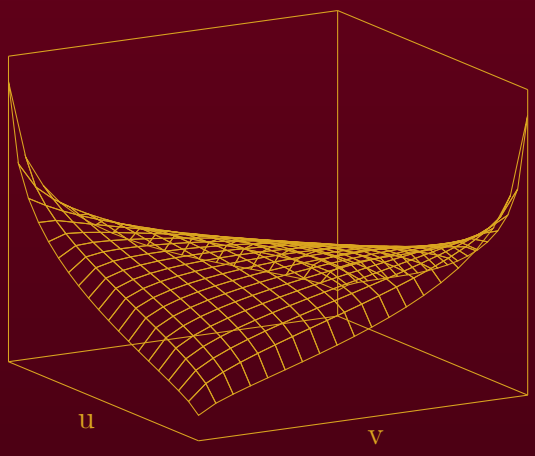
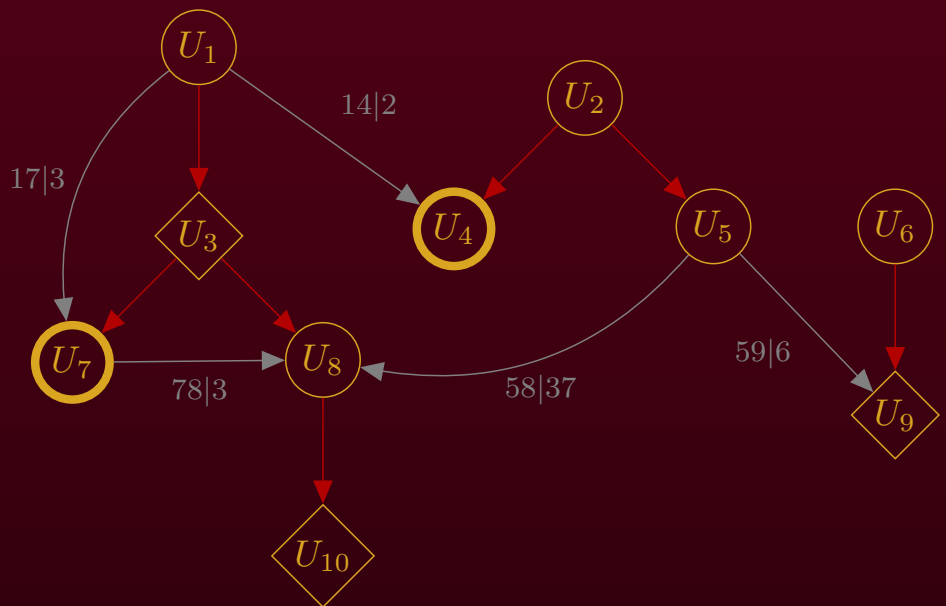
Inference Methods in Pair Copula Bayesian Networks

Panagiotis Basiouras Serrano





TU Delft

Inference Methods in Pair Copula Bayesian Networks

by

Panagiotis Basiouras Serrano

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Friday July 19, 2024 at 10:00 AM.

Student number: 5739055

Project duration: October 6, 2023 – July 19, 2024

Thesis committee: Prof. Dr. D. Kurowicka, TU Delft, supervisor
Dr. A.F.F. Derumigny, TU Delft

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

TU Delft

Abstract

The aim of this thesis is the study of inference problems in Pair Copula Bayesian Networks (PCBN). To this end, certain sub-structures called arteries are identified in the PCBNs and Arterial Sample Propagation, a sample-based extension of Pearl's Belief Propagation Algorithm, is developed for single arteries. This proposed inference methodology incorporates properties unique in PCBNs as well as information on the graph structure, thus avoiding unnecessary computations and boosting the algorithm's performance. Furthermore, an extension of Arterial Sample Propagation is proposed for PCBNs with multiple arteries under some additional assumptions on the graph structure.

This thesis also explores the structural properties of PCBNs, with this examination moving in two separate directions. On the one hand we analyze inference problem reduction through pruning, building up to a pruning algorithm for PCBNs that removes a larger number of variables than existing BN pruning methods. Additionally, we study the implications that the existence of arcs in arteries have on the PCBN's structure. We prove a Theorem used as a background for Arterial Sample Propagation algorithms developed in this thesis, which has potential applications in the development stage of PCBNs.

Preface

This thesis marks the conclusive end of my journey through the Master of Science program in Applied Mathematics at the TU Delft. This journey has been filled with valuable experiences, numerous challenges and learning opportunities that have fundamentally contributed into shaping me as a scientist and an individual.

The choice of PCBN inference as my thesis topic as well as my approach for solving PCBN inference problems are both projections of the evolution of my mathematical interests throughout my studies. Both copulas and graphical models have captivated my attention, offering robust frameworks for understanding complex dependencies and structures within data. Additionally, stochastic sampling has piqued my interest for its ability to capture complex information of mathematical models in a simple form. My exploration into these areas has deepened my appreciation for their theoretical elegance and practical applications, ultimately forming the direction of my research and the methodologies I have employed.

I have spent the last nine months working on this thesis under the diligent supervision of my TU Delft thesis supervisor, Professor Dorota Kurowicka. I feel very lucky for having had such an amazing mentor and scientist guiding me through this journey. I thank her deeply for her ineffable contributions to this thesis; her insightful comments, unwavering support, and the substantial time she dedicated to my work and its structure have been instrumental in shaping the direction and quality of my work. I would like to additionally thank Professor Alexis Derumigny for being a member of my committee and for his valuable feedback during an early stage of my work. Our discussion not only demonstrated how to ensure clarity and comprehension in my work but also provided a significant boost to my self-confidence.

Furthermore, I owe the deepest gratitude to my family for their continuous support and encouragement throughout this chapter of my life. Specifically, my grandparents in Spain, whose unwavering love and comfort have been a constant source of strength. Without their support, I would not have been able to pursue my studies with such dedication. Equally, I am grateful to my parents and sister for their unending love, patience, and sacrifices that have enabled me to focus wholeheartedly on my academic journey.

Lastly, I wish to offer my sincere appreciation for all my close friends, both the ones I made during my studies, as well as those that supported me from Greece. Their presence in my life has been a spring of optimism and strength to keep moving forward and I deeply cherish each and every moment shared and lesson learned together.

Panagiotis Basiouras Serrano
Delft, July 2024

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
PDF	Probability Density Function
PMF	Probability Mass Function
DAG	Directed Acyclic Graph
BN	Bayesian Network
PCBN	Pair Copula Bayesian Network
GBN	Gaussian Bayesian Network
SIR	Sampling Importance Resampling
UBI	Uniformly Backward Instantiated
USBI	Uniformly Secondary Backward Instantiated

Symbols

Symbol	Definition
$\mathcal{N}(\mu, \sigma^2)$	Gaussian Distribution with mean parameter μ and variance parameter σ^2
$\mathcal{U}(\alpha, \beta)$	Uniform Distribution support (α, β)
$\mathscr{G}$	Graph, Bayesian Network
V	Set of nodes/vertices
$\mathscr{G}[V']$	Induced subgraph
A	Set of arcs/edges
$v \to v'$	Arc (v, v')
$v - v'$	Edge $\{v, v'\}$
$pa(v)_{\mathscr{G}}$	Set of parents of v in graph $\mathscr{G}$
$ch(v)_{\mathscr{G}}$	Set of children of v in graph $\mathscr{G}$
$an(v)_{\mathscr{G}}$	Set of ancestors of v in graph $\mathscr{G}$
$de(v)_{\mathscr{G}}$	Set of descendants of v in graph $\mathscr{G}$
$<_v$	Parental order of v
$\mathscr{O}$	Collection of parental orders
$pa(v \downarrow v')$	Parents of v smaller than v' in $<_v$
$pa(v \uparrow v')$	Parents of v larger than v' in $<_v$
$d - sep_{\mathscr{G}}(V_1, V_2 \vert V_3)$	d-separation of V_1 and V_2 given V_3 in graph $\mathscr{G}$
$X_1 \perp\!\!\!\perp X_2$	Independence of X_1 and X_2
$C_{U_1 U_2 \vert U_3}$	Bivariate copula CDF of $(U_1, U_2) \vert U_3$.

Symbol	Definition
$c_{12\|3}$	Bivariate copula PDF of $(U_1, U_2)\|U_3$.
$C_{ij\|pa(j\downarrow i)}$	Copula assigned to arc $U_i \to U_j$.
K_{ij}	Conditioning set $pa(j \downarrow i)$ of copula $C_{ij\|K_{ij}}$
$\mathscr{C}$	Collection of copulas
$\mathscr{F}$	Collection of marginal distributions
u^f	Forward Sample of U
u^b	Backward Sample of U
u^{sb}	Secondary Backward Sample of U
$u^{bil}_{i\to j}$	Bilateral Sample of U_i to be sent to U_j
u^*	Bilateral Sample of U

1

Introduction

The Bayesian Network (BN) is one of the most popular graphical models for representing probabilistic relationships among a set of variables. This model has gained attention across a broad array of domains, including finance, risk management, supply chain management and healthcare. The popularity of this model is due to its ability to incorporate information in an intuitive graphical way. BNs are able to capture dependence and independence relationships through solid and understandable mathematical theory. This, combined with the ever expanding scope of related software implementations, has established the attractiveness of BNs in fields such as reliability, safety etc [64, 70].

A real-life example of a BN application is the computational engine of Bonaparte for Disaster Victim Identification (DVI)[19]. Recent disaster incidents such as the climate change-fueled surge of natural disasters have emphasized the necessity of DNA inference methods for victim identification. The Bonaparte system relies on BNs in order to model the relationship between variables and to perform Bayesian inference given the available evidence. This system was deployed at the Netherlands Forensic Institute after the 2010 Afriqiyah Airways flight crash in Tripoli, Libya and its success in correctly and efficiently identifying the 103 deceased victims of the crash in an automated manner underscores the significance of ongoing research and development in BNs and their relevance in pressing societal matters.

Let us start with a simplistic example in the scope of disease diagnosis in healthcare to explain some main concepts in BN models. Consider we wish to predict whether a patient has the seasonal flu or not based on the symptoms of a patient. The relevant symptoms would be coughing (C), fever (F), sore throat (S), runny nose (R). Additionally the model will consider whether the patient has the flu or not (FL) and whether it is allergy season (A). The directed graph of the BN model for this example is seen in figure 1.1. Note that all variables represented by nodes of the graph are binary taking values True or False (1 or 0).

This particular graph would indicate that the flu influences the probability of runny nose, cough and fever, while runny nose may also be influenced by the allergy season. Furthermore, there is a clear dependence structure between runny nose and a cough, as they may be both caused by some other disease. Additionally, both runny nose and

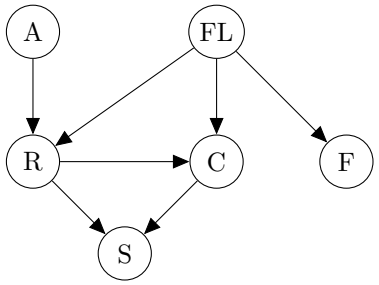


Figure 1.1: A simple Bayesian Network for disease diagnosis.

coughing can lead to a sore throat through post-nasal drip or by directly irritating the larynx or the pharynx.

The dependencies between variables corresponding to nodes in the graph, which can be continuous or discrete, are indicated by arcs. Additionally BNs require specification of conditional densities (for continuous variables) or conditional probability tables (for discrete variables, respectively). For instance, the conditional probability table of fever given flu is:

$$\mathbb{P}(F \mid FL) = \begin{array}{|c|c|c|c|} \hline \multicolumn{2}{|c|}{\text{F}} & 1 & 0 \\ \hline \multirow{2}{*}{FL} & 1 & 0.7 & 0.3 \\ \cline{2-4} & 0 & 0.05 & 0.95 \\ \hline \end{array}$$

Specification of a graph (which needs to be directed and acyclic) and conditional densities (or probability tables) of all nodes given their parents is sufficient to factorize the density (mass function) of the whole network (for details see chapter 2) in terms of those conditional densities or probabilities. The factorization of the mass function for the simple example above is:

$$\begin{aligned} \mathbb{P}(A = x_a, FL = x_{fl}, R = x_r, C = x_c, F = x_f, S = x_s) = \\ \mathbb{P}(S = x_s | R = x_r, C = x_c) \cdot \\ \mathbb{P}(R = x_r | A = x_a, FL = x_{fl}) \cdot \\ \mathbb{P}(C = x_c | R = x_r, FL = x_{fl}) \cdot \\ \mathbb{P}(F = x_f | FL = x_{fl}) \cdot \\ \mathbb{P}(A = x_a) \cdot \mathbb{P}(FL = x_{fl}). \end{aligned} \tag{1.1}$$

If data of patients suffering with respiratory symptoms is available one could estimate parameters of the BN model (conditional probabilities in the tables) for the fixed graphs structure. It is also possible to search an appropriate structure [36].

In this thesis we concentrate on how the BN models are used after they have been built. This part of the process is often referred to as conditionalization or inference. During this phase we are presented with observations of certain variables in the network, also called evidence and represented as $\boldsymbol{E}$, and the goal is to calculate conditional distributions of the form $X|\boldsymbol{E}$ of some other variables given the evidence. For example, we could find the probability of having a flu given that the patient has a sore throat outside of the allergy season (for the BN in figure 1.1). We would calculate:

$$\mathbb{P}(FL = 1 \mid S = 1, A = 0).$$

Since the joint distribution of whole network is specified, one can compute such conditional probability by summing the probability in equation 1.1 over all values of the intermediate nodes R, C, F for $FL = 1, S = 1, A = 0$ to obtain $\mathbb{P}(FL = x, S = 1, A = 0)$, $x \in \{1, 0\}$, then summing out also over possible values of FL to obtain $\mathbb{P}(S = 1, A = 0)$. Then we can simply use the definition of conditional probability to obtain:

$$\mathbb{P}(FL = 1 \,|\, S = 1, A = 0) = \frac{\mathbb{P}(FL = 1, S = 1, A = 0)}{\mathbb{P}(S = 1, A = 0)}$$

However, using this procedure in larger networks with variables that take more values is computationally prohibitive. In just this small example we require $3 \cdot 2^9 + 1$ computations for the inference computed in this way.

For this reason it is essential to study more intricate inference algorithms that achieve the same goal in a more efficient way. These can be divided between ones that operate on BNs that feature either discrete, continuous or both types of variables. The literature of discrete BN algorithms ranges from simplistic approaches such as Variable Elimination to more sophisticated ones like Belief Propagation or Monte-Carlo simulations. The methods for continuous BNs are limited. Continuous variables are often discretized and discrete algorithms to propagate evidences are applied. The exceptions are Gaussian Bayesian Networks (GBNs) where propagation can be performed efficiently. The research of Hybrid BNs is not that advanced, with available models such as Conditionally Gaussian Bayesian Networks (CGBN) being unreasonably restrictive in either efficiency or structure of the BN.

A relatively new and promising family of BNs is the Pair Copula Bayesian Network (PCBN) model. This model was first proposed by Kurowicka and Cooke [46] and further discussed by [31, 32, 30, 3, 4]. The conditional distributions of nodes given their parents are decomposed through a collection of (conditional) bivariate copulas (distributions on unit square with uniform margins). These copulas can be parametric or non-parametric and can be specified freely without constraints.

Despite many appealing properties of PCBNs, these models were not applied in full generality except of the case when all copulas in the decomposition are Gaussian. The recent advances in a master thesis by Horsman [36] has made it possible to perform estimation and simulations in PCBNs when copulas are not Gaussian.

This thesis will focus on the probabilistic inference of PCBNs. We will provide an algorithm that, given evidence in the network in the form of observations of multiple variables, yields the conditional (joint) distributions of subsets of remaining nodes. Our approach to the inference problem will be through application of stochastic sampling.

The motivation behind this is fourfold. Firstly, stochastic sampling can provide with algorithms that exhibit granularity, taking advantage of parallel computing settings. Secondly, we will show that stochastic sampling algorithms require a lower amount and simpler type of integrations present in the algorithm compared to other methods. Furthermore, the presence of certain graph structures called loops poses a problem in most of non-sampling algorithms; we will show that stochastic sampling can be applied to net-

works with loops. Lastly, inference algorithms allow us to tackle more complex inference problems that the rest of the methodologies. These problems arise when the evidence in the network is in the form of a distributional shift instead of a single observation. For instance, in the example of figure 1.1, consider that due to thermometer malfunction we are not certain whether the patient has a fever but we get a 70% probability of it being the case. In this case implementation of stochastic sampling algorithms is straightforward, whereas all other methodologies struggle with tackling the inference task.

The goals of this thesis are the following:

- Create the theoretical foundation upon which PCBN inference will be based on.
- Create an effective and efficient methodology for solving PCBN inference problems for certain DAG structures we define as "arteries".
- Create an algorithm for simplifying general PCBN inference problems into smaller yet equivalent ones.
- Illustrate ways that our methodology can be extended to be applicable to general DAG structures.

In the following section we will provide an overview of the theoretical foundations necessary for our thesis. This includes Graph Theory, Bayesian Networks and a description of the PCBN model. In Chapter 3 we will discuss some of the existing methodologies that are currently available to tackle inference problems in BNs. Chapter 4 contains the detailed formalization of our artery-based inference. There, we present algorithms for conducting efficient and effective inference.

We follow with a Chapter on PCBN pruning, our methodology for simplifying the PCBN based on conditional indepenencies. The thesis continues with Chapter 6, an extensive simulation study where we test the accuracy and computational cost of our algorithms.

Next, Chapter 7 gives a foundation for generalizing our previous PCBN inference approach to general network structures. We illustrate insights on how this can be done and propose a detailed methodology. Lastly, this thesis provides some supportive properties of arteries which are fundamental for both our thesis and the deeper understanding of PCBNs.

2

Prerequisites

In this chapter we provide the necessary theoretical background needed later on in the thesis. The concepts of graph theory that will be used in the rest of the thesis are defined and notation is fixed. Additionally, we will define BNs and build towards the introduction of Pair Copula Bayesian Networks.

All contents of section 2.1 have been extensively discussed and referenced in multiple sources such as Bollobás [8],Daphne Koller [18],West [69] and Bondy, Murty, et al. [9]. These books provide a more in-depth analysis of Graph Theory than the one required in this thesis and include all definitions and results discussed.

In section 2.2 we will cover the background information on Bayesian Networks, which are discussed in detail in Daphne Koller [18], Jensen and Nielsen [39] and Holmes and Jain [35].

Lastly, for section 2.3, we refer to the references [46, 4, 3, 40, 41, 36].

2.1. Graph Theory

Graphs are visual representations of the relationships between different objects represented by nodes or vertices. These relationships are visualized as lines which are called edges or arcs and can be either directed or undirected.

Definition 2.1.1 (Graph)**.** A **graph** is a pair $\mathscr{G} = (V, A)$ where:

- V is a non-empty finite set and,
- $A \subset \{(v_i, v_j); v_i, \in V, v_i \neq v_j\} \cup \{\{v_i, v_j\}; v_i, v_j \in V, v_i \neq v_j\}$.

The set V is the set of **vertices/nodes** while the set A is the set of **arcs/edges**. It is assumed that $\forall v_i, v_j \in V$, of the possible arcs/edges $(v_i, v_j), (v_j, v_i)$ and $\{v_i, v_j\}$ at most one can exist in A.

By taking subsets of V, A, we can get subgraphs of the original graph.

Definition 2.1.2 (Subgraph, Induced Subgraph)**.** Let a graph $\mathscr{G} = (V, A)$.

- A graph $\mathscr{G}' = (V', A')$ is called a **subgraph** of $\mathscr{G}$ if-f $V' \subseteq V$ and $A' \subseteq A \cap (V' \times V')$. Then we write $\mathscr{G}' \subseteq \mathscr{G}$.
- Let $V' \subseteq V$. If $A' = A \cap (V' \times V')$ then the subgraph $\mathscr{G}' = (V', A')$ is called an **induced subgraph** of $\mathscr{G}$ and written as $\mathscr{G}[V']$.

Definition 2.1.1 highlights the distinction between arcs of the form $\{x, y\}$ and those of the form (x, y) for some $x, y \in V$. If only directed arcs are in a graph then there is a natural order of nodes that can be established.

Definition 2.1.3 (Directionality)**.** Let a graph $\mathscr{G} = (V, A)$.

- Edges of the form (x, y) for some $x, y \in V$ are called **directed arcs** and denoted as:
$$x \to y$$
Furthermore, if $A \subset \{(x, y); x, y \in V, x \neq y\}$, then the graph $\mathscr{G}$ is called **directed**.
- Edges of the form $\{x, y\}$ for some $x, y \in V$ are called **undirected arcs** and denoted as
$$x - y$$
Furthermore, if $A \subseteq \{\{x, y\}; x, y \in V, x \neq y\}$, then the graph $\mathscr{G}$ is called **undirected**.

(a) A directed graph. **(b)** An undirected graph.

Figure 2.1: Two examples of graphs.

In figure 2.1 we can see two examples of graphs, one directed and one undirected.

Graphs in essence depict the way different elements are connected to each other. Consequently it is of interest to look at the different ways two nodes of the graph are connected. This is done through the notions of paths and trails.

Definition 2.1.4 (path, trail)**.** Let a graph $\mathscr{G} = (V, A)$.

- A sequence of pairwise distinct vertices $(v_1, \ldots, v_n)$, $v_1, \ldots, v_n \in V, n > 1$ is called a **path** if $v_i \in V \ \forall i \in \{1, \ldots, n\}$ and $(v_i, v_{i+1}) \in A$, $\forall i \in \{1, \ldots, n-1\}$. A path is also denoted as:
$$v_1 \to v_2 \to \cdots \to v_n$$
- A sequence of pairwise distinct vertices $(v_1, \ldots, v_n)$, $v_1, \ldots, v_n \in V, n > 1$ is called a **trail** if $v_i \in V \ \forall i \in \{1, \ldots, n\}$ and either $(v_i, v_{i+1}) \in A$ or $(v_{i+1}, v_i) \in A$, $\forall i \in \{1, \ldots, n-1\}$. A trail is also denoted as:
$$v_1 \leftrightharpoons v_2 \leftrightharpoons \cdots \leftrightharpoons v_n$$

In either of the previous definitions, n is the **length** of the path or trail.

For undirected graphs, the definitions of trails and paths are equivalent to each other and identical to those for directed graphs.

For instance, in figure 2.1a $v_1 \to v_2 \to v_3 \to v_5$ is a path while $v_3 \to v_5 \leftarrow v_4$ is not, because the direction of path $v_4 \to v_5$ is violated. On the other hand, in the same graph the trail $v_3 \leftrightharpoons v_5 \leftrightharpoons v_4$ is viable. While paths need to follow the direction of arcs, trails have no such limitation and only need there to be an arc between two nodes for the path to continue, even if the direction is opposite to that of the arc.

Remark. It should be noted that there cannot be two adjacent identical nodes. This stems from the restriction in definition 2.1.1 which implies $(v, v), \{v, v\} \notin A,\ \forall v \in V$.

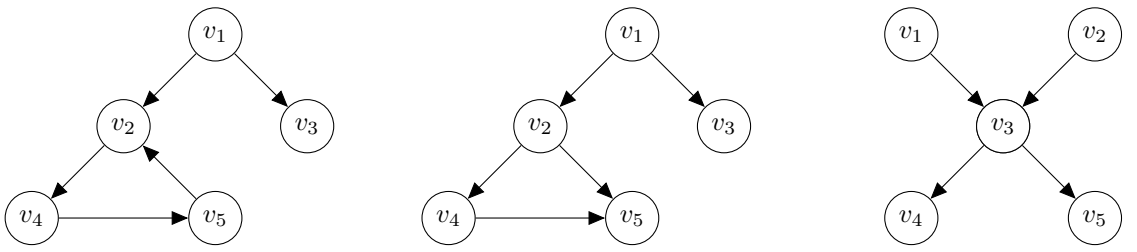
Often in the literature, trails are defined as paths that allow for multiple instances of the same node to appear. However we opt to follow the definition of trails used in Daphne Koller [18] instead.

Directed graphs offer a sense of direction and thus an order between adjacent nodes is introduced. Due to the fact that this thesis focuses mostly on directed graphs, for the following we will be assuming that the graphs are directed.

Definition 2.1.5 (Adjacent, Parent, Child, Ancestors and Descendants nodes)**.** Let a directed graph $\mathscr{G} = (V, A)$ and nodes $v_1, v_2 \in V$.

- If $v_1 \to v_2$ or $v_1 \leftarrow v_2$, then we say that the nodes $v_1,\ v_2$ are **adjacent** and write $v_1 \leftrightharpoons v_2$.
- If $v_1 \to v_2$ then v_1 is a **parent** of v_2 and v_2 is a **child** of v_1. The sets of parents and children of a node v are denoted as $pa(v)$ and $ch(v)$ respectively.
- If there is a path from v_1 to v_2 then we say that v_1 is an **ancestor** of v_2 and v_2 is a **descendant** of v_1. The sets of ancestors and descendants of a node v are denoted as $an(v)$ and $de(v)$ respectively.

For example, in the graph represented by figure 2.1a, v_1 is a parent of v_3 and an ancestor of v_5, while v_5 is a descendant of v_1 and a child of v_2. Also, we can see the following sets: $pa(v_1) = \emptyset,\ pa(v_2) = \{v_1\},\ ch(v_1) = \{v_2, v_3, v_4\},\ an(v_5) = \{v_1, v_2, v_3, v_4\},\ de(v_1) = \{v_2, v_3, v_4, v_5\}$.



(a) Example of a graph that contains the cycle (v_2, v_4, v_5, v_2).

(b) Example of a graph that is acyclical but contains the loop: (v_2, v_4, v_5, v_2).

(c) Example of a polytree

Figure 2.2: Examples illustrating loops and cycles.

Definition 2.1.6 (Cycle, Chord, Loop, Polytree)**.** Consider a graph $\mathscr{G} = (V, A)$.

- A path $(v_1, \ldots, v_n),\ v_1, \ldots, v_n \in V, n > 1$ is called a **cycle** if $v_1 = v_n$. A graph $\mathscr{G}$ is called **acyclic** if it contains no cycles.
- Let a trail $(v_1, \ldots, v_n),\ v_1, \ldots, v_n \in V, n > 1$. An arc $v_i \leftrightharpoons v_j$ between two non-consecutive nodes in the trail is called a **chord** of that trail.
- A trail $(v_1, \ldots, v_n),\ v_1, \ldots, v_n \in V, n > 1$ is called a **loop** if $v_1 = v_n$. A graph $\mathscr{G}$ is called **loopy** if it contains loops.
- A graph $\mathscr{G}$ that does not contain loops is called a **polytree**.

It is noted that a cycle is by definition also a loop. This means that a polytree is by extension also an acyclic graph. The converse is not true however. There can be examples of acyclic graphs that are in fact loopy. Figure 2.2 illustrates three examples of networks of which one contains a cycle, one is acyclic but loopy and one is a polytree.

We can distinguish different types of connections between nodes.

