework where we relax the conditions and consider a broader class of trails.

Throughout this section, we will denote by X, Y and Z three disjoint subsets of V. First, we define the set of all trails from X to Y activated by Z in $\mathcal{G}$.

Definition 5.20. Let $X, Y, Z \subseteq V$ be disjoint subsets of V. We define $TRAILS(X, Y \,|\, Z)$ to be the set of trails from X to Y activated by Z.

Later we show that for a certain X, Y and Z, and some extra conditions, the set $TRAILS(X, Y \,|\, Z)$ contains at least one trail with no converging connections. It will be convenient to define the set of all converging connections on a trail T in $TRAILS(X, Y \,|\, Z)$.

Definition 5.21. For a trail $T \in TRAILS(X, Y \,|\, Z)$, we define $ConvCon(T) := (c_1, \dots, c_C)$ to be the ordered set of nodes corresponding to converging connections in T, ordered by first appearance on the trail from X to Y. Here, we mean that T is of the form

$$x \rightleftharpoons \cdots \to c_1 \leftarrow \cdots \to c_2 \leftarrow \cdots\cdots\cdots \to c_C \leftarrow \cdots \rightleftharpoons y$$

with $x \in X$ and $y \in Y$. The cardinality of the set $ConvCon(T)$ is denoted by $C := C(T) = \big|ConvCon(T)\big|$.

For a trail T to be activated by Z, we must have that for all $i = 1, \dots, C$, c_i is in Z, or one of its descendants is in Z, see Definition 2.22. Later we will need to know if a node c_i is in Z or not. If it is not in Z, then we also require the node in Z that is its **closest descendant**.

Definition 5.22 (Closest descendant)**.** Let T be a trail in $TRAILS(X, Y \,|\, Z)$ and $i \in \{1, \dots, C(T)\}$. If $c_i \notin Z$, then its ***closest descendant*** in Z is a node $Z(c_i) \in Z$ such that there exist a shortest path

$$c_i \to d_1^i \to \cdots \to d_{n_Z(i)}^i \to Z(c_i)$$

with $d_j^i \notin Z$ for all $j = 1, \dots, n_Z(i)$.

Such a path is referred to as a ***descendant path*** of c_i. Its nodes are denoted by the symbol "d" where a superscript i indicates that d_j^i lies on the descendant path of c_i, and the subscript j indicates that it is the j-th node on this path. The length of the descendant path is formally denoted by $n_Z(i)$, but we will often simply write $n := n_Z(i)$. If $c_i \in Z$, we also say that $c_i = Z(c_i)$. Finally, we use the conventions $d_0^i := c_i$ and $d_{n+1}^i := Z(c_i)$.

A trail T in $TRAILS(X, Y \,|\, Z)$ can be seen as a concatenation of trails activated by the empty set. For instance, consider the trail

$$x \rightleftharpoons \cdots \to c_1 \leftarrow \cdots \to c_2 \leftarrow \cdots\cdots\cdots \to c_C \leftarrow \cdots \rightleftharpoons y,$$

then each trail $c_i \leftarrow \cdots \to c_{i+1}$ is a trail between two nodes activated by the empty set. Such trails are referred to as **subtrails**.

Definition 5.23 (Subtrails)**.** Let T be a trail in $TRAILS(X, Y \,|\, Z)$. Suppose that T takes the form

$$x \rightleftharpoons t_1^0 \rightleftharpoons \cdots \rightleftharpoons t_{n_t(0)}^0 \to c_1 \leftarrow t_1^1 \rightleftharpoons \cdots \rightleftharpoons t_{n_t(1)}^1 \to c_2 \leftarrow \cdots\cdots\cdots \to c_C \leftarrow t_1^C \rightleftharpoons \cdots \rightleftharpoons t_{n_t(C)}^C \rightleftharpoons y.$$

The following are referred to as the ***subtrails*** of T:

$$\begin{aligned} &x \rightleftharpoons t_1^0 \rightleftharpoons \cdots \rightleftharpoons t_{n_t(0)}^0 \to c_1, \\ &c_i \leftarrow t_1^i \rightleftharpoons \cdots \rightleftharpoons t_{n_t(i)}^i \to c_{i+1}, \text{ with } i \in \{1,\dots,C-1\}, \\ &c_C \leftarrow t_1^C \rightleftharpoons \cdots \rightleftharpoons t_{n_t(C)}^C \rightleftharpoons y. \end{aligned}$$

The nodes on the subtrails are denoted by the symbol "t" where a superscript i indicates that t_j^i lies in between c_i and c_{i+1} with the conventions $c_0 := x$ and $c_{C+1} := y$. The subscript indicates its location on the subtrail. The length of a subtrail is formally denoted by $n_t(i)$, but we will often simply write $n := n_t(i)$.

Furthermore, we use the conventions $c_0 := x$, $c_{C+1} := y$, $t_0^i := c_i$ and $t_{n+1}^i := c_{i+1}$. Also, we denote the common ancestor among a trail between c_i and c_{i+1} by t_m^i, see Definition 5.2.

In many previous lemmas we picked shortest trails, since this allowed us to exclude the existence of certain arcs. Now, we will need to pick our trails under even stronger assumptions. Instead of picking a shorter trail we will be picking a **better** trail.

Definition 5.24 (Better trail)**.** Let T_1 and T_2 belong to $TRAILS(X, Y \,|\, Z)$. We say that T_1 is a ***better*** trail than T_2, denoted by $T_1 <_{TRAIL} T_2$, if one of the following conditions is satisfied:

1. $\big|ConvCon(T_1) \setminus Z\big| < \big|ConvCon(T_2) \setminus Z\big|$.
2. 1) is an equality and $\big|ConvCon(T_1)\big| := C(T_1) < C(T_2) =: \big|ConvCon(T_2)\big|$.
3. 1) and 2) are equalities and $\sum_{i=1}^{C(T_1)} n_Z(i)(T_1) < \sum_{i=1}^{C(T_2)} n_Z(i)(T_2)$.
4. 1), 2) and 3) are equalities and $\sum_{i=0}^{C(T_1)} n_t(i)(T_1) < \sum_{i=0}^{C(T_2)} n_t(i)(T_2)$.

We remark that the order $<_{TRAIL}$ on the set $TRAILS(X, Y \,|\, Z)$ can be seen as induced by the alphabetical order on the vector $\Big(\big|ConvCon(T) \setminus Z\big|, \big|ConvCon(T)\big|, \sum_{i=1}^{C(T_1)} n_Z(i)(T), \sum_{0=1}^{C(T_1)} n_t(i)(T)\Big)$, for $T \in TRAILS(X, Y \,|\, Z)$. Indeed, we first order trails by number of converging connections not in Z, then by number of converging connections, then by total length of descendant paths and finally by total length of the subtrails. This means that a better trail satisfies the following conditions.

C1. It is a trail from X to Y activated by Z.

C2. It contains a smaller number of converging nodes not contained in Z.

C3. Under the restrictions above, it contains less converging connections.

C4. Under the restrictions above, the paths from converging nodes not contained in Z to its closest descendants are shorter.

C5. Under the restrictions above, it is a shorter such trail.

In Lemma 5.26, we will assume that the subset $Y \sqcup Z$ has a certain property $\mathfrak{A}$ defined below. We remark that in the proofs of Lemmas 5.28 and 5.29 the set $Y \sqcup Z$ will be equal to O_v^k, and the set O_v^k satisfies $\mathfrak{A}$ as a direct consequence of Corollary 5.16.

Definition 5.25. Let $\mathcal{G}$ be a DAG and K a subset of V. We say that K has the property $\mathfrak{A}$ if $\forall\, v_1, v_2 \in K$ such that there exists a trail

$$v_1 \rightleftharpoons x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightleftharpoons v_2$$

with $x_i \notin K$ for all $i = 1,\dots,n$ and no converging connection, then v_1 and v_2 are adjacent.

We will now prove certain properties of a minimal trail in $TRAILS(X, Y \mid Z)$ according to $<_{TRAIL}$.

Lemma 5.26. Let $X, Y, Z \subseteq V$ be three disjoint subsets. Assume that $TRAILS(X, Y \mid Z) \neq \emptyset$. Let

$$x \rightleftharpoons \cdots \to c_1 \leftarrow \cdots \to c_2 \leftarrow \cdots \to c_C \leftarrow \cdots \rightleftharpoons y \tag{5.14}$$

be a minimal element of $TRAILS(X, Y \mid Z)$ for the order $<_{TRAIL}$.

Then, the following properties hold.

(i) For all i, j, $t^i_j \notin Y \sqcup Z$ and $d^i_j \notin Y \sqcup Z$.

(ii) For all $i = 1,\dots,C$, the trails $c_i \to d^i_1 \to \cdots \to d^i_n \to Z(c_i)$ and $t^i_1 \rightleftharpoons \cdots \rightleftharpoons t^i_n$ do not contain a chord. Furthermore, the trails $x \rightleftharpoons t^0_1 \rightleftharpoons \cdots \rightleftharpoons t^0_n$ and $t^C_1 \rightleftharpoons \cdots \rightleftharpoons t^C_n \rightleftharpoons y$ do not contain a chord.

(iii) If $c_i \to c_{i+1}$ and $c_{i+1} \in Z$, then $c_i \in Z$.

(iv) If $c_i \leftarrow c_{i+1}$ and $c_i \in Z$, then $c_{i+1} \in Z$.

(v) For all $i = 1,\dots,C-1$, the i-th subtrail is a shortest trail between c_i and c_{i+1} starting with a leftward pointing arrow, ending with rightward pointing arrow, consisting of nodes in $V \setminus Z$ and with no converging connection. The C-th subtrail is a shortest trail between c_C and y starting with a leftward pointing arrow, consisting of nodes in $V \setminus Z$ and with no converging connection.

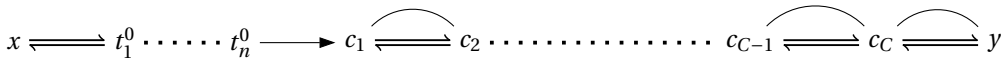
Furthermore, assume that $Y \sqcup Z$ has the property $\mathfrak{A}$. Then, the following hold.

(vi) The final converging node c_C is in Z.

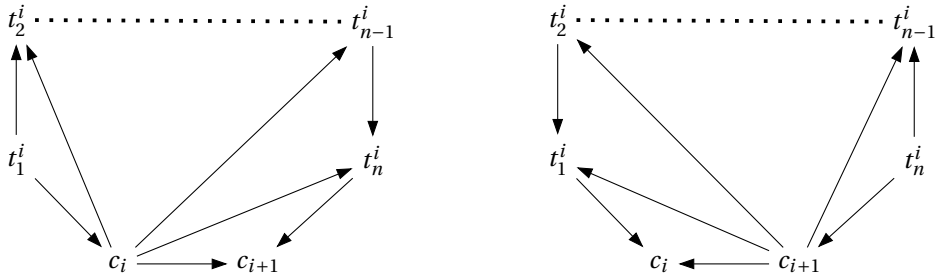
(vii) For all $i = 1,\dots,C-1$, we have $c_i \in Z$ or $c_{i+1} \in Z$.

(viii) For all $i = 1,\dots,C$, the nodes c_i and c_{i+1} are adjacent.

(ix) If this trail contains a total of $C > 0$ converging connections, then $\mathcal{G}$ contains the subgraph below.



Here, the curved lines represent one of the following two subgraphs.



(x) For all $i = 2,\dots,C$, the trail $c_{i-1} \leftarrow c_i \to c_{i+1}$ can not be present in $\mathcal{G}$.

(xi) If $c_{C-1} \to c_C$, then for all $i = 1,\dots,C$, c_i is in Z.

(xii) If $c_1 \leftarrow c_2$ and $c_1 \in Z$, then for all $i = 1, \dots, C$, c_i is in Z.

Proof. **(i):** We want to show that: for all i, j, $t_j^i \notin Y \sqcup Z$ and $d_j^i \notin Y \sqcup Z$.

Assume that there exist i, j such that $t_j^i \in Y \sqcup Z$. Remark first that $t_j^i \notin Z$, otherwise trail (5.14) would be blocked by Z, since t_j^i does not correspond to a converging connection, see Definition 2.22. Hence, t_j^i must be in Y. Now, $x \rightleftharpoons \cdots \rightleftharpoons t_j^i$ is a trail from x to Y activated by Z that is better than (5.14), see Definition 5.24. This is a contradiction with the definition of the minimal trail. We have shown that $t_j^i \notin Y \sqcup Z$.

Now, suppose that there exist i, j such that $d_j^i \in Y \sqcup Z$. By Definition 5.22, this node cannot be in Z. Therefore, d_j^i must be in Y. If this is the case, then the trail

$$x \rightleftharpoons \cdots \rightarrow c_i \rightarrow d_1^i \rightarrow \cdots \rightarrow d_j^i$$

would be a better trail in $TRAILS(X, Y \,|\, Z)$ than (5.14). Indeed, it contains at least one less converging node in Z, since the node c_i now corresponds to a diverging connection. This is a contradiction with the definition of (5.14) which is a minimal trail.

This concluded the proof of (i).

(ii): We want to show that: for all $i = 1, \dots, C$, the trails $c_i \rightarrow d_1^i \rightarrow \cdots \rightarrow d_n^i \rightarrow Z(c_i)$ and $t_1^i \rightleftharpoons \cdots \rightleftharpoons t_n^i$ do not contain a chord. Furthermore, the trails $x \rightleftharpoons t_1^0 \rightleftharpoons \cdots \rightleftharpoons t_n^0$ and $t_1^C \rightleftharpoons \cdots \rightleftharpoons t_n^C \rightleftharpoons y$ do not contain a chord.

First, we consider a descendant path between c_i and $Z(c_i)$ with $i \in \{1, \dots, C\}$. By Definition 5.22, this path is a shortest trail of the form

$$c_i \rightarrow d_1 \rightarrow \cdots \rightarrow d_n \rightarrow Z(c_i)$$

consisting of nodes in $V \setminus Z$. By combining Lemmas 5.5 and 5.6 we know that this descendant path does not contain a chord.

Now, let $i \in \{1, \dots, C\}$ and consider the subtrail

$$c_i \leftarrow t_1 \rightleftharpoons \cdots \rightleftharpoons t_n \rightarrow c_{i+1}.$$

Observe that the trail $t_1 \rightleftharpoons \cdots \rightleftharpoons t_n$ is a shortest trail between t_1 and t_n with no converging connections consisting of nodes in $V \setminus (Z \sqcup Y)$. Indeed, if there would be a shorter such trail T^* between t_1 and t_n, then replacing $t_1 \rightleftharpoons \cdots \rightleftharpoons t_n$ in (5.14) by T^* would result in a better trail than (5.14), and therefore a contradiction. Now, we can apply Lemmas 5.5 and 5.6 to find that $t_1 \rightleftharpoons \cdots \rightleftharpoons t_n$ cannot contain a chord.

Consider the subtrail

$$x \rightleftharpoons t_1^0 \rightleftharpoons \cdots \rightleftharpoons t_n^0 \rightarrow c_1.$$

By similar argument as above, the trail $x \rightleftharpoons t_1^0 \rightleftharpoons \cdots \rightleftharpoons t_n^0$ is a shortest trail activated by the empty set consisting of nodes in $V \setminus (Y \sqcup Z)$. Thus, we can apply Lemmas 5.5 and 5.6 to find that $x \rightleftharpoons t_1^0 \rightleftharpoons \cdots \rightleftharpoons t_n^0$ contains no chords.

The last trail does not contain a chord by symmetry: switch the role of X and Y and apply this result. This concludes the poof of (ii).

(iii): We want to prove that, if $c_i \rightarrow c_{i+1}$ and $c_{i+1} \in Z$, then $c_i \in Z$.

Consider the case when $c_i \notin Z$, then the trail $x \rightleftharpoons \cdots \rightarrow c_i \rightarrow c_{i+1} \leftarrow \cdots \rightleftharpoons y$ would be a better trail than (5.14) as it contains one less converging connection. Because this is a contradiction, we must have $c_i \in Z$.

(iv): This is a direct consequence of (iii) obtained by switching the roles of X and Y.

(v): We must prove that for all $i = 1,\dots,C$, the trail $c_i \leftarrow t_1^i \rightleftharpoons t_2^i \rightleftharpoons \cdots \rightleftharpoons t_{n-1}^i \rightleftharpoons t_n^i \rightleftharpoons c_{i+1}$ is a shortest such trail.

This follows directly form the definition of (5.14). If there would be a shorter trail T^* between c_i and c_{i+1}, then replacing the corresponding subtrail in (5.14) by T^* would result in a better trail, and therefore a contradiction.

(vi): We want to show that the final converging node c_C is in Z.

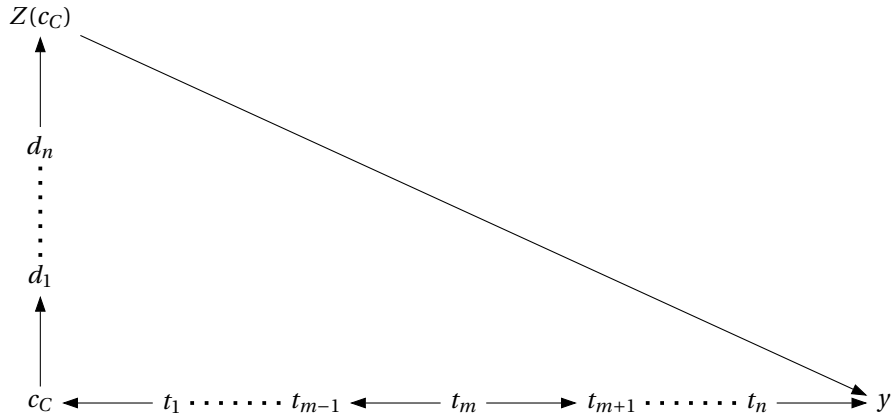
Consider the case when $c_C \neq Z(c_C)$. Then, $\mathcal{G}$ contains the trail

$$Z(c_C) \leftarrow d_n \leftarrow \cdots \leftarrow d_1 \leftarrow c_C \leftarrow t_1 \rightleftharpoons \cdots \rightleftharpoons t_n \rightleftharpoons y.$$

This is a trail between two nodes in $Y \sqcup Z$ consisting of nodes not in $Y \sqcup Z$ by (i). Since, the trail does not contain any converging connections and $Y \sqcup Z$ satisfies property $\mathfrak{A}$ we find that $Z(c_C)$ and y must be adjacent. Assume that the arc $Z(c_c) \leftarrow y$ is present. Consider the trail

$$x \rightleftharpoons \cdots\cdots\cdots \rightarrow c_C \rightarrow d_1 \rightarrow \cdots \rightarrow d_n \rightarrow Z(c_C) \leftarrow y.$$

In this trail, c_C is now not a converging node; but $Z(c_C)$ is a converging node. Therefore, it has the same amount of converging connections C, but one less converging node corresponding to a node not in Z than (5.14). This is because $c_C \notin Z$ while $Z(c_C) \in Z$. So, the trail above is better than (5.14). Since (5.14) is a minimal trail, this is a contradiction, and therefore E must contain the arc $Z(c_C) \rightarrow y$, giving us the subgraph below.