Definition 2.1.7 (Types of Connections)**.** Let $\mathscr{G} = (V, A)$ be a directed graph and $v_1 \leftrightharpoons v_2 \leftrightharpoons v_3$ be a trail in $\mathscr{G}$. Then we say that the trail is a:

- **serial connection** if either $v_1 \to v_2 \to v_3$ or $v_1 \leftarrow v_2 \leftarrow v_3$.
- **diverging connection** if $v_1 \leftarrow v_2 \to v_3$.
- **converging connection** if $v_1 \to v_2 \leftarrow v_3$.

For instance, in Figure 2.2b:

- $v_1 \leftrightharpoons v_2 \leftrightharpoons v_4$ is a serial connection $(v_1 \to v_2 \to v_4)$.
- $v_2 \leftrightharpoons v_1 \leftrightharpoons v_3$ is a diverging connection $(v_2 \leftarrow v_1 \to v_3)$.
- $v_3 \leftrightharpoons v_5 \leftrightharpoons v_4$ is a converging connection $(v_2 \to v_5 \leftarrow v_4)$.

One of the most fundamental aspects of Bayesian Networks is the dependence reasoning these connections offer which requires that cycles not be present in the graph. Hence our family of graphs is restricted further to directed and additionally, acyclic graphs.

Definition 2.1.8 (Directed Acyclic Graphs)**.** A graph $\mathscr{G}$ is called a Directed Acyclic Graph (DAG) if it is both directed and acyclic.

Besides a parent-child, ancestor-descendant hierarchy of the nodes, it is also important to introduce an order on the nodes of the graph. It makes sense that a "good" order follow the previously mentioned hierarchy of the nodes, and thus we come to the following definition:

Definition 2.1.9 (Well-ordering)**.** Let $\mathscr{G} = (V, A)$ be a DAG. A **well-ordering** of $\mathscr{G}$ $<$ is a total order in V for which:

$$v_1 \to v_2 \in A \Rightarrow v_1 < v_2,\ \forall v_1, v_2 \in V$$

The well ordering of a DAG is a very useful notion, but in further sections an order of the parents of each individual node will also be needed.

Definition 2.1.10 (Parental ordering)**.** Let $\mathscr{G} = (V, A)$ be a DAG and $v \in V$. A parental order $<_v$ of v is a total order on the parental set $pa(v)$. For $v' \in V$ we also define the following two sets:

$$pa(v \downarrow v') = \{x \in pa(v) : x <_v v'\}$$
$$pa(v \uparrow v') = \{x \in pa(v) : v' <_v v\}$$

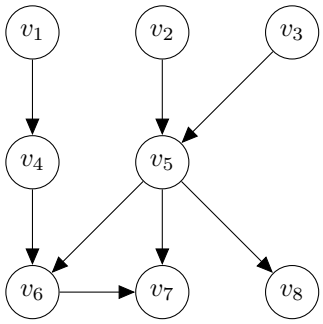


Figure 2.3: Example of a DAG illustrating d-separations.

An important property of BNs is that its nodes can be divided on subsets and one can study whether two sets are separated by the third one.

Definition 2.1.11 (d-separation)**.** Let $\mathscr{G} = (V, A)$ be a DAG and $A, V, K \subset V$ disjoint. We say that K d-separates A and B and write $d - sep_{\mathscr{G}}(A, B|K)$ if for every trail $v_1 \leftrightharpoons v_2 \leftrightharpoons \cdots \leftrightharpoons v_n$ with $v_1 \in A, v_n \in B$ there exists a node $v_i,\ i \in \{2, \ldots, n-1\}$ such that either of the following holds:

- $v_{i-1} \rightarrow v_i \leftarrow v_{i+1}$ (converging connection) and $(\{v_i\} \cup de(v_i)) \cap K = \emptyset$.
- the connection at v_i is not converging and $v_i \in K$.

Each trail that satisfies one of the above conditions is said to be blocked by K.

If $B = \emptyset$ then $d - sep_{\mathscr{G}}(A, \emptyset|K)$ for all $A, K \subset V$.

For instance, in the DAG seen in figure 2.1a, a few (not all) d-separations are:

$$d - sep_{\mathscr{G}}(\{v_1\}, \{v_6\} \,|\, \{v_4\}) \qquad d - sep_{\mathscr{G}}(\{v_1, v_4\}, \{v_2, v_3\} \,|\, \emptyset)$$
$$d - sep_{\mathscr{G}}(\{v_7\}, \{v_8\} \,|\, \{v_5\}) \qquad d - sep_{\mathscr{G}}(\{v_2, v_3\}, \{v_1, v_4, v_6, v_7, v_8\} \,|\, \{v_5\})$$

2.2. Bayesian Networks

BNs are composed of a DAG representing the relationships in the distribution specified by the BN and the set of conditional probability tables (for discrete BNs) or the set of conditional densities (for continuous BNs) of all nodes given their parents.

Definition 2.2.1 (Bayesian Network)**.** A Bayesian network (BN) is a pair comprising of:

- A DAG $\mathscr{G} = (V, A)$ where the nodes in V represent random variables $X_v,\ v \in V$, and the arcs represent direct dependencies that abide by the d-separations in the DAG.

- A collection $\left\{f_{v\,|\,pa(v)} : \forall v \in V\right\}$ of conditional density/mass functions of nodes given their parents.

The DAG provides the information about conditional independencies in the joint distribution represented by the BN, through the concept of d-separation as presented in definition 2.1.11. This result is crucial and can be formalized as: $V_1, V_2, V_3 \subseteq V$ and $\boldsymbol{X_1}, \boldsymbol{X_2}, \boldsymbol{X_3}$ being the random vectors of the variables corresponding to the nodes in V_1, V_2, V_3 respectively:

$$d - sep_{\mathscr{G}}(V_1, V_2|V_3) \iff \boldsymbol{X_1} \perp\!\!\!\perp \boldsymbol{X_2}|\boldsymbol{X_3}$$

When examining the dependencies of a node given other nodes it often suffices to consider a subset of nodes of the BN. Such subsets are called Markov Blankets.

Definition 2.2.2 (Markov Blanket)**.** Let a BN $\mathscr{G} = (V, A)$ and a variable $X \in V$. A **Markov Blanket** of X in a set $V_1 \subset V\backslash\{X\}$ is a subset $V_2 \subseteq V_1$ such that:

$$d - sep(X, V_1\backslash V_2|V_2).$$

If the Markov Blanket of a node X is minimal, then it is referred to as the Markov Boundary of that node.

Definition 2.2.3 (Markov Boundary)**.** Let a BN $\mathscr{G} = (V, A)$ and a variable $X \in V$. A **Markov Boundary** of X for $V_1 = V\backslash\{X\}$ is a Markov Blanket V_2 of X in V_1 such that for every other Markov Blanket V_3 of X in V_1:

$$V_2 \subseteq V_3.$$

It is well known (see e.g. [57]) that the Markov Boundary of a node X is the set of the parents and children of X, as well as all the other parents of X's children:

$$pa(X) \cup ch(X) \cup \bigcup_{Y \in ch(X)} (pa(Y)\backslash\{X\})\,.$$

In what follows, we shall refer to a BN by $\mathscr{G}$, where $\mathscr{G}$ is the DAG of the BN. There is a clear, one-to-one relationship between the nodes of the graph and the variables. The graph represents variables and describes the conditional independencies in the joint distribution of these random variables. We will then treat vertices as the variables themselves, meaning that the vertex set will be of the form $V = \{X_i, i = 1, \ldots, n\}$ and the conditional densities needed to specify the BN will be denoted by $f_{X_i|pa(X_i)}$.

For the sake of simplicity, whenever the variable in V is X_i, then the joint/conditional densities $f_{X_i|X_j}(x_i, x_j)$ will instead be denoted as $f_{i|j}(x_i, x_j)$.

The joint density of all the variables of the BN, can be factorized as the product of all the conditional densities of nodes given their parents.

Proposition 2.2.4. *Let a BN $\mathscr{G} = (V, A)$. Then:*

$$f_{\boldsymbol{X}}(\boldsymbol{x}) = \prod_{v \in V} f_{X_v\,|\,pa(v)}(x_v \,|\, \boldsymbol{x_{pa(v)}})$$

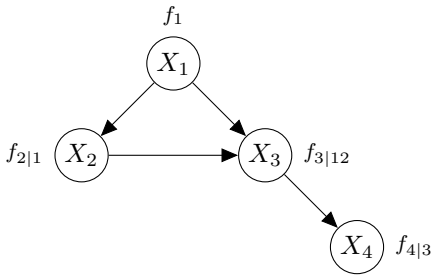


Figure 2.4: A simple example of a BN.

Similarly, the probability mass function corresponding to the discrete BN is computed.

Using the proposition 2.2.4, we can write the density represented by the DAG in figure 2.4:

$$f_{1234}(x_1, x_2, x_3, x_4) = f_1(x_1) f_{2|1}(x_2|x_1) f_{3|12}(x_3|x_1, x_2) f_{4|3}(x_4|x_3)$$

A simple and popular family of Bayesian Networks is **Discrete Bayesian Networks**. As the name suggests, in discrete BNs, all variables follow discrete distributions with finite supports. The conditional distributions of nodes given parents conditional probability tables. For $X \in V$, the conditional probability table corresponding to that variable takes the form of:

$$\left[\mathbb{P}(X = x|pa(X) = \boldsymbol{x}_{pa(X)})\right]_{x \in \mathcal{X}}$$

An example of a Discrete BN is portrayed in figure 2.5 where every conditional density has been replaced with its respective conditional probability table.

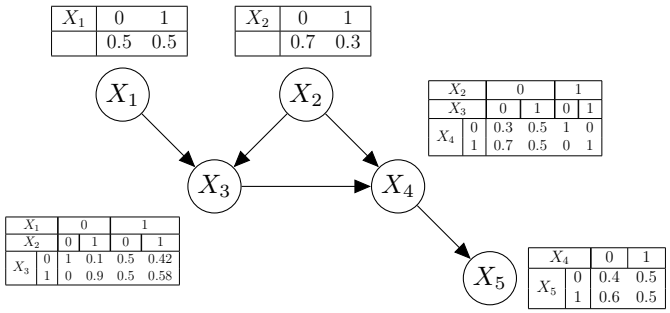


Figure 2.5: A discrete BN and its conditional probability tables.

Another family of BNs with nodes that are continuous is the **Gaussian Bayesian Network (GBN)** model. This model assumes that all conditional distributions in the network $f_{X|pa(X)}$ are linear Gaussian distributed, which means that:

$$(X|\boldsymbol{pa(X)} = \boldsymbol{y}) \sim \mathcal{N}(\beta_0 + \boldsymbol{\beta^T y},\ \sigma^2)$$

As a consequence, in figure 2.4 all conditional densities are Gaussian with a constant variance and a linear regression of the parent variables as the mean. One example for

the BN of the same figure could be:

$$
\begin{aligned}
&f_1(x_1) = \phi(x_1|0,1)\\
&f_{2|1}(x_2|x_1) = \phi(x_2|0.5+2x_1,0.25)\\
&f_{3|12}(x_3|x_1,x_2) = \phi(x_3|-1+0.2x_1-0.7x_2,0.81)\\
&f_{4|3}(x_4|x_3) = \phi(x_4|-0.5x_3,1)
\end{aligned}
$$

A crucial property of GBNs is that all joint distributions in the network are Normally distributed. We will discuss this property in greater detail along with its implications in the next chapter, where we will be examining inference problems.

2.3. Pair Copula Bayesian Networks

The popularity of Discrete and Gaussian families of BNs is attributed to their attractive properties. The computations in these models are relatively simple, as all marginal and conditional distributions in these families belong to the same family. Both Discrete and Gaussian BNs are characterized by the availability of efficient and scalable methods for probabilistic reasoning and inference.

However, the assumption that all nodes are discrete or Gaussian is very restrictive.

The Pair Copula Bayesian Network is a model that utilizes bivariate copulas in order to relax the Gaussianity assumption for continuous BNs. Bivariate Copulas are a rich family of distributions and a powerful statistical tool used for dependence modeling between variables. Through clever use of copulas, it is possible to decompose higher-dimensional distributions into products of bivariate (conditional) copulas.

2.3.1. Copulas

Below we present the definition of a copula.

Definition 2.3.1 (Copula)**.** A **copula** $C : [0,1]^d \to [0,1]$ is a joint CDF of a d-dimensional random vector with uniformly distributed marginal distributions. The density of the copula is c.

The use of copulas for modeling dependence between non-uniformly distributed variables is motivated by the ability to transform any continuous variable into a uniform variable through the use of its CDF function.

Theorem 2.3.2. *Let X be a random variable with a continuous CDF F. Then:*

$$F(X) \sim \mathcal{U}(0,1).$$

As a consequence, for a random variable with a continuous CDF F, the (generalized) inverse F^{-1} of the CDF is well defined and for $V \sim \mathcal{U}(0,1)$:

$$F^{-1}(V) \sim F.$$

For a random variable X_i we denote its transformed counterpart as $U_i = F_i(X_i)$.

The ability to transform a random variable to a uniform variable is of paramount importance for the application of copulas in practice. The following theorem connects the definition of copulas with the general notion of joint CDFs irrespective of marginal distributions.

Theorem 2.3.3 (Sklar's Theorem)**.** *If $F : \mathbb{R}^d \to [0,1]$ is a joint CDF of a d-dimensional random vector, there exists a copula $C : [0,1]^d \to [0,1]$ such that:*

$$F(x_1, \ldots, x_d) = C(F_1(x_1), \ldots, F_d(x_d)).$$

In this case we will say that $X_1, \ldots, X_d$ are joined by copula C.

If additionally X_i are continuous variables, then the copula C from Sklar's theorem is unique.

With Sklar's theorem we can model the dependence between the variables independently from their marginal distributions.

The theorem can be easily rewritten for joint densities, giving the relationship between the joint density and the joint copula density.

Corollary 2.3.4. *Let $F : \mathbb{R}^d \to [0,1]$ be an absolutely continuous joint CDF of a d-dimensional random vector and $C : [0,1]^d \to [0,1]$ the copula CDF with density c. If f is the joint PDF of F and f_i the marginal PDFs of F, then we have:*

$$f(x_1, \ldots, x_d) = c(F_1(x_1), \ldots, F_d(x_d)) \prod_{i=1}^{d} f_i(x_i).$$

Furthermore, we denote conditional copula CDFs and densities in the following way:

$$C_{i|j}(u_i|u_j) = \mathbb{P}(U_i \leq u_i \,|\, U_j = u_j),$$
$$c_{i|j}(u_i|u_j) = \frac{\partial}{\partial u_i} C_{i|j}(u_i|u_j).$$

Bivariate copula models have been rigorously studied and have a rich literature attributed to them [42, 48, 47, 16, 40], and thus gained a lot of attention in many fields, including finance [12].

Motivated by their modeling flexibility, incorporation of bivariate copulas in the conditional probabilities in BNs is a natural step towards copula based BN modeling. To accomplish this, it is essential to be able to decompose the expression of a multivariate density function into one using only bivariate copula densities. More specifically, given a node X in the network, we wish to decompose the conditional density:

$$f_{X|pa(X)}(x|\boldsymbol{x}_{pa(X)})$$

For example, in figure 2.6, we wish to decompose the density $f_{4|123}$.

This is achieved by assigning copulas to arcs of the graph such that one of them is unconditional, then the next one is conditional on the first variable, the third conditional

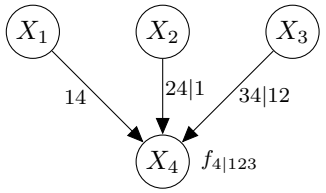


Figure 2.6: An example illustrating a Bivariate copula decomposition of conditional densities.

on the first two etc. Joe [41] and Bedford and Cooke [5, 6]. This process requires specification of the parental ordering $<_X$ of a variable X and assigning copulas according to this order.

In the example of figure 2.6, we show a decomposition the density of interest,

$$f_{4|123}(x_4 \,|\, x_1, x_2, x_3).$$

Given the parental ordering $<_4$ such that $X_1 <_4 X_2 <_4 X_3$ the factorization is:

$$f_{4|123}(x_4 \,|\, x_1, x_2, x_3) = f_4(x_4)c_{14}(u_1, u_4)c_{24|1}(u_{2|1}, u_{4|1}|u_1)c_{34|12}(u_{3|12}, u_{4|12}|u_1, u_2),$$

where, based on the conditional independencies of the graph:

$$\begin{aligned} & u_{2|1} \coloneqq C_{2|1}(u_2|u_1), \\ & u_{4|1} \coloneqq C_{4|1}(u_4|u_1), \\ & u_{3|12} \coloneqq C_{3|12}(u_3|u_1, u_2) \\ \text{and} \quad & u_{4|12} \coloneqq C_{4|12}(u_4|u_1, u_2). \end{aligned}$$

For the general case, the following decomposition of the conditional density of a node given its parents holds:

$$f_{X|pa(X)}(x|\boldsymbol{x}_{pa(X)}) = f_X(x) \prod_{Y \in pa(X)} c_{Y,X|pa(X\downarrow Y)}(u_{Y|pa(X\downarrow Y)}, u_{X|pa(X\downarrow Y)}|\boldsymbol{x}_{pa(X\downarrow Y)}). \tag{2.1}$$

where:

$$\begin{aligned} & u_{Y|pa(X\downarrow Y)} \coloneqq C_{Y|pa(X\downarrow Y)}(y|\boldsymbol{x}_{pa(X\downarrow Y)}) \\ \text{and} \quad & u_{X|pa(X\downarrow Y)} \coloneqq C_{X|pa(X\downarrow Y)}(x|\boldsymbol{x}_{pa(X\downarrow Y)}). \end{aligned}$$

As shown in the network of figure 2.6, we will denote the assignment of conditional copula $C_{ij|pa(i\downarrow j)}$ by writing $ij|pa(i \downarrow j)$ next to the arc it corresponds to.

Equation 2.1 implies that only copulas $c_{Y,X|pa(X\downarrow Y)}$ are required for decomposing the conditional densities of a BN into distinct bivariate copula models. Each of these models corresponds to an arc in the graph, which is a very intuitive and elegant property that makes PCBNs all the more attractive.

In the calculation of each conditioned copula density, the calculation of conditional margins is required.

In the presented example of figure 2.6 all v-structures are uncoupled, meaning that there is no arc between the parents, which in-turn implies the independence between any two parents of X_4. Because of this, all copulas C_{12}, C_{13}, C_{23} can be assumed to be the independence copula. With this in mind, the u-values are calculated as follows:

- $u_{2|1} = u_2$ due to the independence between U_1 and U_2.
- $u_{4|1} = C_{4|1}(u_4|u_1)$ is computed using copula C_{14}.
- $u_{3|12} = u_3$ due to the independence between U_1, U_2 and U_3.
- $u_{4|12} = \frac{\partial}{\partial u_{2|1}} C_{24|1}(u_{2|1}, u_{4|1}|u_1)$.

In the calculation of $u_{4|12}$ we needed to compute the derivative of copula $C_{24|1}$ with respect to the margin $u_{2|1}$. As we saw, the arguments of that copula were also conditional margins, creating a recursion of such calculations. The definition of h-functions help generalize this recursive calculation of conditional margins.

Definition 2.3.5 (h-functions)**.** The first partial derivatives of a bivariate copula $C(u, v)$ are called **h-functions**:

$$h_{\underline{U},V}(u, v) = \frac{\partial C(u, v)}{\partial v} \quad \text{and} \quad h_{U,\underline{V}}(u, v) = \frac{\partial C(u, v)}{\partial u}.$$

Therefore, for copula C joining the random variables U, V:

$$h_{\underline{U},V}(u, v) = C(u|v),$$
$$h_{U,\underline{V}}(u, v) = C(v|u).$$

To extend h-functions for conditioned copulas and to illustrate how any given conditional u value is calculated, if K is the conditioning set we have for each $Y \in K$ and $X \notin K$:

$$u_{X|K} = F_{X|K}(x|\boldsymbol{x}_K) = h_{\underline{X},Y|K_{-Y}}(F_X(x|\boldsymbol{x}_{K_{-Y}}), F_Y(y|\boldsymbol{x}_{K_{-Y}})).$$

Going back to the example in figure 2.6, let us depict the calculation of $u_{3|12}$. By definition we have:

$$u_{4|12} = F_{4|12}(x_3 \,|\, x_1, x_2) = h_{\underline{4},2|1}(u_{4|1}, u_{2|1}|u_1)$$

where:

$$u_{4|1} = h_{\underline{4},1}(u_4, u_1),$$
$$u_{2|1} = h_{\underline{2},1}(u_2, u_1).$$

The values u_1, u_2, u_4 are known and the h-functions $h_{\underline{4},2|1}, h_{\underline{4},1}, h_{\underline{2},1}$ are the conditional CDF's corresponding to the copulas $C_{42|1}, C_{14}$ and the independence copula C_{12}. If we had an arc between X_1 and X_2 with a corresponding copula C_{12} we would use that copula instead for the calculation of the h-function $h_{\underline{2},1}$

2.3.2. The model

Having stated the decomposition of conditional densities into bivariate copula densities along with h-functions, we can proceed with the formal introduction of Pair-Copula Bayesian Networks.

Definition 2.3.6 (Pair-Copula Bayesian Network)**.** A Pair-Copula Bayesian Network (PCBN) is the collection $(\mathscr{G}, \mathscr{O}, \mathscr{F}, \mathscr{C})$ where:

- $\mathscr{G} = (V, A)$ is a DAG.
- $\mathscr{O} = \{<_v : v \in V\}$ is the collection of all parental orders of $\mathscr{G}$.
- $\mathscr{F} = \{f_v : v \in V\}$ is the collection of the marginal distributions of all variables in the network.
- $\mathscr{C} = \{C_{uv|pa(v\downarrow u)} : u \to v \in A\}$ is the collection of copulas that are assigned to each arc in A.

In contrast to the general case of BNs where individual arcs only indicate a parent-child relationship, PCBNs specifically model the relationship between a node and each of its parents.

Let us consider the graph structure of the BN of the made-up example in figure 1.1. The variables are now continuous and we will use this as a PCBN presented in figure 2.7 to identify the forming of a hurricane in a specific geographic location. The relevant variables are the current month (X_1), Hurricane formation (X_2), sea surface temperature (X_3), atmospheric pressure (X_4), upper-level steering wind speed (X_5) and atmospheric moisture (X_6).

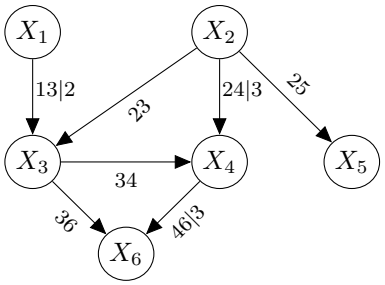


Figure 2.7: PCBN on hurricane formation prediction.

In this example, the density of the PCBN is given by:

$$f(x_1, \ldots, x_6) = \prod_{i=1}^{6} f_i(x_i) c_{23}(u_2, u_3) c_{25}(u_2, u_5) c_{34}(u_2, u_4) c_{36}(u_3, u_6)$$
$$c_{13|2}(u_{1|2}, u_{3|2}|u_2) c_{24|3}(u_{2|3}, u_{4|3}|u_3) c_{46|3}(u_{4|3}, u_{6|3}|u_3).$$

For estimation purposes, the so called **simplifying assumption** is in practice assumed to hold in equation 2.1, ignoring the term $\boldsymbol{x}_{pa(X\downarrow Y)}$ and turning the equation to:

$$f_{X|pa(X)}(x|\boldsymbol{x}_{pa(X)}) = f_X(x) \prod_{Y \in pa(X)} c_{Y,X|pa(X\downarrow Y)}(u_{Y|pa(X\downarrow Y)}, u_{X|pa(X\downarrow Y)}). \tag{2.2}$$

This assumption however, is not required for the work of our thesis and we will not incorporate it further.

The copulas in $\mathscr{C}$ are said to be **specified** by the PCBN. Copulas specified by the PCBN include, besides the copulas in $\mathscr{C}$, the independence copulas that are implied by the d-separations in graph $\mathscr{G}$. For example in figure 2.7, copulas $C_{23}, C_{46|3} \in \mathscr{C}$ and the independence copula C_{12} are all specified by the PCBN. On the other hand, the copula C_{45} is not specified by the PCBN.

Remark. In order to be able to effectively work with a BN, we need to have an analytic expression of the conditional densities of nodes given their parents to be able to calculate the network's density. However, it has been shown that there are two families of network structures that, if present, conditional density calculation is impossible in a PCBN without the need of integration. This is due to the reliance on recursive calculation of h-functions which, in some structures requires calculation of copulas that are not specified by the PCBN.

2.3.3. Restricted PCBNs

Horsman [36] has shown that there are only two problematic structures that always necessitate integration. It is proven that no such structure is present if and only if there exists a collection of parental orders $\mathscr{O}$ for which calculating the density of a network without integration is possible. Moreover, an algorithm is presented that allows to choose the parental order which avoids integration in density evaluation or sampling.

The first problematic structure is called an active cycle.

Definition 2.3.7 (Active Cycle)**.** Let $\mathscr{G} = (V, A)$ be a DAG. If the loop $x \leftarrow v_1 \leftrightharpoons \cdots \leftrightharpoons v_n \rightarrow x$ is present in $\mathscr{G}$ for $n > 2$, we call this path an **active cycle** if the following conditions hold:

1. all connections in $(v_1, \ldots, v_n)$ are either diverging or serial.
2. $(x, v_1, \ldots, v_n, x)$ contains no chords

The second structure is called an interfering v-structure.

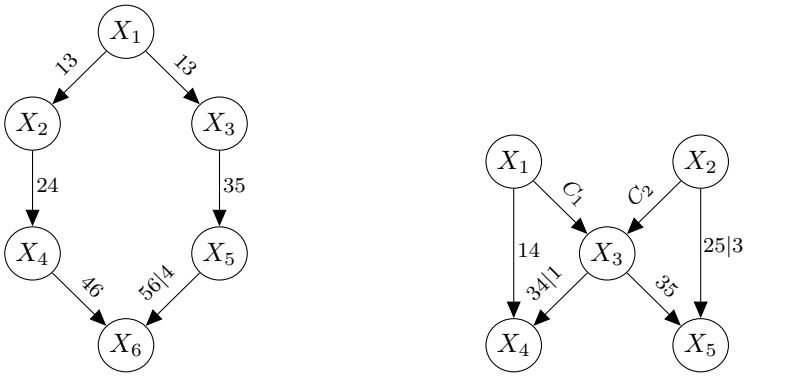
Definition 2.3.8 (Interfering v-structures)**.** Let $\mathscr{G} = (V, A)$ be a DAG. Five nodes $v_1, v_2, v_3, v_4, v_5 \in V$, are said to form an **interfering v-structure** if:

- $\{v_3, v_4\} \in pa(v_1)$ and $\{v_4, v_5\} \in pa(v_2)$
- $\{v_3, v_5\} \in pa(v_4)$

Both structures can be seen in figure 2.8. In the active cycle of figure 2.8a, in order to calculate the density of the network we would need to calculate the decomposed conditional density:

$$f_{6|45}(x_6|x_4, x_5) = f_6(x_6)c_{46}(u_4, u_6)c_{56|4}(u_{5|4}, u_{6|4}|u_4)$$

However, for the calculation of $u_{5|4} = h_{\underline{5},4}$ we need the copula C_{45} which is not specified by the PCBN. In order to evaluate this copula using the information specified in $\mathscr{C}$ we would need to resort to integration. Integration for calculating the density is something we wish



(a) Active Cycle. **(b)** Interfering V-structure.