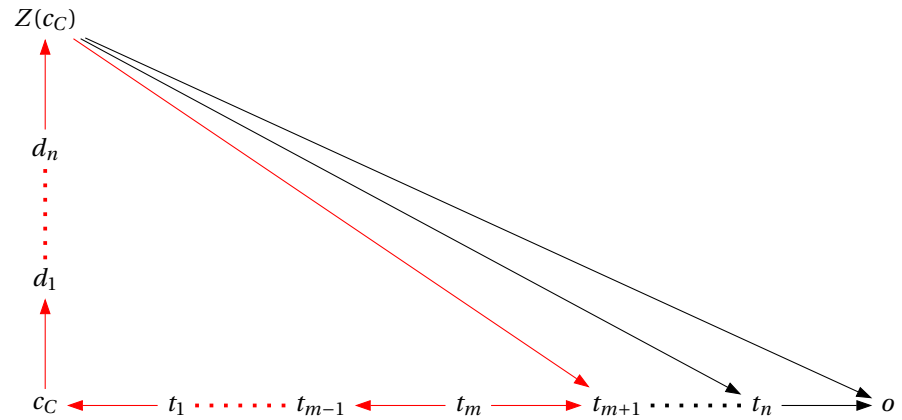


The undirected cycle above has one converging connection (at y); therefore it is an active cycle, unless E contains the appropriate chords. Several chords can be excluded:

- The trails $c_C \rightarrow d_1 \rightarrow \cdots \rightarrow d_n \rightarrow Z(c_C)$ and $t_1 \rightleftharpoons \cdots \rightleftharpoons t_n \rightarrow y$ do not contain any chords by (ii).
- $\forall\, j = 0,\dots,n+1$, $\forall\, l = 1,\dots,m$, $d_j \rightarrow t_l$ results in a cycle.
- $\forall\, j = 0,\dots,n$, $\forall\, l = m+1,\dots,n+1$, $d_j \rightarrow t_l$ results in a trail with less converging connections.
- $\forall\, j = 1,\dots,n$, $\forall\, l = 1,\dots,n+1$, $t_l \rightarrow d_j$ results in a trail with shorter descendant paths.
- $\forall\, l = 2,\dots,n+1$, $t_l \rightarrow c_C$ results in a shorter trail.

- $\forall l = 1,\dots,n$, $t_l \to Z(c_C)$ results in a trail with less converging nodes not in Z.

Therefore, the only allowed chords are arcs from the node $Z(c_C)$ to nodes in $\{t_j\}_{j=m+1,\dots,n}$. The lack of any of them would result in an active cycle; therefore they all have to be present, giving us the subgraph below.



The undirected cycle displayed in red is an active cycle, unless it is of length smaller than 4. It consists of the nodes c_C, $Z(c_C)$, $d_1,\dots,d_n$ and $t_1,\dots,t_{m+1}$ and is therefore of length $2+n+m+1 = n+m+3$. This means that $n+m+3 \leq 3$, and therefore $n+m=0$. However, this means that $t_m = t_0 := c_C$, and therefore $c_C \to t_1$. This is a contradiction with the definition of c_C since it is a converging node in (5.14).

Because the assumption that $c_C \notin Z$ provided a contradiction, we now know that the node c_C must be in Z, completing the proof of (vi).

(vii): We want to prove that: for all $i \in \{1,\dots,C-1\}$, we have $c_i \in Z$ or $c_{i+1} \in Z$.

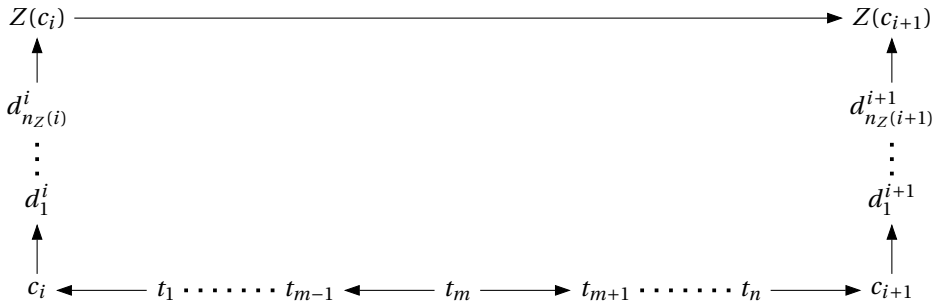
Assume that there exists an $i \in \{1,\dots,C-1\}$ such that $c_i, c_{i+1} \notin Z$. Then, we have the descendant paths

$$c_i \to d_1^i \to \cdots \to d_n^i \to Z(c_i) \text{ and } c_{i+1} \to d_1^{i+1} \to \cdots \to d_n^{i+1} \to Z(c_{i+1}).$$

Therefore, $Z(c_i)$ and $Z(c_{i+1})$ are two nodes in $Y \sqcup Z$ joined by a trail

$$Z(c_i) \leftarrow \cdots \leftarrow c_i \leftarrow \cdots \to c_{i+1} \to \cdots \to Z(c_{i+1})$$

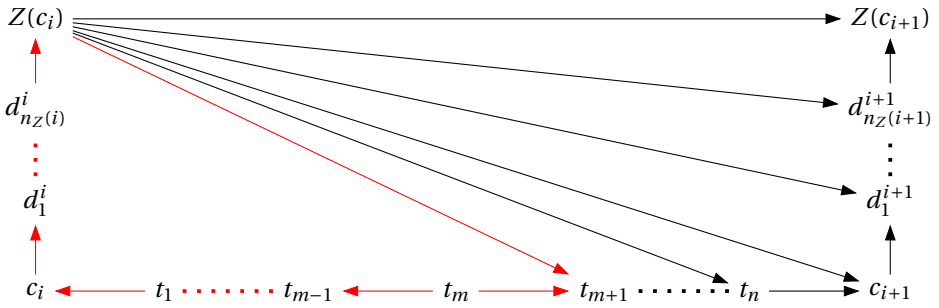
which is activated by the empty set and contains no nodes in $Y \sqcup Z$ (by (i)). Because $Y \sqcup Z$ has the property $\mathfrak{A}$, the nodes $Z(c_i)$ and $Z(c_{i+1})$ must be adjacent. Without loss of generality, we can assume that $Z(c_i) \to Z(c_{i+1})$, otherwise we switch the roles of X and Y. Remark that $\mathcal{G}$ contains the subgraph below.



The undirected cycle above has one converging connection (at $Z(c_{i+1})$; therefore it is an active cycle, unless E contains the appropriate chords. Several chords can be excluded:

- The trails $c_i \to d_1^i \to \cdots \to d_n^i \to Z(c_i)$, $t_1 \rightleftharpoons \cdots \rightleftharpoons t_n$ and $c_{i+1} \to d_1^{i+1} \to \cdots \to d_n^{i+1} \to Z(c_{i+1})$ do not contain no chords by (ii).
- $\forall j = 0,\dots,n_Z(i)+1$, $\forall l = 1,\dots,m$, $d_j^i \to t_l$ results in a cycle.
- $\forall j = 0,\dots,n_Z(i)$, $\forall l = m+1,\dots,n+1$, $d_j^i \to t_l$ results in a trail with less converging connections.
- $\forall j = 0,\dots,n_Z(i)$, $\forall l = 0,\dots,n_Z(i+1)+1$, $d_j^i \to d_l^{i+1}$ results in a trail with less converging connections.
- $\forall j = 0,\dots,n_Z(i+1)+1$, $\forall l = m,\dots,n$, $d_j^{i+1} \to t_l$ results in a cycle.
- $\forall j = 0,\dots,n_Z(i+1)$, $\forall l = 0,\dots,m-1$, $d_j^{i+1} \to t_l$ results in a trail with less converging connections.
- $\forall j = 0,\dots,n_Z(i+1)$, $\forall l = 0,\dots,n_Z(i)+1$, $d_j^{i+1} \to d_l^i$ results in a trail with less converging connections.
- $\forall j = 1,\dots,n_Z(i)$, $\forall l = 1,\dots,n$, $t_l \to d_j^i$ results in a trail with shorter descendant paths (because d_j^i becomes the new converging connection instead of c_i).
- $\forall j = 1,\dots,n_Z(i+1)$, $\forall l = 1,\dots,n$, $t_l \to d_j^{i+1}$ results in a trail with shorter descendant paths (because d_j^{i+1} becomes the new converging connection instead of c_{i+1}).
- $\forall l = 1,\dots,n$, $t_l \to Z(c_i)$ results in a trail with less converging nodes not in Z.
- $\forall l = 1,\dots,n$, $t_l \to Z(c_{i+1})$ results in a trail with less converging nodes not in Z.
- $\forall l = 1,\dots,n$, $t_l \to c_i$ and $t_l \to c_{i+1}$ result in a shorter trail (the arcs $t_1 \to c_1$ and $t_n \to c_{i+1}$ are not chords).
- $\forall l = 0,\dots,n+1$, $Z(c_{i+1}) \to t_l$ results in a cycle.
- $\forall j = 0,\dots,n_Z(i)+1$, $Z(c_{i+1}) \to d_j^i$ results in a cycle.

Therefore, the only allowed chords are of the form $Z(c_i) \to t_l$ with $l \in \{m+1,\dots,n+1\}$ and $Z(c_i) \to d_j^{i+1}$ with $j \in \{0,\dots,n_Z(i+1)\}$. It is evident that all these arcs must be present to prevent an active cycle from occurring, giving us the subgraph below.



This graph contains an undirected cycle with one converging connection (at t_{m+1}), coloured in red. There are no more chords which could be present. Therefore, this undirected cycle must be of length smaller than 4. The undirected cycle is made up of the nodes c_i, $Z(c_i)$, $d_1^i,\dots,d_{n_Z(i)}^i$ and $t_1,\dots,t_{m+1}$; it is of length $2+n_Z(i)+m+1 = n_Z(i)+m+3$. This means that $n_Z(i)+m+3 \leq 3$, and therefore $n_Z(i) = m = 0$.

Thus, $t_m = t_0 := c_i$ must be the first diverging node on the subtrail between c_i and c_{i+1}. However, this means that $c_i \to t_1$. This is a contradiction with the definition of c_i as it corresponds to a converging connection in

(5.14).

(viii): We want to show that: for all $i = 1,\dots,C$, the nodes c_i and c_{i+1} are adjacent.

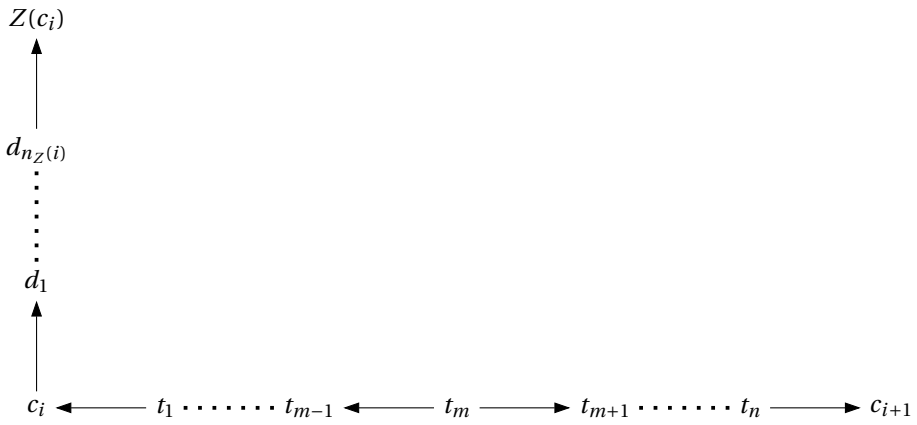
First, we consider the case when $i = C$. Here, c_C is in Z by (vi), and $c_{C+1} := y$ is in Y. Moreover, the nodes c_C and c_{C+1} are connected by the trail

$$c_C \leftarrow t_1 \rightleftharpoons \cdots \rightleftharpoons t_n \rightleftharpoons c_{C+1}$$

with no converging connections and containing no nodes in $Y \sqcup Z$ by (i). Hence, we can apply property $\mathfrak{A}$ to find that they are adjacent, completing the proof for the case when $i = C$.

Now, we prove (viii) for $i \in \{1,\dots,C-1\}$. Note that, by (vii), at least one of the nodes c_i and c_{i+1} belongs to Z, giving us three cases.

Case 1: $c_i \notin Z$ and $c_{i+1} \in Z$.
First, we remark that the arc $c_i \rightarrow c_{i+1}$ is not possible by (iii). Therefore, we must show that $c_i \leftarrow c_{i+1}$. Suppose that this arc is not present in E. This means that c_i and c_{i+1} are not adjacent. Furthermore, $\mathcal{G}$ contains the subgraph below.

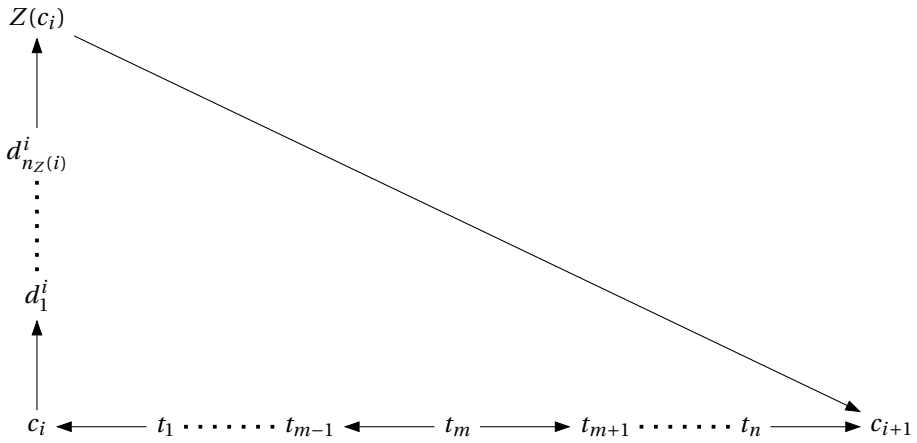


Thus, $Z(c_i)$ and $c_{i+1} \in Z$ are joined by a trail

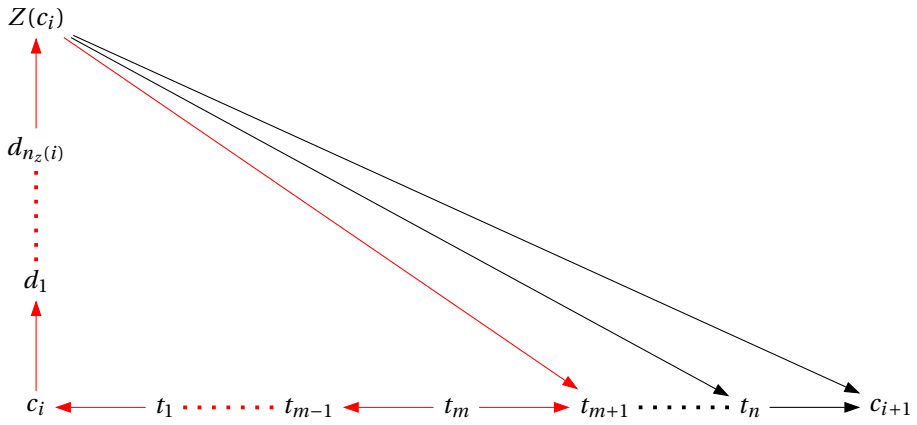
$$Z(c_i) \leftarrow d_{n_Z(i)} \leftarrow \cdots \leftarrow d_1 \leftarrow c_i \leftarrow t_1 \rightleftharpoons \cdots \rightleftharpoons t_n \rightarrow c_{i+1}$$

which is activated by the empty set and consists of nodes not in $Y \sqcup Z$ by (i), and hence they are adjacent, since $Y \sqcup Z$ satisfies $\mathfrak{A}$. We consider both cases; when $Z(c_i) \rightarrow c_{i+1}$ and when $Z(c_i) \leftarrow c_{i+1}$.

First, let us assume that $Z(c_i) \rightarrow c_{i+1}$, giving us the subgraph below.

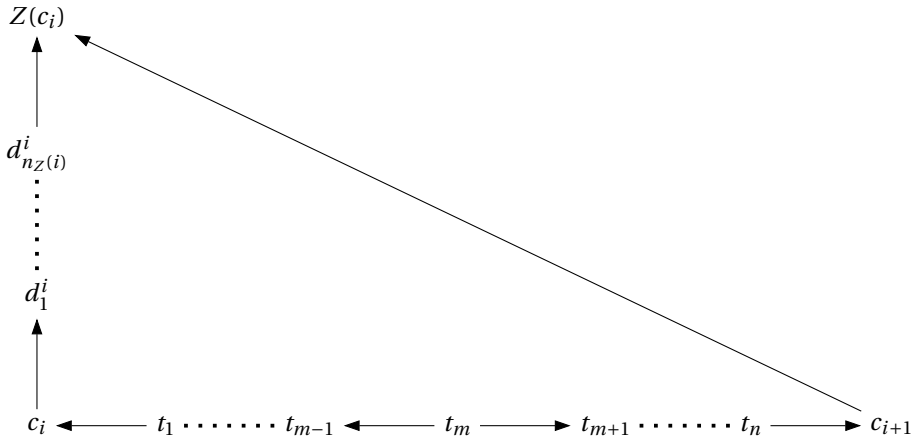


This subgraph contains an undirected cycle with one converging connection (at c_{i+1}), hence the appropriate chords must be present. The same arcs which provided a contradiction in the proof of (vii) still do[1]. This means that the only possible chords are $Z(c_i) \rightarrow t_l$ with $l \in \{m+1, \ldots, n\}$. It is evident that all such arcs are required to be present to prevent an active cycle, giving us the subgraph below.



This provides us with the same undirected cycle as displayed in the proof of (vii), and therefore we have a contradiction in the same way.

Now, suppose that $Z(c_i) \leftarrow c_{i+1}$, giving us the subgraph below.



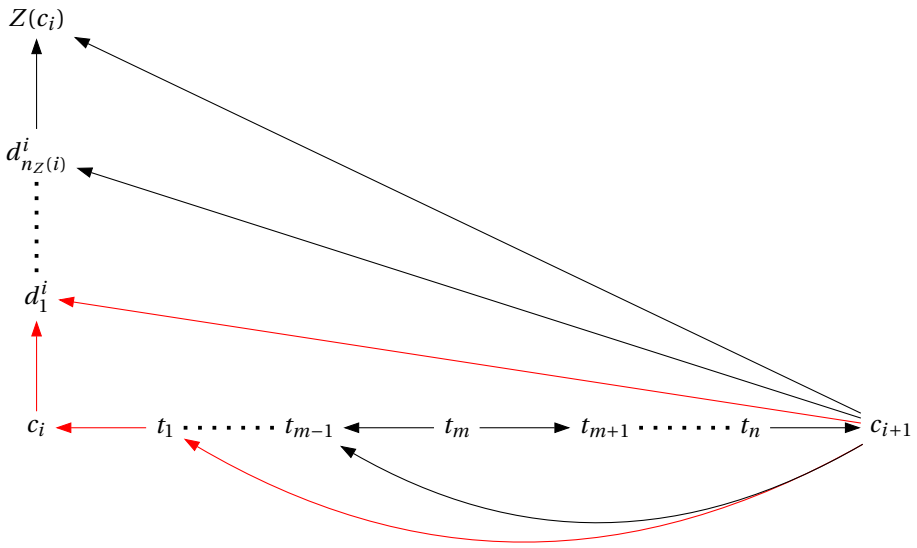
[1] This statement holds because $c_{i+1} = Z(c_{i+1})$

This undirected cycle is an active cycle, unless the appropriate chords are present. We can exclude several chords:

- The trails $c_i \to d_1^i \to \cdots \to d_{n_Z(i)}^i \to Z(c_i)$ and $t_1 \leftarrow \cdots \to t_n$ do not contain chords by (ii).
- $\forall\, j = 0,\ldots,n_Z(i)+1$, $\forall\, l = 1,\ldots,m$, $d_j^i \to t_l$ results in a cycle.
- $\forall\, j = 0,\ldots,n_Z(i)$, $\forall\, l = m+1,\ldots,n+1$, $d_j^i \to t_l$ results in a trail with less converging connections.
- $\forall\, j = 1,\ldots,n_Z(i)$, $\forall\, l = 1,\ldots,n$, $t_l \to d_j^i$ results in a trail with shorter descendant paths (because d_j^i becomes the new converging connection instead of c_i).
- $\forall\, l = 1,\ldots,n$, $t_l \to c_i$ and $t_l \to c_{i+1}$ results in a shorter trail whenever these are chords.
- $\forall\, l = 1,\ldots,n$, $t_l \to Z(c_i)$ result in a trail with less converging connections not in Z.
- $\forall\, l = m,\ldots,n$, $Z(c_i) \to t_l$ results in a cycle.
- $\forall\, l = m+1,\ldots,n$, $c_i \to t_l$ results in a cycle.

Therefore, the only possible chords are $c_{i+1} \to d_j^i$ with $j \in \{1,\ldots,n_Z(i)\}$, $c_{i+1} \to t_l$ with $l \in \{1,\ldots,m-1\}$ and $c_{i+1} \to c_i$.

We will now show that the arc $c_{i+1} \to c_i$ must be present to prevent the occurrence of an active cycle. Consider the case where all possible chords are present except $c_{i+1} \to c_i$, giving us the subgraph below.



This subgraph contains an undirected cycle with one converging connection (at d_1^i). It is made up of the nodes c_i, c_{i+1}, t_1 and d_1^i; therefore it is of length 4. To prevent the occurrence of an active cycle it must have a chord. The only possible chord is the arc $c_{i+1} \to c_i$, and hence this arc must be present.

Case 2: $c_i \in Z$ and $c_{i+1} \notin Z$.
This case is a direct consequence of the previous case by the symmetry of the set $TRAILS(X, Y \,|\, Z)$ with respect to X and Y.

Case 3: $c_i, c_{i+1} \in Z$.

The nodes c_i and c_{i+1} are two nodes in $Y \sqcup Z$ joined by a trail

$$c_i \leftarrow t_1 \rightleftharpoons \cdots \rightleftharpoons t_n \rightarrow c_{i+1}$$

with no converging connections and containing no nodes in $Y \sqcup Z$ by (i). Because $Y \sqcup Z$ has the property $\mathfrak{A}$, we know that c_i and c_{i+1} are adjacent.

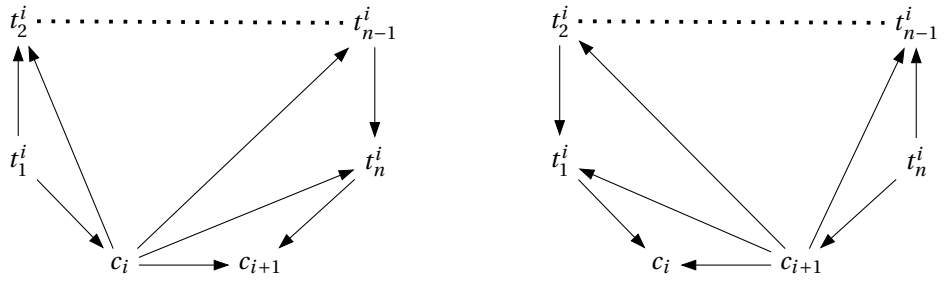
Thus, for each case we have found that c_i and c_{i+1} must be adjacent, completing the proof of (viii).

(ix): We want to show that for all $i = 1, \ldots, C$, $\mathcal{G}$ contains one of the considered two subgraphs.

By (viii) we know that for all $i = 1, \ldots, C$, the nodes c_i and c_{i+1} are adjacent. Moreover, by (v), the trails

$$c_i \leftarrow t_1^i \rightleftharpoons \cdots \rightleftharpoons t_n^i \rightleftharpoons c_{i+1},$$

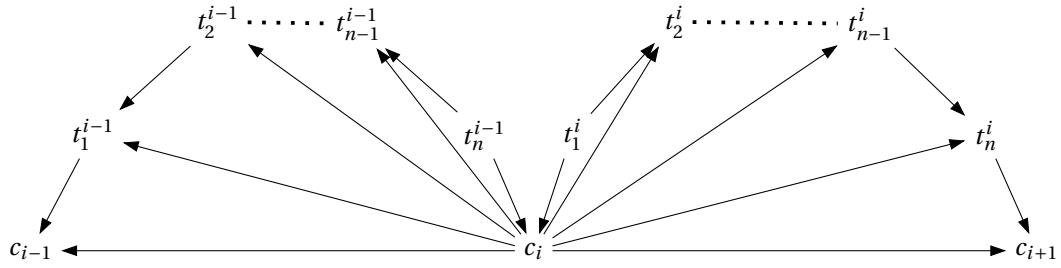
with $t_n^i \rightarrow c_{i+1}$ if $i < C$, are shortest such trails consisting of nodes in $V \setminus Z$. Therefore, we can apply Lemmas 5.5 and 5.9 to find that $\mathcal{G}$ contains one of the two subgraphs below depending on the direction of the arc between c_i and c_{i+1}.



We remark that if $i = C$, then the second subgraph reduces to the arc $c_C \leftarrow y$.

(x): We want to prove that the trail $c_{i-1} \leftarrow c_i \rightarrow c_{i+1}$ can not be present in $\mathcal{G}$.

Suppose that there exists such a diverging connection. By (ix), $\mathcal{G}$ contains the subgraph below.



Remark that

$$t_n^{i-1} \in B(c_i, t_{n-1}^{i-1}) = pa(c_i) \cap pa(t_{n-1}^{i-1}),$$
$$t_1^i \in B(c_i, t_2^i) = pa(c_i) \cap pa(t_2^i).$$

Since $\mathcal{G}$ does not contain any interfering v-structures, we must have $t_1^i \in B(c_i, t_{n-1}^{i-1})$ or $t_n^{i-1} \in B(c_i, t_2^i)$. This means that $t_1^i \rightarrow t_{n-1}^{i-1}$ or $t_n^{i-1} \rightarrow t_2^i$. However, both arcs result in the existence of trails between x and y that have less converging connections than (5.14), and therefore better trails than (5.14). Because this is a contradiction, there cannot be a diverging connection $c_{i-1} \leftarrow c_i \rightarrow c_{i+1}$, proving (x).

(xi): We want to prove that: if $c_{C-1} \to c_C$, then for all $i = 1,\dots,C$, c_i is in Z.

By (viii), the graph contains the trail

$$c_1 \rightleftharpoons c_2 \rightleftharpoons \cdots \rightleftharpoons c_{C-1} \to c_C.$$

This trail can not contain a diverging connection by (x), and therefore, it takes the form

$$c_1 \to c_2 \to \cdots \to c_{C-1} \to c_C.$$

Node c_C is in Z by (vi), and we have that $c_i \to c_{i+1}$ for all $i = 1,\dots,C-1$. Consequently, (iii) implies that $c_i \in Z$ for all $i = 1,\dots,C$. This concludes the proof of (xi).

(xii): If $c_1 \leftarrow c_2$ and $c_1 \in Z$, then for all $i = 1,\dots,C$, c_i is in Z.

This claim follows by an analogous proof as for (xi).

□

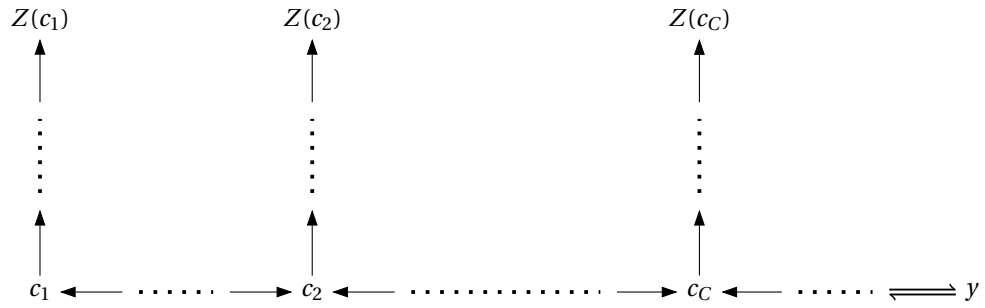
5.4. Lemmas to construct sequences of nodes

For the proof in Section 4.2.3, we use Lemma 4.6 and Lemma 4.8 to construct the sequences of nodes. In both lemmas we assume that there exist a trail with no converging connections in a set $TRAILS(X, Y \,|\, Z)$ where X is equal to a single node w in $B(O_v^k) \setminus O_v^k$ and $Y \sqcup Z \subseteq O_v^k$. Lemma 5.28 shows that such a trail exists under the conditions imposed in Lemma 4.6, and Lemma 5.29 establishes the same claim under the conditions of Lemma 4.8. The proof of both lemmas will rely heavily on the properties we have proven in Lemma 5.26.

But first, we establish another result which is required for the proof of Lemma 5.29.

Lemma 5.27. Under Assumption 5.11, let Y, Z be two subsets such that $Y \sqcup Z = O_v^k$, and let $y \in Y$. Consider the subgraph below where:

- The trail
$$c_1 \leftarrow \cdots \rightarrow c_2 \leftarrow \cdots\cdots\cdots \rightarrow c_C \leftarrow \cdots \rightleftharpoons y$$
has C converging connections corresponding to the nodes $\{c_i\}_{i=1}^C$ with $C > 1$.
- Each c_i is either contained in Z or it has a closest descendant $Z(c_i)$ in Z.
- All nodes on the trail and descendant paths not equal to y or $Z(c_i)$ with $i \in \{1, \ldots, C\}$ are in $V \setminus O_v^k$.



Then, there exists a node $\tilde{o} \in O_v^k$ such that $y \rightarrow \tilde{o}$ and for all $i = 1, \ldots, C$, $Z(c_i) \rightarrow \tilde{o}$ whenever this does not result in the self-loop $\tilde{o} \rightarrow \tilde{o}$. Indeed, the node $\tilde{o}$ may be equal to any node in O_v^k including y and $Z(c_i)$ with $i \in \{1, \ldots, C\}$.

Proof. Remark that $\forall i = 1, \ldots, C$, $Z(c_i) \in Z \subseteq O_v^k$, and the node $y \in Y \subseteq O_v^k$. Therefore, the set $\{Z(c_i)\}_{i=1}^C \cup \{y\}$ must have a highest node according to the ordered set O_v^k. This highest node o_{max} must have been a possible candidate to some partial order $\tilde{O} \subseteq O_v^k$ which contains all other nodes in the set, i.e. $\left[\{Z(c_i)\}_{i=1}^C \cup \{y\}\right] \setminus \{o_{max}\}$.

For convenience of the proof we use the conventions $c_{C+1} := y$ and $Z(c_{C+1}) := y$.

Now, we pick $j \in \{1, \ldots, C+1\}$ such that $Z(c_j) = o_{max}$. Consequently, the node $Z(c_j)$ must be a possible candidate to a set $\tilde{O} \subseteq O_v^k$ which contains $\{Z(c_i)\}_{i=1}^{C+1} \setminus \{Z(c_j)\}$. We now prove that it cannot be a candidate by independence. Observe that at least one of the following trails exist:

$$Z(c_j) \leftarrow \cdots \leftarrow c_j \leftarrow \cdots \rightarrow c_{j+1} \rightarrow \cdots \rightarrow Z(c_{j+1}),$$
$$Z(c_j) \leftarrow \cdots \leftarrow c_j \leftarrow \cdots \rightarrow c_{j+1} \rightarrow \cdots \rightarrow Z(c_{j-1}).$$

These are trails with no converging connections between $Z(c_j)$ and $\{Z(c_{j-1}), Z(c_{j+1})\} \subseteq \tilde{O}$ consisting of nodes in $V \setminus O_v^k \subseteq V \setminus \tilde{O}$. Therefore, we have that $\not{d}\text{-}sep_{\mathcal{G}}\left(Z(c_j), \tilde{O} \,|\, \emptyset\right)$, and therefore $Z(c_j) \notin Poss.Cand_{Ind}(\tilde{O})$, see

Definition 4.4. This means that $Z(c_j)$ must be in $Poss.Cand_{In}(\tilde{O})$ or $Poss.Cand_{Out}(\tilde{O})$. We consider both cases.

Case 1: Suppose that $Z(c_j) \in Poss.Cand_{In}(\tilde{O})$. Then, by Definition 4.4 there exists an $\tilde{o} \in \tilde{O}$ such that $Z(c_j) \to \tilde{o}$ satisfying

1. $pa(\tilde{o} \downarrow Z(c_j)) \subseteq \tilde{O}$,
2. $d\text{-}sep_{\mathcal{G}}\big(Z(c_j), \tilde{O} \setminus \underline{pa}(\tilde{o} \downarrow Z(c_j)) \,\big|\, \underline{pa}(\tilde{o} \downarrow Z(c_j))\big)$.

We will show that this implies that for all $i \neq j$, $Z(c_i) \in \underline{pa}(\tilde{o} \downarrow Z(c_j)) := pa(\tilde{o} \downarrow Z(c_j)) \sqcup \{\tilde{o}\}$. This means that each $Z(c_i)$ with $i \neq j$ points towards $\tilde{o}$ or is equal to $\tilde{o}$. Moreover, by the construction above we also know that $Z(c_j) \to \tilde{o}$. This finishes the proof of Lemma 5.27.

Consider the nodes $Z(c_{j-1})$ and $Z(c_{j+1})$ (assuming that they exist). Suppose that the nodes $Z(c_{j-1})$ and $Z(c_{j+1})$ are not in $\underline{pa}(\tilde{o} \downarrow Z(c_j))$. They are connected to $Z(c_j)$ by the trails

$$Z(c_j) \leftarrow \cdots \leftarrow c_j \leftarrow \cdots \rightarrow c_{j+1} \rightarrow \cdots \rightarrow Z(c_{j+1}),$$
$$Z(c_j) \leftarrow \cdots \leftarrow c_j \leftarrow \cdots \rightarrow c_{j-1} \rightarrow \cdots \rightarrow Z(c_{j-1}),$$

which contain no converging connections nor nodes in $\underline{pa}(\tilde{o} \downarrow Z(c_j)) \subseteq \tilde{O} \subseteq O_v^k$. Therefore, these trails are activated by $\underline{pa}(\tilde{o} \downarrow Z(c_j))$. Thus, they are trails from $Z(c_j)$ to $\tilde{O} \setminus \underline{pa}(\tilde{o} \downarrow Z(c_j))$ activated by $\underline{pa}(\tilde{o} \downarrow Z(c_j))$. This means that $\cancel{d\text{-}sep}_{\mathcal{G}}\big(Z(c_j), \tilde{O} \setminus \underline{pa}(\tilde{o} \downarrow Z(c_j)) \,\big|\, \underline{pa}(\tilde{o} \downarrow Z(c_j))\big)$ which contradicts the assumption that the second restriction is satisfied. Therefore, we must have $Z(c_{j-1}), Z(c_{j+1}) \in \underline{pa}(\tilde{o} \downarrow Z(c_j))$.

Now, the trails

$$Z(c_j) \leftarrow \cdots \leftarrow c_j \leftarrow \cdots \rightarrow c_{j+1} \leftarrow \cdots \rightarrow c_{j+2} \rightarrow \cdots \rightarrow Z(c_{j+2}),$$
$$Z(c_j) \leftarrow \cdots \leftarrow c_j \leftarrow \cdots \rightarrow c_{j-1} \leftarrow \cdots \rightarrow c_{j-2} \rightarrow \cdots \rightarrow Z(c_{j-2})$$

are activated by $\underline{pa}(\tilde{o} \downarrow Z(c_j))$ since the converging connections at c_{j-1} and c_{j+1} have a descendant ($Z(c_{j-1})$ and $Z(c_{j+1})$, respectively) in $\underline{pa}(\tilde{o} \downarrow Z(c_j))$. Thus, by the same argument $Z(c_{j-2})$ and $Z(c_{j+2})$ are in $\underline{pa}(\tilde{o} \downarrow Z(c_j))$.

We conclude this proof by induction. Indeed, the same argument can be repeated to show that for any k, $Z(c_{j+k}) \in \underline{pa}(\tilde{o} \downarrow Z(c_j))$ (resp. $Z(c_{j-k}) \in \underline{pa}(\tilde{o} \downarrow Z(c_j))$) whenever $1 \leq j+k \leq C+1$ (resp. $1 \leq j-k \leq C+1$). Therefore, we have proved that for all $i \neq j$, $Z(c_i) \in \underline{pa}(\tilde{o} \downarrow Z(c_j))$, which completes the proof in this case.

Case 2: Suppose that $Z(c_j) \in Poss.Cand_{Out}(\tilde{O})$. Then, there exists an $\tilde{o} \in \tilde{O}$ with $\tilde{o} \to Z(c_j)$ satisfying

1. $pa(Z(c_j) \downarrow \tilde{o}) \subseteq \tilde{O}$,
2. $d\text{-}sep_{\mathcal{G}}\big(Z(c_j), \tilde{O} \setminus \overline{pa}(Z(c_j) \downarrow \tilde{o}) \,\big|\, \overline{pa}(Z(c_j) \downarrow \tilde{o})\big)$.

We will show that for all $i \neq j$, $Z(c_i) \in \overline{pa}(Z(c_j) \downarrow \tilde{o})$. Therefore, $i \neq j$, $Z(c_i) \to Z(c_j) \in O_v^k$, so $Z(c_j)$ is our desired $\tilde{o}$. This finishes the proof of 5.27 in this case.

First, consider the nodes $Z(c_{j+1})$ and $Z(c_{j-1})$ (assuming that they exist). If they are both in $\tilde{O} \setminus \overline{pa}(Z(c_j) \downarrow o)$, then

$$Z(c_j) \leftarrow \cdots \leftarrow c_j \leftarrow \cdots \rightarrow c_{j+1} \leftarrow \cdots \rightarrow Z(c_{j+1}),$$
$$Z(c_j) \leftarrow \cdots \leftarrow c_j \leftarrow \cdots \rightarrow c_{j-1} \leftarrow \cdots \rightarrow Z(c_{j-1}),$$

contain no converging connections nor nodes in $\overline{pa}(Z(c_j) \downarrow \tilde{o}) \subseteq O_v^k$. Therefore, they are activated by $\overline{pa}(Z(c_j) \downarrow \tilde{o})$. This means that $\not{d\text{-}sep}_{\mathcal{G}}\big(Z(c_j), \tilde{O} \setminus \overline{pa}(Z(c_j) \downarrow \tilde{o}) \,\big|\, \overline{pa}(Z(c_j) \downarrow \tilde{o})\big)$ which contradicts the assumption that the second restriction is satisfied. Therefore, we must have $Z(c_{j-1}), Z(c_{j+1}) \in \overline{pa}(Z(c_j) \downarrow \tilde{o})$.

Now, the trails

$$Z(c_j) \leftarrow \cdots \leftarrow c_j \leftarrow \cdots \rightarrow c_{j+1} \leftarrow \cdots \rightarrow c_{j+2} \rightarrow \cdots \rightarrow Z(c_{j+2}),$$
$$Z(c_j) \leftarrow \cdots \leftarrow c_j \leftarrow \cdots \rightarrow c_{j-1} \leftarrow \cdots \rightarrow c_{j-2} \rightarrow \cdots \rightarrow Z(c_{j-2}),$$

are activated by $\overline{pa}(Z(c_j) \downarrow \tilde{o})$. Thus, by the same argument $Z(c_{j+2}), Z(c_{j-2}) \in \overline{pa}(Z(c_j) \downarrow \tilde{o})$.

Similarly as in the first case, the proof is finished by an induction argument, showing that for all $i \neq j$, $Z(c_i) \in \overline{pa}(Z(c_j) \downarrow \tilde{o})$, as claimed above. □

We will now prove the existence of a trail with no converging connection in $TRAILS\big(w, O_v^k \setminus \underline{pa}(o \downarrow w) \,\big|\, \underline{pa}(o \downarrow w)\big)$ in the setting of Lemma 4.6.

Lemma 5.28. Under Assumption 5.11, let $w \in B(O_v^k) \setminus O_v^k$ and $o \in O_v^k$ such that

- $w \rightarrow o \in E$,
- $pa(o \downarrow w) \subseteq O_v^k$,
- $pa(o \uparrow w) \cap O_v^k = \emptyset$.