Figure 2.8: Problematic structures.

to avoid, however. Having a ready, analytic expression of the density of the PCBN is an essential condition for ensuring computational efficiency of PCBN applications.

In the interfering v-structure of figure 2.8b, the reasoning is similar to that of the active cycle. In order to assess which of the copulas C_1 or C_2 is unconditional, we must focus on the v-structures formed at the nodes X_4 and X_5. In either of their conditional density decompositions we have a copula density containing both of their respective parents: $c_{34|1}$ and $c_{25|3}$.

For the evaluation of these two densities the following conditional marginals need to be calculated through h-function recursion:

$$u_{3|1} = h_{1,\underline{3}}(u_1, u_3),$$
$$u_{2|3} = h_{\underline{2},3}(u_2, u_3).$$

This means that both $C_1 = C_{13}$ and $C_2 = C_{23}$ are required. However, in a PCBN both copulas can't be specified due to one of the two being conditioned. If for instance we choose $C_1 = C_{13}$ and $C_2 = C_{23|1}$, then we would need to calculate $C_{23|1}$ though integration.

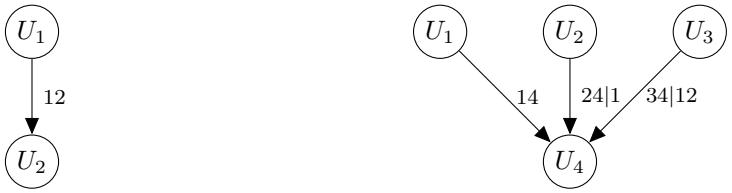
Therefore, in order to be able to calculate all conditional margins and in extent the PCBN's density without integrations, we restrict the model to **restricted DAG**s, meaning DAGs that do not feature either active cycles or interfering v-structures.

If a PCBN has a restricted DAG, and the collection of parental orderings $\mathscr{O}$ allows for the direct calculation of all conditional margins, then it is called a **restricted PCBN**.

2.3.4. PCBN Sampling

A basic utility of PCBNs is the ability to simulate the PCBN by obtaining samples of the nodes of the network. These samples showcase the dependence properties implied by the copulas in $\mathscr{C}$ and the dependence structure of the DAG.

First we will illustrate how one may sample from simpler copula structures. In the most primitive scenario, we are interested in sampling from a single copula, conditioned on the observation of one of its margins. Let us denote this copula as $C_{12}(u_1, u_2)$ and let



(a) Simple Copula Sampling.

(b) Sampling a node from its parents.

Figure 2.9: Copula sampling examples.

U_1 and U_2 be the variables being joint by C_{12}. We can see an representation of this dependence structure in figure 2.9a. We wish to sample from the distribution:

$$U_1|U_2 = u_2$$

We may do so by introducing a uniform sample $U \sim \mathcal{U}(0,1)$ and obtain our target sample as:

$$\hat{u}_1 = C_{12}^{-1}(U|u_2) = h_{\underline{12}}^{-1}(U, u_2)$$

Let us now consider a variable with multiple parents whose density has been factorized according to equation 2.1, the example of figure 2.9b can be considered. In this scenario, to sample u_4 given its parents we introduce a uniform sample $U \sim \mathcal{U}(0,1)$ and obtain our target sample through the following h-function recursion:

$$\hat{u}_4 = h_{1\underline{4}}^{-1}(u_1, h_{2\underline{4}|1}^{-1}(u_{2|1}, h_{3\underline{4}|12}^{-1}(u_{3|12}, U|u_1, u_2)|u_1))$$

Note that in this simple example, due to independence we have $u_{2|1} = u_2$ and $u_{3|12} = u_3$. This may not be the case in other scenarios however.

For simulating the PCBN, we sample all roots in the network from the uniform $\mathcal{U}(0,1)$ distribution independently. Then, at each iteration we sample the next node according to the well-ordering of the PCBN, given the samples of its parents. At the end of the sampling procedure, we transform all samples to their respective marginal distributions in $\mathscr{F}$ using the corresponding quantile functions.

Let us show this using the PCBN of figure 2.7. The steps are the following:

Step 1 The PCBN has two roots, X_1, X_2. Therefore we sample $V_1, V_2 \overset{iid}{\sim} \mathcal{U}(0,1)$ and set the root samples as $\hat{u}_1 = V_1, \hat{u}_1 = V_2$.

Step 2 Next node U_3 is sampled and its parents are U_1 and U_2. Using $V_3 \sim \mathcal{U}(0,1)$, we obtain our sample from the recursion:

$$\hat{u}_3 = h_{2\underline{3}}^{-1}(u_2, h_{1\underline{3}|2}^{-1}(u_{1|2}, V_3|u_1))$$

Step 3 Next node U_4 is sampled and its parents are U_2 and U_3. Using $V_4 \sim \mathcal{U}(0,1)$, we obtain our sample from the recursion:

$$\hat{u}_4 = h_{3\underline{4}}^{-1}(u_3, h_{2\underline{4}|3}^{-1}(u_{2|3}, V_4|u_3))$$

Step 4 Next node U_5 is sampled and it has one parent U_2. Using $V_5 \sim \mathcal{U}(0,1)$, we obtain our sample from the recursion:

$$\hat{u}_5 = h^{-1}_{2\underline{5}}(u_2, V_5)$$

Step 5 Lastly, U_6 is sampled and its parents are U_3 and U_4. Using $V_6 \sim \mathcal{U}(0,1)$, we obtain our sample from the recursion:

$$\hat{u}_6 = h^{-1}_{3\underline{6}}(u_3, h^{-1}_{4\underline{6}|3}(u_{4|3}, V_6|u_3))$$

This concludes our review on PCBN simulation. There is a plethora of literature that delves into more details on copula sampling, including sampling from vine structures. For more information on copula sampling the reader can refer to Kurowicka and Cooke [46].

3

Established Methods in Bayesian Network Inference

Inference problems of BNs have been studied extensively in the past, with proposed algorithms covering a vast range of mathematical approaches and graph sub-families [54, 65]. In this section we will delve into the inference methodologies that are most relevant to our thesis and which help us highlight the advantages and disadvantages of the proposed algorithm.

Whenever there is an observation of a random variable X, we call this assignment **evidence**. We denote the set of evidence nodes as $\mathbf{E}$ and $\mathbf{e}$ as the observations of the evidence nodes. This new information involving variables of the BN entails that certain probability distributions of the remaining variables will change when conditioned on the evidence. The calculation of the conditional (on the observed nodes) joint or marginal probability distributions in the BN is called **inference**, **conditionalization** or **belief updating**.

It has been shown that the general problem of probabilistic inference in BNs is NP hard [14]. Furthermore, this result has been expanded to include the problem of approximate probabilistic inference [17]. It is thus not possible to find an "optimally efficient" algorithm for either the exact calculation or approximate inference of conditional distributions in the network. This is the reason why researching approximative algorithms is important; while exact algorithms may be impractical in many applications due to their computational complexity, approximate algorithms can offer a trade-off between efficiency and accuracy.

In the general case of BN $\mathscr{G} = (V, A)$ inference, we are interested in calculating the conditional probability/density:

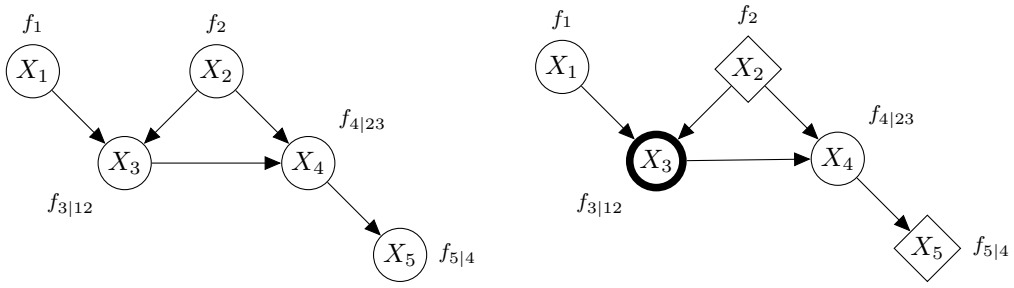
$$f_{\boldsymbol{X_Q}|\boldsymbol{X_E}}\left(\boldsymbol{x_Q}|\boldsymbol{x_E}\right)$$

Where $\boldsymbol{X_Q}, \boldsymbol{X_E}$ are distinct random vectors consisting of variables in V. This implies a partition $\{Q, I, E\}$ of V where:

- Q corresponds to the set of variables which we are interested to find a joint conditional density of, also called **query variables**.

- E corresponds to the set of **evidence variables** of which we have observed values.
- I corresponds to the set of variables which are neither query nor evidence, meaning that $I = V \backslash (Q \cup E)$. We call these variables **intermediate**.

Variables that belong to $Q \cup E$ will also be called **nodes/variables of interest**.



(a) Example of a BN.

(b) Notation of the inference problem $f_{3|2,5}$.

Figure 3.1: Figures illustrating a BN and the notation related to conditionalization.

For example, let us consider the BN structure of figure 3.1a. An example of an inference problem based on this BN could be:

$$f_{3|25}(x_3|0,1) \tag{3.1}$$

in which case $Q = \{X_3\},\ I = \{X_1, X_4\},\ E = \{X_2, X_5\}$ with $e = \{0, 1\}$.

In the graphical representation of the BN, we visualize the inference problem by assigning thicker lines to variables in Q and using diamond shapes for variables in E. The intermediate nodes, i.e. those in I are the only ones that do not change shape. We can see an example of this representation in figure 3.1b.

Yuan and Druzdzel [71] cover a great number of methodologies that are available for BN inference. The authors mention that the algorithms can be divided between those that address the task of exact inference and those that find approximate solutions. Daphne Koller [18] provides a detailed overview of some of the basic methodologies that have been studied.

The two algorithms that we will discuss are Variable Elimination and Peal's Algorithm for Belief Propagation. The choice of these algorithms is motivated by their fundamental relevance in our own algorithm for PCBN inference, which will be presented in chapter 4.

3.1. Variable Elimination

Variable Elimination is a simple, yet effective method of exact inference in a BN, first proposed by Zhang and Poole [73]. Daphne Koller [18] illustrates the sum-product algorithm of Variable elimination. Inference is conducted by marginalizing the $\boldsymbol{X_I}$ variables from the network density f_V through integration to obtain $f_{Q,E}$. The same procedure is done for the variables $\boldsymbol{X_E}$ to obtain f_E.

Then, using the definition of conditional probability we can obtain:

$$f_{Q|E}\left(\boldsymbol{x}_A|\boldsymbol{x}_E\right) = \frac{f_{Q,E}\left(\boldsymbol{x}_A, \boldsymbol{x}_E\right)}{f_E\left(\boldsymbol{x}_E\right)} \tag{3.2}$$

In the most basic approach, marginalization by integrating the density directly can become exponentially expensive for large and connected networks. Variable elimination optimizes the integration procedure by focusing on integrating only the relevant factors in the factorization of the BN density as in proposition 2.2.4. In Variable Elimination, only the factors containing the variable being eliminated are integrated.

The Variable Elimination algorithm for only one variable includes the notation of factors ϕ, functions that represent the conditional distributions corresponding to the BN factorization. For the BN factorization we have the set of factors:

$$\Phi = \{\phi_X \coloneqq f_{X|Pa(X)}\}_{X \in V}.$$

As we only need to eliminate one intermediate variable Y, we need to define a factor τ by integrating over Y from the product of all factors that contain Y.

$$\tau = \int_Y \prod_{\phi_X \in \Phi : Y \in D_X} \phi_X \, dy, \qquad \text{where } D_X = \{X\} \cup pa(X)$$

The density of the model without Y will be:

$$f_{V \setminus Y} = \tau \cdot \prod_{\phi_X \in \Phi : Y \notin D_X} \phi_X$$

By repetitive use of this algorithm we can eliminate multiple variables in a network and subsequently calculate any joint distribution and solve inference problems through equation 3.2. There are also many extensions of this algorithm such as Message Passing in Clique Trees [18] and the Clustering Algorithm [50].

3.2. Pearl's Belief Propagation Algorithm

The next algorithm that we will analyze is one of the most monumental methodologies for BN inference as it provides foundation for many other algorithms and introduces a unique approach for inference problems. This is the Belief Propagation for polytrees proposed by Pearl [57]. This algorithm is also referred to as Pearl's algorithm, named after the author.

Pearl's algorithm assumes that the DAG is a polytree, thus not containing loops. Belief Propagation treats evidence updating as signals in the network, to be propagated towards nodes of interest through the trails in the polytree. The restriction to polytrees is an essential requirement for belief propagation due to potential conflict between signals created by merging signals that take different trails to reach the same node.

In belief propagation we utilize **causal** and **diagnostic parameters**. These depict the information of the evidence arriving at the target node from either its parents or its children. The casual parameter of a node X is written as $\pi(x)$ while its diagnostic parameter as $\lambda(x)$.

We divide the evidence vector into $\boldsymbol{E}_X^+ = \boldsymbol{E} \cap an(X)$ and $\boldsymbol{E}_X^- = \boldsymbol{E} \cap de(X)$ which indicate the evidence variables that are ancestors or descendants of X respectively. Given these, the parameters can be defined by:

$$\lambda(x) = f_{\boldsymbol{E}_X^-|X}(\boldsymbol{e}_X^-|x) \quad \text{and} \quad \pi(x) = f_{X|\boldsymbol{E}_X^+}(x|\boldsymbol{e}_X^+).$$

Given these signals the conditional density that we want to arrive at can be written as:

$$f_{X|\boldsymbol{E}}(x|\boldsymbol{e}) = \alpha\pi(x)\lambda(x),$$

where α is a normalizing constant.

Let X be any node in the polytree with parents $\boldsymbol{U} = \{U_1, \ldots, U_n\}$ and children $\boldsymbol{C} = \{C_1, \ldots, C_m\}$. We can further decompose the causal and diagnostic parameters into conditional probabilities corresponding to each of the children and the parents.

First we decompose the evidence into evidences that arrive to X from each of the parents and children as $\boldsymbol{E}_X^- = \{\boldsymbol{E}_{X,C_1}^-, \ldots, \boldsymbol{E}_{X,C_m}^-\}$ and $\boldsymbol{E}_X^+ = \{\boldsymbol{E}_{U_1,X}^+, \ldots, \boldsymbol{E}_{U_n,X}^+\}$, respectively. Then, we can define the factors of the target decomposition which are called **causal** and **diagnostic messages**:

- The diagnostic message that each child C_i sends to X is:

$$\lambda_{C_i}(x) = f_{\boldsymbol{E}_{X,C_i}^-|X}(\boldsymbol{e}_{X,C_i}^-|x).$$

- The causal message that each parent U_i sends to X:

$$\pi_X(u_i) = f_{U_i|\boldsymbol{E}_{U_i,X}^+}(u_i|\boldsymbol{e}_{U_i,X}^+).$$

A graphical representation of messages being sent between X and its children and parents is illustrated in figure 3.2.

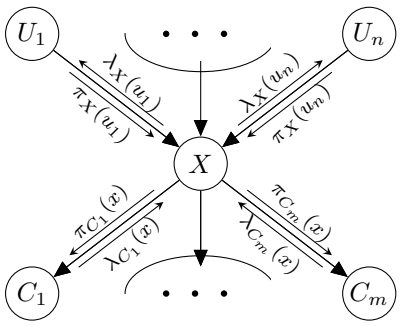


Figure 3.2: Parents and children of X.

The decomposition of the diagnostic parameter as an expression of messages is:

$$\lambda(x) = f_{\boldsymbol{E}_X^-|X}(\boldsymbol{e}_X^-|x) = \prod_{i=1}^{m} f_{\boldsymbol{E}_{X,C_i}^-|X}(\boldsymbol{e}_{X,C_i}^-|x) = \prod_{i=1}^{m} \lambda_{C_i}(x). \tag{3.3}$$

On the other hand, the decomposition of the causal parameter is:

$$\begin{aligned}\pi(x) &= f_{X|\boldsymbol{E}_X^+}(x|\boldsymbol{e}_X^+) \\ &= \int_{U_1,\dots,U_n} f_{X|pa(X)}(x|\boldsymbol{u}) f_{pa(X)|\boldsymbol{E}^+_{U_1,X},\dots,\boldsymbol{E}^+_{U_n,X}}(\boldsymbol{U}|\boldsymbol{e}^+_{U_1,X},\dots,\boldsymbol{e}^+_{U_n,X})\, du_1 \dots du_n \\ &= \int_{\boldsymbol{U}} f_{X|pa(X)}(x|\boldsymbol{u}) \prod_{i=1}^{n} \pi_X(u_i)\, d\boldsymbol{u}.\end{aligned} \tag{3.4}$$

Subsequently, the conditional density incorporating both types of information is:

$$f_{X|\boldsymbol{E}}(x|e) = \alpha \left(\prod_{i=1}^{m} \lambda_{C_i}(x) \right) \left(\int_{\boldsymbol{U}} f_{X|pa(X)}(x|\boldsymbol{u}) \prod_{i=1}^{n} \pi_X(u_i)\, d\boldsymbol{u} \right)$$

Each of the messages requires derivation. To keep notation clear, for updating we always denote as X the node that sends the message. For calculating the diagnostic and causal messages, it is derived that:

$$\pi_{C_i}(x) = \alpha_1 \prod_{k \neq i} \lambda_{C_k}(x)\pi(x), \tag{3.5}$$

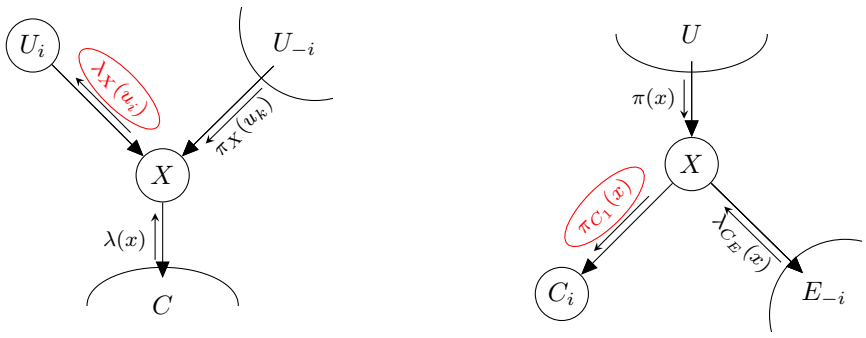
$$\lambda_X(u_i) = \alpha_2 \int_X \lambda(x) \int_{\boldsymbol{U}\setminus\{U_i\}} f_{X|pa(X)}(x|\boldsymbol{u}) \prod_{k \neq i} \pi_X(u_k)\, d\boldsymbol{u}_{-i} dx. \tag{3.6}$$

Where α_1, α_2 are normalizing constants. The derivation of these formulas, while simple, does not fall under the scope of this thesis. However, an easy way to intuitively capture the logic behind them is by looking at figures 3.3a,3.3b. For calculating $\lambda_X(u_i)$ we need to join the diagnostic parameter $\lambda(x)$ that arrives to X from below with the causal messages $\pi_X(u_k)$ that arrive individually from each of the parents of X except U_i. In order to join the messages we need to integrate first the rest of the parents of X and lastly, having joined the signals through $\mathbb{P}(x|pa(x))$, join the diagnostic parameter and integrate X.

For the causal message calculation, the case is simpler. We simply need to join the causal parameter of X with all of the diagnostic messages that arrive to X from each of its children except C_i.

Using Pearl's algorithm for inference with evidence $\boldsymbol{E}$ is done through the following steps:

1. **Initialization:** In this step, all peripheral nodes (roots, leafs and evidence nodes) are activated in the following way:
   - For $X = e \in \boldsymbol{E}$ we set $\lambda(x) = \pi(x) = 1$ if-f $x = e$ and 0 otherwise.
   - For X being a root node we set $\pi(x) = f_X(x)$, the prior probability.
   - For X being a leaf node we set $\lambda(x) = 1$ and then normalize.



(a) Diagnostic message calculation.

(b) Causal message calculation.

Figure 3.3: Illustration of Diagnostic and Causal belief propagation.

2. **Bottom-up propagation:** A node X calculates the new diagnostic messages $\lambda_X(u_i)$ to be sent to the parents, using the messages that X has received from both its children and its other parents.
3. **Top-down propagation:** A node X calculates the new causal messages $\pi_{C_i}(x)$ to be sent to its children, using the messages that X has received from both its parents and its other children.
4. **Iteration:** Steps 2 and 3 are iterated for all nodes until all causal and diagnostic parameters are calculated:
    - If all causal messages that arrive to X are known, calculate the causal parameter $\pi(x)$ using equation 3.5.
    - If all diagnostic messages that arrive to X are known, calculate the diagnostic parameter $\lambda(x)$ using equation 3.6.
    - If the causal parameter $\pi(x)$ has been calculated for X along all the diagnostic messages that arrive to X, except perhaps that of one of X's children Y, calculate the causal message from X to Y: $\pi_Y(x)$.
    - If the diagnostic parameter $\lambda(x)$ has been calculated for X along all the causal messages that arrive to X, except perhaps that of one of X's parents Z, calculate the diagnostic message from X to Z: $\lambda_X(z)$.

Pearl's Belief Propagation Algorithm has served as a fundamental tool in BN inference. The algorithm's significance extends beyond its ability to provide exact inference in polytrees, as it has provided the scientific community with an alternative formal mathematical foundation for tackling BN inference problems.

Its advantages over Variable Elimination include the fact that in sparse networks it's performance is significantly superior and that it is possible to tackle many inference problems at the same time. For instance, by continuing the message propagation in the previous example for one more iteration, $f_{1|25}$ and $f_{4|25}$ are also calculated, whereas in Variable Elimination we would need to go through the whole process once per required conditional probability.

Additionally, Pearl's algorithm can easily take advantage of parallelized computations to boost its efficiency exponentially with respect to the available CPU threads.

Remark. In spite of the algorithm's significance, there are certain problems that should be addressed.

1. The algorithm does not account for the BN's structure and conditional independencies present in the model. Consider the inference problem illustrated by figure 3.4. In this scenario, due to $X_8 \perp\!\!\!\perp X_9|X_1$, it is not required for pearl's algorithm to wait for the diagnostic message to arrive to X_1 before it propagates towards X_8. In larger graphs this problem may become even more evident

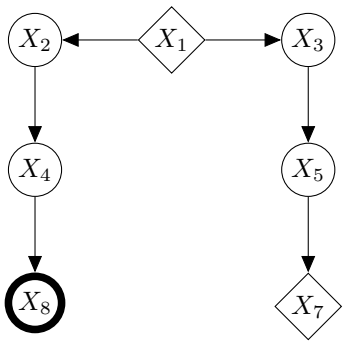


Figure 3.4: Example of a BN where Pearl's Algorithm is inefficient.

2. The algorithm propagates the messages away from the roots, leafs and evidence, but not only towards the variables in X_Q. The previous example showcased this behavior on the last steps, where we calculated $\pi_4(x_3)$ and $\pi(x_4)$, which are not useful for the inference problem at hand. In bigger and more connected BNs, this blind propagation results in exponential loss of performance.

3. By not incorporating factors as in Variable Elimination, Pearl's algorithm implements integration in a non-strategic way. Each variable is integrated once per parent in the calculation of the diagnostic message towards that parent. Even more costly is the integration over all the combinations of parental values in the calculation of the causal parameter through equation 3.4. These combinations are exponential in the number of parents and this is why "if there are more than four or five parents, approximation techniques must be invoked that make use of the special structure of the link matrix $f_{X|pa(X)}(x|\boldsymbol{u})$", Pearl [57, p.183].

4. It is impossible to calculate joint conditional probabilities through Pearl's algorithm. This means that the random vector $\boldsymbol{X_Q}$ can only be one dimensional. Variable Elimination, on the other hand, is able to calculate all types of inference problems.

5. The requirement that the graph model be a polytree is a very significant restriction. While the existence of large loops in graphical models may not always be intuitive, this is not often the case with triangles, which are also loops.

An extension of Pearl's algorithm that works in general BN structures is Loopy Belief Propagation. This method of belief propagation is an approximative algorithm has been discussed by Murphy, Weiss, and Jordan [53]. When loops are present in the BN, Pearl's algorithm produces endless circulation of propagating signals through each loop. Disregarding this problem, Loopy Belief propagation employs the algorithm consecutively

in hopes of the signals converging to an equilibrium which is equivalent to the true posterior probability.

In the previous example, if we had kept the BN in figure 3.1 without removing the edge $X_2 \to X_4$, due to the loop (X_2, X_3, X_4) we would continuously be re-evaluating the messages:

$$\lambda_4(x_2), \lambda_4(x_3)\lambda_3(x_2), \pi_4(2), \pi_4(3), \pi_3(2),$$

along with the parameters:

$$\pi(x_3), \pi(x_4), \lambda(x_2), \lambda(x_3).$$

General convergence of loopy signals seems to be the case in BNs according to empirical results. However, it has been shown that Loopy Belief Propagation is not consistent as there have been results that illustrate convergence to the wrong posteriors[53]. It is also shown that for BNs with the same DAG structure, convergence to the correct distribution depends on the parameters chosen for the conditional distributions in ways that research has not been able to infer thus far.

The number of available BN inference algorithms is vast, with extensions of Pearl's Belief Propagation Algorithm such as Localized Partial Evaluation [21] which works in tandem with the Annihilation/Reinforcement algorithm [66, 67], as well as distinctly alternative algorithms like Bounded Conditioning method [37]. Some other algorithms exist such as an approximative extension of the clique tree propagation algorithm [44] and a proposed algorithm by Moral, Rumi, and Salmerón [52] for BNs with Mixtures of Truncated Exponentials.