If $w \notin Poss.Cand_{In}(O_v^k)$, then there exists a trail from w to $O_v^k \setminus \underline{pa}(o \downarrow w)$ which is activated by $\underline{pa}(o \downarrow w)$ and contains no converging connections.

In particular we will show that for a minimal trail

$$w \rightleftharpoons \cdots \rightarrow c_1 \leftarrow \cdots \rightarrow c_2 \leftarrow \cdots\cdots\cdots \rightarrow c_C \leftarrow \cdots \rightleftharpoons y$$

in the set $TRAILS\big(X, Y \,\big|\, Z\big)$ according to $<_{TRAIL}$ with $X := \{w\}$, $Y := O_v^k \setminus \underline{pa}(o \downarrow w)$ and $Z := \underline{pa}(o \downarrow w)$, the following statements hold:

(i) If the node o is included in the trail, then it must be the first node, i.e. $o := c_1$. If this is the case, then $w \rightarrow o$ is the first subtrail.

(ii) If $c_1 \in Z := \underline{pa}(o \downarrow w)$ then $\mathcal{G}$ contains one of the three subgraphs below.

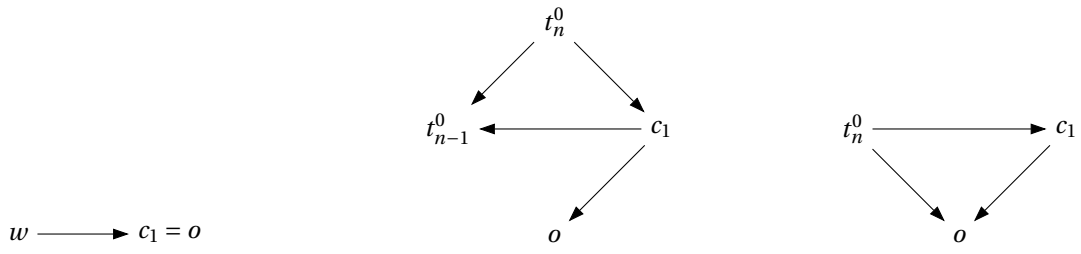


Figure 5.4: Subgraphs for which one must be included in $\mathcal{G}$ in case that $c_1 \in Z$.

(iii) If $C > 1$, then without loss of generality we can assume that $c_{C-1} \leftarrow c_C$, in the sense that there exists a minimal trail (for $<_{TRAIL}$) in $TRAILS\big(X, Y \,\big|\, Z\big)$ such that $c_{C-1} \leftarrow c_C$.

(iv) The node y is not in $pa(o)$.

(v) There exists a node $\tilde{y} \in Y := O_v^k \setminus \underline{pa}(o \downarrow w)$ such that $o \to \tilde{y}$ and $\forall i = 1,\dots,C$, $Z(c_i) \to \tilde{y}$. Furthermore, if $\tilde{y} \neq y$, then $y \to \tilde{y}$.

(vi) The total number of converging nodes C cannot be strictly larger than 1.

(vii) The trail has no converging connections, i.e. $C = 0$.

Remark that (vii) immediately implies that a minimal trail in $TRAILS\big(X, Y \,\big|\, Z\big)$ contains no converging connections. This is equivalent to the first statement of the lemma if the set $TRAILS\big(X, Y \,\big|\, Z\big)$ is not empty, i.e. there must be a trail in $TRAILS\big(X, Y \,\big|\, Z\big)$ containing no converging connections.

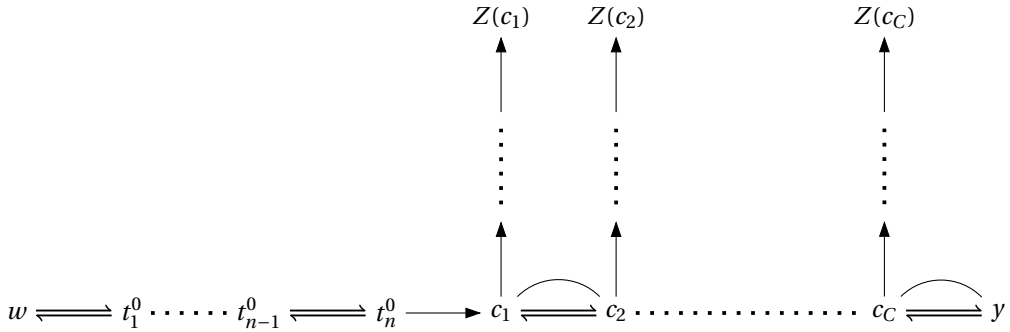
Proof. Since $w \notin Poss.Cand_{In}(O_v^k)$, w cannot be a possible candidate by the incoming arc $w \to o$, see Definition 4.4. This means that one of the two restrictions must be violated:

1. $pa(o \downarrow w) \subseteq O_v^k$.
2. $d\text{-}sep_{\mathcal{G}}\big(w, O_v^k \setminus \underline{pa}(o \downarrow w) \,\big|\, \underline{pa}(o \downarrow w)\big)$.

The first restrictions is satisfied by the assumption of the lemma. Therefore, the second restrictions must be violated, meaning that $\cancel{d\text{-}sep_{\mathcal{G}}}\big(w, O_v^k \setminus \underline{pa}(o \downarrow w) \,\big|\, \underline{pa}(o \downarrow w)\big)$. Hence, the set $TRAILS\big(X, Y \,\big|\, Z\big)$ is not empty. Consequently, there exists a minimal trail

$$w \rightleftharpoons \cdots \to c_1 \leftarrow \cdots \to c_2 \leftarrow \cdots\cdots\cdots \to c_C \leftarrow \cdots \rightleftharpoons y \tag{5.15}$$

in $TRAILS\big(X, Y \,\big|\, Z\big)$ according to $<_{TRAIL}$ with $X := \{w\}$, $Y := O_v^k \setminus \underline{pa}(o \downarrow w)$ and $Z := \underline{pa}(o \downarrow w)$. If this trail contains no converging connections, then the proof of the main statement of the lemma is complete. Therefore, we assume that it contains at least one converging connection. By Lemma 5.26, $\mathcal{G}$ contains the subgraph below with $y \in Y := O_v^k \setminus \underline{pa}(o \downarrow w)$. Moreover, all properties proven in Lemma 5.26 hold for the trail (5.15). This is because $Y \sqcup Z = O_v^k$ satisfies the property $\mathfrak{A}$ by Corollary 5.16.



(i): We must prove that if the node o is included in the trail (5.15), then it must be the first node, i.e. $o := c_1$. If this is the case, then $w \to o$ is the first subtrail.

Proof of (i). Naturally, if the trail (5.15) contains the node $o \in Z := \underline{pa}(o \downarrow w)$, then it must correspond to a converging connection. Otherwise, (5.15) would be blocked by Z.

Consider the case when a node c_i with $i \in \{2,\dots,C\}$ is equal to o. Then, the trail

$$w \to o \leftarrow \cdots \to c_{i+1} \leftarrow \cdots\cdots\cdots \to c_C \leftarrow \cdots \rightleftharpoons y$$

would be a better trail than (5.15) which is a contradiction. Hence, if o is located along the trail it must be equal to c_1.

In this case the subtrail

$$w \rightleftharpoons t_1^0 \rightleftharpoons \cdots \rightleftharpoons t_n^0 \to o$$

takes the form $w \to o$, since we picked a minimal trail. This concludes the proof of (i).

(ii): We show that if $c_1 \in Z := \underline{pa}(o \downarrow w)$ then $\mathcal{G}$ contains one of the three claimed subgraphs.

Proof of (ii). Consider the subtrail

$$w \rightleftharpoons t_1^0 \rightleftharpoons \cdots \rightleftharpoons t_n^0 \to c_1. \tag{5.16}$$

If o is in trail (5.15), then by (i), it is the first node along the trail, i.e. $\mathcal{G}$ contains subgraph in Figure 5.4a.

Suppose that o is not located along the trail. If t_n^0 is in $pa(o)$, then we find the subgraph in Figure 5.4c.

We will now show that if t_n^0 is not in $pa(o)$ and o is not in (5.15), then $\mathcal{G}$ must contain the graph in Figure 5.4b. First, we define an integer

$$p := \max\{i \in \{1,\dots,n\};\, t_i^0 \in pa(o)\}$$

such that t_p^0 is the furthest node from w in trail (5.16) contained in the set $pa(o)$. By Lemma 5.26(ii), the subtrail

$$t_p^0 \rightleftharpoons \cdots \rightleftharpoons t_n^0 \to c_1$$

contains no chords. Moreover, the nodes t_p^0 and c_1 are in $pa(o)$, and the nodes $t_{p+1}^0,\dots,t_n^0$ are not in $pa(o)$.

Remark that $t_p^0 \rightleftharpoons \cdots \rightleftharpoons t_n^0 \to c_1$ is a shortest trail activated by the empty set from t_p^0 to c_1 ending with a rightward pointing arrow consisting of nodes in $V \setminus Z$. Therefore, we may apply Lemma 5.5 and Lemma 5.10 (with $v_1 = t_p^0$, $v_2 = c_1$ and $v_3 = o$ in the notation of Lemma 5.10) to find that $\mathcal{G}$ contains the second subgraph.

(iii): To prove is that if $C > 1$, then without loss of generality we can assume that $c_{C-1} \leftarrow c_C$, in the sense that there exists a minimal trail (for $<_{TRAIL}$) in $TRAILS(X, Y \,|\, Z)$ such that $c_{C-1} \leftarrow c_C$.

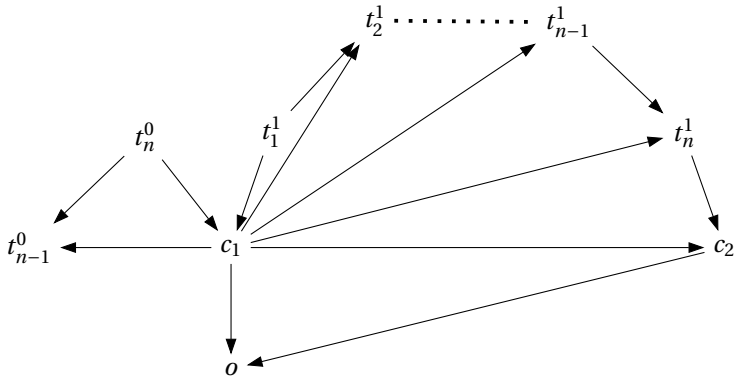
Proof of (iii). By Lemma 5.26(viii), we know that c_{C-1} and c_C are adjacent. Suppose that $c_{C-1} \to c_C$. Combining Lemma 5.26(viii), (x) and (xi) we find that $\forall\, i = 1,\dots,C-1$, $c_i \to c_{i+1}$ and $c_i \in Z := \underline{pa}(o \downarrow w)$. In particular we have that $c_1 \to c_2$ where $c_1, c_2 \in Z$.

We will show that the arc $c_1 \to c_2$ with $c_1, c_2 \in Z$ leads to either a contradiction or the existence of another minimal trail, that satisfies $c_{C-1} \leftarrow c_C$. Since $c_1 \in Z$ we can apply (ii), to find that $\mathcal{G}$ contains one of three subgraphs in Figure 5.4. We consider each case.

Case 1: Subgraph 5.4a.
In this case $c_1 = o$. Since $c_1 \to c_2$, we have that $o \to c_2$. Moreover, because $c_2 \in Z := \underline{pa}(o \downarrow w) \subseteq pa(o)$, we know that $c_2 \to o$, and therefore we have the cycle $c_2 \to o \to c_2$ which is a contradiction.

Case 2: Subgraph 5.4b.

In this case, by Lemma 5.26(ix), $\mathcal{G}$ contains the subgraph below.



Here, we have

$$t_1^1 \in B(c_1, t_2^1),$$
$$t_n^0 \in B(c_1, t_{n-1}^0).$$

Since $\mathcal{G}$ does not contain any interfering v-structures, this means that $t_n^0 \to t_2^1$ or $t_1^1 \to t_{n-1}^0$. These arcs provide us with the following respective trails

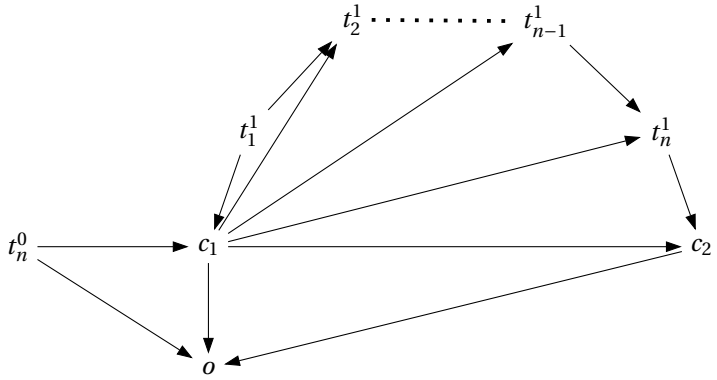
$$w \rightleftharpoons \cdots \rightleftharpoons t_{n-1}^0 \rightleftharpoons t_n^0 \to t_2^1 \rightleftharpoons \cdots\cdots\cdots \rightleftharpoons y,$$
$$w \rightleftharpoons \cdots \rightleftharpoons t_{n-1}^0 \leftarrow t_1^1 \rightleftharpoons t_2^1 \rightleftharpoons \cdots\cdots\cdots \rightleftharpoons y,$$

which are both better than (5.15); indeed, (5.15) can be rewritten as

$$w \rightleftharpoons \cdots \rightleftharpoons t_{n-1}^0 \rightleftharpoons t_n^0 \to c_1 \leftarrow t_1^1 \rightleftharpoons t_2^1 \rightleftharpoons \cdots\cdots\cdots \rightleftharpoons y.$$

We therefore get a contradiction, as claimed.

Case 3: Subgraph 5.4c.
In this case, by Lemma 5.26(ix), $\mathcal{G}$ contains the subgraph below, where $c_2 \to o$ because $c_2 \in Z := \underline{pa}(o \downarrow w)$.



Here, we have $t_n^0 \in B(c_1, o)$ and $t_1^1 \in B(c_1, t_2^1)$. By the same argument as above, this means that $t_n^0 \to t_2^1$ or $t_1^1 \to o$. The former arc results in a trail from w to y, which is a better trail than (5.15), and therefore a contradiction. Hence, we must have the arc $t_1^1 \to o$. This arc provides us with the trail

$$w \to o \leftarrow t_1^1 \rightleftharpoons \cdots\cdots\cdots \rightleftharpoons y \tag{5.17}$$

which is better than the trail (5.15), that is

$$w \rightleftharpoons \cdots \rightleftharpoons t^0_{n_t(0)} \to c_1 \leftarrow t^1_1 \rightleftharpoons \cdots\cdots\cdots \rightleftharpoons y,$$

unless $n_t(0) = 0$. In this case, (5.17) is also a minimal trail in $TRAILS\big(X, Y \,\big|\, Z\big)$.

Note that the converging connections of the trail (5.17) are $o, c_2, \ldots, c_C$. Combining the fact that $o \leftarrow c_2$ with Lemma 5.26(x) we must have

$$o \leftarrow c_2 \leftarrow c_3 \leftarrow \cdots \leftarrow c_C.$$

So, we have proven that there exists a minimal trail in $TRAILS\big(X, Y \,\big|\, Z\big)$ containing an arc $c_{C-1} \leftarrow c_C$, completing the proof of (iii).

(iv): We must show that the node y is not in $pa(o)$.

Proof of (iv). By definition, $y \in Y := O^k_v \setminus \underline{pa}(o \downarrow w)$. Observe that

$$\begin{aligned} Y \cap pa(o) &= \big(O^k_v \setminus \underline{pa}(o \downarrow w)\big) \cap \big(\underline{pa}(o \downarrow w) \sqcup pa(o \uparrow w)\big) \\ &= \big(O^k_v \setminus \underline{pa}(o \downarrow w)\big) \cap pa(o \uparrow w) \\ &\subseteq O^k_v \cap pa(o \uparrow w) = \emptyset, \end{aligned}$$

by assumption of Lemma 5.28, hence $y \notin pa(o)$.

(v): To prove: There exists a node $\tilde{y} \in Y := O^k_v \setminus \underline{pa}(o \downarrow w)$ such that $o \to \tilde{y}$ and $\forall i = 1, \ldots, C$, $Z(c_i) \to \tilde{y}$. Furthermore, if $\tilde{y} \neq y$, then $y \to \tilde{y}$.

Proof of (v). First, we show that there exists an $\tilde{y} \in O^k_v$ such that $o \to \tilde{y}$, $y \to \tilde{y}$ and $\forall i = 1, \ldots, C$, $Z(c_i) \to \tilde{y}$.

Suppose that o is located on (5.15). By (i), this means that $o = c_1$, and therefore $\mathcal{G}$ contains the trail

$$o \leftarrow \cdots \to c_2 \leftarrow \cdots\cdots\cdots \to c_C \leftarrow \cdots \rightleftharpoons y.$$

that satisfies all conditions for Lemma 5.27 (by applying Lemma 5.26(i)). If $o \neq c_1$, then the trail

$$o \leftarrow w \rightleftharpoons \cdots \to c_1 \leftarrow \cdots \to c_2 \leftarrow \cdots\cdots\cdots \to c_C \leftarrow \cdots \rightleftharpoons y$$

also satisfies the conditions of Lemma 5.27 (by applying Lemma 5.26(i)).

Therefore, in both cases we can apply Lemma 5.27 to find that there exists an $\tilde{y}$ in O^k_v such that $o \to \tilde{y}$, $y \to \tilde{y}$ and for all $i = 1, \ldots, C$, $Z(c_i) \to \tilde{y}$; as before if $\tilde{y}$ is equal to one of them (o, y or $Z(c_i)$), then the self-loop is not present. It remains to show that this node $\tilde{y}$ is in $Y := O^k_v \setminus \underline{pa}(o \downarrow w)$, i.e. that $\tilde{y}$ cannot be in $\underline{pa}(o \downarrow w) := pa(o \downarrow w) \sqcup \{o\}$.

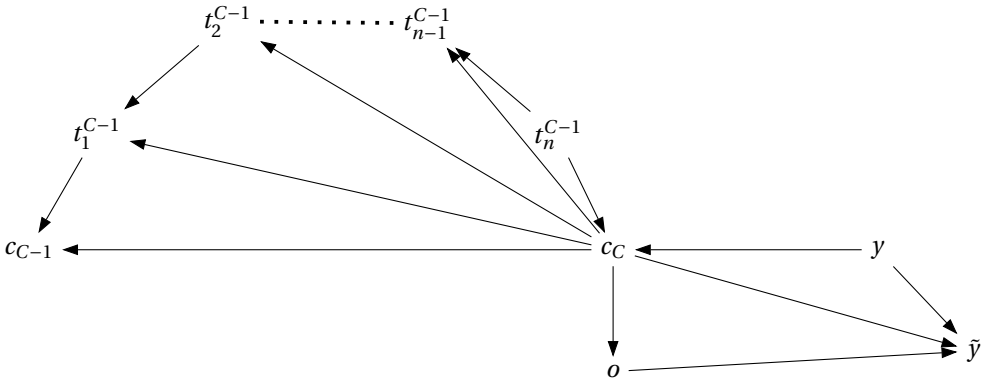
If $\tilde{y} = o$, then $\mathcal{G}$ contains the arc $y \to o$. This is in contradiction with (iv) which states that $y \notin pa(o)$. If $\tilde{y} \in pa(o \downarrow w)$ then $\tilde{y} \to o$, and hence $\mathcal{G}$ contains the cycle $\tilde{y} \to o \to \tilde{y}$ which is a contradiction (because we showed above that $o \to \tilde{y}$). Therefore, $\tilde{y}$ must be in $Y := O^k_v \setminus \underline{pa}(o \downarrow w)$ which concludes the proof of (v).

(vi): We have to prove that the total number of converging nodes C cannot be strictly larger than 1.

Proof of (vi). Assume that (5.15) has $C > 1$ converging connection. The end of the trail can have several different types of structures. By (iii) we can assume that $c_{C-1} \leftarrow c_C$ without loss of generality. If $c_C \to y$, we obtain a contradiction by Lemma 5.26(x). Therefore $c_C \leftarrow y$.

Remark that by Lemma 5.26(vi) the node c_C is in the set $Z := \underline{pa}(o \downarrow w)$, and therefore $c_C \to o$. Consequently, the graph contains the trail $y \to c_C \to o$. This means that the node $\tilde{y} \in Y$ from (v) cannot be equal to y. Indeed, this would lead to the cycle $y \to c_C \to o \to y$ which is a contradiction.

Furthermore, $c_C \in Z$, therefore $Z(c_C) = c_C \to \tilde{y}$ by (v). Combining the previous results with Lemma 5.26(ix) gives the subgraph below.



Here, we have $t_n^{C-1} \in B(c_C, t_{n-1}^{C-1})$ and $y \in B(c_C, \tilde{y})$. Since $\mathcal{G}$ does not contain interfering v-structures, we must have $t_n^{C-1} \to \tilde{y}$ or $y \to t_{n-1}^{C-1}$. Both arcs provide a trail from w to a node in $Y := O_v^k \setminus \underline{pa}(o \downarrow w)$ which is a better trail than (5.15). Indeed, the trails

$$w \rightleftharpoons \cdots\cdots \leftarrow t_n^{C-1} \to \tilde{y},$$
$$w \rightleftharpoons \cdots\cdots \leftarrow t_{n-1}^{C-1} \leftarrow y,$$

contain one fewer converging connection than (5.15) which is

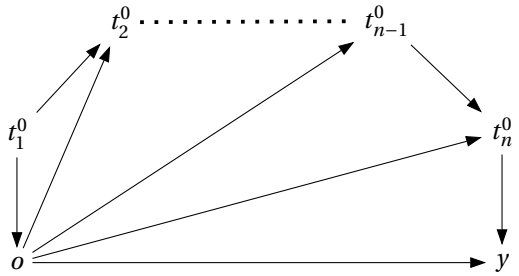
$$w \rightleftharpoons \cdots\cdots \leftarrow t_n^{C-1} \to c_C \leftarrow y$$

We obtain a contradiction because (5.15) was assumed to be a minimal trail. This finishes the proof of (vi).

(vii): We show that the trail has 0 converging connections, i.e. $C = 0$.

Proof of (vii). By (vi), the trail (5.15) has either 0 or 1 converging connection. If it has zero converging connection, then the existence of this trail completes the proof of Lemma 5.28. Therefore we assume that (5.15) has exactly one converging connection, i.e. $C = 1$. Furthermore, by Lemma 5.26(vi) we know that $c_C = c_1 \in Z := \underline{pa}(o \downarrow w)$. This means that we can apply (ii) to find that $\mathcal{G}$ contains one of the three subgraphs in Figure 5.4. We consider each subgraph separately. Furthermore, for each case we will consider two sub-cases; when $c_1 \to y$ and when $c_1 \leftarrow y$, since c_1 and y are adjacent by Lemma 5.26(viii).

Case 1: Subgraph 5.4a.
In this case the node c_1 is equal to o. Since $y \notin pa(o)$ by (iv), the arc $c_1 \leftarrow y$ cannot be present. Therefore, we have $c_1 \to y$. Thus, by Lemma 5.26(ix) we know that $\mathcal{G}$ contains the subgraph below.



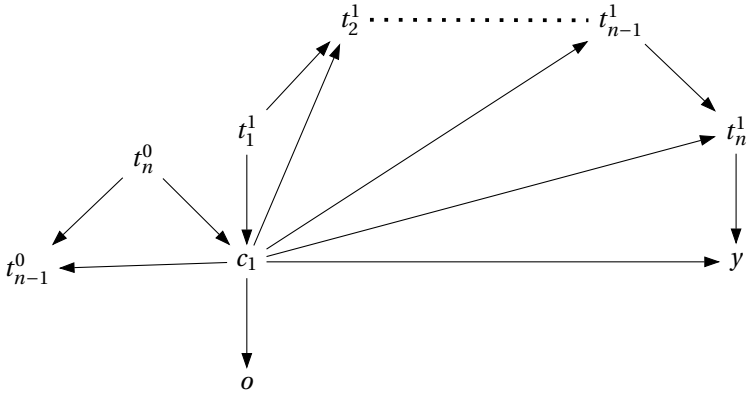
The node t_1^0 is in $V \setminus O_v^k$ by Lemma 5.26(i), and it is also in $pa(o)$. This means that $t_1^0 \in pa(o) \setminus O_v^k$. By the assumptions of the lemma we have that $pa(o \downarrow w) \subseteq O_v^k$ and $o \in O_v^k$, and thus $pa(o) \setminus O_v^k \subseteq pa(o \uparrow w)$. So, $t_1^0 \in pa(o \uparrow w)$, and therefore $w <_o t_1^0$.

Because the parental order $<_o$ has been determined by our algorithm, it abides by the B-sets, see Lemma 5.17. Therefore, any B-set corresponding to the node o which contains t_1^0 must also contain w. Remark that $t_1^0 \in B(o, t_2^0)$. Consequently, we have that $w \in B(o, t_2^0)$. This means that $w \to t_2^0$ which provides us with the trail

$$w \to t_2^0 \to \cdots \to t_n^0 \to y$$

with contains no converging connections. Thus, this trail is better than the trail (5.15) which is a contradiction.

Case 2: Subgraph 5.4b.
Remember that we must consider the cases when $c_1 \to y$ and when $y \to c_1$. For both cases we have that $c_1 \in Z := pa(o \downarrow w)$, and therefore $c_1 \to o$. First, let us assume that $c_1 \to y$, then by Lemma 5.26(ix) we know that $\mathcal{G}$ contains the subgraph below.



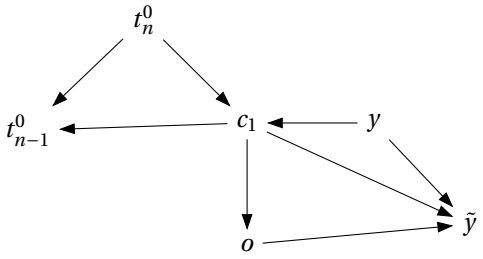
Here, we have that $t_n^0 \in B(c_1, t_{n-1}^0)$ and $t_1^1 \in B(c_1, t_2^1)$. Since $\mathcal{G}$ does not contain any interfering v-structures, we must have $t_n^0 \to t_2^1$ or $t_1^1 \to t_{n-1}^0$. Both arcs result in a trail from w to Y without converging connections, and therefore contradictions. Indeed, we find the trails

$$w \rightleftharpoons \cdots \rightleftharpoons t_n^0 \to t_2^1 \rightleftharpoons \cdots \rightleftharpoons y,$$
$$w \rightleftharpoons \cdots \rightleftharpoons t_{n-1}^0 \leftarrow t_1^1 \rightleftharpoons \cdots \rightleftharpoons y,$$

which are better than (5.15).

Because the arc $c_1 \to y$ leads to a contradiction, we can assume that $c_1 \leftarrow y$. In this case the $\tilde{y}$ whose existence has been established from (v) cannot be equal to y since this would provide the cycle $o \to y \to c_1 \to o$, and

therefore a contradiction. Thus, $\mathcal{G}$ contains the subgraph below.



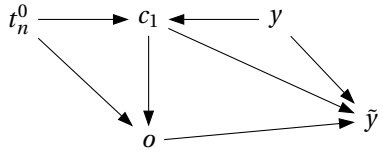
Here, we have $t_n^0 \in B(c_1, t_{n-1}^0)$ and $y \in B(c_1, \tilde{y})$. By the same argument this means that $t_n^0 \to \tilde{y}$ or $y \to t_{n-1}^0$. Both arcs result in a trail from w to Y without converging connections, and therefore contradictions. Indeed, we find the trails

$$w \rightleftharpoons \cdots \rightleftharpoons t_n^0 \to \tilde{y},$$
$$w \rightleftharpoons \cdots \rightleftharpoons t_{n-1}^0 \leftarrow y,$$

where the node $\tilde{y}$ is in Y by (v). This gives us the existence of the trail as claimed.

Case 3: Subgraph 5.4c.

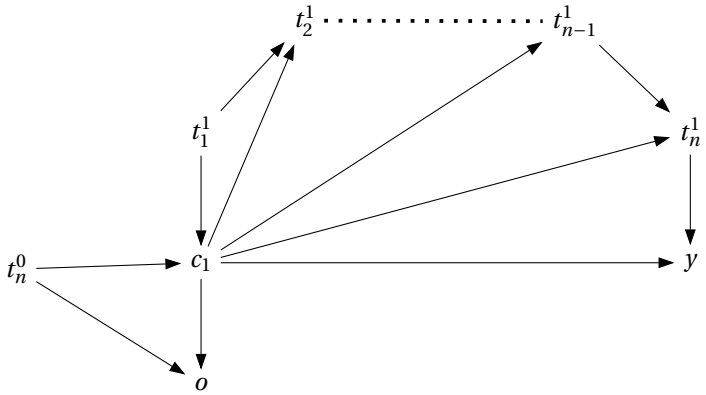
We must consider the two cases $c_1 \leftarrow y$ and $c_1 \to y$. First, let us assume that $c_1 \leftarrow y$, giving us the subgraph below. Again, $\tilde{y}$ cannot be equal to y, since this would provide a cycle. Therefore, $\mathcal{G}$ contains the subgraph below.



Here, we have $t_n^0 \in B(c_1, o)$ and $y \in B(c_1, \tilde{y})$. Therefore, E must contain $t_n^0 \to \tilde{y}$ or $y \to o$. The former arc results in a trail

$$w \rightleftharpoons \cdots \rightleftharpoons t_n^0 \to \tilde{y}$$

from w to Y without converging connections (and therefore a better trail than (5.15)) and the latter is in contradiction with $y \notin pa(o)$ ((iv)). Since both are contradictions, we can assume that $c_1 \to y$. Therefore, by combining subgraph 5.4c with Lemma 5.26(ix) we obtain the subgraph below.



Here, we have $t_n^0 \in B(c_1, o)$ and $t_1^1 \in B(c_1, t_2^1)$. Therefore, we have $t_n^0 \to t_2^1$ or $t_1^1 \to o$. The arc $t_n^0 \to t_2^1$ provides the trail

$$w \rightleftharpoons \cdots \rightleftharpoons t_n^0 \to t_2^1 \rightleftharpoons \cdots \rightleftharpoons y$$

between w and Y without converging connections, and thus a contradiction with the definition of (5.15). The arc $t_1^1 \to o$ provides us with the trail

$$w \to o \leftarrow t_1^1 \rightleftharpoons \cdots \rightleftharpoons t_n^1 \to y \tag{5.18}$$

which is better than the trail (5.15) according to $<_{TRAIL}$, unless $n_t(0) = 0$. In that case, (5.18) is also a minimal trail in $TRAILS\big(X, Y \,\big|\, Z\big)$ with one converging node which is equal to o. Therefore, we can apply the same argument as in Case 1 to the trail (5.18), which leads to a contradiction.

□

In the setting of Lemma 4.8, we will now prove the existence of a trail with no converging connection in the set $TRAILS\big(w, O_v^k \setminus \overline{pa}(w \downarrow o) \,\big|\, \overline{pa}(w \downarrow o)\big)$.

Lemma 5.29. Under Assumption 5.11, let $w \in B(O_v^k) \setminus O_v^k$ and $o \in O_v^k$ such that

- $o \to w$.
- $pa(w \downarrow o) \setminus O_v^k = \emptyset$.
- There is no arc from $B(O_v^k) \setminus O_v^k$ to O_v^k.

If $w \notin Poss.Cand_{Out}(O_v^k)$, then there exists a trail from w to a node in $O_v^k \setminus \overline{pa}(w \downarrow o)$ which is activated by $\overline{pa}(w \downarrow o)$ and contains no converging connections.

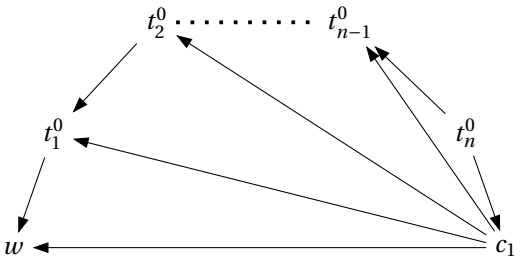
In particular we will show that for a minimal trail

$$w \rightleftharpoons \cdots \to c_1 \leftarrow \cdots\cdots\cdots \to c_C \leftarrow \cdots \rightleftharpoons y$$

in the set $TRAILS\big(X, Y \,\big|\, Z\big)$ with $X := \{w\}$, $Y := O_v^k \setminus \overline{pa}(w \downarrow o)$ and $Z := \overline{pa}(w \downarrow o)$, the following statements hold:

(i) $c_1 = Z(c_1)$.

(ii) The graph contains the subgraph below.



(iii) $c_1 \leftarrow c_2$.

(iv) For all $i = 1, \ldots, C$, we have that $c_i = Z(c_i)$ and $c_i \leftarrow c_{i+1}$.

(v) The number of converging connections is equal to zero, i.e. $C = 0$.

Remark that (v) is equivalent to the first statement of the lemma if the set $TRAILS\big(X, Y \,\big|\, Z\big)$ is not empty.

Proof. By assumption we have $w \notin Poss.Cand_{Out}(O_v^k)$. Therefore, w is not a possible candidate by the outgoing arc $o \to w$. By Definition 4.4, this means that one of the following conditions must be violated.

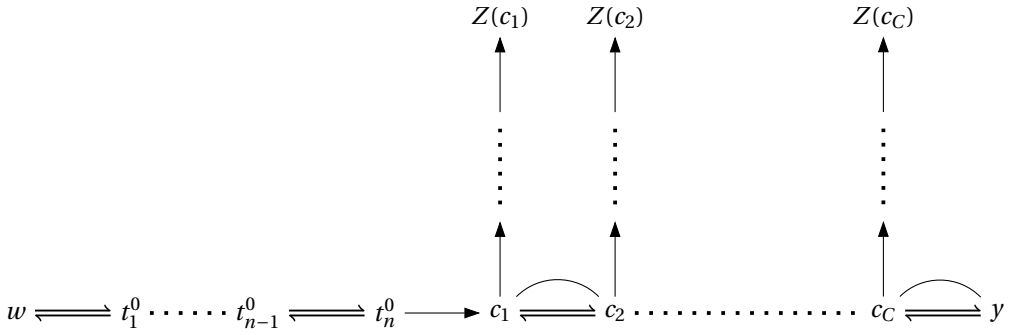
1. $pa(w \downarrow o) \subseteq O_v^k$.
2. $d\text{-}sep_{\mathcal{G}}\big(w, O_v^k \setminus \overline{pa}(w \downarrow o) \,\big|\, \overline{pa}(w \downarrow o)\big)$.

The first condition is satisfied, because $pa(w \downarrow o) \setminus O_v^k = \emptyset$ by the assumptions of the lemma. Therefore, the second restriction must be violated, i.e. $\cancel{d\text{-}sep_{\mathcal{G}}}\big(w, O_v^k \setminus \overline{pa}(w \downarrow o) \,\big|\, \overline{pa}(w \downarrow o)\big)$. This means that there exists a trail from w to $O_v^k \setminus \overline{pa}(w \downarrow o)$ activated by $\overline{pa}(w \downarrow o)$.

Consequently, the set $TRAILS\big(X, Y \,\big|\, Z\big)$ with $X := \{w\}$, $Y := O_v^k \setminus \overline{pa}(w \downarrow o)$ and $Z := \overline{pa}(w \downarrow o)$ is not empty, and thus we can pick a minimal trail in this set:

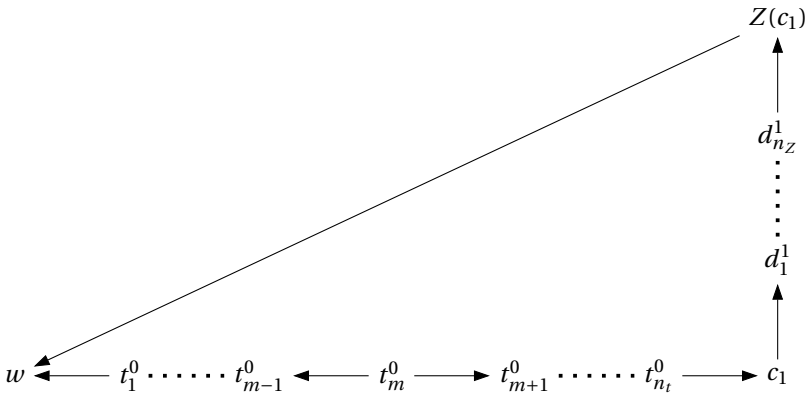
$$w \rightleftharpoons \cdots \to c_1 \leftarrow \cdots\cdots\cdots \to c_C \leftarrow \cdots \rightleftharpoons y. \tag{5.19}$$

If this trail contains no converging connections, then the proof is complete. Therefore, we assume that it contains at least one converging connection, that is $C \geq 1$. By Lemma 5.26, $\mathcal{G}$ contains the subgraph below, and all properties in Lemma 5.26 hold. This lemma can be applied because $Y \sqcup Z = O_v^k$ satisfies the property $\mathfrak{A}$ by Corollary 5.16.



(i): To prove: $c_1 = Z(c_1)$.

Proof of (i). Let us assume that $c_1 \neq Z(c_1)$. Since $Z := \overline{pa}(w \downarrow o)$, we have that $Z(c_i) \to w$ for all $i = 1, \ldots, C$. In particular, we get that $Z(c_1) \to w$, and therefore $\mathcal{G}$ contains the subgraph below.

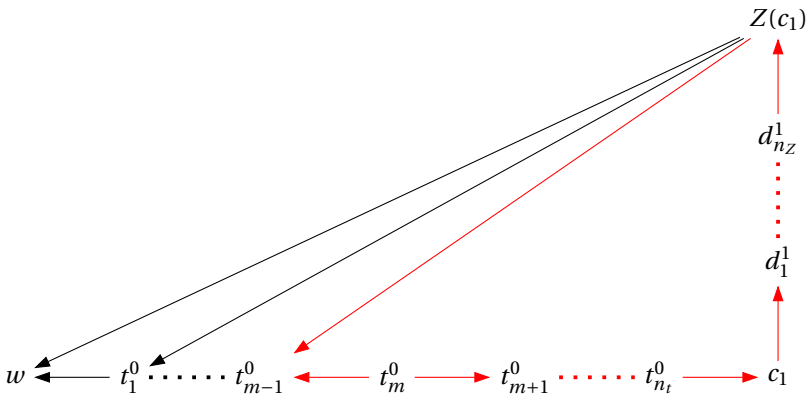


The undirected cycle above is an active cycle, unless the appropriate chords are present in $\mathcal{G}$. Several chords can be excluded:

- The trails $c_1 \to d_1 \to \cdots \to d_{n_t} \to Z(c_1)$ and $w \rightleftharpoons t_1^0 \rightleftharpoons \cdots \rightleftharpoons t_{n_t}^0$ do not contain any chords by Lemma 5.26(ii).