3.3. Gaussian Bayesian Networks

The Gaussian Bayesian Network (GBN) model is a continuous BN model that assumes that all conditional distributions in the network $f_{X|pa(X)}$ are linear Gaussian. This means that:

$$(X|\boldsymbol{pa(X)} = \boldsymbol{y}) \sim \mathcal{N}(\beta_0 + \boldsymbol{\beta^T y},\ \sigma^2)$$

For $pa(X) \sim \mathcal{N}(\boldsymbol{\mu_p}, \boldsymbol{\Sigma_p})$, it follows that:

- $\mu_X = \beta_0 + \boldsymbol{\beta^T \mu_p}$
- $\sigma_x^2 = \sigma^2 + \boldsymbol{\beta^T \Sigma_p \beta}$
- $\sigma_{X,Y_i} = \sum_{j=1}^{n} \beta_j \left(\boldsymbol{\Sigma_p}\right)_{i,j}$

Using these equations we can directly calculate the joint distribution of all the variables in the GBN which would be a multivariate Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Thus, by utilizing the conditional distribution formulas for multivariate Gaussian distributions we can tackle any inference problem. Given a partition $\{\boldsymbol{X_1}, \boldsymbol{X_2}\}$ of V and:

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu_1} \\ \boldsymbol{\mu_2} \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{bmatrix} \boldsymbol{\Sigma_{11}} & \boldsymbol{\Sigma_{12}} \\ \boldsymbol{\Sigma_{21}} & \boldsymbol{\Sigma_{22}} \end{bmatrix}$$

Then the conditional distribution of $\boldsymbol{X_1}|\boldsymbol{X_2} = \boldsymbol{x_2}$ is multivariate Gaussian with parameters:

$$\boldsymbol{\mu'} = \boldsymbol{\mu_1} + \boldsymbol{\Sigma_{12}}\boldsymbol{\Sigma_{22}^T}(\boldsymbol{x_2} - \boldsymbol{\mu_1})$$
$$\boldsymbol{\Sigma'} = \boldsymbol{\Sigma_{11}} - \boldsymbol{\Sigma_{12}}\boldsymbol{\Sigma_{22}^T}\boldsymbol{\Sigma_{21}}$$

GBNs are very attractive in that they offer an extremely efficient and simple approach to inference problems, albeit to the expense of severely restricting the BN's conditional distributions to only allowed to be linear Gaussian. While this strict reliance on linear Gaussian conditional distributions turns the GBN approach away from the goals of this thesis, the GBN will serve as a point of reference for testing our algorithm.

A notable extension of GBNs are Conditional Gaussian Bayesian Networks (CG BNs). CG BNs are a a family of hybrid BNs that extend the theory of GBNs to incorporate some discrete variables [33, 51, 10, 15].

In CG BNs the restriction is that continuous parents cannot have discrete children, thus all discrete variables appear "before" the continuous. The conditional distributions of continuous variables are once again linear Gaussian, while the conditional distributions of discrete variables given their (only discrete) parents are multinomial. Extensions relaxing these assumptions exist, however it is not possible to conduct inference in such models.

In contrast, multiple algorithms exist for CG BNs including Most Probable Explanation (MPE) approaches such as the Viterbi Algorithm, with some restrictions on the inference problem [23] and Maximum A Posteriori (MAP) approaches [58, 59].

We deem however that CG BNs restrict both the distributions and the network structure too much, hence we will not analyze any of the CG BN methodologies. This thesis aims to provide an inference methodology that is applicable to PCBNs, an extremely flexible family of BNs which is not bound by assumptions such as that of CG BNs.

3.4. Applications

In this subsection we will illustrate in practice how Variable Elimination, Pearl's Algorithm and Gaussian Bayesian Networks are applied.

The theoretical foundations of Variable Elimination and Pearl's Algorithm in Belief Propagation apply for both discrete and continuous BNs. However, their application as exact inference methodologies are mostly restricted to discrete variables. This is because, in the case of discrete BNs, the integrations of each respective algorithm are much easier to evaluate because they become summations. On the other hand, in many cases of continuous BNs, it is not possible to evaluate those integrals without approximation. Therefore, for the sake of simplicity we chose to show applications of both of these algorithms in discrete BNs.

A very common approach to inference problems in continuous BNs is Discretization. This is one of the most common techniques in which all continuous variables have their supports partitioned in order to become discrete. Thus, the continuous BN inference problem becomes a discrete one.

Due to the simplicity of this approach, Discretization has been studied extensively [20, 13, 24, 45, 22, 55]. Subsequently, Discretization algorithms were used in most commercial BN tools [49]. However, issues arise such as choosing an optimal cardinality of the partitions. Furthermore, in larger and more connected networks, loss of information from the discretization of the variables can propagate through the network and become quite significant.

3.4.1. Example of Variable Elimination

In figure 3.5 we have an graphical representation of a discrete BN inference problem inspired by the one in figure 3.1, with included conditional probability tables. We want to solve the inference problem $X_3|X_2 = 0, X_5 = 1$ or equivalently find:

$$f_{3|2,5}(x_3|0,1). \tag{3.7}$$

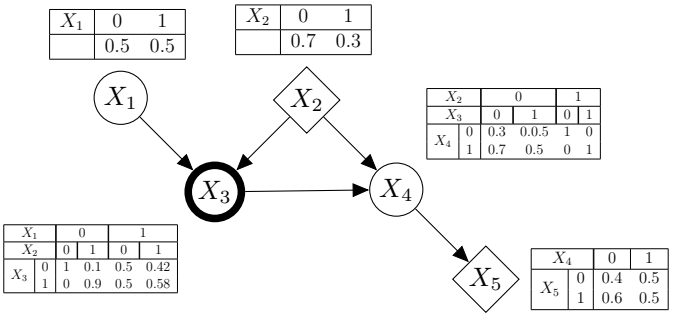


Figure 3.5: A discrete BN in the context of the inference problem $f_{3|2,5}$.

The factorization of the network PMF can be rewritten in factor terms as:

$$\begin{aligned}\mathbb{P}(x_1, x_2, x_3, x_4, x_5) &= f_{5|4}(x_5|x_4) f_{4|23}(x_4|x_2, x_3) f_{3|12}(x_3|x_1, x_2) f_2(x_2) f_1(x_1) \\ &=: \phi_5(x_4, x_5)\phi_4(x_2, x_3, x_4)\phi_3(x_1, x_2, x_3)\phi_2(x_2)\phi_1(x_1).\end{aligned}$$

As a first step we shall eliminate variable X_4. Only factors containing this variable are

needed for the integration and we calculate:

$$\phi_4(x_2, x_3, x_4)\phi_5(x_4, x_5) =$$

X_2	X_3	X_4	X_5	$\phi_4\phi_5$
0	0	0	0	0.12
			1	0.18
		1	0	0.35
			1	0.35
	1	0	0	0.2
			1	0.3
		1	0	0.25
			1	0.25
1	0	0	0	0.4
			1	0.6
		1	0	0
			1	0
	1	0	0	0
			1	0
		1	0	0.5
			1	0.5

Then we calculate the new factor:

$$\tau_1(x_2, x_3, x_5) = \sum_{x_4} \phi_4(x_2, x_3, x_4)\phi_5(x_4, x_5) =$$

X_1	X_2	X_4	τ_1
0	0	0	0.47
		1	0.53
	1	0	0.45
		1	0.55
1	0	0	0.4
		1	0.6
	1	0	0.5
		1	0.5

Finally, the reduced PMF takes the form:

$$f_{1235}(x_1, x_2, x_3, x_5) = \tau_1(x_2, x_3, x_5)\phi_3(x_1, x_2, x_3)\phi_2(x_2)\phi_1(x_1).$$

Similarly, we can sum over the values of X_1 to find $f_{235}(0, x_3, 1)$ and once more sum over values of X_3 to find $f_{25}(0, 1)$. By combining the last two PMFs we get the conditional probability of the inference problem 3.7:

$$f_{3|25}(x_3|0, 1) = 0.763\mathbb{1}_{\{x_3=0\}} + 0.237\mathbb{1}_{\{x_3=1\}}.$$

Despite its primitive nature, the performance of variable elimination is a significant improvement compared to just integrating the entire density over all values. Keeping a factor structure means that we avoid many unnecessary calculations. In our example, for the first step of Variable Elimination, we required 24 calculations, while if we where to integrate the whole density over X_4 we would need 48 calculations. The algorithm is relatively consistent and the main aspects that limit its efficiency are the number of variables and the number of values each variable can take. Furthermore, Variable Elimination does not restrict the graph structure of the BN, as opposed to other algorithms.

3.4.2. Example of Pearl's Algorithm

The previous example where we applied Variable Elimination assumed a BN whose DAG was not a poly tree, as it contains the triangular loop (X_2, X_3, X_4) (see in figure 3.5). Therefore, for the application of Pearl's Algorithm we need to simplify the network by removing an edge. Figure 3.6 illustrates a BN where we have removed the edge $X_2 \to X_4$.

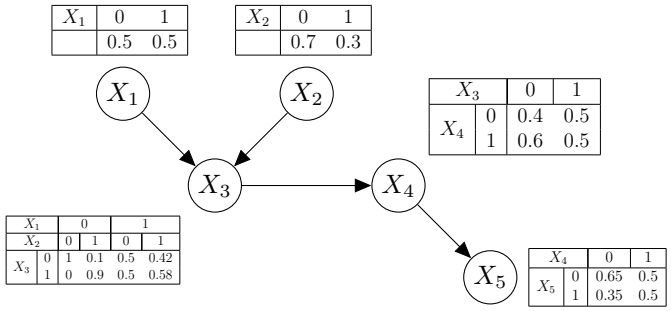


Figure 3.6: A discrete BN.

Let us use the algorithm to solve the same inference problem $f_{3|25}(x_3|0,1)$ for the network in figure 3.6. The evidence in this case is $\boldsymbol{E} = \{X_2 = 0, X_5 = 1\}$.

Step 1: In the initialization step we set:

- $\pi(x_2) = \lambda(x_2) = \mathbb{1}_{\{x_2=0\}}$ and $\pi(x_5) = \lambda(x_5) = \mathbb{1}_{\{x_5=1\}}$.
- $\pi(x_1) = f_1(x_1)$.

Step 2: Diagnostic messages:

$$\lambda_5(x_4) = \beta \sum_{x_5} \lambda(x_5) f_{5|4}(x_5|x_4) = 0.42\mathbb{1}_{\{x_4=0\}} + 0.58\mathbb{1}_{\{x_4=1\}}.$$

Step 3: Causal messages::

$$\pi_3(x_1) = \beta\pi(x_1) = f_1(x_1),$$

$$\pi_3(x_2) = \beta\pi(x_2) = \mathbb{1}_{\{x_2=0\}}.$$

Then we can calculate the causal parameter of X_3 and the diagnostic parameter of X_4:

$$\pi(x_3) = \sum_{x_1,x_2} f_{3|12}(x_3|x_1,x_2)\pi_3(x_1)\pi_3(x_2) = 0.75\mathbb{1}_{\{x_3=0\}} + 0.25\mathbb{1}_{\{x_3=1\}},$$
$$\lambda(x_4) = \lambda_5(x_4) = 0.41\mathbb{1}_{\{x_4=0\}} + 0.59\mathbb{1}_{\{x_4=1\}}.$$

Step 2: Second iteration:

$$\lambda_4(x_3) = \beta \sum_{x_4} \lambda(x_4) f_{4|3}(x_4|x_3) = 0.52\mathbb{1}_{\{x_3=0\}} + 0.48\mathbb{1}_{\{x_3=1\}}.$$

Step 3: Second iteration:

$$\pi_4(x_3) = \beta\pi(x_3) = 0.75\mathbb{1}_{\{x_3=0\}} + 0.25\mathbb{1}_{\{x_3=1\}}.$$

Then we can calculate the causal parameter of X_4 and the diagnostic parameter of X_3:

$$\pi(x_4) = \sum_{x_3} f_{4|3}(x_4|x_3)\pi_4(x_3) = 0.45\mathbb{1}_{\{x_4=0\}} + 0.55\mathbb{1}_{\{x_4=1\}},$$
$$\lambda(x_3) = \lambda_4(x_3) = 0.52\mathbb{1}_{\{x_3=0\}} + 0.48\mathbb{1}_{\{x_3=1\}}.$$

Now we have both the causal and diagnostic parameters of X_3 ready, meaning that we can calculate:

$$\mathbb{P}(X_3 = x_3|\boldsymbol{E}) = \alpha\lambda(x_3)\pi(x_3) = 0.765\mathbb{1}_{\{x_3=0\}} + 0.235\mathbb{1}_{\{x_3=1\}}.$$

Note that this result differs from the one obtained by variable elimination. This is clearly due to the removal of arc $X_2 \to X_4$, changing the BN.

3.4.3. Example of Gaussian Bayesian Network

To illustrate how inference is tacked for GBNs we will use a simpler DAG structure.

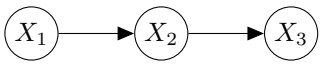


Figure 3.7: Example of a GBN.

for the conditional densities of the network we assume the following:

$$X_1 \sim \mathcal{N}(1, 1)$$
$$X_2|X_1 = X_1 \sim \mathcal{N}(0.5X_1 + 2, 0.25)$$
$$X_3|X_2 = X_2 \sim \mathcal{N}(X_2 - 1, 2)$$

From the conditional distributions we can directly derive that the distribution of the random vector (X_1, X_2, X_3) is multivariate Gaussian $(X_1, X_2, X_3) \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ where:

$$\mu = \begin{pmatrix} 1 \\ 2.5 \\ 1.5 \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{bmatrix} 1 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 2.5 \end{bmatrix}$$

Using the conditional formula of joint normal distributions we can directly derive that:

$$X_2|X_1 = 0, X_2 = 2 \sim \mathcal{N}(2.11, 0.22)$$

As we saw, the implementation of GBNs is very effective and efficient. We chose a three node network for the sake of simplicity. For larger networks the process would be equally as simple.

3.5. Stochastic Sampling

To address the shortcomings of the aforementioned methodologies and their extensions, we chose to conduct inference through sampling. While maintaining core aspects of both Variable Elimination and Belief Propagation, sampling allows for many additional benefits including advanced parallelization and granularity, as well as the ability to propagate information in the presence of loops.

A recurring technique used in existing sampling-based inference algorithms is that of Importance Sampling, a strategy for calculating an expected value of a distribution using samples from a different distribution.

Let f, g be two probability densities corresponding to two distributions with the same support $\mathcal{X}$. Furthermore, assume that both distributions have finite moments and that we are able to sample from g. Then, if h is an integrable function and wish to evaluate the integral:

$$\int_{\mathcal{X}} h(x)f(x)dx = \mathbb{E}_f\left(h(x)f(x)\right),$$

we can instead use Monte-Carlo simulation to estimate the value:

$$\mathbb{E}_g\left(h(X)\frac{f(X)}{g(X)}\right).$$

In importance sampling, f is referred to as the **target function/distribution**, while g is called the **importance function/distribution**. To evaluate the integral we choose samples from the density g, also called **importance samples**, and take the sample average of the expression $h(X)\frac{f(X)}{g(X)}$ for all the importance samples. The factor $\frac{f(x)}{g(x)}$ is also referred to as the **weight** of each importance sample and in measure theoretic terms it is the Radon-Nikodym derivative of f with respect to g.

For more details on Importance Sampling and Stochastic Sampling methodologies we refer to Glasserman [29],Robert, Casella, and Casella [61] or Gelman et al. [27].

Stochastic Sampling algorithms have been used before in BN inference problems, albeit the most popular are also quite inefficient. We will briefly cover the most used methods before presenting our approach.

The first family of sampling algorithms has the prior distribution of the network as the importance function. Probabilistic Logic Sampling is the most straightforward and inefficient of the algorithms in this family. It essentially simulates the network unconditionally and then keeps the samples that are compatible with the evidence [34]. It is clear that this simple algorithm is increasingly more inefficient the more unlikely the evidence is.

Likelihood Weighing was an algorithm proposed by Fung and Chang [25] as an augmentation to Probabilistic Logic Sampling. After the same sampling procedure, each sample gets attributed a weight, which is the product of conditional densities of evidence nodes given its parents.

For instance in sample i and for $E_1, \ldots, E_m$ being the evidence nodes with observations $e_1, \ldots, e_m$, the weight would be equal to:

$$w_i = \prod_{j=1}^{m} \mathbb{P}\left(E_j = e_j | X_{pa(E_j)} = \widehat{x_{pa(E_j)}}^{i}\right)$$

Then the posterior probabilities are are equal to the weighted observation rates in the samples. This algorithm is an obvious improvement over Probabilistic Logic Sampling. Nevertheless, for unlikely evidence it is still highly inefficient and its use is reserved for improving other algorithms that do not incorporate weighting.

More sophisticated approaches to Sampling algorithms exist. Instead of using the prior distribution of the BN, these algorithms employ methodologies to learn an "optimal" importance function, which would be as close as possible to the posterior distribution. Self Importance Sampling (SIS), Adaptive Importance Sampling (AIS-BN) and Hamiltonian Monte-Carlo (HMC) simulations are all algorithms that fall under this category [62, 11]. Most of these algorithms have a common drawback, they do not include information about BN structure in the importance function like in SIS and HMC. AIS-BN has multiple modifications to the prior using heuristics which results in over two orders of magnitude improvement in convergence over likelihood weighting and self-importance sampling algorithms according to Yuan and Druzdzel [71].

This highlights the importance on incorporating clearly the network structure into our algorithm, instead of relying on the prior to cover the conditional independencies [49]. Some methods have been proposed that can be applied to existing algorithms to exploit causal independencies [72], but these methods are limited by the framework of the original algorithm they is applied to, and there is a strict limit to the possible improvement.

3.5.1. Sampling Importance Resampling

In the scope of our inference algorithm there will be cases in which to get the sample of interest, we will need to sample from a density which is difficult to sample from. One may use any sampling methodology to tackle this task, but in this thesis we opted to employ Sampling Importance Resampling (SIR)[60].

Let us consider the problem of sampling from a univariate density f. For employing SIR, we may choose a simpler to sample density g and exploit proportionality:

$$f(x) \propto g(x)h(x)$$

then the first step of the SIR algorithm is to obtain L importance samples $\hat{x}_i^g$ from the density g. Each of these samples $\hat{x}_i^g$ then gets an unnormalized weight:

$$w_i' = \frac{f(\hat{x}_i^g)}{g(\hat{x}_i^g)} = h(\hat{x}_i^g), \quad i = 1, \ldots, L$$

These weights are normalized as to provide weights whose sum is 1:

$$w_i = \frac{w_i'}{\sum_{j=1}^{L} w_j'}, \quad i = 1, \ldots, L$$

To obtain the final samples from the SIR algorithm we resample from the importance samples $\hat{x}_i^g$ using the normalized weights w_i. Each sample is chosen with probability equal to its assigned weight.

One could choose any existing methodology to sample from a density instead of SIR. However, we deemed SIR to be an especially beneficial choice to solve inference problems in PCBNs due to the ability to work with the proportional expression of the density as well as to exploit the copula density factorization to obtain importance samples.

4

PCBN Inference

PCBN models distinguish the marginal distributions of all node variables and their dependence structures. The former are included in the collection of marginal distributions $\mathscr{F}$, while the latter are part of the collection of specified copulas $\mathscr{C}$. Hence, the inference can be presented for uniform margins, that is on the level of copulas, and at the last stage the inverses of the original marginal CDFs can be applied to the conditional margins obtained through conditionalization to reveal the desired conditional distributions.

Let us first observe that each PCBN can be decomposed into subgraphs that contain nodes connected by arcs assigned with unconditional copulas. For the PCBN in figure 4.1 these are the subgraphs containing the nodes $\{U_1, U_3, U_4, U_7, U_8, U_9, U_{11}\}$ and $\{U_2, U_6, U_{10}\}$.

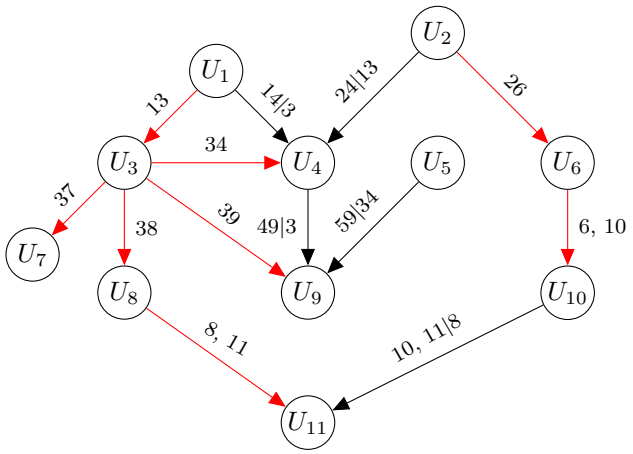


Figure 4.1: Example of a PCBN that illustrates "artery" trails.

Definition 4.0.1 (Arterial subgraph, artery, extended artery, arterial arcs)**.**
Let $(\mathscr{G}, \mathscr{O}, \mathscr{F}, \mathscr{C})$ be a PCBN. For U being a root of $\mathscr{G}$, we call a subgraph $\mathscr{A} = (V_{\mathscr{A}}, A_{\mathscr{A}}) \subset \mathscr{G}$ an **artery** of the graph $\mathscr{G}$ generated by U if the following conditions are satisfied.

- $U \in V_{\mathscr{A}}$.

- $\{U_2 \in V \mid \exists\, U_1 \in V_{\mathscr{A}} : C_{12} \in \mathscr{C}\} \subseteq V_{\mathscr{A}}$.
- $\forall\, (U_1, U_2) \in A_{\mathscr{A}},\ \exists\, C_{12} \in \mathscr{C}$.

Furthermore, we introduce the following definitions:

❑ The artery $\mathscr{A}$ is referred to as a **singleton artery** if $|V_{\mathscr{A}}| = 1$

❑ The induced subgraph $\mathscr{G}_{\mathscr{A}} = \mathscr{G}[V_{\mathscr{A}}]$ is called **extended artery** of $\mathscr{G}$.

❑ The arcs in $A_{\mathscr{A}}$ will be referred to as the **arterial arcs** of $\mathscr{G}_{\mathscr{A}}$, and the arcs in $A \backslash A_{\mathscr{A}}$ will be called as the **non-arterial arcs** of $\mathscr{G}_{\mathscr{A}}$.

We extend the definitions of descendants, ancestors, children and parents of node U to arterial and extended arterial counterparts, each corresponding to the respective sets in the artery or extended artery that U belongs to. In the example of figure 4.1, for $U = U_9$ we have:

$$
\begin{aligned}
&pa(U_9) = \{U_3, U_4, U_5\},\\
&pa_{\mathscr{G}_{\mathscr{A}_1}}(U_9) = \{U_3, U_4\},\\
&pa_{\mathscr{A}_1}(U_9) = \{U_3\}.
\end{aligned}
$$

Arteries make up subgraphs of the BN containing only arcs for which unconditional copulas are specified. In the original graph there may also be arcs between nodes of the same artery that are connected through conditional copulas. These are part of the extended artery. On the other hand, arcs between nodes of different arteries belong to neither arteries nor extended arteries.

Remark. Note that due to the process of assigning unconditional and conditional copulas in the PCBN, for every $U_i \in V$, U_i is either a root variable or there exists a unique $U_j \in pa(U_i)$ such that:

$$C_{ji} = C_{ji|\emptyset} \in \mathcal{C}.$$

This means that every node is either a root node or has a unique parent to whom it is connected through an unconditioned copula. This in turn implies that the number of arteries of a PCBN equals the number of roots, and that each variable in the PCBN belongs to exactly one artery.

Hence, if there are m root nodes in a PCBN then there are m arteries $\mathscr{A}_i = (V_{\mathscr{A}_i}, A_{\mathscr{A}_i})$ and $V_{\mathscr{A}_1}, \ldots, V_{\mathscr{A}_m}$ form a partition of V.

Since extended arteries are induced subgraphs of $\mathscr{G}$ on $V_{\mathscr{A}_i}$ then $\{\mathscr{G}_{\mathscr{A}_i}\}_{i=1\ldots m}$ is the unique **extended arterial partition** of $\mathscr{G}$.

In figure 4.2 there are three arteries $\mathscr{A}_1, \mathscr{A}_2, \mathscr{A}_5$ of the PCBN represented in 4.1, corresponding to each of the three roots of the network $U_1,\ U_2,\ U_5$. Note that the artery with root U_5 is a singleton artery.

In figure 4.3 the arteries in figure 4.1 are supplied with extra arks assigned with conditional copulas and we can see extended arteries of the PCBN represented in 4.1.

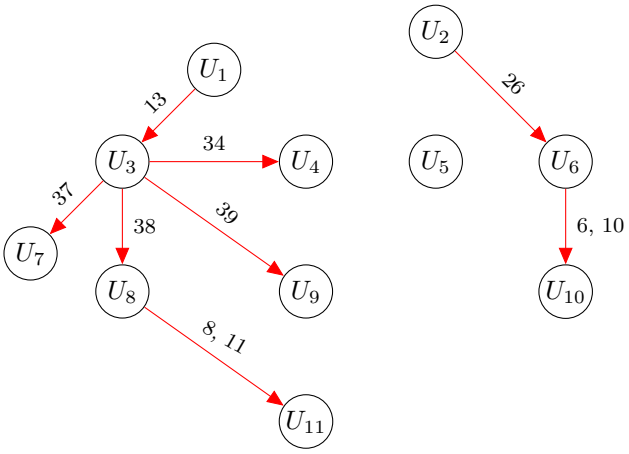


Figure 4.2: Arteries of PCBN in figure 4.1.

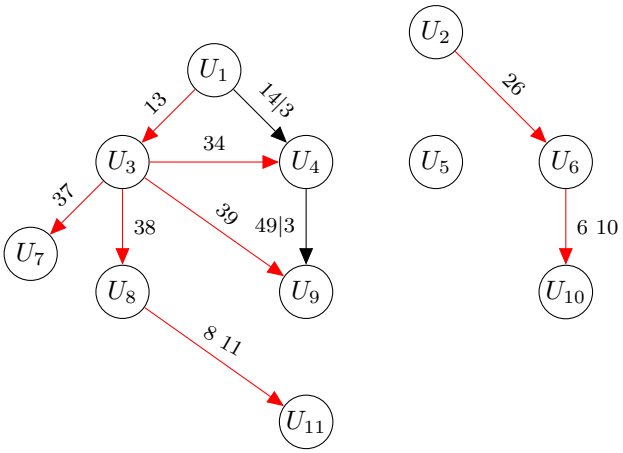


Figure 4.3: Extended arteries of PCBN in figure 4.1.

Our proposed methodology for solving PCBN inference problems relies on conducting inference on the sub-PCBN networks that correspond to the arterial and extended arterial DAGs.

Definition 4.0.2 (Arterial PCBN, Extended Arterial PCBN)**.** Let $(\mathscr{G}, \mathscr{O}, \mathscr{F}, \mathscr{C})$ be a PCBN with extended arterial partition $\{\mathscr{G}_{\mathscr{A}_i}\}_{i=1\ldots m}$.

- The **arterial PCBN** of an artery $\mathscr{A}$ is the PCBN

$$(\mathscr{A}, \mathscr{O}', \mathscr{F}', \mathscr{C}')$$

  where:

  - $\mathscr{O}'$ is a collection of parental orderings of the nodes $V_{\mathscr{A}}$ in $\mathscr{A}$ such that every parental ordering contains only the first parent in $\mathscr{G}$ of each node, according to $\mathscr{O}$. Equivalently, $\forall U_1 \in V_{\mathscr{A}}$ and $o \in \mathscr{O}$, $o' \in \mathscr{O}'$ the parental orders of U_1 in the respective collection, then o' is a total order on the singleton $\{U_2\}$ where U_2 is the smallest element in o that belongs to $V_{\mathscr{A}}$. If o has no nodes in $V_{\mathscr{A}}$, then $o' = \emptyset$ instead.
  - $\mathscr{F}'$ is the sub-collection of $\mathscr{F}$ including exactly the marginal distributions of

the nodes in $V_{\mathscr{A}}$.