- $\forall l = 0,\dots,n_t-1$, $t_l^0 \to c_1$ results in a shorter trail (and $t_{n_t}^0 \to c_1$ is not a chord).
- $\forall l = 1,\dots,n_t$, $t_l^0 \to Z(c_1)$ results in a trail with less converging connections not in Z.
- $\forall j = 1,\dots,n_Z$, $\forall l = 1,\dots,n_t$, $t_l^0 \to d_j^1$ results in a trail with shorter descendant paths.
- $\forall j = 0,\dots,n_Z+1$, $\forall l = m,\dots,n_t$, $d_j^1 \to t_l^0$ results in a cycle.
- $\forall j = 0,\dots,n_Z$, $\forall l = 0,\dots,m-1$, $d_j^1 \to t_l^0$ result in a trail with less converging connections.
- $\forall j = 1,\dots,n_Z$, $w \to d_j^1$ results in a cycle.
- $\forall l = 0,\dots,m-1$, $c_1 \to t_l^0$ results in a trail with less converging connections.

Therefore, the only remaining chords are $Z(c_1) \to t_l^0$ with $l = 1,\dots,m-1$. It is evident that all such arcs must be present to prevent the appearance of an active cycle in $\mathcal{G}$, giving us the subgraph below.



The subgraph above contains an undirected cycle with one converging connection (at t_{m-1}^0), coloured in red. Because there are no more chords which could be present, this undirected cycle must be of length smaller than 4, see Definition 3.9. The undirected cycle consists of the nodes c_1, $Z(c_1)$, $t_{m-1}^0, t_m^0, \dots, t_{n_t}^0$ and $d_1^1,\dots,d_{n_Z}^1$; therefore it is of length $2 + n_t - (m-1) + 1 + n_Z = n_Z + n_t - m + 4$. This means that $n_Z + n_t - m + 4 \leq 3$, and therefore $n_Z + n_t - m \leq -1$. The equality can only hold if $n_Z = 0$ and $m = n_t + 1$. This is not possible because $t_{n_t+1}^0 = c_1$ is a converging connection, and therefore $t_{n_t}^0 \to c_1$.

So, if $c_1 \neq Z(c_1)$, we have shown that $\mathcal{G}$ contains an active cycle, and therefore we have proven that c_1 must be in Z.

(ii): To prove is that the graph contains the subgraph as claimed.

Proof of (ii). We know that $\mathcal{G}$ contains the trail

$$w \rightleftharpoons t_1^0 \rightleftharpoons \cdots \rightleftharpoons t_n^0 \to c_1.$$

By (i), $c_1 \in Z = \overline{pa}(w \downarrow o)$ Because this is a shortest trail activated by the empty set ending with a rightward arrow ($t_n^0 \to c_1$) consisting of nodes in $V \setminus Z$ and $c_1 = Z(c_1) \to w$ by definition of the set Z, we can apply Lemma 5.5 and Lemma 5.9 (with $v_1 = c_1$ and $v_2 = w$ in the notation of Lemma 5.9) to find that $\mathcal{G}$ contains the subgraph as claimed.

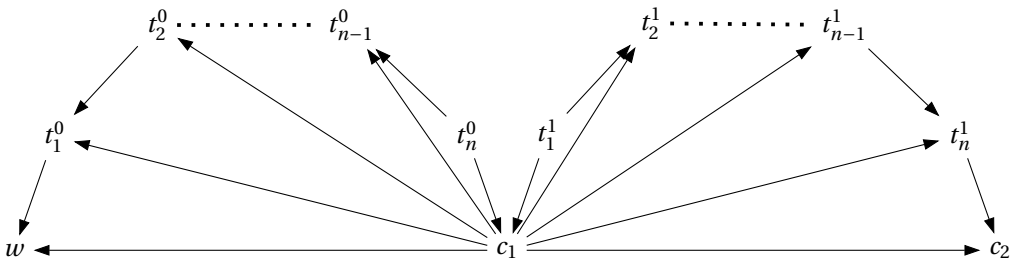
Furthermore, the length n of this trail must be strictly larger than zero. If it were of length zero, then it would simply be the arc $w \to c_1$. However, this would result in a cycle, as we have shown that the arc $c_1 \to w$ must

be present.

(iii): We must prove that $c_1 \leftarrow c_2$.

Proof of (iii). By Lemma 5.26(viii), we know that c_1 and c_2 are adjacent. Therefore, it suffices to show that $c_1 \not\rightarrow c_2$.

Suppose that $c_1 \rightarrow c_2$. Combining (ii) with Lemma 5.26(ix) tells us that $\mathscr{G}$ contains the subgraph below.



Here, we have that $t_n^0 \in B(c_i, t_{n-1}^0)$ and $t_1^1 \in B(c_1, t_2^1)$. Since $\mathscr{G}$ does not contain any interfering v-structures, we must have $t_1^1 \rightarrow t_{n-1}^0$ or $t_n^0 \rightarrow t_2^1$. However, both these arcs result in a better trail than (5.19). Indeed, the trails

$$w \leftarrow \cdots \leftarrow t_{n-1}^0 \leftarrow t_1^1 \rightleftharpoons \cdots\cdots\cdots y,$$
$$w \leftarrow \cdots \leftarrow t_n^0 \rightarrow t_2^1 \rightarrow \cdots\cdots\cdots \rightleftharpoons y,$$

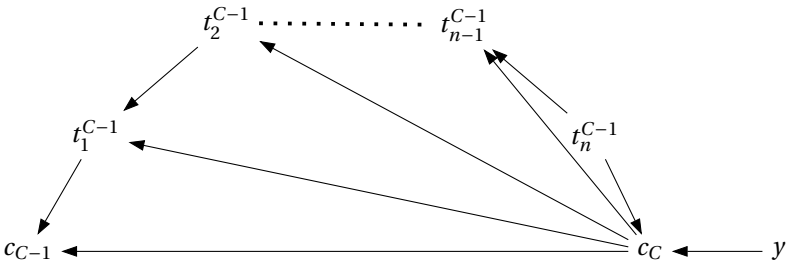
contain one fewer converging connection than (5.19), which gives a contradiction as claimed.

(iv): We must show that or all $i = 1, \ldots, C$, we have that $c_i = Z(c_i)$ and $c_i \leftarrow c_{i+1}$.

Proof of (iv). If $C = 1$, then the statement follows immediately by (i) and (iii). If $C > 1$, by (i) and (iii), we have that $c_1 = Z(c_1)$ and $c_1 \leftarrow c_2$. Consequently, we can apply 5.26(xii) to find that for all $i \in \{1, \ldots, C\}$, $c_i \leftarrow c_{i+1}$ and $c_i = Z(c_i)$.

(v): We must prove that the number of converging connections is equal to zero, i.e. $C = 0$.

Proof of (v). By the facts established above and the properties in Lemma 5.26, we know that $\mathscr{G}$ contains the subgraph below with the convention $c_0 := w$ in the case that $C = 1$.

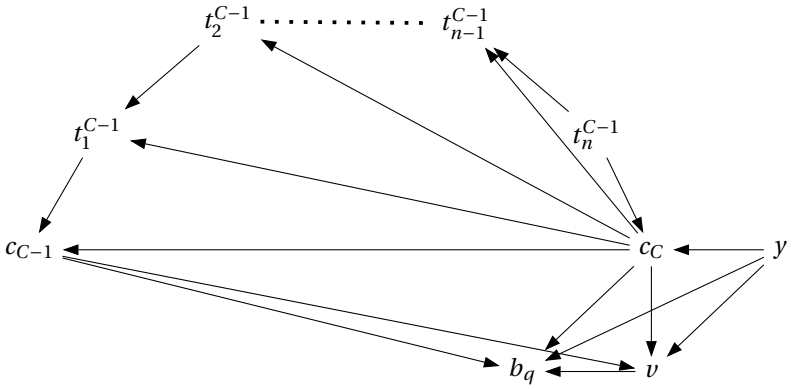


We consider two cases; when $B(O_v^k) \neq pa(v)$ and when $B(O_v^k) = pa(v)$.

Case 1: Let us assume that $B(O_v^k) \neq pa(v)$ and let b_q be its corresponding node, see Definition 3.18. Remark that the nodes c_C and y are in $Y \sqcup Z := O_v^k$, and therefore they are in $B(O_v^k)$. Furthermore, if $C > 1$, then the

node c_{C-1} is in $Z \subseteq O_v^k \subset B(O_v^k)$, and if $C = 1$, then $c_{C-1} = c_0 := w$ where w is in $B(O_v^k)$ by the assumptions of the lemma.

By Definition 3.18, any node in $B(O_v^k)$ has an arc pointing towards both v and b_q, giving us the subgraph below.



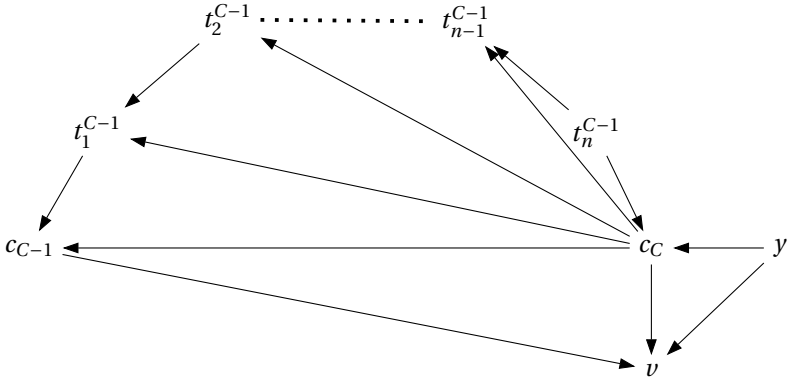
Here, we have that $t_n^{C-1} \in B(c_C, t_{n-1}^{C-1})$, $y \in B(c_C, v)$ and $y \in B(c_C, b_q)$. Since $\mathcal{G}$ does not contain any interfering v-structures, we must have $y \to t_{n-1}^{C-1}$, or both $t_n^{C-1} \to v$ and $t_n^{C-1} \to b_q$. The arc $y \to t_{n-1}^{C-1}$ results in a trail

$$w \rightleftharpoons \cdots \rightleftharpoons t_{n-1}^{C-1} \leftarrow y$$

with less converging connections than (5.19). Since this is a contradiction, the arcs $t_n^{C-1} \to v$ and $t_n^{C-1} \to b_q$ must be present, and therefore $t_n^{C-1} \in pa(v) \cap pa(b_q) = B(v, b_q) = B_q$. Because $B_q = B(O_v^k)$, we obtain $t_n^{C-1} \in B(O_v^k)$. Moreover, by Lemma 5.26(i) we know that $t_n^{C-1} \notin O_v^k$, and thus $t_n^{C-1} \in B(O_v^k) \setminus O_v^k$.

The arc $t_n^{C-1} \to c_C$ is now an arc from a node in $B(O_v^k) \setminus O_v^k$ to a node in O_v^k which is not possible by the assumptions of Lemma 5.29. Therefore, this case provides us with a contradiction.

Case 2: If $B(O_v^k) = pa(v)$, then by a similar argument as for the first case we find that $\mathcal{G}$ must contain the subgraph



Remark that there are potential interfering v-structures at the nodes y and t_n^{C-1} to c_C. As in the previous case, we find that the arc $t_n^{C-1} \to v$ must be present. This means that $t_n^{C-1} \in pa(v) = B(O_v^k)$, and therefore by Lemma 5.26(i) we have that $t_n^{C-1} \in B(O_v^k) \setminus O_v^k$. Hence, we again find the arc $t_n^{C-1} \to c_C$ from a node in $B(O_v^k) \setminus O_v^k$ to a node in O_v^k which is a contradiction.

Thus, for both cases we find a contradiction when $C > 0$, completing the proof of Lemma 5.29. □

6

Estimation and Structure learning

In this chapter we present the process of estimating and structure learning in the subclass restricted PCBNs studied in Chapter 3 and Chapter 4. First, in Section 6.1, we establish how the estimation of the copulas and their assignment has to be curried out, which is then followed by the construction of a structure learning algorithm in Section 6.2. Both topics are complemented with a simulation study in Sections 6.1.1 and 6.2.1, respectively.

6.1. Estimation

A PCBN is characterized by the (conditional) copulas attached to its arcs. Hence, estimating the "optimal" density for given data involves determining the families and corresponding parameters of these copulas and the order in which they are assigned.

Each copula takes the form $c_{wv|pa(v\downarrow w)}$, where $v \in V$ and $w \in pa(v)$. Under the simplifying assumption, such copulas are characterized by a constant vector $\boldsymbol{\theta}_{w\to v}$ which specifies the family and parameters for the arc $w \to v$. The information of all these vectors is stored in a parameter vector denoted by $\boldsymbol{\theta}$[1].

By Theorem 3.15, we know that a PCBN $(\mathscr{G},\mathscr{O})$ with neither active cycles nor interfering v-structures does not necessitate integration. Moreover, in Chapter 4 we have shown that to prevent integration the assignment of copulas, $\mathscr{O}$, must be determined by Algorithm 2. Therefore, we impose the both restrictions on the class of PCBNs. Finally, we assume that we are modeling a random vector with uniform margins, giving us the model below.

Definition 6.1 (Restricted PCBN model)**.** Let $(\mathscr{G},\mathscr{O})$ be a PCBN where $\mathscr{G}$ contains no active cycles and interfering v-structures and $\mathscr{O}$ is chosen by Algorithm 2. Let $\mathscr{P}_{v\to w} = \left\{c_{wv|pa(v\downarrow w);\boldsymbol{\theta}_{w\to v}}; \boldsymbol{\theta}_{w\to v} \in \boldsymbol{\Theta}_{w\to v}\right\}$ be a collection of bivariate (conditional) pair-copula densities for each arc $w \to v \in E$. Then, the collection of densities

$$\mathscr{P} = \Big\{ \prod_{v\in V} \prod_{w\in pa(v)} c_{\boldsymbol{\theta}_{w\to v}}; c_{\boldsymbol{\theta}_{w\to v}} \in \mathscr{P}_{w\to v} \Big\}$$

is a model corresponding to $(\mathscr{G},\mathscr{O})$. The triplet $(\mathscr{G},\mathscr{O},\mathscr{P})$ is called a ***restricted PCBN model***. To reduce heavy notation, we will often simply write $\mathscr{P} = \{c_{\boldsymbol{\theta}}; \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$. Here, $\boldsymbol{\theta}$ contains all information from the vectors in the set $\left\{\boldsymbol{\theta}_{w\to v} \in w \to v \in E\right\}$. If it is clear from context that a PCBN model is restricted, we will simply say PCBN model.

[1] Without the simplifying assumptions the vectors $\boldsymbol{\theta}_{w\to v}$ are functions $\boldsymbol{\theta}_{w\to v}(\cdot)$ which take observations $\mathbf{x}_{pa(v\downarrow w)}$ as inputs.

A joint density $c_{\boldsymbol{\theta}}$ corresponding to a PCBN model can be decomposed by

$$c_{\boldsymbol{\theta}}(\mathbf{u}_V) = \prod_{v \in V} \prod_{w \in pa(v)} c_{wv|pa(v\downarrow w)}\big(u_{w|pa(v\downarrow w)}, u_{v|pa(v\downarrow w)}; \boldsymbol{\theta}_{w\to v}\big). \tag{6.1}$$

Here, the terms $u_{w|pa(v\downarrow w)}$ and $u_{v|pa(v\downarrow w)}$ are computed with a recursion of h-functions, see Section 2.5.5 and Chapter 3. For example, computing a conditional margin $u_{2|1}$ requires the copula c_{12}, and therefore the parameter vector $\boldsymbol{\theta}_{1\to 2}$ (assuming that $1 \to 2 \in E$). Thus, formally we should write $u_{2|1;\boldsymbol{\theta}_{1\to 2}}$ in the joint density. However, to simplify the equation, this is omitted.

Estimating a PCBN given data involves finding the graph, assignment of copulas and parameters. Since this is a highly complicated procedure, we will first assume that the graphical structure and assignment of copulas are known, hence we are tasked with estimating $\boldsymbol{\theta}$. Hereafter, we concern ourselves with the assignment of copulas $\mathcal{O}$ whereas the estimation of $\mathcal{G}$ will be discussed in Section 6.2.

Similar as was done in Section 2.3, we introduce log-likelihood, AIC and BIC for PCBNs.

Definition 6.2 (Likelihood-based selection criteria for PCBNs)**.** Let $(\mathcal{G},\mathcal{O})$ be a PCBN with a corresponding density f_V and parameter vector $\boldsymbol{\theta}$. For a random sample $\mathcal{D} = (\mathbf{u}_V^{(m)})_{m=1,\dots,M}$ of size M, the ***log-likelihood*** is defined by

$$\begin{aligned}\ell(\boldsymbol{\theta};\mathcal{G},\mathcal{O},\mathcal{D}) &= \log\Big[\prod_{m=1}^{M} c_{\boldsymbol{\theta}}\big(\mathbf{u}_V;\boldsymbol{\theta}\big)\Big] \\ &= \sum_{m=1}^{M}\sum_{v\in V}\sum_{w\in pa(v)} \log c_{wv|pa(v\downarrow w)}\big(u^{(m)}_{w|pa(v\downarrow w)}, u^{(m)}_{v|pa(v\downarrow w)};\boldsymbol{\theta}_{w\to v}\big).\end{aligned}$$

Furthermore, the ***AIC*** and ***BIC*** are given by

$$\begin{aligned}AIC(\boldsymbol{\theta};\mathcal{G},\mathcal{O},\mathcal{D}) &= -2\cdot\ell(\boldsymbol{\theta};\mathcal{G},\mathcal{O},\mathcal{D}) + 2k,\\ BIC(\boldsymbol{\theta};\mathcal{G},\mathcal{O},\mathcal{D}) &= -2\cdot\ell(\boldsymbol{\theta};\mathcal{G},\mathcal{O},\mathcal{D}) + \log(M)k,\end{aligned}$$

where k is the number of parameters in $\boldsymbol{\theta}$.

Estimating the joint density is done by maximizing the log-likelihood, $-AIC$ or $-BIC$. These functions can be optimized efficiently, since they are the sum of smaller optimization problems. Indeed, maximizing the log-likelihood reduces to maximizing the terms "$\sum_{m=1}^{M}\log c_{wv|pa(v\downarrow w)}\big(u^{(m)}_{w|pa(v\downarrow w)}, u^{(m)}_{v|pa(v\downarrow w)};\boldsymbol{\theta}_{w\to v}\big)$". Because the arguments of one copula may depend on other copulas, these optimization problems are not independent. However, by estimating the bivariate copulas in a specific order this problem is mitigated. For example, in Figure 3.8b, the copula $c_{34|2}$ takes the argument $u_{4|2}$ which depends on c_{24}. Therefore, we simply estimate c_{24} first, and then compute $u_{4|2}$ using this copula.

Now that we have established how to estimate $\boldsymbol{\theta}$, we move on to the estimation of $\mathcal{O}$. For larger graphs, the number of possible orderings can be extremely large. Therefore, it would be beneficial to find a heuristic which finds the best choice of $\mathcal{O}$ in an intelligent manor. However, we simply check every possible choice of $\mathcal{O}$, and choose $\mathcal{O}$ such that it has the highest maximum likelihood or the lowest minimal AIC or BIC. The steps described above provide us with the algorithm displayed below. The estimated assignment of copulas and parameter vector are denoted by $(\hat{\boldsymbol{\theta}}_\ell,\hat{\mathcal{O}}_\ell)$, $(\hat{\boldsymbol{\theta}}_{AIC},\hat{\mathcal{O}}_{AIC})$ or $(\hat{\boldsymbol{\theta}}_{BIC},\hat{\mathcal{O}}_{BIC})$ depending on the chosen selection criterion.

Algorithm 3 estimation of $\boldsymbol{\theta}$ and $\mathscr{O}$

Input: restricted DAG $\mathscr{G}$, data $\mathscr{D}$, selection criterion $s()$
Output: estimated set of orders and parameter vector $(\hat{\mathscr{O}}, \hat{\boldsymbol{\theta}})$

```
for each set of orders O found by Algorithm 2 do
    θ(O) ← arg max_θ [ s(θ; G, O, D) ]
end for
Ô ← arg max_O [ s(θ(O); G, O, D) ]
θ̂ ← θ(Ô)
return (Ô, θ̂)
```

6.1.1. Simulation study

In this section, we show that Algorithm 3 can accurately estimate the order and parameters given a data set generated from a known PCBN. Moreover, we compare its performance against a GBN.