– $\mathscr{C}'$ is the sub-collection of all unconditioned copulas in $\mathscr{C}$ between nodes in $V_{\mathscr{A}}$, meaning that:

$$\mathscr{C}' = \{C_{12|K} \in \mathscr{C} : U_1, U_2 \in V_{\mathscr{A}}, K = \emptyset\}.$$

- The **extended arterial PCBN** of an extended artery $\mathscr{G}_{\mathscr{A}}$ is the PCBN

$$(\mathscr{G}_{\mathscr{A}}, \mathscr{O}'', \mathscr{F}'', \mathscr{C}'')$$

where:

– $\mathscr{O}''$ is a collection of parental ordering of the nodes $V_{\mathscr{A}}$ in $\mathscr{G}_{\mathscr{A}}$ such that it abides by the parental orderings in $\mathscr{O}$, meaning that for all $U_1, U_2, U_3 \in V_{\mathscr{A}}$ such that $U_1, U_2 \in pa_{\mathscr{A}}(U_3)$, and for o, o'' denoting the parental orderings of U_3 in $\mathscr{O}$ and $\mathscr{O}''$ respectively:

$$U_1 <_3 U_2 \in o'' \iff U_1 <_3 U_2 \in o.$$

– $\mathscr{F}''$ is the sub-collection of $\mathscr{F}$ including exactly the marginal distributions of the nodes in $V_{\mathscr{A}}$.

– $\mathscr{C}''$ is the sub-collection of all copulas in $\mathscr{C}$ between nodes in $V_{\mathscr{A}}$, meaning that:

$$\mathscr{C}'' = \{C_{1,2|K} \in \mathscr{C} : U_1, U_2 \in V_{\mathscr{A}}\}.$$

Next we will identify two different types of children a node can have. This distinction will be crucial in later stages of our algorithm.

Definition 4.0.3 (Children Types)**.** Let $\mathscr{G}_{\mathscr{A}}$ be an extended arterial PCBN and $U_i \in ch_{\mathscr{G}_{\mathscr{A}}}(U_1)$, then:

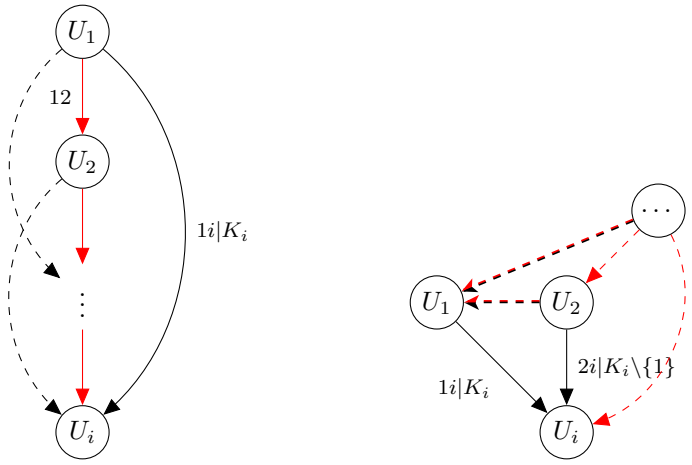
- We call U_i a **type 1 child** of U_1 if $U_i \in de_{\mathscr{A}}(U_1)$
- We call U_i a **type 2 child** of U_1 if $U_i \notin de_{\mathscr{A}}(U_1)$

Remark. Figure 4.4 illustrates the two types of children. In network 4.4a, U_i is a type 1 child of U_1 as it is also an arterial descendant. On the other hand, in network 4.4b, U_i is not an arterial descendant and thus is a type 2 child. In the case of the type 2 child, the other parents of U_i are shown and one of them has to be the arterial parent of U_1.

With the required definitions and concepts, we proceed to present the methodology for PCBN inference. We show this process for single arterial PCBNs. After illustrating our approach for single-arterial PCBN inference, we will provide insights on how this methodology can be extended to cover the general case. This means how to combine inference for multiple arteries, which may be connected with each other.

4.1. Node Sampling

The inference method proposed in this thesis is based on stochastic sampling of the nodes of the network, given previous samples of their parents and their children. Each sampling iteration is node through either of the following three types of sampling:



(a) Type 1 **(b)** Type 2

Figure 4.4: The two types of children in extended arteries.

1. Forward Sampling
2. Backward Sampling
3. Bilateral Sampling

Forward Sampling is performed when we wish to sample a variable given the values of this variable's parents. Backward Sampling, is conducted in the opposite direction, hence a variable is sampled given values of its children. Lastly, Bilateral Sampling is conducted when we wish to sample a node using values of both parents and children. The last type of sampling is used to combine the information coming from the parents and children of the node.

Next we provide a detailed overview of each of the three types of sampling.

4.1.1. Forward Sampling

Out of the three sampling types, Forward sampling is the simplest as it falls in line with the directionality of the BN.

Forward sampling a node U_1, when the values of the node's parents $\{U_2, \ldots, U_m\}$ are given, is identical with the PCBN sampling presented in subsection 2.3.4, where we obtain our sample through the use of all incoming copulas.

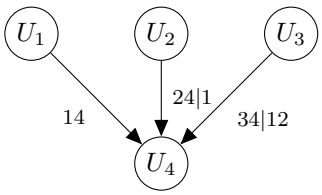


Figure 4.5: Illustration of Forward Sampling iteration.

In figure 4.5 an example of a Forward Sampling iteration is illustrated. In order to sample U_4 using the values of $U_1 = u_1, U_2 = u_2, U_3 = u_3$, a uniformly distributed sample:

$$V \sim \mathcal{U}(0, 1).$$

is needed. This sample is an argument of the recursion of inverse h-functions to obtain a sample of $U_4|U_1 = u_1, U_2 = u_2, U_3 = u_3$:

$$h^{-1}_{1\underline{4}}\left(u_1, h^{-1}_{2\underline{4}|1}\left(u_{2|1}, h^{-1}_{3\underline{4}|12}\left(u_{3|12}, V|u_1, u_2\right)|u_1\right)\right).$$

Note that all required conditional margins needed to perform the recursion above are computable without integration in restricted PCBNs. Forward samples of variables in the inference algorithm will be denoted with a superscript f. Thus, instead of presenting the whole recursion for the sample in example of figure 4.5, we will denote it as u_4^f.

The pseudo-code of Forward Sampling algorithm is given in algorithm 1.

Algorithm 1 Forward.Sample($\mathscr{G}_{\mathscr{A}}, U_1, \{u_2, \ldots, u_m\}$)

Input:
- $\mathscr{G}_{\mathscr{A}}$ an Extended Artery
- U_1 the node to be Forward Sampled
- $\{u_2, \ldots, u_m\}$ values of U_1's parents $\{U_2, \ldots, U_m\}$ indexed according to the parental ordering of U_1

Output: Sample $\hat{u_1}$ of U_1

1: $V \sim \mathcal{U}(0,1)$ ▷ Uniform Sample
2: $\hat{u_1} \leftarrow h^{-1}_{2\underline{1}}(u_2, h^{-1}_{3\underline{1}|2}(u_{3|2}, \ldots h^{-1}_{m\underline{1}}(u_{m|2\ldots m-1}, V|\boldsymbol{u}_{K_i})\ldots|u_2))$
3: **return** $\hat{u_1}$

4.1.2. Backward Sampling

In contrast to Forward Sampling, sampling given the values of a node's children can be more complicated due to the sampling direction being opposite to the direction of the arcs of the DAG.

Two types of Backward sampling are considered:

1. Conditioned on only type 1 children,
2. Conditioned on both type 1 and type 2 children.

We will refer to **Backward Sampling** in case when only type 1 children are included. When also type 2 children are involved then the procedure is referred to as **Secondary Backward Sampling**.

Backward Sampling

Two cases are considered below which are differentiated by the copula-arc structure of U_1 and its children. In both cases we will denote the type 1 children of U_1 as $\{U_2, \ldots, U_m\}$.

Case 1: The first case of Backward Sampling assumes that the following condition holds:

(B1) There is a type 1 child $U_i \in \{U_2, \ldots, U_m\}$ of U_1 with specified copula $C_{1i|K_i} \in \mathscr{C}$ such that:

$$\{U_2, \ldots, U_m\} \backslash \{U_i\} = K_i.$$

Let us assume that this node is U_m and that $U_2, \ldots, U_{m-1}$ are ordered according to the parental order of U_1. Then the Backward Sample U_1 is calculated through the use of copula $C_{1m|K_m}$ and the recursion of h-values, similarly to Forward Sampling.

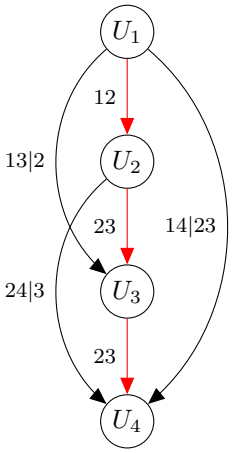


Figure 4.6: First Type of Backward Sampling.

Example 4.1.1. In figure 4.6 we see an example of a DAG with four nodes. The goal is to backward sample U_1 given values of its children U_2, U_3 and U_4. The condition (B1) is satisfied and to sample U_1 using the values of $U_2 = u_2, U_3 = u_3, U_4 = u_4$, a uniformly distributed sample $V \sim \mathcal{U}(0,1)$ is obtained and the recursion of inverse h-function gives a Backward Sample of $U_1|U_2 = u_2, U_3 = u_3, U_4 = u_4$, which is denoted simply as u_1^b:

$$u_1^b = h_{\underline{1}2}^{-1}\left(h_{\underline{1}3|2}^{-1}\left(h_{\underline{1}4|23}^{-1}\left(V, u_{4|23}|u_2, u_3\right), u_{3|2}|u_2\right), u_2\right).$$

The pseudocode of Backward Sampling in case 1 is presented below in algorithm 2.

Algorithm 2 Backward.Sample.1($\mathscr{G}_{\mathscr{A}}, U_1, \{u_2, \ldots, u_m\}$)

Input:
- $\mathscr{G}_{\mathscr{A}}$ an Extended Artery
- U_1 the node to be Backward Sampled
- $\{u_2, \ldots, u_m\}$ values of U_1's type 1 children $\{U_2, \ldots, U_m\}$ indexed according to the parental ordering of U_m.

Output: Sample $\hat{u_1}$ of U_1

1: $V \sim \mathcal{U}(0,1)$ ▷ Uniform Sample
2: $\hat{u_1} \leftarrow h_{\underline{1}2}^{-1}(h_{\underline{1}3|2}^{-1}(\ldots(h_{\underline{1}m|K_2}^{-1}(V, u_{m|K_m}|\boldsymbol{u}_{K_m})\ldots), u_{3|2}|\boldsymbol{u}_2), u_2)$
3: **return** $\hat{u_1}$

Case 2: When the condition (B1) is not satisfied, the Backward Sampling is a bit more complicated and is referred to as case 2.

In this case there is no single h-function recursion to obtain a sample from U_1 given its children. Instead we will choose a subset of children for which the condition (B1) is satisfied, and use the case 1 Backward Samples as importance samples.

$$\mathscr{C}_1 \coloneqq \{C_{ij|K_{ij}} \in \mathscr{C} : i, j = 1, \ldots, m \ \wedge \ U_1 \in K_{ij} \cup \{U_i\}\}.$$

This is a collection of copulas assigned to arcs between nodes $U_1, U_2, \ldots, U_m$ such that U_1 is either a marginal variable or part of the conditioning set of these copulas.

The second case of Backward Sampling requires the sample from the conditional density $c_{1|2\ldots m}(u_1|u_2 \ldots u_m)$ which can be rewritten as follows:

$$\begin{aligned} c_{1|2\ldots m}(u_1|u_2 \ldots u_m) &= \frac{c_{1,2\ldots m}}{c_{2\ldots m}} \\ &= \frac{\prod_{c\in\mathscr{C}_1} c}{\int_0^1 \left(\prod_{c\in\mathscr{C}_1} c\right) du_1} \\ &\propto \prod_{c\in\mathscr{C}_1} c. \end{aligned}$$

Note that only copulas containing U_1 in the conditioning or the conditioned sets are needed.

In order to sample from this conditional density we use SIR. We choose the child U_i such that for the specified copula $C_{1i|K_i}$, the following set has the largest possible cardinality.

$$K_i^1 = \{U_{j_1}, \ldots, U_{j_{m_i}}\} \coloneqq K_i \cap \{U_2, \ldots, U_m\}, \tag{4.1}$$

where $U_{j_1}, \ldots, U_{j_{m_i}}$ are indexed according to the parental order of U_i. We do this in order to maximize the information on the children of U_1 in the importance sample. This choice will be investigated in the simulation study in Chapter 6.

The L importance samples of U_1 are obtained using the following h-function recursion for copula $C_{1i|K_i}$:

$$\hat{u_1}_k^{imp} = h^{-1}_{1j_1}(h^{-1}_{1j_2|K_{j_2}}(\ldots h^{-1}_{1i|K_i}(V_k, u_{i|K_i}|\boldsymbol{u}_{K_{j_{m_i}}}) \ldots, u_{j_2|K_{j_2}}|\boldsymbol{u}_{K_{j_2}}), u_{j_1}),$$

where $V_1, \ldots, V_L \overset{iid}{\sim} \mathcal{U}(0,1)$.

The weight for each sample will be equal to the product of the densities of all copulas that were not used in the calculation of the importance sample. These belong to the sub-collection:

$$\mathscr{C}_1' \coloneqq \{C_{ij|K_{ij}} \in \mathscr{C}_1 : (\{U_2 \ldots U_m\} \backslash (\{U_i\} \cup K_i^1)) \cap (\{i, j\} \cup K_{ij}) \neq \emptyset\}.$$

Then the weight for each importance sample is equal to:

$$w_k' = \prod_{C\in\mathscr{C}_1'} c.$$

In order to be used in SIR the weights must be normalized:

$$w_k = \frac{w_k}{\sum_{j=1}^L w_j}.$$

Lastly, we resample using the normalized weights w_k in order to obtain the Backward Sample of U_1, denoted as u_1^b.

The pseudocode for case 2 Backward Sampling is presented in algorithm 3:

Algorithm 3 Backward.Sample.2($\mathscr{G}_{\mathscr{A}}, L, U_1, \{u_2, \ldots, u_m\}$)

Input:
- $\mathscr{G}_{\mathscr{A}}$ an Extended Artery
- L the number of importance samples.
- U_1 the node to be Backward Sampled
- $\{u_2, \ldots, u_m\}$ values of U_1's type 1 children $\{U_2, \ldots, U_m\}$.

Output: Sample $\hat{u_1}$ of U_1

$V_1, \ldots, V_L \sim \mathcal{U}(0,1)$ ▷ Uniform Samples
Choose U_i that maximizes the cardinality of $\{U_{j_1}, \ldots, U_{j_{m_i}}\}$ as defined in (4.1)
for $k \in \{1, \ldots, L\}$ **do**
    $\hat{u_1}_k^{imp} \leftarrow h_{\underline{1}j_1}^{-1}(h_{\underline{1}j_2|K_{j_2}}^{-1}(\ldots(h_{\underline{1}j_{m_i}|K_{j_{m_i}}}^{-1}(V_k, u_{j_{m_i}|K_{j_{m_i}}}|\boldsymbol{u}_{K_{j_{m_i}}})\ldots), u_{j_2|K_{j_2}}|\boldsymbol{u}_{K_{j_2}}), u_{j_1})$
    $w'_k \leftarrow \prod_{C \in \mathscr{C}'_1} c$
end for
for $k \in \{1, \ldots, L\}$ **do**
    $w_k = \frac{w'_k}{\sum_{l=1}^{N} w_l}$
end for
Sample $\hat{u_1}$ from $\{\hat{u_1}_1^{imp}, \ldots, \hat{u_1}_L^{imp}\}$ with weights $w_1, \ldots, w_L$
return $\hat{u_1}$

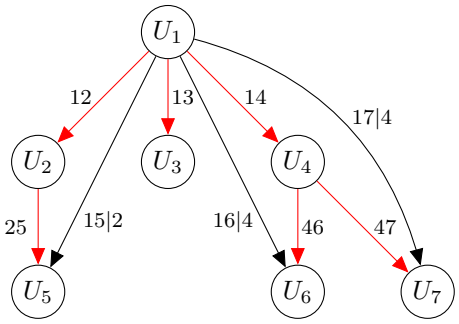


Figure 4.7: Second Type of Backward Sampling.

Example 4.1.2. In figure 4.7 a PCBN of seven nodes is presented. If U_1 is to be Backward Sampled then this node does not meet the requirements for case 1 Backward Sampling. We choose a child of U_1 whose respective copula has the largest conditioning set. In the example the choices are U_5, U_6 and U_7, as each of them have one variable in the conditioning set of the copula assigned to the arc between them and U_1. If we choose U_5, the importance sample is obtained through the h-function recursion of copula $C_{15|2}$:

$$\hat{u_1}^{imp} = h_{\underline{1}2}^{-1}\left(h_{\underline{1}5|2}^{-1}\left(V, u_{5|2}|u_2\right), u_2\right).$$

and its weight is equal to:

$$w' = c_{13}c_{14}c_{16|4}c_{17|4}.$$

Finally, the case 2 Backward Sample of U_1, denoted as u_1^b is obtained by resampling from the importance sample using the corresponding normalized weights.

Secondary Backward Sampling:

The steps of Secondary Backward Sampling are similar to the second case of Backward Sampling presented above. The sampling from the conditional density needs to be obtained through SIR. As before, we divide Secondary Backward Sampling into two cases, depending on the number and types of children that the node has.

Case 1: In the first scenario we assume that U_1 satisfies the following condition:

(B2)
- 0 type 1 children and,
- $m \geq 1$ type 2 children $\{U_2, \ldots, U_m\}$.

Let us denote the copulas corresponding to the arcs $U_1 \to U_i$ as $C_{1i|K_i}$. As before, we wish to sample from the density:

$$c_{1|2\ldots m} = \frac{c_{1,2\ldots m}}{c_{2\ldots m}}.$$

However, now due to existence of type 2 children of U_1, we do not have a closed form expression of the numerator. Indeed, to compute the numerator it is required to integrate over the remaining extended arterial parents of the children $\{U_2, \ldots, U_m\}$, which are part of the conditioning sets K_i. These nodes are denoted as:

$$\left(\bigcup_{i=1}^{m} K_i\right) \setminus \{U_2, \ldots, U_m\} = \{U_{\iota_1}, \ldots, U_{\iota_M}\}. \tag{4.2}$$

The nodes U_{ι_j} could be connected to U_1 through in or outgoing arc. There will be a copula assigned to each of these arcs and we will denote these copulas as $C^1_{\iota_j|K_{\iota_j}}$ (avoiding specifying the direction of the arcs).

Additionally all copulas that contain U_{ι_j} nodes in either their marginals or the conditioning sets are collected in the set $\mathscr{C}_1$. These are copulas specified by arcs between:

- two U_{ι_j} variables,
- one U_{ι_j} variable and one type 2 child U_i,
- one U_{ι_j} variable and U_1 or
- between two type 2 children U_i, U_j for which the conditioning set K_{ij} of copula $C_{ij|K_{ij}}$ contains a U_{ι_j} node.

$$\begin{aligned}\mathscr{C}_1 \coloneqq \Big\{ C_{kl|K_{kl}} \in \mathscr{C} :\ & U_k, U_l \in \{U_1, U_2, \ldots, U_m, U_{\iota_1,\ldots,U_{\iota_M}}\} \text{ and} \\ & (\{U_k, U_l\} \cup K_{kl}) \cap \{U_{\iota_1}, \ldots, U_{\iota_M}\} \neq \emptyset \Big\}.\end{aligned} \tag{4.3}$$

Using the introduced notation we can rewrite the needed conditional density as follows:

$$\begin{aligned} c_{1|2\ldots m} &= \frac{c_{1,2\ldots m}}{c_{2\ldots m}} \\ &= \frac{\int_{[0,1]^M} c_{1,2\ldots m,\iota_1\ldots\iota_M} du_{\iota_1}\ldots du_{\iota_M}}{\int_{[0,1]^{M+1}} c_{1,2\ldots m,\iota_1\ldots\iota_M} du_1 du_{\iota_1}\ldots du_{\iota_M}} \\ &= \frac{\int_{[0,1]^M} \prod_{C\in\mathscr{C}_1} c\ du_{\iota_1}\ldots du_{\iota_M}}{\int_{[0,1]^{M+1}} \prod_{C\in\mathscr{C}_1} c\ du_1 du_{\iota_1}\ldots du_{\iota_M}} \\ &\propto \int_{[0,1]^M} \prod_{C\in\mathscr{C}_1} c\ du_{\iota_1}\ldots du_{\iota_M}. \end{aligned}$$

Since the numerator does not have a closed form, it is therefore impossible to conduct importance sampling as all copula densities are involved in the integration of U_{ι_j} nodes.

SIR is still possible by choosing L iid uniformly distributed samples

$$V_1,\ldots,V_L \overset{iid}{\sim} \mathcal{U}(0,1)$$

as the importance samples. Then each such sample has an unnormalized weight calculated by:

$$w' = \int_{[0,1]^M} \prod_{C\in\mathscr{C}_1} c\ du_{\iota_1}\ldots du_{\iota_M}.$$

The pseudo code for the first case of Secondary Backward Sampling is presented in algorithm 4.

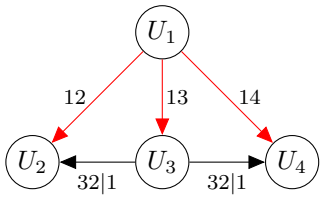


Figure 4.8: First Type of Secondary Backward Sampling.

Example 4.1.3. The steps of the Secondary Backward Sampling are illustrated on the example DAG shown in figure 4.8. We can see that the node U_3 may be Secondary Backward Sampled as it has no type 1 children and two type 2 children, U_2 and U_4. To obtain the Secondary Backward Sample of U_3, the sample from the conditional density

$$c_{3|24} \propto \int_0^1 c_{32|1}c_{34|1}c_{12}c_{13}c_{14}du_1$$

Algorithm 4 Sec.Backward.Sample.1($\mathscr{G}_{\mathscr{A}}, L, U_1, \{u_2, \ldots, u_{m_1}, u_{\eta_1}, \ldots, u_{\eta_{m_2}}\}$)

Input:
- $\mathscr{G}_{\mathscr{A}}$ an Extended Artery
- L the number of importance samples.
- U_1 the node to be Backward Sampled
- $\{u_2, \ldots, u_m\}$ values of U_1's type 2 children $\{U_2, \ldots, U_m\}$.

Output: Sample $\hat{u_1}$ of U_1.

1: $V_1, \ldots, V_L \sim \mathcal{U}(0,1)$ ▷ Uniform Samples
2: Identify the nodes $\{U_{\iota_1}, \ldots, U_{\iota_M}\}\}$ as defined in (4.2).
3: Identify the copula collection $\mathscr{C}_1$ as defined in (4.3).
4: **for** $k \in \{1, \ldots, L\}$ **do**
5: Estimate $w'_k \leftarrow \int_{[0,1]^M} \prod_{C \in \mathscr{C}_1} c \; du_{\iota_1} \ldots du_{\iota_M}$
6: **end for**
7: **for** $k \in \{1, \ldots, L\}$ **do**
8: $w_k = \frac{w'_k}{\sum_{l=1}^{N} w_l}$
9: **end for**
10: Sample $\hat{u_1}$ from $\{V_1, \ldots, V_L\}$ with weights $w_1, \ldots, w_L$
11: **return** $\hat{u_1}$

is required. The importance sample will be uniformly distributed. The unnormalized weight of each sample is equal to:

$$w' = \int_0^1 c_{32|1} c_{34|1} c_{12} c_{13} c_{14} du_1.$$

Then we resample from the importance samples using the normalized weights and get the required samples denoted as u_1^{sb}.

Because the importance samples of case 1 Secondary Backward Sampling contain no information on the evidence, we can expect this case to be less efficient. Furthermore, while the integral can be effectively estimated through Monte-Carlo simulation, estimation has to be done for each individual importance sample, thus increasing computational cost.

Case 2: In the second case we shall examine all other scenarios that do not meet the condition (B2), in which case, U_1 has at least one type 1 child.

In this scenario U_1 has:

- $m_1 - 1$ type 1 children $U_2, \ldots, U_{m_1}$ and
- m_2 type 2 children $U_{\eta_1}, \ldots, U_{\eta_{m_2}}$.

Furthermore, if K_{η_i} are the conditioning sets of the copulas $C_{1\eta_i|K_{\eta_i}}$, we collect all copulas for the nodes that are to be integrated:

$$\left(\bigcup_{i=1}^{m_2} K_{\eta_i} \right) \setminus \{U_{\eta_1}, \ldots, U_{\eta_{m_2}}\} = \{U_{\iota_1}, \ldots, U_{\iota_M}\}. \tag{4.4}$$

Moreover, we define the collection of copulas $\mathscr{C}_2$ which contains the copulas included in $\mathscr{C}_1$ as per equation 4.3 and all copulas between U_{ι_j} and type 1 children of U_1:

$$\mathscr{C}_2 \coloneqq \Big\{ C_{kl|K_{kl}} \in \mathscr{C} : \; U_k, U_l \in \{U_1, U_2, \dots, U_{m_1}, U_{\eta_1}, \dots, U_{\eta_{m_2}}, U_{\iota_1}, \dots, U_{\iota_M}\} \text{ and } (\{U_k, U_l\} \cup K_{kl}) \cap \{U_{\iota_1}, \dots, U_{\iota_M}\} \neq \emptyset \Big\}. \tag{4.5}$$

In this scenario we wish to sample form the density:

$$\begin{aligned} c_{1|2\dots m_1,\eta_1\dots\eta_{m_2}} &= \frac{c_{1,2\dots m_1,\eta_1\dots\eta_{m_2}}}{c_{2\dots m_1,\eta_1\dots\eta_{m_2}}} \\ &= \frac{\displaystyle\int_{[0,1]^M} c_{1,2\dots m_1,\eta_1\dots\eta_{m_2},\iota_1\dots\iota_M} du_{\iota_1} \dots du_{\iota_M}}{\displaystyle\int_{[0,1]^{M+1}} c_{1,2\dots m_1,\eta_1\dots\eta_{m_2},\iota_1\dots\iota_M} du_1 du_{\iota_1} \dots du_{\iota_M}} \\ &= \frac{\displaystyle\prod_{j=2}^{m_1} c_{1j|K_j} \int_{[0,1]^M} \prod_{C\in\mathscr{C}_2} c \; du_{\iota_1} \dots du_{\iota_M}}{\displaystyle\int_{[0,1]^{M+1}} \prod_{j=2}^{m_1} c_{1j|K_j} \prod_{C\in\mathscr{C}_2} c \; du_1 du_{\iota_1} \dots du_{\iota_M}} \\ &\propto \prod_{j=2}^{m_1} c_{1j|K_j} \int_{[0,1]^M} \prod_{C\in\mathscr{C}_2} c \; du_{\iota_1} \dots du_{\iota_M}. \end{aligned}$$

In contrast to case 1 Secondary Backward Sampling, in case 2 we have some copulas that are outside of the integral, making it possible to obtain an importance sample conditioned on type 1 children of U_1. We choose a type 1 child U_i of U_1 such that the for the specified copula being $C_{1i|K_i}$, the following set has the largest possible cardinality.

$$K_i^1 = \{U_{j_1}, \dots, U_{j_{m_i}}\} \coloneqq K_i \cap \{U_2, \dots, U_m\}, \tag{4.6}$$

where $U_{j_1}, \dots, U_{j_{m_i}}$ are indexed according to the parental order of U_i.