For this purpose we specify a restricted PCBN as defined in Definition 6.1 and simulate data according to the process described in Appendix B. Naturally, this data set will have uniform margins. Since in applications the true marginal distributions of the data are not known, we rank the margins with the function pobs() from the VineCopula package ([24]) to obtain pseudo-observations.

Because the GBN requires normal margins, the pseudo-observations are then scaled to standard normal margins. To this meta-Gaussian data we fit a GBN using the function bn.fit() from the bnlearn package ([32]).

As discussed in Section 2.5.2, the usual approach of fitting copulas is to apply a two-step process. First, we re-estimate the margins of the meta-Gaussian data by normal distributions using the function fitdistr() from the MASS package ([37]). Hereafter, the PCBN is fitted to the pseudo-observations. The log-likelihood, AIC and BIC of the PCBN will be the criteria as in Definition 6.2 with the addition of the log-likelihood, AIC and BIC of the estimated margins.

The main purpose of this simulation study is to see if the PCBN provides better results than the GBN when applied to data with a non-Gaussian dependence structure. Hence, we will exclusively use non-Gaussian copulas. Moreover, to reduce computational time, only a limited set of copulas is applied; Gumbel, Joe and Frank.

It should be noted that the function BiCopSelect() from the VineCopula package is used to estimate all copulas. When provided with a set of copula families, this function fits all their rotations as well.

Thus, our simulation study follows the steps:

1. Simulate uniform data $\mathbf{U}$ from a PCBN.
2. Rank $\mathbf{U}$ to obtain the pseudo-observations $\hat{\mathbf{U}}$.
3. Scale the margins of $\hat{\mathbf{U}}$ to standard normals to find the meta-Gaussian data $\hat{\mathbf{X}}$.
4. Fit margins to $\hat{\mathbf{X}}$, and fit a PCBN to $\hat{\mathbf{U}}$.
5. Fit a GBN to $\hat{\mathbf{X}}$.
6. Compare the estimated PCBN to the true PCBN.
7. Compare the results of the estimated PCBN and GBN.

Consider a PCBN with graphical structure and assignment of copulas as in Figure 6.1 and parameters from Table 6.1. From this model we generate 1000 data sets containing 1000 samples each. An example of a generated data set can be seen in Figure 6.2.

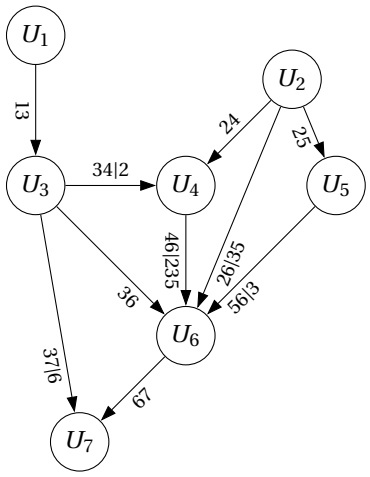


Figure 6.1: PCBN used for the simulation studies.

Arc	Copula	Family	Kendall's τ
$1 \to 3$	c_{13}	Gumbel	0.6
$2 \to 4$	c_{24}	Joe	0.8
$3 \to 4$	$c_{34\|2}$	Gumbel	0.6
$2 \to 5$	c_{25}	Frank	0.7
$3 \to 6$	c_{36}	Joe	0.9
$5 \to 6$	$c_{56\|3}$	Frank	0.6
$2 \to 6$	$c_{26\|35}$	Frank	0.85
$4 \to 6$	$c_{46\|235}$	Gumbel	0.75
$6 \to 7$	c_{67}	Gumbel	0.65
$3 \to 7$	$c_{37\|6}$	Joe	0.55

Table 6.1: The copula families and parameters of the PCBN in Figure 6.1.

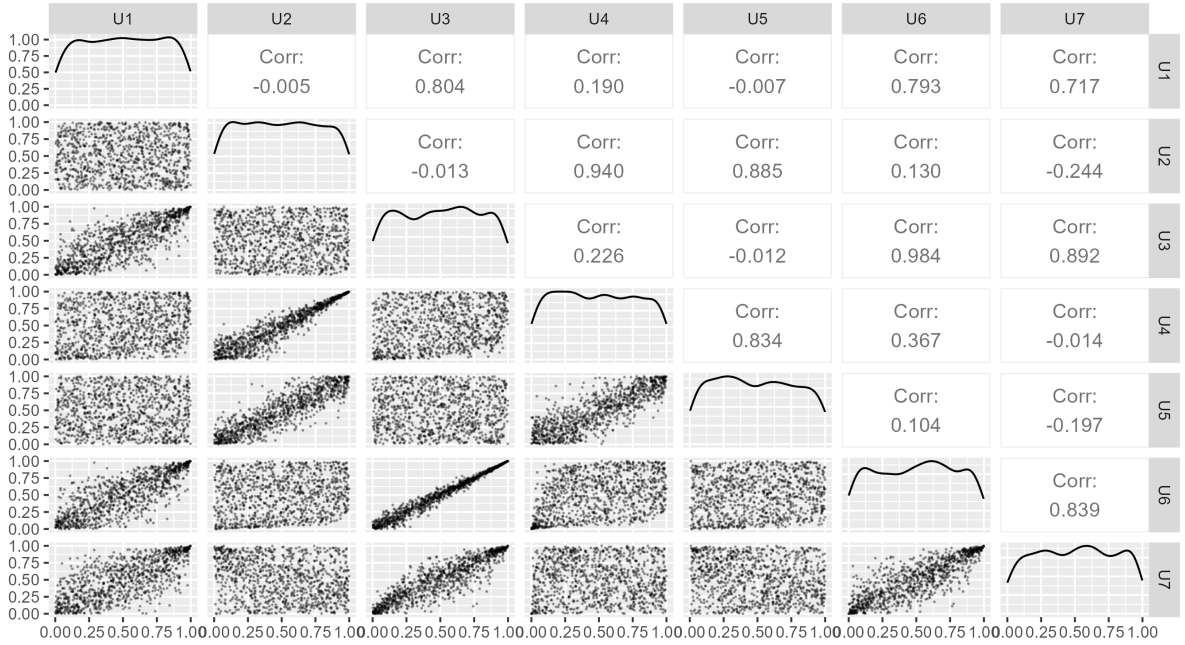


Figure 6.2: Uniform data simulated from the PCBN in Figure 6.1 with parameters from Table 6.1 consisting of 1000 samples.

We fit a PCBN and GBN to the 1000 data sets simulated from specified model above using the BIC as the selection criterion. The PCBN was able to recover the correct order $\mathcal{O}$ for every replication, and for most arcs it was able to estimate the correct copula reasonably well, see Table 6.2. Observe that the unconditional copulas are accurately estimated. The more conditioning variables a copula has, the less accurate results become. Indeed, for the copula $c_{46|235}$ the family is not correctly estimated in most cases and the mean of the estimated Kendall's τ is significantly different to the true Kendall's τ. However, this is to be expected as we are propagating errors when computing conditional copulas.

The average Kendall's τ estimated by the GBN is significantly less accurate than the estimation of the PCBN. In particular for the conditional copulas, the differences are quite large.

Arc	Copula	Correct family	True τ	Estimated τ PCBN	Estimated τ GBN
$1 \to 3$	c_{13}	100%	0.6	0.601 ± 0.0132	0.581
$2 \to 4$	c_{24}	100%	0.8	0.798 ± 0.0078	0.723
$3 \to 4$	$c_{34\|2}$	100%	0.6	0.585 ± 0.0139	0.435
$2 \to 5$	c_{25}	100%	0.7	0.7 ± 0.00903	0.634
$3 \to 6$	c_{36}	100%	0.9	0.898 ± 0.00404	0.815
$5 \to 6$	$c_{56\|3}$	100%	0.6	0.578 ± 0.0128	0.352
$2 \to 6$	$c_{26\|35}$	100%	0.85	0.731 ± 0.014	0.356
$4 \to 6$	$c_{46\|235}$	12%	0.75	0.394 ± 0.0301	0.233
$6 \to 7$	c_{67}	100%	0.65	0.651 ± 0.0124	0.629
$3 \to 7$	$c_{37\|6}$	98.3%	0.55	0.515 ± 0.0154	0.377

Table 6.2: Results of the estimated PCBN and GBN compared to the true model. The estimated Kendall's τ by the PCBN is given as the mean ± standard deviation for the 1000 replications. For the estimated Kendall's τ by the GBN only the mean is given[2].

Let us now compare the performance of the GBN and PCBN by investigating Table 6.3. Here, the Kullback Leibler divergence of the two models with respect to the true distribution has been approximated by Monte Carlo simulation with 10000 samples[3]. When comparing the GBN to the PCBN by the log-likelihood, AIC, BIC and KL divergence, we find that on average the PCBN outperforms the GBN. Furthermore, for each replication the PCBN was the preferred model by a significant margin. For example, the maximum found KL divergence for the PCBN was 0.35, whereas the minimum value for the GBN was 3.11. Thus, the PCBN is clearly the more accurate model. However, its computational time was significantly higher compared to the GBN.

Model	Computation time (s)	log-lik	AIC	BIC	KL
PCBN	5.36 ± 0.358	-2226 ± 107	4499 ± 214	4547± 214	0.0707 ± 0.0946
GBN	0.00041 ± 0.0026	-4975 ± 92.1	10117±184	9999±184	3.44 ± 0.0997

Table 6.3: Performance metrics given as mean ± standard deviation for the 1000 replications. Here, the computation time is given in seconds.

[2]The estimated Kendall's τ for the GBN was a later addition to this thesis. Consequently, the necessary information to compute the standard deviation was not stored.

[3]The sample size of 10000 was chosen because increasing the number of samples did not significantly affect the found result. Furthermore, computing the KL divergence with 10000 samples already took roughly 1.5 seconds per replication.

6.2. Structure learning

The previous section explained the process of estimating the assignment of copulas and parameter vectors. In this setting, the graph was assumed to be known. To estimate the graphical structure we apply a similar approach as was described for the GBN in Section 2.4.

First, we define three likelihood-based score functions in the same fashion as was done in Definition 2.33.

Definition 6.3 (Likelihood-based score functions for PCBNs)**.** Consider a data set $\mathscr{D}$ and a DAG $\mathscr{G}$. We define the following ***likelihood-based score functions***

$$\begin{aligned} score_{\ell}(\mathscr{G};\mathscr{D}) &= \hat{\mathscr{O}}_{\ell}(\hat{\boldsymbol{\theta}}_{\ell};\mathscr{G},\hat{\mathscr{O}}_{\ell},\mathscr{D}),\\ score_{AIC}(\mathscr{G};\mathscr{D}) &= \hat{\mathscr{O}}_{AIC}(\hat{\boldsymbol{\theta}}_{AIC};\mathscr{G},\hat{\mathscr{O}}_{AIC},\mathscr{D}),\\ score_{BIC}(\mathscr{G};\mathscr{D}) &= \hat{\mathscr{O}}_{BIC}(\hat{\boldsymbol{\theta}}_{BIC};\mathscr{G},\hat{\mathscr{O}}_{BIC},\mathscr{D}). \end{aligned}$$

Here, $(\hat{\boldsymbol{\theta}}_{\ell},\hat{\mathscr{O}}_{\ell})$, $(\hat{\boldsymbol{\theta}}_{AIC},\hat{\mathscr{O}}_{AIC})$, $(\hat{\boldsymbol{\theta}}_{BIC}$ and $\hat{\mathscr{O}}_{BIC})$ are the estimators determined by Algorithm 3.

The conditional copulas assigned to the arcs take conditional margins as arguments which must be computed with other copulas. For this reason, the score functions defined above are not decomposable. Furthermore, the score functions are not score equivalent. Indeed, distinct but equivalent graphs can have a different assignment of copulas. In this case, their scores will not be the same.

Remark 6.4. The score functions in Definition 6.3 are not decomposable nor score equivalent, see Section 2.4.1.

Graphs whose distance as defined in Definition 2.35 is equal to zero will have an equal score. Because they have the same v-structures and skeleton which means that they allow for the same assignment of copulas.

With the score functions in Definition 6.3 we can almost immediately apply the Hill climbing algorithm defined in Algorithm 1. However, this algorithm traverses all possible DAGs. Therefore, we must make a slight adjustment such that the algorithm only traverses DAGs without active cycles and interfering v-structures, giving us the algorithm displayed below.

In Example 2.34, it was noted that in the implementation of bnlearn the Hill climbing algorithm chooses between equal arc operations based on the order of the columns in the data-frame. Therefore, for the sake of comparison, we do the same.

Algorithm 4 Hill climbing for restricted PCBNs

Input: restricted DAG $\mathscr{G}$, data $\mathscr{D}$, score function $score(\mathscr{G};\mathscr{D})$
Output: the DAG $\mathscr{G}_{max}$ which locally maximizes $score(\mathscr{G};\mathscr{D})$

```
G_max ← G
S_max ← score(G; D)
while S_max increases do
    for each arc operation e on G_max resulting in a restricted DAG G_e do
        compute the score delta Δ(e) = score(G_e; D) − S_max
    end for
    if max{Δ(e)} > 0 then
        e* = arg max_e {Δ(e)}
        G_max ← G_{e*}
        S_max ← S_max + Δ(e*)
    end if
end while
return G_max
```

6.2.1. Simulation study

Again we consider data simulated from the PCBN in Figure 6.1. We apply Algorithm 4, starting with the empty graph, to see if it finds the correct graphical structure or not. The same is done for the Hill climbing algorithm for GBNs, using the function hc() from the bnlearn package ([32]). In similar fashion as in Section 6.1, the function the function hc() is applied to the meta-Gaussian data and Algorithm 4 to the pseudo-observations. Because of the high computational cost, the simulation study is performed with just 40 replications as this already took roughly 24 hours. For each replication we randomly shuffle the columns of the data-frame for a more realistic result, since the column order is used to break ties in both algorithms.

Furthermore, we also fit a PCBN to the found graph by the hc(). Naturally, this graph may contain an active cycle or interfering v-structure. If this is the case, then we apply Algorithm 5 from Appendix C to transform the graph into a restricted graph. The PCBN estimated with this graph is referred to as "PCBNtoGBN".

Let us investigate the results displayed in Table 6.4. Here, the distance of two graphs is as in Definition 2.35. Again, it is clear that the PCBN outperforms the GBN in terms of accuracy as its found distribution and graph are closer to the true model.

Fitting a PCBN to the found GBN graph also provides us with more accurate results. The benefit of this approach is that it requires significantly less computational time. Another interesting observation is that although on average PCBNtoGBN provided worse results than the PCBN, it did perform better 22.5% of the times in terms of its KL divergence, see Table 6.5. Furthermore, the PCBN outperformed the GBN in 92.5% of the cases whereas the PCBNtoGBN always did.

Model	Computation time (s)	log-lik	AIC	BIC	KL	Distance
PCBN	2065 ± 3027	-2347 ± 213	4747 ± 426	4807 ± 426	0.89 ± 1.13	10.3 ± 6.07
GBN	0.009 ± 0.009	-4852 ± 110	9884 ± 224	9756 ± 222	3.34 ± 0.0974	14.9 ± 4.28
PCBNtoGBN	78 ± 126	-2824 ± 486	5702 ± 970	5764 ± 971	1.03 ± 0.598	15 ± 4.74

Table 6.4: Performance metrics given as mean ± standard deviation for the 40 replications. Here, the computation time is in seconds.

	Lower than PCBN KL	Lower than GBN KL	Lower than PCBNtoGBN KL
PCBN KL	×	92.5 %	67.5 %
GBN KL	7.6 %	×	0 %
PCBNtoGBN KL	22.5 %	100 %	×

Table 6.5: Percentage of the time where the KL divergence of one model was better than another for the 40 replications.

7

Conclusion and future work

In this chapter, we commence by giving a comprehensive summary of the thesis, followed by its key conclusions. Hereafter, we highlight some interesting topics for future research.

7.1. Summary

The main goal of this thesis was to define necessary and sufficient conditions a PCBN must satisfy such that computations of likelihood and simulations do not require integration. It was found that presence of an active cycle or interfering v-structures will necessitate integration, see Theorems 3.11 and 3.14. Furthermore, it was shown that if neither structure is present, then the copulas can be assigned such that integration is not required, see Theorem 3.15. However, this assignment must be chosen in a specific manner. Indeed, in Theorem 4.5, it has been proven that an assignment of copulas will not lead to integration if and only if it is determined by Algorithm 2.

Thus, we have developed a strict subclass of PCBNs for which computations are efficient. For this subclass, estimation and sampling require relatively low computational cost. This has been one of the major drawbacks of PCBNs so far, since the possible need for numerical integration makes them not scalable to larger graphs and data sets.

The proof of Theorem 4.5 required a significant amount of work. We needed to establish a robust framework and prove a long list of supporting lemmas, see Chapter 5. Most of the results established in this chapter hold intrinsic value beyond their role in the proof of Theorem 4.5.

In Section 6.1, we have established how to fit a restricted PCBN to a data set given a fixed graph. This encompasses finding the optimal parameters and assignment of copulas. Here, we opted for a simple heuristic to find the optimal copula assignment; simply try all of them. To evaluate the performance of the PCBN, we simulated 1000 data sets from a PCBN model with a non-Gaussian dependence structure and fitted both the restricted PCBN and a GBN to these data sets. The PCBN was able to find the correct parental orders for each replication, and the parameters were estimated rather accurately. When comparing the results to the GBN, the PCBN was the clear winner. However, it must be mentioned that the simulation study was performed for only one structure. A more comprehensive study is needed before drawing a general conclusion.

In Section 6.2, we have shown how to apply a score-based structure learning algorithm to the restricted PCBN. Here, we applied the Hill climbing algorithm which searched over the space of restricted DAGs, and used likelihood-based score functions. To investigate the performance of the algorithm, we again simulated data

from the same PCBN as was used in Section 6.1. This time only 40 data sets were simulated, as the high computational time did not allow for more repetitions. It was found that the PCBN was better able to recover the graphical structure than the GBN in most cases. In 92.5% of the replications the PCBN outperformed the GBN, according to the log-likelihood, AIC, BIC an KL divergence. Furthermore, we fitted a PCBN to the graph found by the GBN, after removing the active cycles and interfering v-structures, using Algorithm 5 in Appendix C. The resulting PCBN was always more accurate than the GBN, which of course not surprising.

In both estimation and structure learning the PCBN that was found was a more accurate model, but it did so in a significantly longer amount of time. Indeed, the estimation of the restricted PCBN took roughly 5 seconds whereas the GBN is fitted nearly instantaneous; 0.00041 seconds on average. The Hill climbing algorithm for PCBNs took an average of 2065 seconds to complete in contrast with the 0.009 seconds for the GBN. It should be noted that our code has not been properly optimized yet. Although, the computational cost of the PCBN will always be significantly higher than that of the GBN, we expect that the significant improvements in this respect are possible. Furthermore, the simulation were run on a simple laptop with an "AMD Ryzen 7 4800HS with Radeon Graphics" processor.

7.2. Conclusion

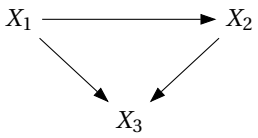
We have established a major theoretical result in the form of the restricted pair-copula Bayesian network. For this subclass of PCBNs the joint density can be computed without integration, making the model scalable to larger graphs and data sets. This is a notable advancement to the previously established PCBNs. Furthermore, through two small simulation studies we have shown that the restricted PCBN is able to model non-Gaussian more accurately than the Gaussian Bayesian network. Therefore, the restricted PCBN is certainly a viable alternative for the GBN and general PCBN given a continuous data set.