We get the importance samples of U_1 using the corresponding recursion of h-functions for the copula $C_{1i|K_i}$:

$$\hat{u1}_k^{imp} = h^{-1}_{\underline{1}j_1}(h^{-1}_{\underline{1}j_2|K_{j_2}}(\dots h^{-1}_{\underline{1}i|K_i}(V_k, u_{i|K_i}|\boldsymbol{u}_{K_{j_{m_i}}}) \dots, u_{j_2|K_{j_2}}|\boldsymbol{u}_{K_{j_2}}), u_{j_1}),$$

where V is a uniformly distributed variable.

The unnormalized weight of each sample is:

$$w' = \prod_{\substack{l=2 \\ j\notin K_i\cup\{i\}}}^{m_1} c_{1l|K_l} \int_{[0,1]^M} \prod_{C\in\mathscr{C}_2} c \; du_{\iota_1} \dots du_{\iota_M}.$$

Resampling the importance samples using the normalized weights we get the Secondary Backward Sample of U_1.

Algorithm 5 Sec.Backward.Sample.2($\mathscr{G}_{\mathscr{A}}, L, U_1, \{u_2, \ldots, u_{m_1}, u_{\eta_1}, \ldots, u_{\eta_{m_2}}\}$)

Input:
- $\mathscr{G}_{\mathscr{A}}$ an Extended Artery
- L the number of importance samples.
- U_1 the node to be Backward Sampled
- $\{u_2, \ldots, u_{m_1}, u_{\eta_1}, \ldots, u_{\eta_{m_2}}\}$ values of U_1's type 1 and type 2 children.

Output: Sample $\hat{u_1}$ of U_1.

$V_1, \ldots, V_L \sim \mathcal{U}(0,1)$ ▷ Uniform Samples
Identify the nodes $\{U_{\iota_1}, \ldots, U_{\iota_M}\}$ as defined in (4.4).
Identify the copula collection $\mathscr{C}_2$ as defined in (4.5).
Choose U_i that maximizes the cardinality of $\{U_{j_1}, \ldots, U_{j_{m_i}}\}$ as defined in (4.6)
for $k \in \{1, \ldots, L\}$ **do**
    $\hat{u_1}_k^{imp} \leftarrow h^{-1}_{\underline{1}j_1}(h^{-1}_{\underline{1}j_2|K_{j_2}}(\ldots(h^{-1}_{\underline{1}j_{m_i}|K_{j_{m_i}}}(V_k, u_{j_{m_i}|K_{j_{m_i}}}|\boldsymbol{u}_{K_{j_{m_i}}})\ldots), u_{j_2|K_{j_2}}|\boldsymbol{u}_{K_{j_2}}), u_{j_1})$
    $w'_k \leftarrow \prod_{\substack{l=2 \\ l \notin K_i \cup \{i\}}}^{m_1} c_{1l|K_l} \int_{[0,1]^M} \prod_{C \in \mathscr{C}_2} c \, du_{\iota_1} \ldots du_{\iota_M}$
end for
for $k \in \{1, \ldots, L\}$ **do**
    $w_k = \frac{w'_k}{\sum_{l=1}^{N} w_l}$
end for
Sample $\hat{u_1}$ from $\{\hat{u_1}_1^{imp}, \ldots, \hat{u_1}_L^{imp}\}$ with weights $w_1, \ldots, w_L$
return $\hat{u_1}$

The pseudo-code for the second case of Secondary Backward Sampling is presented in algorithm 5.

Example 4.1.4. The Secondary Backward Sampling for case 2 is illustrated by sampling U_2 in PCBN in Figure 4.9. The node U_2 has two type 2 children U_3, U_5 and one type 1 child, U_4.

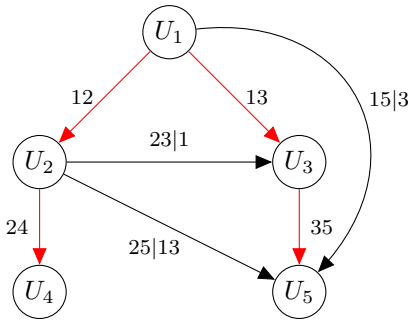


Figure 4.9: Second Type of Secondary Backward Sampling.

We want to sample from the density:

$$\begin{aligned} c_{2|345} &= \frac{c_{2345}}{c_{345}} \\ &= \frac{\int_0^1 c_{12345} du_1}{\int_{[0,1]^2} c_{12345} du_1 du_2} \\ &= \frac{c_{24} \int_0^1 c_{12} c_{13} c_{35} c_{23|1} c_{25|13} c_{15|3} du_1}{\int_{[0,1]^2} c_{24} c_{12} c_{13} c_{35} c_{23|1} c_{25|13} c_{15|3} du_1 du_2} \\ &\propto c_{24} \int_0^1 c_{12} c_{13} c_{35} c_{23|1} c_{25|13} c_{15|3} du_1. \end{aligned}$$

The copula C_{24} allows us to generate a sample from the distribution $U_2|U_4 = u_4$ which is used as an importance sample. Then each sample has the unnormalized weight

$$w' = \int_0^1 c_{12} c_{13} c_{35} c_{23|1} c_{25|13} c_{15|3} du_1,$$

which can be easily and efficiently estimated. We resample from the importance sample using the normalized weights to obtain our Secondary Backward Sample.

Finally the last type of sampling process will be explained.

4.1.3. Bilateral Sampling

Bilateral Sampling is used to combine information coming from children and parents of a node. A node U_1 is conditionalized on both extended arterial parents and extended arterial children.

In contrast to Secondary Backward Sampling, due to the addition of U_1's parents in the conditionalization for Bilateral Sampling, the extended arterial parents of type 2 children which are also extended arterial parents of U_1 will not be integrated.

In this scenario U_1 has:

- m_1 type 1 children $U_2, \ldots, U_{m_1}$,
- m_2 type 2 children $U_{\eta_1}, \ldots, U_{\eta_{m_2}}$ and
- m extended arterial parents $U_{\pi_1}, \ldots, U_{\pi_m}$

Furthermore, if K_{η_i} is the conditioning set of copula $C_{1\eta_i|K_{\eta_i}}$, then we define as previously:

$$\left(\bigcup_{i=1}^{m_2} K_{\eta_i} \right) \setminus (\{U_{\eta_1}, \ldots, U_{\eta_{m_2}}\} \cup pa_{\mathscr{G}_{\mathscr{A}}}(U_1)) = \{U_{\iota_1}, \ldots, U_{\iota_M}\}. \tag{4.7}$$

Note that we have also removed the extended arterial parents of U_1 as they will not be integrated over.

The collection of copulas $\mathscr{C}_2$ as defined in equation 4.5 is required at this sampling scheme.

Now the target density can be rewritten:

$$\begin{aligned}
c_{1|2\ldots m_1,\eta_1\ldots\eta_{m_2},\pi_1\ldots,\pi_m} &= \frac{c_{1,2\ldots m_1,\eta_1\ldots\eta_{m_2},\pi_1\ldots,\pi_m}}{c_{2\ldots m_1,\eta_1\ldots\eta_{m_2},\pi_1\ldots,\pi_m}} \\
&= \frac{\displaystyle\int_{[0,1]^M} c_{1,2\ldots m_1,\eta_1\ldots\eta_{m_2},\pi_1\ldots,\pi_m,\iota_1\ldots\iota_M} du_{\iota_1}\ldots du_{\iota_M}}{\displaystyle\int_{[0,1]^{M+1}} c_{1,2\ldots m_1,\eta_1\ldots\eta_{m_2},\pi_1\ldots,\pi_m,\iota_1\ldots\iota_M} du_1 du_{\iota_1}\ldots du_{\iota_M}} \\
&= \frac{\displaystyle\prod_{i=1}^{m} c_{\pi_i 1|K_{\pi_i}} \prod_{j=2}^{m_1} c_{1j|K_j} \int_{[0,1]^M} \prod_{C\in\mathscr{C}_2} c\ du_{\iota_1}\ldots du_{\iota_M}}{\displaystyle\int_{[0,1]^{M+1}} \prod_{j=2}^{m_1} c_{1j|K_j} \prod_{i=1}^{m} c_{\pi_i 1|K_{\pi_i}} \prod_{C\in\mathscr{C}_2} c\ du_1 du_{\iota_1}\ldots du_{\iota_M}} \\
&\propto \prod_{j=2}^{m_1} c_{1j|K_j} \prod_{i=1}^{m} c_{\pi_i 1|K_{\pi_i}} \int_{[0,1]^M} \prod_{C\in\mathscr{C}_2} c\ du_{\iota_1}\ldots du_{\iota_M}.
\end{aligned}$$

We may choose to obtain the importance sample of U_1 by Forward Sampling from the distribution $U_1|U_{\pi_1} = u_{\pi_1}\ldots U_{\pi_m} = u_{\pi_m}$. Therefore, each importance sample is calculated by:

$$\hat{u_1}^{imp} = Forward.Sample(\mathscr{G}_{\mathscr{A}}, U_1, \{u_{\pi_1},\ldots,u_{\pi_m}\}).$$

The unnormalized weight that corresponds to each of the importance samples is:

$$w = \prod_{j=2}^{m_1} c_{1j|K_j} \int_{[0,1]^M} \prod_{C\in\mathscr{C}_2} c\ du_{\iota_1}\ldots du_{\iota_M}.$$

Then the final Bilateral sample is obtained by resampling from the importance samples using the normalized weights:

Remark. To ensure efficiency of SIR implementation, it would be wise to compare the number of parents m to the cardinality of a maximally conditioned copula set K_j of an outgoing copula $C_{1j|K_j}$. If $|K_j| > m$ then one may choose that copula instead to procure the importance sample with unnormalized weights:

$$w' = \prod_{\substack{i=2 \\ U_i\notin K_j\cup\{U_j\}}}^{m_1} c_{1i|K_i} \prod_{i=1}^{m} c_{\pi_i 1|K_{\pi_i}} \int_{[0,1]^M} \prod_{C\in\mathscr{C}_2} c\ du_{\iota_1}\ldots du_{\iota_M}.$$

The notation of a Bilateral Sample of U_1 will depend on the node that will use it in the Sample Propagation stage. The notation will be $u^{bil}_{1\to i}$ for U_i being the node using the bilateral sample of U_1. The meaning of U_i using the bilateral sample of U_1 will be introduced in the next section.

The pseudo-algorithm for Bilateral Backward Sampling is presented in algorithm 5.

Algorithm 6 Bilateral.Sample($\mathscr{G}_{\mathscr{A}}, L, U_1, \{u_2, \dots, u_{m_1}, u_{\eta_1}, \dots, u_{\eta_{m_2}}, u_{\pi_1}, \dots, u_{\pi_m}\}$)

Input:
- $\mathscr{G}_{\mathscr{A}}$ an Extended Artery
- L the number of importance samples.
- U_1 the node to be Backward Sampled
- $\{u_2, \dots, u_{m_1}, u_{\eta_1}, \dots, u_{\eta_{m_2}}, u_{\pi_1}, \dots, u_{\pi_m}\}$ values of U_1's extended arterial children and parents.

Output: Sample $\hat{u_1}$ of U_1.

1: Identify the nodes $\{U_{\iota_1}, \dots, U_{\iota_M}\}$ as defined in (4.7).
2: Identify the copula collection $\mathscr{C}_2$ as defined in (4.5).
3: **for** $k \in \{1, \dots, L\}$ **do**
4: $\quad \hat{u_1}_k^{imp} \leftarrow Forward.Sample(\mathscr{G}_{\mathscr{A}}, U_1, \{u_{\pi_1}, \dots, u_{\pi_m}\})$
5: $\quad w'_k \leftarrow \prod_{j=2}^{m_1} c_{1j|K_j} \int_{[0,1]^M} \prod_{C \in \mathscr{C}_2} c \, du_{\iota_1} \dots du_{\iota_M}$
6: **end for**
7: **for** $k \in \{1, \dots, L\}$ **do**
8: $\quad w_k = \frac{w'_k}{\sum_{l=1}^{N} w_l}$
9: **end for**
10: Sample $\hat{u_1}$ from $\{\hat{u_1}_1^{imp}, \dots, \hat{u_1}_L^{imp}\}$ with weights $w_1, \dots, w_L$
11: **return** $\hat{u_1}$

Example 4.1.5. An example where Bilateral Sampling is applied is sampling U_1, using values of $U_{\pi_1}, U_{\pi_2}, U_3, U_5$ in PCBN in Figure 4.10.

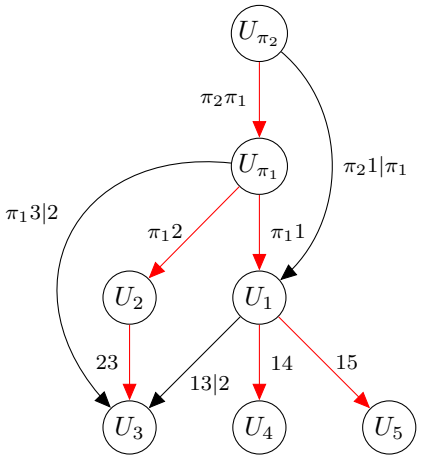


Figure 4.10: Example of Bilateral Sampling.

In this example the following density is required:

$$
\begin{aligned}
c_{1|345\pi_1\pi_2} &= \frac{c_{1345\pi_1\pi_2}}{c_{345\pi_1\pi_2}} \\
&= \frac{c_{14}c_{15}c_{\pi_1 1}c_{\pi_2 1|\pi_1}\int_0^1 c_{13|2}c_{23}c_{\pi_1 2}c_{\pi_1 3|2}\ du_2}{\int_0^1 c_{14}c_{15}c_{\pi_1 1}c_{\pi_2 1|\pi_1}\int_0^1 c_{13|2}c_{23}c_{\pi_1 2}c_{\pi_1 3|2}\ du_2\ du_1} \\
&\propto c_{14}c_{15}c_{\pi_1 1}c_{\pi_2 1|\pi_1}\int_0^1 c_{13|2}c_{23}c_{\pi_1 2}c_{\pi_1 3|2}\ du_2.
\end{aligned}
$$

If we choose to use the Forward Sample as an importance sample, we sample from the parents U_{π_1}, U_{π_2} with assigned weights:

$$w_i = c_{14}c_{15}\int_0^1 c_{13|2}c_{23}c_{\pi_1 2}c_{\pi_1 3|2}\ du_2.$$

Then we can resample with the normalized weights.

4.2. Single Arterial Propagation

In this section the main steps of propagation in the single extended artery are presented. The propagation is done via sampling which is sequential, local and falls in line with the theoretical foundation of Pearl's Algorithm on Belief Propagation. The main difference being that in our algorithm, messages take the form of samples instead of distributions.

During the procedure a node may be sampled multiple times through different types of sampling. For this reason we introduce the following notation for the four types of Node Sampling:

- u_1^f : Forward Sample of U_1.
- u_1^b : Backward Sample of U_1.
- u_1^{sb} : Secondary Backward Sample of U_1.
- $u_{1\to i}^{bil}$: Bilateral Sample of U_1 excluding U_1's child U_i and U_i's arterial descendants.
- u_1^* : Bilateral Sample of U_1 using all parents and children.

The reasoning behind the distinction between $u_{1\to i}^{bil}$ and u_1^* will be made clear in the following presentation.

The Algorithm for Single Arterial Inference consists of the following four steps:

Stage 1: Initialization - In this stage, all evidence nodes in the artery are identified, and their Forward, Backward, Secondary Backward and Bilateral Samples are set to their respective observed values. Furthermore, nodes that can be removed from the Backward Propagation procedure are identified.

Stage 2: Backward Propagation - This stage is composed of a sequence of Backward and Secondary Backward Samples. The Backward Propagation starts at evidence nodes and moves upwards, towards the root of the artery.

Stage 3: Forward Propagation - consists of a sequence of Forward and Bilateral Samples which follow the directionality of the DAG and moves from the root of the artery towards the leaves.

Stage 4: Finalization - combines all necessary and previously calculated samples to obtain the final samples of the nodes of the PCBN, each being conditionalized on all of the evidence in the PCBN.

Example 4.2.1. Let us start with a simple example of inference in PCBN in figure 4.11. The evidence is $\mathbf{E} = \{U_1 = u_1, U_5 = u_5\}$. The Pearl's algorithm would follow the following steps:

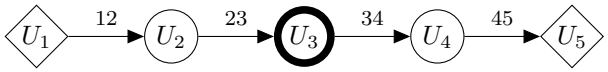


Figure 4.11: Arterial Inference Problem

- Initialization step, in which the evidences' parameters are set:

$$\pi(u_1),\ \lambda(u_5).$$

- Then the messages are calculated : $\pi_2(u_1),\ \lambda_5(u_4)$.
- Next, the causal and diagnostic parameters for U_2 and U_4 are calculated from the messages: $\pi(u_2),\ \lambda(u_4)$.
- This is followed by computing messages : $\pi_3(u_2),\ \lambda_4(u_3)$.
- Finally using parameters $\pi(u_3),\ \lambda(u_3)$, the updated distribution of U_3 can be calculated.

Thus, by combining the causal and diagnostic messages to U_3, we can arrive to the conditional distribution of U_3 given $U_1 = u_1$ and $U_5 = u_5$.

Having Pearl's Algorithm in Belief Propagation as a point of reference, an overview of our proposed methodology applied to this simple example is the following:

Stage 1: Initialization Due to all messages and parameters taking the form of samples, the casual and diagnostic parameters in the initialization phase are exactly the evidence values. The counterparts to the parameters $\pi(u_1),\ \lambda(u_5)$ would be just the evidence values $u_1,\ u_5$.

Stage 2: Backward Propagation The next step would be to sample from the distribution $U_4|U_5 = u_5$. The obtained sample is:

$$u_4^b.$$

Note that the sample u_4^b is a sample from the distribution whose density corresponds to diagnostic parameter:

$$\lambda(u_4) = c_{5|4}(u_5|u_4).$$

Stage 2: Forward Propagation The next step would be to sample from the distribution $U_2|U_1 = u_1$. The obtained sample is:

$$u_2^f.$$

This sample comes from the distribution whose density corresponds to the causal parameter:

$$\pi(u_2) = c_{2|1}(u_2|u_1).^1$$

Stage 3: Finalization The last step is to sample from the distribution $U_3|U_1 = u_1,\ U_5 = u_5$ to obtain the sample u_3^*. This is done by Bilateral Sampling from the distribution $U_3|U_2 = u_2^f,\ U_4 = u_4^b$, where the conditionalization on $U_2,\ U_4$ is on their previously sampled values.

As seen in this simple example, belief propagation through sampling requires the three different types of sampling. Backward Sampling was used to sample u_4^b, while Forward Sampling was used for sampling u_2^f. Lastly, Bilateral sampling was used to combine both of the previous Forward and Backward Samples to get u_3^*.

Next, each of the presented steps is discussed in more details.

4.2.1. Initialization

During the Initialization stage, the preparatory steps for sample propagation are taken. The first step to our algorithm is to identify nodes which directly provide information on the evidence.

Clearly the evidence nodes fall under this category. An evidence node $U_1 \in \mathbf{E}$ will not be sampled in the following stages, but it will need to propagate the information of its observed value to its parents and children. Therefore, for all evidence variables $U_i = u_i \in \mathbf{E}$ we fix their values:

- The Forward Sample: $u_1^f = u_i$.
- The Backward Sample: $u_1^b = u_i$.
- The Secondary Backward Sample: $u_1^{sb} = u_i$.
- The Bilateral Samples: $u_{i\to j}^{bil} = u_i$, for every child $U_j \in ch_{\mathscr{G}_{\mathscr{A}}}(U_i)$.

Apart from evidence nodes, the initialization phase of the algorithm also identifies and groups nodes according to whether they have descendants in the evidence. This is a preparatory step required for choosing the nodes that take part in future Backward, Secondary Backward and Bilateral Sampling.

Let us consider a node U_1 belonging to an extended artery $\mathscr{G}_{\mathscr{A}}$.

- If $de_{\mathscr{A}}(U_1) \cap E = \emptyset$ then U_1 is called **Uniformly Backward Instantiated (UBI)**.
- If $de_{\mathscr{G}_{\mathscr{A}}}(U_1) \cap E = \emptyset$ then U_1 is called **Uniformly Secondary Backward Instantiated (USBI)**.

[1] Note that in this example there are no counterparts of causal and diagnostic messages, but only of parameters. Later, in more complex examples we will see how $u_{i\to j}^{bil}$ samples are counterparts to messages.

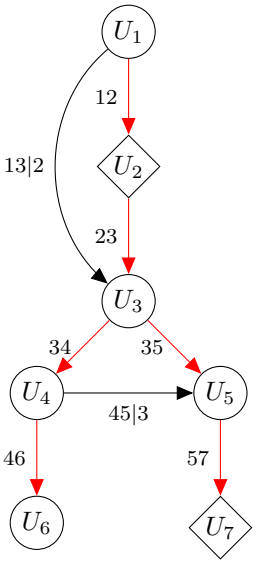


Figure 4.12: Example of the Initialization stage.

Let us consider as an example the PCBN inference problem in figure 4.12.

While any node that is $USBI$ is also UBI, the opposite is not true as can be easily seen in this example. U_6 has evidence in neither its arterial nor extended arterial descendants. This means U_6 is both UBI and USBI. On the other hand, U_4 has no evidence in its arterial descendants but has the evidence U_7 in its extended arterial descendants. Therefore U_4 is $USBI$ but not UBI.

Whether a node is UBI or USBI is determined by the DAG structure. If a node is both USBI, then it will not participate in any type of Backward or Secondary Backward Sampling during the Backward Propagation stage. If a node is UBI but not USBI, depending on the DAG structure, there may be cases where a Secondary Backward Sample of the node is be necessary. We will discuss the proper use of Secondary Backward Samples and UBI nodes in greater detail next.

4.2.2. Backward Propagation

The Backward Propagation procedure starts after the Initialization stage. This propagation follows reverse sampling order of the extended arterial PCBN, starting from the last evidence nodes. The process then moves towards the arterial parents of the earlier nodes in the propagation procedure.

At each iteration of Backward Propagation, we wish to Backward Sample or Secondary Backward Sample a variable U_1 conditioned on all of the evidence variables that are arterial descendants or extended arterial descendants of U_1, respectively.

Backward Propagation is **finalized** when all of the nodes U that have not taken part in Backward Propagation satisfy the following conditions:

1. $U \in E$
2. $pa_{\mathscr{G}_{\mathscr{A}}}(U) \subseteq E$

(BP) 3. U is USBI

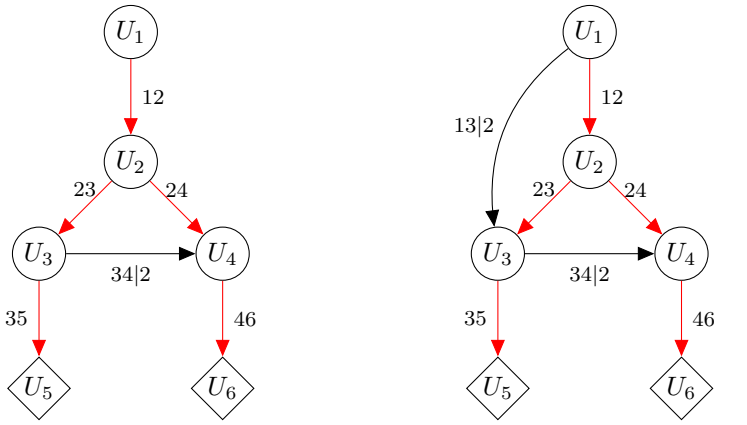
4. U is UBI and there does not exist $U' \in pa_{\mathscr{G}_{\mathscr{A}}}(U)$ with specified copula $C_{U',U|K}$ such that there exists an arterial diverging node in K

Thus, the process is finalized when all remaining nodes are not required to be sampled in order to propagate the evidence to their parents.

In the first condition, U is an evidence node and it is not sampled. In the second condition, U does not have any extended arterial parents that require a Backward Propagated Sample of U. The third condition means that U is USBI and its Propagated Sample would contain no information on the evidence. The fourth condition refers to the cases when it is relevant to Backward Propagate information on type 2 children.

As a result of initialization of Backward Propagation at the last in the order evidence nodes and propagating towards the root, USBI nodes do not participate in the procedure of Backward Propagation. Furthermore, as they do not include any information on descendants in the evidence (as they do not have any), USBI nodes are also omitted when Sampling of their parents is performed.

Next we will elaborate on the distinction between Backward and Secondary Backward Sampling and the motivation behind keeping these two sampling types separate.



(a) Example where Secondary Backward Sampling is not required.

(b) Example where Secondary Backward Sampling is required.

Figure 4.13: Examples illustrating Secondary Backward Sampling.

Backward Sampling uses only samples of U_1's type 1 children.

The reason for the exclusion of type 2 children is the fact that for U_i being a type 2 child of U_1, the copula $C_{1i|K}$ specified by the arc $U_1 \to U_i$ has a conditioning set K that includes the arterial parent of U_1. For example, in figure 4.13a, U_4 is a type 2 child of U_3 and the copula $C_{34|2}$ assigned to arc $U_3 \to U_4$ is conditioned on U_2. Integrating over the values of that parent in some way will be required in Secondary Backward Sampling. However, for the Backward Sample of the parent, we cannot use a sample obtained through integration of that same parent.

In chapter 8 we generalize this statement and prove that for any type 2 child U_2 of a node U_1, the assigned copula $C_{12|K_{12}}$ is conditioned on the arterial parent of U_1. Therefore, we are never able to use type 2 children in Backward Sampling.

In such cases the inclusion of type 2 children through Secondary Backward Sampling is required for upcoming Backward Sampling. The condition for Secondary Backward Sampling to be used is that U_1 has a parent U_2 such that:

(BS) The copula $C_{21|K}$ which is specified by the arc $U_2 \to U_1$ has a conditioning set K that includes a diverging node in the artery.

Note that Secondary Backward Sampling does not serve as a substitute to regular Backward Sampling. Whenever Secondary Backward Sampling is required, we will be using two samples of a node, each applied in the Backward Propagation towards different parents.

Example 4.2.2. The examples of PCBNs in figure 4.13 help us illustrate the Secondary Backward Sampling in the context of Backward sampling of U_3. In both networks 4.13a and 4.13b, U_3 has a type 2 child U_4 and a type 1 child U_5, while U_2 is its arterial parent. Furthermore, we notice that U_2 is a diverging node in the artery. In the network in figure 4.13a, there is no parent of U_3 satisfying the condition (BS). On the other hand, in the network illustrated in figure 4.13b, U_1 is an extended arterial parent of U_3 such that

- The specified copula $C_{13|2}$ is conditioned on node U_2 which is a diverging node in the artery.

Therefore in the second network, U_3 satisfies the condition (BS) and will need to be both Backward Sampled and Secondary Backward Sampled including the type 2 child U_4. This Secondary Backward Sample will be used later on for the Backward Sample of U_1.

Next we will analyze the types of samples used during each Backward and Secondary Backward Sampling. As mentioned, USBI nodes are completely omitted from the Sampling, thus we are left with UBI and non-UBI nodes.

For calculating either the Backward Sample or the Secondary Backward Sample of a node U_1, we will use the following samples of its children.

(Ch)
- A type 1 child U_2 will be:
  - (Ch1) Secondary Backward Sampled if the specified copula $C_{12|K}$ includes a diverging arterial node in the conditioning set K that is a parent of U_1.
  - (Ch2) Backward Sampled if it is not UBI and the condition (Ch1) is not satisfied.
  - (Ch3) Omitted if it is UBI and the condition (Ch1) is not satisfied.
- All type 2 children will be Secondary Backward Sampled.

Example 4.2.3. Let us illustrate the procedure of Backward Propagation in the context of the inference problem in figure 4.13a. In this case Secondary Backward Sampling of U_3 is not required. The procedure is the following:

$$u_4^b \rightsquigarrow u_3^b \rightsquigarrow u_2^b \rightsquigarrow u_1^b$$

On the other hand, in the inference problem seen in figure 4.13b, we require the Secondary Backward Sample of U_3 to propagate to U_1, while the regular Backward Sample of U_3 is used for Backward Propagation towards U_2. The Backward Propagation procedure in this scenario is:

$$u_4^b \rightsquigarrow u_3^b,\ u_3^{sb} \rightsquigarrow u_2^b \rightsquigarrow u_1^b$$

The pseudo-code corresponding to the Backward Propagation Procedure is shown in algorithm 7.

Larger Example of Backward Propagation Let us consider the example of figure 4.14.

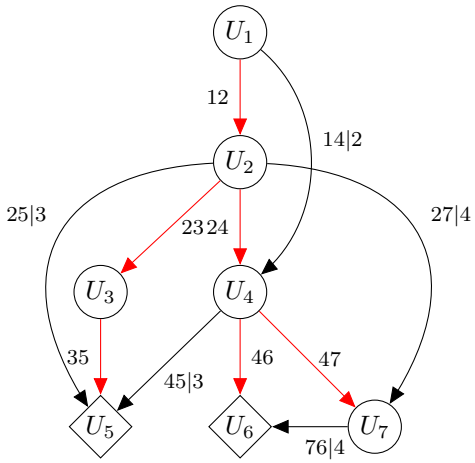


Figure 4.14: Example of Backward Propagation.

In this example we have two evidence nodes U_5, U_6, no USBI nodes and one UBI node U_7.

The Backward Propagation starts at the evidence nodes, which have no descendants. The iterations of Backward Propagation are as follows:

Step 1: Set $u_5^b = u_5^{sb} = u_5$ and $u_6^b = u_6^{sb} = u_6$ to the observed values of the evidence.

Step 2: U_3 is not UBI and (B1) holds. Therefore we sample u_3^b using case 1 Backward Sampling from the distribution:

$$U_3|U_5 = u_5^b.$$

Step 3: U_7 is UBI, therefore we do not find its Backward Sample. On the other hand, condition (BS) holds along with condition (B2). Therefore we sample u_7^{sb} with case 1 Secondary Backward Sampling from the distribution:

$$U_7|U_6 = u_6^{sb}.$$

Algorithm 7 Backward.Propagation($\mathscr{G}_{\mathscr{A}}, \boldsymbol{E}, \boldsymbol{e}, A^{UBI}, A^{USBI}$)

Input:
- $\mathscr{G}_{\mathscr{A}}$ an Extended Artery
- $\boldsymbol{E}$ the evidence nodes.
- $\boldsymbol{e}$ the evidence values.
- A^{UBI} the set of UBI nodes.
- A^{USBI} the set of USBI nodes.

Output: $\{u_i^b\}_{i=1...m}, \{u_i^{sb}\}_{i=1...m}$: collections of Backward and Secondary Backward samples.

$\{U_1, \ldots, U_m\} \leftarrow$ the nodes in $V_{\mathcal{A}} \backslash (A^{USBI})$ ordered in the reverse PCBN sampling order.
$i = m$
while (BP) is not satisfied **do** ▷ Propagation Loop
  if $U_i \in \boldsymbol{E}$ **then** ▷ Assign evidence values
    $u_i^b \leftarrow e_i$
    $u_i^{sb} \leftarrow e_i$
  else if $pa_{\mathscr{G}_{\mathscr{A}}}(U_i) \cap \boldsymbol{E} \neq \emptyset$ **then** ▷ Exclude nodes with evidence parents
    $Ch^1 \leftarrow$ type 1 children of U_m that are not USBI
    $Ch^2 \leftarrow$ type 2 children of U_m that are not USBI
    $\forall U_l \in Ch^1$ choose u_l from $u_l^b, u_l^{sb}, \boldsymbol{e}$ according to (Ch).
    if (B1) holds and U_i is not UBI **then** ▷ Type 1 Backward Sampling
      $u_i^b \leftarrow Backward.Sample.1(\mathscr{G}_{\mathscr{A}}, U_i, \{\boldsymbol{u}_{Ch^1}\})$
    else if U_i is not UBI **then** ▷ Type 2 Backward Sampling
      $u_i^b \leftarrow Backward.Sample.2(\mathscr{G}_{\mathscr{A}}, L, U_i, \{\boldsymbol{u}_{Ch^1}\})$
    end if
    if (BS) holds **then**
      $\forall U_l \in Ch^1$ choose $u_l \leftarrow u_l^{sb}$.
      if (B2) **then** ▷ Type 1 Secondary Backward Sampling
        $u_i^{sb} \leftarrow Sec.Backward.Sample.1(\mathscr{G}_{\mathscr{A}}, L, U_i, \{\boldsymbol{u}_{Ch^2}\})$
      else ▷ Type 2 Secondary Backward Sampling
        $u_i^{sb} \leftarrow Sec.Backward.Sample.2(\mathscr{G}_{\mathscr{A}}, L, U_i, \{\boldsymbol{u}_{Ch^1}, \boldsymbol{u}_{Ch^2}\})$
      end if
    end if
  end if
  $i = i - 1$
end while
return $\{u_i^b\}_{i=1...m}, \{u_i^{sb}\}_{i=1...m}$

Step 4: U_4 is not UBI and condition (B1) is not satisfied. Therefore we sample u_4^b using case 2 Backward Sampling from the distribution:

$$U_4|U_6 = u_6^b, U_7 = u_7^b.$$

Furthermore, condition (BS) holds with condition (B2) not being satisfied. Therefore we sample u_4^{sb} with case 2 Secondary Backward Sampling from the distribution:

$$U_4|U_5 = u_5^{sb}, U_6 = u_6^b, U_7 = u_7^b.$$

Step 5: U_2 is not UBI and condition (B1) is not satisfied. Therefore we sample u_2^b using case 2 Backward Sampling from the distribution:

$$U_2|U_3 = u_3^b, U_4 = u_4^b, U_5 = u_5^b, U_7 = u_7^{sb}.$$

Step 6: U_1 is not UBI and condition (B1) is satisfied. Therefore we sample u_1^b using case 1 Backward Sampling from the distribution:

$$U_1|U_2 = u_2^b, U_4 = u_4^{sb}.$$

Then the Backward Propagation Procedure is finalized due to the conditions (BP) being satisfied.

The sampling procedure can be illustrated in the following sequence of samples:

$$u_5^b, u_5^{sb}, u_6^b, u_6^{sb} \rightsquigarrow u_3^b \rightsquigarrow u_7^{sb} \rightsquigarrow u_4^b \rightsquigarrow u_4^{sb} \rightsquigarrow u_2^b \rightsquigarrow u_1^b.$$

4.2.3. Forward Propagation

Forward Propagation is a sequence of Forward Samples and Bilateral Samples. We initialize the Forward Propagation procedure after the Backward Propagation is finalized. It starts from the root of the artery and follows the ordering of nodes similarly as the PCBN simulation.

If the next node U_i in the Forward Propagation process is part of the evidence, we simply skip it and continue by sampling the next node U_{i+1} in the process, using $u_i^f = u_i$.

Forward Propagation is **finalized** when all of the nodes U that have not taken part in Forward Propagation satisfy either of the following two conditions:

(FP)
- $U \in E$ or
- $ch(U) \subseteq E$

Whether the Forward or Bilateral Sampling of a node U_1 is conducted depends on the arc-copula structure of U_1 and its children. The following condition is checked:

U_1 is Forward Sampled if for each extended arterial child U_2 of U_1:

(FS)
$$ch_{\mathcal{G}_{\mathcal{A}}}(U_1) \subseteq an_{\mathcal{G}_{\mathcal{A}}}(U_2) \cup de_{\mathcal{G}_{\mathcal{A}}}(U_2) \cup \{U_2\}.$$

This means that all children of U_1 belong to the same arterial branch.

(Bil) U_1 is Bilaterally Sampled if the previous condition is violated.

In the second case when the Bilateral Sampling is applied, the continuation of Forward Propagation procedure towards the children may require different Bilateral Samples of U_1.

In Bilateral Sampling of U_1, later used in Forward Propagation to child U_2, we use all parents of U_1 as well as all of U_1's children that are neither ancestors nor descendants of U_2. This means that the following set of nodes is used:

$$ch_{\mathscr{G}_{\mathscr{A}}}(U_1)\backslash(an_{\mathscr{G}_{\mathscr{A}}}(U_2) \cup de_{\mathscr{G}_{\mathscr{A}}}(U_2)).$$

The reason is that each of U_1's arterial children's forward samples, require U_1's sample to be conditionalized also on the evidence that is part of the descendants of the remaining children, but not on those that are already part of the Forward Sample of U_2 (i.e. the descendants and ancestors of U_2).

The procedure of each Forward Propagation iteration at a node U_1 is as follows:

1. If U_1 is an evidence node, keep the observed value as the sample u_1^f of U_1.
2. If U_1 is not an evidence node and satisfies the condition (FS), obtain a sample u_1^f by Forward Sampling using all of U_1's parents´ Forward Samples.
3. If U_1 is not an evidence node and does not satisfy condition (FS), for each of U_1's extended arterial child U_2 obtain a Bilateral Sample $u_1^{\rightarrow 2}$ of U_1 using all of U_1's parents and all children nodes in:

$$ch_{\mathscr{G}_{\mathscr{A}}}(U_1)\backslash(an_{\mathscr{G}_{\mathscr{A}}}(U_2) \cup de_{\mathscr{G}_{\mathscr{A}}}(U_2) \cup \{U_2\}).$$

In order to avoid multiple evaluations of the same samples we may only calculate one sample for each arterial branch stemming from U_1 and use that sample as a Bilateral Sample towards all other nodes of the same branch.

Thus we only need to calculate the Bilateral samples for the children that belong to the node set:

(Bil.F) $$ch_{\mathscr{A}}(U_i)\backslash A^{USBI} \bigcup \{U_j \in Ch^2 : C_{1j|K} \in \mathscr{C},\ |K| = 1\}.$$

The pseudocode of Forward Propagation is presented in algorithm 8.

Example 4.2.4. Let us consider a PCBN in figure 4.15a, where $E = \{U_1\}$. The Forward Propagation in this case would be conducted without the use of Bilateral Sampling until reaching the leaves of the PCBN. The Forward Propagation in this example is:

$$u_1 \rightsquigarrow u_2^f \rightsquigarrow u_3^f \rightsquigarrow u_4^f \rightsquigarrow u_5^f \rightsquigarrow u_6^f.$$

If $E \subseteq \{U_1, U_2, U_3\}$, then the same holds, and the sampling process does not stop in the arterial diverging node U_4.

The PCBN shown in figure 4.15b also does not require Bilateral Sampling but only Forward Sampling of node U_4. This is because U_4 belongs to the evidence. In this case the Forward Propagation procedure is:

Algorithm 8 Forward.Propagation($\mathscr{G}_{\mathscr{A}}, \boldsymbol{E}, \boldsymbol{e}, A^{UBI}, A^{USBI}$)

Input:
- $\mathscr{G}_{\mathscr{A}}$ an Extended Artery
- $\boldsymbol{E}$ the evidence nodes.
- $\boldsymbol{e}$ the evidence values.
- A^{UBI} the set of UBI nodes.
- A^{USBI} the set of USBI nodes.

Output: $\{u_i^f\}, \{u_{i\to j}^{bil}\}$: collections of Backward and Secondary Backward samples.

$\{u_i^b\}, \{u_i^{sb}\} \leftarrow Backward.Propagation(\mathscr{G}_{\mathscr{A}}, \boldsymbol{E}, \boldsymbol{e}, A^{UBI}, A^{USBI})$
$\{U_1, \dots, U_m\} \leftarrow V_{\mathscr{A}}$ ordered in the PCBN sampling order.
$i = 1$
while (FP) is not satisfied **do** ▷ Propagation Loop
  if $U_i \in \boldsymbol{E}$ **then** ▷ Assign evidence values
    $u_i^f \leftarrow e_i$
    for $U_j \in ch_{\mathscr{G}_{\mathscr{A}}}(U_i)$ **do**
      $u_{i\to j}^{bil} \leftarrow e_i$
    end for
  else if $i = 1$ and $|ch_{\mathscr{A}}(U_i)| = 1$ **then** ▷ Root not in evidence
    Sample $u_i^f \sim U(0,1)$
  else if $ch_{\mathscr{G}_{\mathscr{A}}}(U_i) \cap \boldsymbol{E} \neq \emptyset$ **then** ▷ Exclude nodes with evidence children
    $P \leftarrow$ extended arterial parents of U_i
    $Ch^1 \leftarrow$ type 1 children of U_m that are not USBI.
    $Ch^2 \leftarrow$ type 2 children of U_m that are not USBI.
    $\forall U_l \in P$ choose u_l according to (FS) and (Bil).
    $\forall U_i \in Ch^1, Ch^2$ choose u_i from u_i^b, u_i^{sb} according to (Ch).
    if (FS) holds **then**
      $u_i^f \leftarrow Forward.Sample(\mathscr{G}_{\mathscr{A}}, U_i, \boldsymbol{u}_P)$ ▷ Forward Sampling
    else
      for $U_j \in$ (Bil.F) **do** ▷ Bil.Sampling for each arterial branch
        $Ch \leftarrow (Ch^1 \cup Ch^2) \backslash (\{U_j\} \cup de_{\mathscr{A}(U_j)})$
        $u_{i\to j}^{bil} \leftarrow Bilat.Sampling(\mathscr{G}_{\mathscr{A}}, U_i, \boldsymbol{u}_P, \boldsymbol{u}_{Ch})$ ▷ Bilateral Sample
        for $U_l \in de_{\mathscr{A}}(U_j) \cap ch_{\mathscr{G}_{\mathscr{A}}}(U_i)$ **do**
          $u_{i\to l}^{bil} \leftarrow u_{i\to j}$
        end for
      end for
    end if
  end if
  $i = i + 1$
end while
return $\{u_i^f\}, \{u_{i\to j}^{bil}\}$

$$u_1 \rightsquigarrow u_2^f \rightsquigarrow u_3^f \rightsquigarrow u_4^f = u_4 \rightsquigarrow u_5^f \rightsquigarrow u_6^f.$$

Finally, for the PCBN in figure 4.15c, the Forward Propagation would reach U_3 with only Forward Samples but then it would require the arterial diverging node U_4 to be Bilaterally Sampled. The Forward Propagation procedure is:

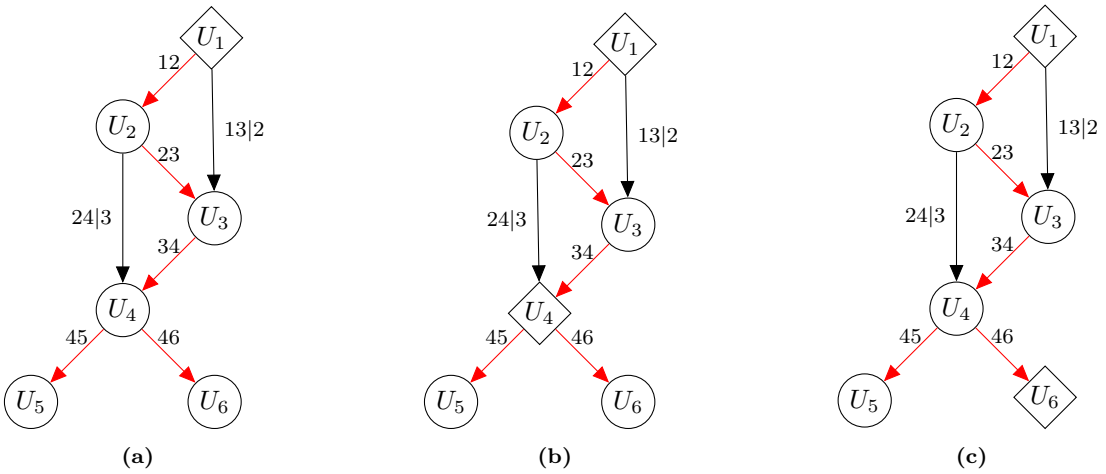


Figure 4.15: Examples illustrating Forward Propagation.

$$u_1 \rightsquigarrow u_2^f \rightsquigarrow u_3^f \rightsquigarrow u_4^{\rightarrow 5} \rightsquigarrow u_5^f.$$

Let us analyze the Forward Propagation procedure for the example in figure 4.15c. Assuming that the results of Backward Propagation are available, we begin Forward Propagation from the root U_1.

Step 1: Set $u_1^f = u_{1\rightarrow 2}^{bil} = u_{1\rightarrow 3}^{bil} = u_1$ as the observed value of the evidence node U_1.

Step 2: U_2 satisfies the condition (FS). Therefore we sample u_2^f using Forward Sampling from the distribution:

$$U_2|U_1 = u_1^f.$$

Step 3: U_3 satisfies the condition (FS). Therefore we sample u_3^f using Forward Sampling from the distribution:

$$U_3|U_1 = u_1^f, U_2 = u_2^f.$$

Step 4: U_4 does not satisfy the condition (FS). Therefore we need to create Bilateral Samples to be sent to each of U_4's children. The child U_6 is part of the evidence, therefore, the sample $u_{4\rightarrow 6}^{bil}$ is not needed. For the child U_5 we sample $u_{4\rightarrow 5}^{bil}$ using Bilateral Sampling from the distribution:

$$U_4|U_2 = u_2^f, U_3 = u_3^f, U_6 = u_6^b.$$

Step 5: U_5 satisfies the condition (FS). Therefore we sample u_5^f using Forward Sampling from the distribution:

$$U_5|U_4 = u_{4\rightarrow 5}^{bil}.$$

Then the Forward Propagation Procedure is finalized due to the conditions (FP) being satisfied.

4.2.4. Finalization

During the Finalization stage, for all $U_1 \in Q$ samples of the form

$$U_1|\boldsymbol{E}$$

are calculated. Three cases are considered at this step depending on the query node U_1:

1. If U_1 has no extended arterial descendants in the evidence, then its Forward Samples are used.
2. If U_1 has no extended arterial ancestors in the evidence, then the Secondary Backward Samples are used.
3. If U_1 has both extended arterial ancestors and descendants in the evidence, then its Bilateral Samples are used.

This concludes our algorithm on Single-Arterial PCBN Sample Propagation.

4.3. Implementation

In this section we will illustrate a full example of Arterial Sample Propagation in PCBNs. We implement our methodology in `R` and compare the results with the theoretical conditional distributions.

To make this comparison we only Gaussian Copulas, so as to create a GBN, whose inference solution is known. We discuss this choice and the calculation of the theoretical conditional distribution in greater detailed in chapter 6 of our simulation study.

In figure 4.16 the PCBN is illustrated. As seen in the figure, nodes U_1, U_7 and U_8 form the evidence in our problem.

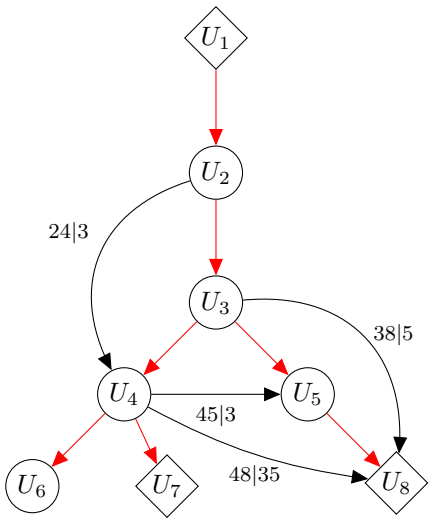


Figure 4.16: PCBN structure of example in which we apply Arterial Sample Propagation.

In the collection $\mathscr{F}$, all variables in the PCBN are assigned with a Gaussian $\mathcal{N}(0,1)$ marginal distribution.

The inference problem that we will solve for is conditioned on the evidence

$$X_1 = -0.75,\ X_7 = 0.5 \text{ and } X_8 = -0.5,$$

where X_i are the (Gaussian) transformed variables in the PCBN.

When transforming those values back to their uniform counterparts, the evidence takes the form

$$U_1 = -0.2266,\ U_7 = 0.6915 \text{ and } U_8 = -0.3085.$$

Both Gaussian copula parameters and evidence observation values were chosen manually for this example, with an effort to introduce some randomness, though not truly random. A restriction was placed to avoid extreme values. We talk more about this restriction in chapter 6.

In table 4.1 the chosen copula parameters for this example are shown for each of the specified copulas in the PCBN.

Copula	**Parameter**
C_{12}	-0.6
C_{23}	0.8
C_{34}	0.5
C_{35}	0.6
C_{56}	-0.4
C_{47}	0.75
C_{48}	-0.5
$C_{24\|3}$	0.4
$C_{36\|5}$	0.5
$C_{45\|3}$	-0.4
$C_{46\|35}$	0.4

Table 4.1: Gaussian Copula Parameters

Let us start the Arterial Sample Propagation. The implementation was done for both Sample and Importance Sample Sizes of 10^4:

Initialization We initialize by setting:

$$u_1^f = u_1^b = u_1^{sb} = u_1^{bil} = 0.2266$$
$$u_7^f = u_1^b = u_7^{sb} = u_7^{bil} = 0.6915$$
$$u_8^f = u_8^b = u_8^{sb} = u_8^{bil} = 0.3085$$

We additionally identify UBI and USBI nodes. U_7 is a USBI node and will not take part in Backward Propagation.

Backward Propagation U_4 has an incoming arc $U_2 \to U_4$ assigned with a copula $C_{24|3}$ conditioned on the arterial diverging node U_3. Hence we must also Secondary Backward Sample U_4.

The Backward Propagation Process is:

$$u_4^b, u_5^b \rightsquigarrow u_4^{sb}, u_3^b$$

Forward Propagation U_3 is an arterial diverging node with evidence in either side of its arterial branches it creates. Thus we must Bilateral Sample U_3 to send its sample to either U_4 or U_5

The Forward Propagation Process is:

$$u_2^f \rightsquigarrow u_{3\to4}^{bil}, u_{3\to5}^{bil} \rightsquigarrow u_4^f \rightsquigarrow u_6^f$$

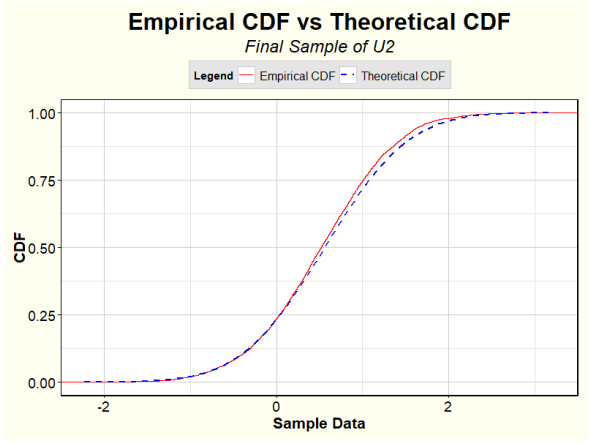
Finalization Here the final samples for nodes U_2, U_3, U_4, U_5 and U_6 are calculated.

Note that the final sample of U_6 is equivalent to u_6^f due to it only having parents.

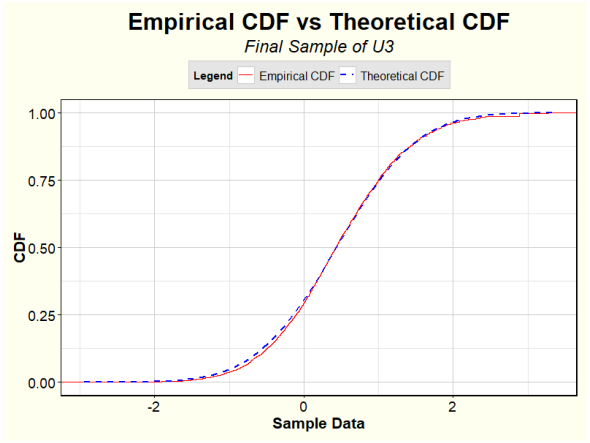
Lastly all final samples are transformed into their Gaussian counterparts, which marks the end of Arterial Sample Propagation.

Let us now discuss the performance of Arterial Sample Propagation in this example. To compare the empirical distributions of our samples to the theoretical ones provided by GBN theory, we plot the empirical CDF of the samples against the theoretical Gaussian CDFs.

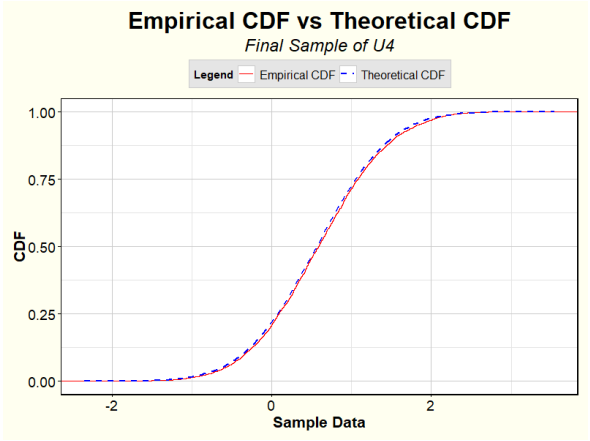
We can see the excellent performance of Arterial Sample Propagation in the plots presented in figure 4.17. For all nodes, the distributions of the samples followed closely the respective theoretical distributions. Furthermore, the total execution time for the propagation was 47.1 seconds, of which 47 were spent during Secondary Backward Sampling of U_4.