7.3. Future work

The main goal of this thesis was to develop a subclass of PCBNs which admit integration-free computations. Although this goal has been achieved, there are many interesting questions left to answer. As the list is quite extensive, we have divided them into several categories below.

Theoretical questions:
We have proven many results concerning properties of trails and d-separation, most of which in Chapter 5. Some of these results are generally applicable. They do not hold only in the restricted graphs. For instance, Lemma 5.6 can be applied for any DAG $\mathcal{G}$. However, other lemmas require a very specific framework, e.g. all lemmas concerning the B-sets and partial orders. It is unclear if these results can be of use elsewhere. Furthermore, we consider the following points:

- The number of possible DAGs given a node set V is known, see [30]. It would be interesting to establish a similar result for the restricted DAGs.
- We have seen that for a DAG, the assignment of copulas $\mathcal{C}$ must be chosen by Algorithm 2, to guarantee integration-free computations. It is unknown how many suitable sets of orders $\mathcal{C}$ there are given a restricted DAG $\mathcal{G}$.
- We defined a heuristic which is able to remove the active cycles and interfering v-structures from a DAG in Algorithm 5, see Appendix C. However, the construction of this algorithm received limited attention. Therefore, it would be of interest to explore other heuristics.
- In Theorem 3.8, it is has been established that multitrees do not require integration for any choice of $\mathcal{C}$. This property is not exclusive to multitrees as one can easily find graphs for which the same is true, e.g. the graph below. Therefore, it would be intriguing to find a more general subclass than the multitrees satisfying this property.



- In Lemma 5.26 we needed the property $\mathfrak{A}$ to hold for the subset $Y \sqcup Z$. Therefore, it would be interesting to see which subsets of DAGs satisfy the property $\mathfrak{A}$ from Definition 5.25.

We have shown that the PCBN provides an obvious benefit when compared to the widely used GBN. However, it is important to note that that there is still a major theoretical gap between the two models. Many subjects which have been excessively investigated for the GBN have not been explored for the PCBN. For instance:

- In GBNs, conditioning on evidences can be done analytically, the same is not possible for the restricted PCBN. It is known that already for Gaussian PCBNs (PCBNs where all copulas are Gaussian) the conditionalization has to be performed by sampling. This might be also a solution for restricted PCBNs.
- Mixed models allowing continuous and discrete data are available for GBNs. It would be interesting to incorporate discrete random variables into the restricted PCBN.

Escaping local maxima:
A known issue of the Hill climbing algorithm is that it can get stuck in a local maximum (local maximal score).

This problem certainly affects Algorithm 4. Furthermore, there could be cases where the best arc operation results in an active cycle or interfering v-structures, hence we cannot escape the local maximum. Also, we have seen that Algorithm 4 is susceptible to different column orders. We provide several possible solutions to these problems:

- Try to escape the local maximum by using random restarts or the tabu search, see Section 2.4.4
- Whenever there is a choice to be made between arc operations resulting in equal score, the algorithm can try both. That is, when for the current graph $\mathcal{G}$ we have arc operations e_1 and e_2 resulting in an equal score delta, simply run the algorithm for both $\mathcal{G}_{e_1}$ and $\mathcal{G}_{e_2}$ as starting graphs.
- Use the output of a structure learning search for GBNs as a starting graph for the PCBN Hill climbing algorithm.
- Use an alternative definition of the neighbouring graphs. For instance, the removal of an arc $w \to v$ may not be allowed because it provides an active cycle. Perhaps it would be better to let the arc operation be; remove the arc $w \to v$ and apply the minimum amount of arc operations which prevent the active cycle from occurring. Idem for arc operations resulting in interfering v-structures.
- If it was possible to determine first the nodes which are most likely to be the member of a large v-structure, then we could change the score function such that arcs pointing towards such nodes are valued more than away pointing arcs.

Simulation studies:
Obvious improvements for future simulation studies are:

- Performing a much large simulation study with more replications, copula families, marginal distributions and larger graphs.
- Comparing the computational time between the restricted PCBN and the general PCBN.

Optimization of code:
The R code used to produce the results of the simulation studies has not been optimized properly. Some measures have been taken to lower the computational cost. For example, any copula and conditional margin which is computed is stored for later usage. However, there are still many improvements which could be made. We note various improvements:

- In Algorithm 3, the optimal assignment of copulas given a graph was determined by simply trying every possible order. Deciding upon a heuristic which estimates the order more intelligently would reduce the computational cost significantly.
- The implementation which checks if a graph has active cycles or not is very much not optimized and takes a significant amount of time for more complex graphs. Therefore, an efficient algorithm is warranted.
- As mentioned before, the score functions for the PCBN in Definition 6.3 are not decomposable. Hence, computing the score delta of an arc operation will not be as efficient as for the GBN. However, our current implementation could still be improved.

Bibliography

[1] Kjersti Aas et al. “Pair-copula constructions of multiple dependence”. In: Insurance: Mathematics and Economics 44.2 (2009), pp. 182–198. DOI: 10.1016/j.insmatheco.2007.02.001.

[2] Elif F. Acar, Christian Genest, and Johanna Nešlehová. “Beyond simplified pair-copula constructions”. In: Journal of Multivariate Analysis 110 (2012), pp. 74–90. DOI: 10.1016/j.jmva.2012.02.001.

[3] Alexander Bauer and Claudia Czado. “Pair-Copula Bayesian Networks”. In: Journal of Computational and Graphical Statistics 25.4 (2016), pp. 1248–1271. DOI: 10.1080/10618600.2015.1086355.

[4] Alexander Bauer, Claudia Czado, and Thomas Klein. “Pair-copula constructions for non-Gaussian DAG models”. In: Canadian Journal of Statistics 40 (2012), pp. 86–109. DOI: 10.1002/cjs.10131.

[5] Tim Bedford and Roger Cooke. “Probability Density Decomposition for Conditionally Dependent Random Variables Modeled by Vines”. In: Annals of Mathematics and Artificial Intelligence 32 (2001), pp. 245–268. DOI: 10.1023/A:1016725902970.

[6] Remco Ronaldus Bouckaert. “Bayesian belief networks: from construction to inference”. PhD thesis. Utrecht University, 1995.

[7] Roger Cooke and Dorota Kurowicka. Uncertainty Analysis With High Dimensional Dependence Modelling. John Wiley & Sons, 2006. ISBN: 0-470-86306-4. DOI: 10.1002/0470863072.

[8] Robert G. Cowell et al. Probabilistic Networks and Expert Systems: Exact Computational Methods for Bayesian Networks. Information Science and Statistics. Springer New York, 1999. ISBN: 978-0-387-98767-5. DOI: 10.1007/b97670.

[9] M.L. Eaton. Multivariate Statistics: A Vector Space Approach. Institute of Mathematical Statistics. Lecture notes-monograph series. Institute of Mathematical Statistics, 2007. ISBN: 9780940600690. DOI: 10.1214/lnms/1196285102.

[10] Geoffrey Grimmet and Dominic Welsh. Probability: An Introduction. Oxford University Press, 1986. ISBN: 9780198709978.

[11] Anca M. Hanea, Dorota Kurowicka, and Roger Cooke. “Hybrid Method for Quantifying and Analyzing Bayesian Belief Nets”. In: Quality and Reliability Engineering International 22 (2006), pp. 709–729. DOI: 10.1002/qre.808.

[12] Anca M. Hanea, Dorota Kurowicka, and Roger Cooke. “Mixed Non-Parametric Continuous and Discrete Bayesian Belief Nets”. In: Advances in Mathematical Modeling for Reliability. Ed. by Tim Bedford et al. Amsterdam: IOS Press, 2008. Chap. 1, pp. 9–16.

[13] David Heckerman, Dan Geiger, and David M. Chickering. “Learning Bayesian Networks: The Combination of Knowledge and Statistical Data”. In: Machine Learning 20 (1995), pp. 197–243. DOI: 10.1023/A:1022623210503.

[14] Ingrid Hobæk Haff, Kjersti Aas, and Arnoldo Frigessi. “On the simplified pair-copula construction — Simply useful or too simplistic?” In: Journal of Multivariate Analysis 101.5 (2010), pp. 1296–1310. DOI: 10.1016/j.jmva.2009.12.001.

[15] Finn V. Jensen and Thomas D. Nielsen. Bayesian Networks and Decision Graphs. Springer, 2007. DOI: 10.1007/978-0-387-68282-2.

[16] Harry Joe. Dependence Modeling with Copulas. Chapman & Hall/CRC, 2014. ISBN: 9780429103186. DOI: 10.1201/b17116.

[17] Harry Joe. “Families of m-variate distributions with given margins and $m(m-1)/2$ bivariate dependence parameters”. In: Distributions with Fixed Marginals and Related Topics. Ed. by L. Rüschendorf, B. Schweizer, and M. D. Taylor. Vol. 28. Institute of Mathematical Statistics. Hayward, CA: Institute of Mathematical Statistics, 1996, pp. 120–141. DOI: 10.1214/lnms/1215452614.

[18] Neville K. Kitson et al. “A survey of Bayesian Network structure learning”. In: Artificial Intelligence Review 56 (8 2023), pp. 8721–8814. DOI: 10.1007/s10462-022-10351-w.

[19] Daphne Koller and Nir Friedman. Probabilistic Graphical Models: Principles and Techniques. Adaptive computation and machine learning. MIT Press, 2009. ISBN: 9780262013192.

[20] Solomon Kullback and Richard A Leibler. “On information and sufficiency”. In: The Annals of Mathematical Statistics 22.1 (1951), pp. 79–86. DOI: 10.1214/aoms/1177729694.

[21] Dorota Kurowicka and Roger Cooke. “Distribution-free continuous Bayesian belief nets”. In: Modern Statistical and Mathematical Methods in Reliability 10 (2005), pp. 309–322. DOI: https://doi.org/10.1142/9789812703378_0022.

[22] Marloes Maathuis et al. Handbook of Graphical Models. 1st. USA: Chapman & Hall/CRC, 2018. ISBN: 1498788629. DOI: 10.1201/9780429463976.

[23] Hans Manner. Estimation and model selection of copulas with an application to exchange rates. Research Memorandum 056. Maastricht University, Maastricht Research School of Economics of Technology and Organization (METEOR), 2007. DOI: 10.26481/umamet.2007056.

[24] Thomas Nagler et al. VineCopula: Statistical Inference of Vine Copulas. R package version 2.4.3. 2021. URL: https://CRAN.R-project.org/package=VineCopula.

[25] Martin Neil et al. “Modelling Operational Risk in Financial Institutions using Hybrid Dynamic Bayesian Networks”. In: Journal of Operational Risk 4 (2009). DOI: 10.21314/JOP.2009.057.

[26] Roger B. Nelsen. An Introduction to Copulas. Springer Series in Statistics. New York: Springer, 2006. ISBN: 978-0-387-28659-4. DOI: 10.1007/0-387-28678-0.

[27] Judea Pearl. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, 1988. ISBN: 978-0-08-051489-5. DOI: 10.1016/C2009-0-27609-4.

[28] Judea Pearl, D. Geiger, and T.S. Verma. “The Logic of Influence Diagrams”. In: Kybernetica 25.2 (1990), pp. 33–44.

[29] Oliver Pourret, Patrick Naim, and Bruce Marcot. Bayesian Networks. A Practical Guide to Applications. John Wiley & Sons, 2008. ISBN: 9780470060308. DOI: 10.1002/9780470994559.

[30] Robert W. Robinson. “Counting unlabeled acyclic digraphs”. In: Combinatorial Mathematics V. Ed. by Charles H. C. Little. Berlin: Springer Berlin Heidelberg, 1977, pp. 28–43. ISBN: 978-3-540-37020-8.

[31] Stuart. Russel and Peter. Norvig. Artificial intelligence : a modern approach. Prentice Hall, 2010.

[32] Marco Scutari and Jean-Baptiste Denis. Bayesian Networks with Examples in R. 2nd. Boca Raton: Chapman and Hall, 2021. ISBN: 9780429347436. DOI: 10.1201/9780429347436.

[33] Marco Scutari, Catharina Elisabeth Graafland, and José Manuel Gutiérrez. “Who learns better Bayesian network structures: Accuracy and speed of structure learning algorithms”. In: International Journal of Approximate Reasoning 115 (2019), pp. 235–253. DOI: https://doi.org/10.1016/j.ijar.2019.10.003.

[34] Ross D. Shachter and C. Robert Kenley. “Gaussian influence diagrams”. In: Management Science 35.5 (1989), pp. 527–550. DOI: 10.1287/mnsc.35.5.527.

[35] Abe Sklar. “Fonctions de repartition an dimensions et leurs marges”. In: Publications de l’Institut de Statistique de l’Université de Paris 8.3 (1959), pp. 229–231.

[36] Ioannis Tsamardinos, Laura Brown, and Constantin Aliferis. “The Max-Min Hill-Climbing Bayesian Network Structure Learning Algorithm”. In: Machine Learning 65 (2006), pp. 31–78. DOI: 10.1007/s10994-006-6889-7.

[37] William N. Venables and Brian D. Ripley. Modern Applied Statistics with S. Fourth. Stastics and Computing. New York: Springer, 2002. ISBN: 978-0-387-95457-8. DOI: 10.1007/978-0-387-21706-2.

[38] Thomas Verma and Judea Pearl. “Equivalence and synthesis of causal models”. In: UAI ’90: Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence, MIT, Cambridge, MA, USA, July 27-29, 1990. Ed. by Piero P. Bonissone et al. Elsevier, 1990, pp. 255–270.

A

Possible candidates sets are disjoint

Lemma A.1. Let O_v^k be a partial order determined by Algorithm 2 with $k < \left|pa(v)\right|$. Then, the sets $Poss.Cand_{Ind}(O_v^k)$, $Poss.Cand_{In}(O_v^k)$ and $Poss.Cand_{Out}(O_v^k)$ are mutually disjoint.

Proof. By Definition 4.4, the set $Poss.Cand_{Ind}(O_v^k)$ must be disjoint from both sets $Poss.Cand_{In}(O_v^k)$ and $Poss.Cand_{Out}(O_v^k)$. Indeed, an element in $w \in Poss.Cand_{Ind}(O_v^k)$ is d-separated from O_v^k given the empty set, and therefore there cannot be an incoming arc $w \to o$ or outgoing arc $o \to w$ with $o \in O_v^k$.

It remains to show that $Poss.Cand_{In}(O_v^k) \cap Poss.Cand_{Out}(O_v^k) = \emptyset$. Suppose that $w \in Poss.Cand_{In}(O_v^k) \cap Poss.Cand_{Out}(O_v^k)$. Since $w \in Poss.Cand_{In}(O_v^k)$, there must exist an $o_1 \in O_v^k$ such that $w \to o_1$ satisfying:

1. $pa(o_1 \downarrow w) \subseteq O_v^k$,
2. $d\text{-}sep_{\mathcal{G}}\big(w, O_v^k \setminus \underline{pa}(o_1 \downarrow w) \,\big|\, \underline{pa}(o_1 \downarrow w)\big)$.

Furthermore, since $w \in Poss.Cand_{In}(O_v^k)$, there must exist an $o_2 \in O_v^k$ such that $o_2 \to w$ satisfying:

(i) $pa(w \downarrow o_2) \subseteq O_v^k$,

(ii) $d\text{-}sep_{\mathcal{G}}\big(w, O_v^k \setminus \underline{pa}(w \downarrow o_2) \,\big|\, \underline{pa}(w \downarrow o_2)\big)$.

The two nodes o_1 and o_2 cannot be the same node. Indeed, this would provide the cycle $w \to o_1 = o_2 \to w$, and therefore a contradiction. Thus, we have $o_1 \neq o_2$.

By combining (ii) with Lemma 5.18, we have that w cannot be adjacent to $O_v^k \setminus \overline{pa}(w \downarrow o_2)$. But, because $o_1 \in O_v^k$ and $w \to o_1$, we know that $o_1 \in O_v^k \setminus pa(w) \subseteq O_v^k \setminus \underline{pa}(w \downarrow o_2)$, and therefore the set $O_v^k \setminus \underline{pa}(w \downarrow o_2)$ is adjacent to w by the arc $w \to o_1$. Since this is a contradiction, the two sets $Poss.Cand_{In}(O_v^k)$ and $Poss.Cand_{Out}(O_v^k)$ are disjoint, proving the lemma. □

B

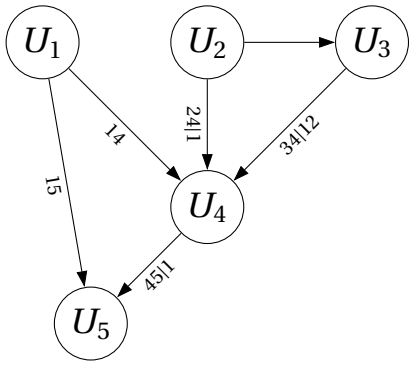
Simulating from a restricted PCBN

The sampling procedure for a PCBN is described in [21]. Given a restricted PCBN the sampling procedure is rather straightforward. For a random vector $\mathbf{U}_V$, we sample in order of an arbitrary well-ordering. Each realization $\mathbf{u}_v$ is then sampled according to the conditional distribution $F_{v|pa(v)}$.

Let us consider the PCBN below. First, let v_1, v_2, v_3, v_4 and v_5 be realizations of independent uniform random variables. A realization $\mathbf{u}_V$ of $\mathbf{U}_V$ can be obtained as follows.

- $u_1 = v_1$.
- $u_2 = v_2$.
- $u_3 = C^{-1}_{3|2;u_2}(v_3)$.
- $u_4 = C^{-1}_{4|1;u_1}\left(C^{-1}_{4|12;u_{2|1}}\left(C^{-1}_{4|123;u_{3|12}}(v_4)\right)\right)$.
- $u_5 = C^{-1}_{5|1;u_1}\left(C^{-1}_{5|14;u_{4|1}}(v_5)\right)$.

For general PCBNs, sampling may require integration, as is seen in the example given in [21]. However, for restricted PCBNs all conditional margins needed as arguments for the inverse CDFs can be computed analytically ($u_{2|1}$, $u_{3|12}$ and $u_{4|1}$ in the example above).



C

DAG to restricted DAG

To turn a general DAG into a restricted DAG, we must remove its active cycles and interfering v-structures. This can be done by either removing or adding arcs. We opted to only add arcs in order to preserve the existing dependence structures.

Consider an active cycle

$$v \leftarrow x_1 \rightleftharpoons \cdots \rightleftharpoons x_n \rightarrow v$$

with its converging connection at v. Removing the active cycle by adding arcs means adding the appropriate chords. Naturally, we have many choices. We opt for the simplest solution; point an arc from each node x_i to v. This can create a large v-structure which may not be desirable.

In case the graph contains interfering v-structures, then there must be a node v for which two B-sets, $B_1(v)$ and $B_2(v)$, are not contained in one another. Thus, we have $w_1 \in B_1$ and $w_2 \in B_2$ such that $w_1 \notin B_2$ and $w_2 \notin B_1$. To fix this problem one can either add an arc from w_1 to b_2, from w_2 to b_1 or both. We decide to add only one arc which will be chosen arbitrarily.

The steps described above provide the following algorithm.

Algorithm 5 DAG to restricted DAG

Input: DAG $\mathcal{G}$
Output: restricted graph $\mathcal{G}^*$

```
G* ← G
for each active cycle in G do
    Add arcs from all nodes in the active cycle to its converging connection in G*
end for
for each v in V do
    for each pair of B-sets B1(v) and B2(v) do
        if B1 ⊈ B2 and B2 ⊈ B1 then
            Add the arcs from B1 \ B2 to b2 or add arcs from B2 \ B1 to b1 to G*
        end if
    end for
end for
return G*
```