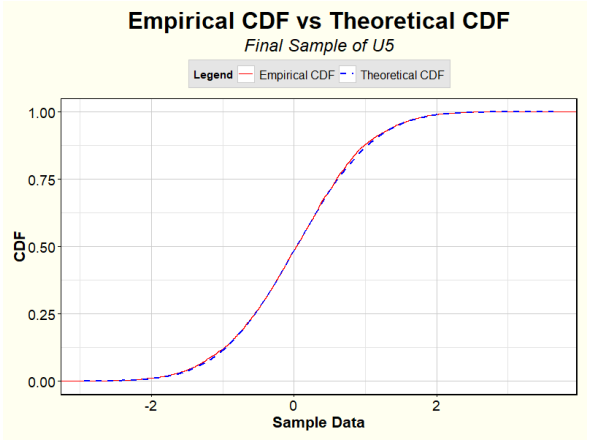
(a)



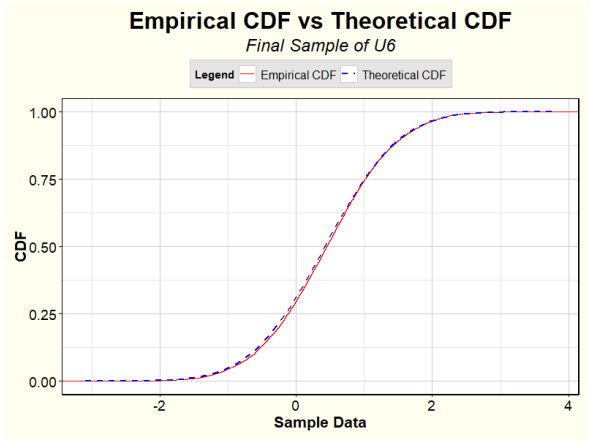
(b)



(c)



(d)



(e)

Figure 4.17: Comparison of empirical CDFs of final samples and theoretical CDFs.

5

Pruning

The initial step in PCBN inference is the reduction and simplification of the PCBN. The goal is to simplify a PCBN and produce a sub-model with the least possible number of nodes such that all nodes of interest are included and that all conditional densities of query variables given the evidence stay the same. This process of BN simplification is called **pruning**.

The pruning of a PCBN is the removal of specific nodes in the graph. The pruned PCBN model is a sub-model resulting from the removal of all marginal distributions, parental orderings and copulas corresponding to the removed arcs and nodes.

First, pruning will be illustrated with an example. Let us consider the inference problem $f_{4|6,8}$ applied to the PCBN in figure 4.1. The graphical representation of this problem is shown in figure 5.1.

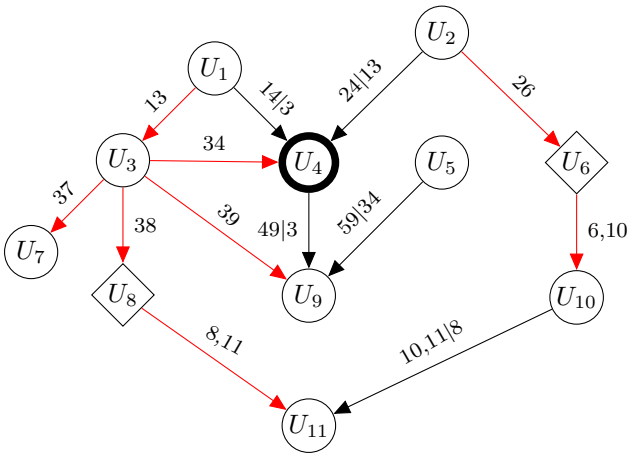


Figure 5.1: Example of inference problem on 11 nodes.

To solve the inference problem $f_{4|6,8}$ we are interested in trails between the query variable U_4 and the evidence variables U_6 and U_8.

In this example, the trail $U_6 \to U_{10} \to U_{11} \leftarrow U_8 \leftarrow U_3 \to U_4$ is blocked by E as it contains the v-structure $U_8 \to U_{11} \leftarrow U_{10}$ and U_{11} is not in E, as well as due to the serial

connection $U_3 \to U_8 \to U_{11}$, where $U_8 \in E$. Furthermore, the trail $U_8 \to U_{11} \leftarrow U_{10} \leftarrow U_6 \leftarrow U_2 \to U_4$ is also blocked by E. Since node U_{11} is not part of any unblocked trail between evidence and query variables, it could be considered for pruning..

Another condition to consider when deciding whether to prune a node is whether its removal necessitates recalculation of copulas. If in example 5.1 when we assume that U_8 is not an evidence variable nodes U_1 or U_3 are candidates to be pruned, as there is no un-blocked trail from U_6 to U_4 including these nodes. However, U_2 is included in the unblocked trail $U_6 \leftarrow U_2 \to U_4$, where the last arc is assigned with the conditional copula $C_{24|13}$. Both U_1 and U_3 are included in the conditioning set of that copula, hence we cannot prune them. If we were to prune U_1 and U_3 we would need to compute copula C_{24}, which is not specified by the original PCBN. We will examine this in greater detail in the section PCBN pruning.

In principle one could examine all trails in the PCBN and see which nodes can be pruned. However in what follows we propose a more efficient way of simplifying the PCBN inference problem.

5.1. Leaf Pruning

The easiest kind of PCBN pruning is leaf pruning, which is applied in the general framework of BNs [1]. The following proposition is helpful in the leaf pruning process.

Proposition 5.1.1. *Let $U \in I$ be a leaf. Then there exists no trail from E to Q containing U, that is not blocked by E.*

Proof. Let us first consider the case $|pa(U)| = 1$. Denote this single parent of U as V. Since U has no children and only one parent V and U belongs neither to Q nor to E then there is no trail from Q to E that contains U.

Let $|pa(U)| > 1$. Since $ch(U) = \emptyset$ then a trail containing U must be of the form:

$$\cdots \leftrightharpoons U_i \to U \leftarrow U_j \leftrightharpoons \dots$$

The connection at U is a converging connection and $U \in I$ while $U \cap K = \emptyset$. Therefore, the trail is blocked. □

Due to Proposition 5.1.1 all leaves in I can be pruned. This is because, in addition to the independency statement of the proposition, they can be put as last in the ordering and their removal does not change the distribution of remaining variables[1].

Example 5.1.2. Let us see how the algorithm 9 works for the PCBN in figure 5.1.

Step 1 : The original PCBN contains three intermediate (neither in query nor in evidences) leaf nodes:

$$L = \{U_7, U_9, U_{11}\}.$$

These are added to the selection of variables to be pruned:

$$L_{pr} = L = \{U_7, U_9, U_{11}\}.$$

Algorithm 9 Leaf.Prune($\mathscr{G}, I$)

Input: $\mathscr{G}$ a DAG,
I the set of intermediate nodes
Output: Pruned DAG $\mathscr{G}_R$

$L \leftarrow \{U \in I : ch(U) = \emptyset\}$ ▷ Intermediate Leaves of the network
$V_{pr} \leftarrow L$ ▷ Variables to be pruned
$B \leftarrow L$
while $B \neq \emptyset$ **do**
  $B \leftarrow \{U \in I : ch(U) \backslash V_{pr} = \emptyset\} \backslash V_{pr}$ ▷ Intermediate Leaves of the pruned network
  $V_{pr} \leftarrow V_{pr} \cup B$
end while
$\mathscr{G}_R \leftarrow \mathscr{G}[V \backslash V_{pr}]$
return $\mathscr{G}_R$

By removing these variables we also remove the arcs with assigned copulas:

$$C_{37}, C_{39}, C_{49|3}, C_{59|34}, C_{8,11}, C_{10,11|8}.$$

The resulting PCBN after one iteration of pruning is presented in figure 5.2.

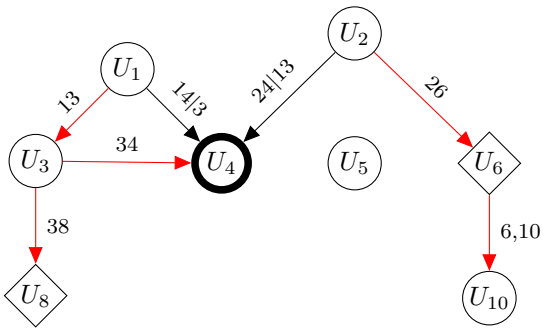


Figure 5.2: First iteration of Leaf Pruning.

Step 2 : After the first iteration we see that the PCBN contains two additional leaf nodes

$$L = \{U_5, U_{10}\}$$

and they are added to the final selection of variables to be pruned:

$$L_{pr} = \{U_5, U_7, U_9, U_{10}, U_{11}\}.$$

Additionally the arc with assigned copula

$$C_{6,10}$$

is removed. The resulting PCBN after the second iteration of pruning is presented in figure 5.3.

The twice pruned PCBN does not contain any more intermediate leaf nodes that can be removed, and thus, leaf pruning is concluded having removed the nodes L_{pr} from the original PCBN.

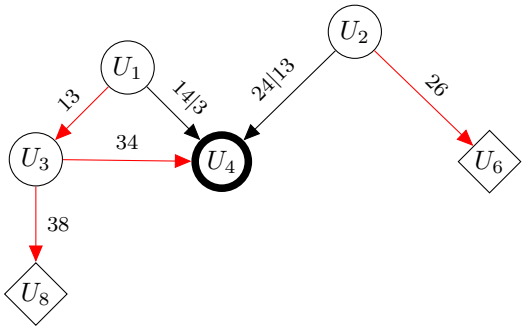


Figure 5.3: Second iteration of Leaf Pruning.

5.2. PCBN Pruning

Next, we will show that pruning certain non leaf nodes is also possible in PCBNs. In the general BN setting, pruning nodes that are not leaves is not performed because it would require recalculation of the conditional distributions of all children of the pruned node, or their marginal distributions when the pruned node is their only parent.

In PCBNs, however, it is possible to also prune nodes that are not leaves. This is because of the specification of PCBNs through margins and copulas instead of the conditional distributions. Furthermore, the bivariate copula factorization of the conditional densities in PCBNs makes it possible to remove the last parent according to the parental ordering without requiring to re-compute copulas..

Similarly to Leaf Pruning, our attention lies in the unblocked trails between evidence and query variables. Let us consider some examples. In each of the three PCBNs seen in figure 5.4, it is possible to prune the variables U_1 and U_2. In figure 5.4a, all trails from U_1 and U_2 to U_4 are blocked by $U_3 \in E$. In figure 5.4b, we observe that the shortest trail from the evidence to U_4 does not contain the nodes U_1 and U_2, nor are these nodes part of a conditioning set of a copula specified by that trail. In figure 5.4c, similarly as in figure 5.4b , U_1 and U_2 can be removed but we do not remove U_3 as it forms part of a trail from the evidence U_5 to U_4.

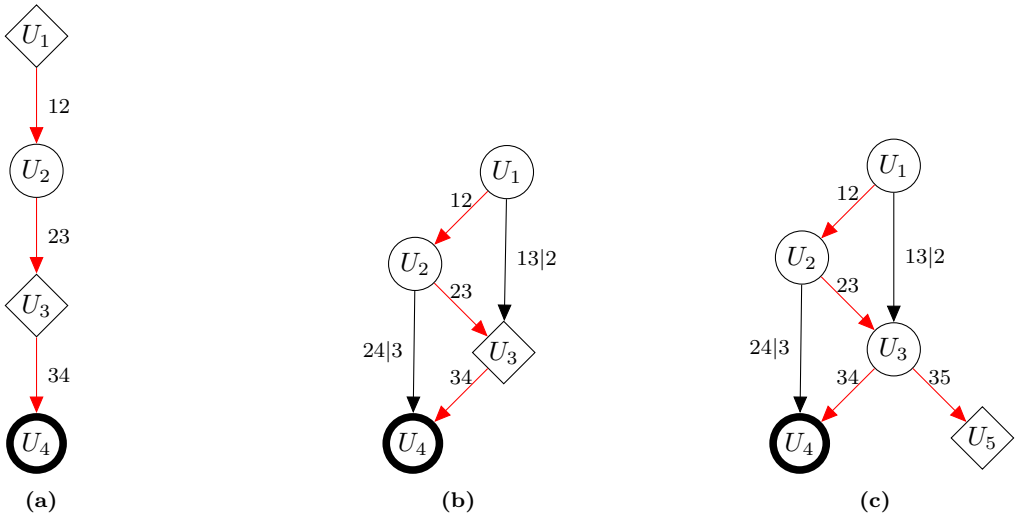


Figure 5.4: Examples for PCBN pruning.

In the general case, an un-blocked trail between a query variable U_q and an evidence variable U_e may contain a number of converging connections $U_1, \ldots, U_n$ which are activated by the evidence. The trail takes the form:

$$U_q \leftrightharpoons \cdots \rightarrow U_1 \leftarrow \cdots \rightarrow U_n \leftarrow \cdots \leftrightharpoons U_e.$$

In the initial stage, our algorithm finds the evidences that are connected to a given query variable through trails that do not contain converging connections. In the general setting of trails containing converging connections, this step is equivalent to finding the trail connecting U_q with the evidence variable that activates the first converging connection U_1.

Our algorithm continues iteratively by finding evidence nodes connected to U_q through trails containing n converging connections. Due to the converging connections being activated, each of them imply the existence of at least one evidence variable that is their descendant or the connection itself. Therefore, the n'th converging connection's evidence descendant is connected to U_q though a trail with n-1 converging connections identified previously.

Motivated by these observations, for a given PCBN we examine the types of unblocked trails between a query node $U_q \in Q$ and an evidence node $U_e \in E$. Despite this trail possibly containing multiple converging connections that are unblocked by E, at an initial phase we will restrict our scope to trails without converging nodes, and expand our framework later.

Hence, in each iteration we search for an unblocked trail between two variables, such that it does not contain a single converging variable; we are interested in trails of the form:

$$U_1 \leftarrow \cdots \leftarrow U_m \rightarrow \cdots \rightarrow U_n$$

As illustrated in the example of figure 5.4b, in order to find such a trail while pruning as many nodes as possible, we must analyze the shortest trails between nodes of interest, and to keep only nodes that are specified by the copulas of the shortest trail. In the PCBN of figure 5.4b, despite the existence of the unblocked trails:

$$U_4 \leftarrow U_2 \leftarrow U_1 \rightarrow U_3$$

and

$$U_4 \leftarrow U_2 \rightarrow U_3,$$

the shortest trail $U_4 \leftarrow U_3$ assigned with an unconditional copula directly implies that we do not need to consider the other two trails.

By including the nodes that are part of the conditioning sets of the copulas specified by the trail we make sure that all required copulas are specified by the original PCBN. In the example of figure 5.4c, we kept U_3 as it is a part of the shortest trail between U_5 and U_3. In figure 5.5, an alternative problem is presented in which the arc $3 \rightarrow 5$ is assigned with a conditional copula, $C_{35|6}$. In this scenario we must keep U_6, as it appears in the conditioning set of this copula.

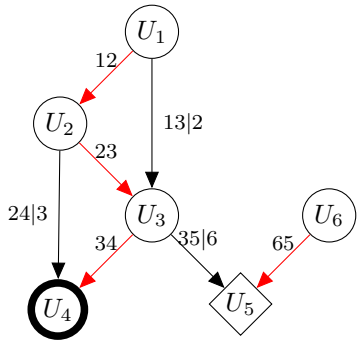


Figure 5.5

Definition 5.2.1 (Earliest Common Ancestor)**.** Let $(\mathscr{G}, \mathscr{O}, \mathscr{F}, \mathscr{C})$ be a PCBN and $U_i, U_j \in V$ and consider a set $A = an(U_i) \cup an(U_j) \cup \{U_i\} \cup \{U_j\}$. A node $U' \in A$ is called **earliest common ancestor** of U_i and U_j if $\forall\, U \in A$, the shortest trail from U_i to U_j passing though U' is not longer than the shortest trail passing though U.

Remark. Note that the earliest common ancestor of U_i and U_j might be U_i or U_j. This is done to include the case of $U_i \leftrightharpoons U_j$. For example, in figure 5.4b, the earliest common ancestor of U_3 and U_4 was U_3, with trail $U_3 \to U_4$. On the other hand, in figure 5.4c, the earliest common ancestor of U_4 and U_5 was U_3, with trail $U_4 \leftarrow U_3 \to U_5$.

In our methodology, for every $U_q \in Q$ and $U_e \in E$ we find the earliest common ancestor U_{qe} and the shortest trail including U_{qe}:

$$U_{i_1} \leftrightharpoons \cdots \leftrightharpoons U_{i_n}. \tag{5.1}$$

where $i_1 = q$ and $i_n = e$.

If U_q and U_e do not posses an earliest common ancestor then all trails between the two contain converging connections. Thus, such trail does not provide with any variables that are possible candidates to be pruned. If we denote the variables to be kept based on such trails between U_q and U_e as V^*_{qe}, we get that $V^*_{qe} = \emptyset$.

Assume that there exists an earliest common arterial ancestor between U_q and U_e. The trail 5.1 is made up of a sequence of variables and specified copulas. Let V_{qe} and $\mathscr{C}_{qe}$ denote the sets of all these variables and copulas. In this case the nodes we must not allow to be pruned are:

$$V^*_{qe} \coloneqq V_{qe} \cup \left(\bigcup_{C_{ij|K_{ij}} \in \mathscr{C}_{qe}} K_{ij} \right)$$

By conducting this procedure, we find the following sets of evidence nodes based on whether they are connected to the query variable without converging connections:

- $E^1_1 \coloneqq \{U_e \in E \mid V^*_{qe} \neq \emptyset\}$.
- $E^1_2 \coloneqq E \backslash E^1_1$.

This shows that all nodes not allowed to be pruned at this stage are:

$$V^q_1 \coloneqq \bigcup_{e \in E^1_1} V^*_{qe}$$

The next step is to examine whether there are nodes in E_2^1 with a trail to Q unblocked by E, by allowing a single converging connection which is unblocked by E_1^1. This means that we look for nodes in E_2^1 with a trail to E_1^1 unblocked by E and containing no converging connection.

Just like in the first step, for every $U_{e_1} \in E_1^1$, $U_{e_2} \in E_2^1$, we find the earliest common ancestor $U_{e_1e_2}$ and the implied trail. We then find the specified nodes and copulas of that trail which provide us with the nodes $V^*_{e_1u_2}$ which cannot prune.

Then, the next sets of evidence nodes are defined:

- $E_1^2 \coloneqq \{U_e \in E_2^1 \mid \exists U_{e_1} \in E_1^1 : V^*_{e_1u_2} \neq \emptyset\}$.
- $E_2^2 \coloneqq E \backslash E_1^2$.

The nodes to be kept after this stage are:

$$V_2^q \coloneqq \bigcup_{e_1 \in E_1^1} \bigcup_{e_2 \in E_1^2} V^*_{e_1e_2}$$

The process is repeated until we obtain $E_1^{m+1} = \emptyset$. Then, the search for evidence nodes connected to $U_q \in Q$ is finished, and the nodes that are kept based on those trails are:

$$V^{q,keep} \coloneqq \bigcup_{i=1}^{m} V_i^q$$

For every $U_q \in Q$ the above process is performed and all the nodes that need to be kept from pruning are:

$$V^{q,keep} \coloneqq \bigcup_{U_q \in Q} V^{q,keep} \tag{5.2}$$

As an output, this algorithm calls for removing all nodes in $V \backslash V^{keep}$ from the PCBN, along with all of their connected (incoming and outgoing) arcs. The pseudocode corresponding to our PCBN pruning algorithm is depicted in algorithm 10.

Remark. Note that this algorithm also incorporates leaf pruning. An intermediate leaf node will never be part of the ancestors of query or evidence nodes and thus, by default, it will never become part of an V_i^{keep} set.

Next, we will prove the algorithm's ability to retain the conditional distributions of query variables given the evidence. We start with a very simple lemma.

Lemma 5.2.2. *Let $\mathcal{G}$ be a DAG and consider the following trail in $\mathcal{G}$:*

$$U_1 \leftrightharpoons \cdots \leftrightharpoons U_n.$$

If the trail contains no node with converging connections, it contains at most one diverging connection.

Proof. If this trail is composed only of serial connections, then obviously it has no nodes with diverging connections.. To show that there can be only one node with diverging

Algorithm 10 PCBN.prune($\mathscr{G}, I, E, R_{keep}$)

Input: $\mathscr{G}$ a PCBN
I the set of intermediate nodes
E the set of evidence nodes **Output:** Pruned PCBN $\mathscr{G}_{pr}$

$Q \leftarrow V \backslash (I \cup E)$
for $U_q \in Q$ **do**
  $V_0^q \leftarrow \{q\}$
  $E_1^0 \leftarrow \{q\}$
  $E_2^0 \leftarrow E$
  $i \leftarrow 1$
  while $E_1^{i-1} \neq \emptyset$ **do**
    for $U_{e_1} \in E_1^{i-1}$ **do**
      for $U_{e_2} \in E_2^{i-1}$ **do**
        $U_{e_1e_2} \leftarrow$ earliest common ancestor of U_{e_1} and U_{e_2} (def.5.2.1).
        $V_{e_1e_2} \leftarrow$ nodes in trail 5.1.
        $\mathscr{C}_{e_1e_2} \leftarrow$ copulas specified by trail 5.1.
        $V^*_{e_1e_2} \leftarrow V_{e_1e_2} \cup \left(\bigcup_{C_{ij\mid K_{ij}} \in \mathscr{C}_{e_1e_2}} K_{ij} \right)$.
      end for
    end for
    $E_1^i \leftarrow \{U_{e_2} \in E_2^{i-1} \mid \exists U_{e_1} \in E_1^{i-1} : V^*_{e_1e_2} \neq \emptyset\}$.
    $E_2^i \leftarrow E_2^{i-1} \backslash E_1^i$.
    $V_i^q \leftarrow \bigcup_{e_1 \in E_1^1} \bigcup_{e_2 \in E_1^2} V^*_{e_1e_2}$
    $i \leftarrow i + 1$
  end while
  $V^{q,keep} \leftarrow \bigcup_{j=0}^{i-2} V_j^q$
end for
$V^{keep} \leftarrow \bigcup_{U_q \in Q} V^{q,keep}$
$\mathscr{G}^{pr} \leftarrow$ sub-PCBN $\mathscr{G}$ having removed $V \backslash V^{keep}$.
return $\mathscr{G}^{pr}$

connection assume that there be two diverging connections in a trail. This trail is of the form:

$$U_1 \leftarrow \cdots \leftarrow U_{i_1} \rightarrow \cdots \leftarrow U_{i_2} \rightarrow \cdots \rightarrow U_n.$$

After the first diverging node all following arcs are rightwards facing, while all of the arcs before the second diverging node are leftwards facing. Hence a node with converging connection must be on the trail between U_{i_1} and U_{i_2}, which contradicts the initial assumption of no converging connections. □

Next, we show that the PCBN obtained after application of the algorithm leads to the same solution of the inference problem as the initial PCBN.

Theorem 5.2.3. *Let the PCBN $(\mathscr{G}', \mathscr{O}', \mathscr{F}', \mathscr{C}')$ be a sub-model of PCBN $(\mathscr{G}, \mathscr{O}, \mathscr{F}, \mathscr{C})$ obtained as the output of algorithm 10. The conditional distribution $f_{Q|E}$ computed in PCBN $\mathscr{G}'$ is the same as the one computed in PCBN $\mathscr{G}$.*

Proof. In order to prove this theorem it suffices to show that for every $U_q \in Q$ and $U_e \in E$, every minimal activated by E trail in $\mathscr{G}$ between U_q and U_e is also present in $\mathscr{G}'$ and remains activated.

Let the minimal trail in $\mathscr{G}$ between U_q and U_e activated by E be of the form:

$$U_q \leftrightharpoons U_1 \leftrightharpoons \cdots \leftrightharpoons U_n \leftrightharpoons U_e \tag{5.3}$$

This trail is not blocked by $\boldsymbol{E}$ and is minimal in length, meaning that there are no chords present between its elements in $\mathscr{G}$.

The following connections $U_{i-1} \leftrightharpoons U_i \leftrightharpoons U_{i+1}$ are possible in this trail :

1. Serial or a diverging connection with $U_i \notin \boldsymbol{E}$.
2. Converging connection with U_i or at least one of its descendants belongs to the evidence.

Let $U_{d_1}, \ldots, U_{d_m}$ be the converging nodes in trail 5.3.

The proof will be done by induction on the number m of converging nodes in the trail 5.3.

For $m = 0$, there is no converging connection in the trail, and the trail is composed of only serial and diverging connections. Due to lemma 5.2.2, there is at most one diverging connection in this trail and it is the earliest common ancestor of U_q and U_e.

If there is no diverging connection, the trail is built of only serial connections, implying either $U_q \in an(U_e)$ or $U_e \in an(U_q)$. All nodes on such a trail are included and are not allowed to be pruned. Furthermore, if this trail is activated by E in $\mathscr{G}$ this implies that none of the nodes are part of E, hence it is also activated in $\mathscr{G}'$.

If there is one diverging connection at a node U_a, then the trail takes the following form:

$$U_q \leftarrow \cdots \leftarrow U_a \rightarrow \cdots \rightarrow U_e$$

Then, U_a is the earliest common ancestor of U_q and U_e and the algorithm includes all nodes in the pruned graph. Like before, due to all connections being serial and diverging, if the trail is activated by E then it is also active in $\mathscr{G}'$

Let us assume that minimal trails activated by $\boldsymbol{E}$ between nodes U_q and U_e with m or less converging connections are active in the pruned graph obtained by our algorithm. That is all nodes required for such trails to be active are kept in the pruned graph. We will prove that trails with $m+1$ converging connections are also active in pruned graph.

Such a trail takes the following form and we assume that $U_{d_{m+1}}$ is the closest to U_e.

$$U_q \leftrightharpoons \cdots \rightarrow U_{d_{m+1}} \leftarrow \cdots \leftrightharpoons U_e$$

This last converging connection, $U_{d_{m+1}}$ is unblocked by $\boldsymbol{E}$, hence either $U_{d_{m+1}}$ or one of its descendants is in $\boldsymbol{E}$. Let us denote this node that belongs to $\boldsymbol{E}$ as $U^e_{d_{m+1}}$. Then the following trails between U_q and $U_{d_{m+1}}$ as well as between $U_{d_{m+1}}$ and U_e are present:

$$U_q \leftrightharpoons \cdots \rightarrow U_{d_{m+1}} \rightarrow \cdots \rightarrow U^e_{d_{m+1}}$$
$$U^e_{d_{m+1}} \leftarrow \cdots \leftarrow U_{d_{m+1}} \leftarrow \cdots \leftrightharpoons U_e$$

The first trail has m converging connections and $U^e_{d_{m+1}}$ is part of the evidence and by the inductive assumption, the trail is included in the pruned graph and it is activated by E.

Furthermore, $U^e_{d_{m+1}}$ is added to the set E_1^{m+1} specified in the algorithm. Since the last converging connection is at $U_{d_{m+1}}$, the second trail from $U^e_{d_{m+1}}$ to U_e does not contain any converging connections. Therefore, by lemma 5.2.2, this sub-trail contains at most one diverging connection. Similarly to the argument for $m = 0$, this entails that $U^e_{d_{m+1}}$ and U_e have an earliest common ancestor and that the algorithm keeps all nodes required for the minimal active trail between the two nodes. $U_{d_{m+1}}$ is part of this trail, hence the sub-trail between $U_{d_{m+1}}$ and U_e is included in $\mathscr{G}'$.

Therefore, the algorithm includes all nodes required to keep the minimal trail between U_q and U_e activated by $\boldsymbol{E}$ in pruned graph and the proof is concluded. □

Let us now have a closer look at an example with multiple arteries to illustrate how the pruning algorithm works.

Example 5.2.4. Let us apply algorithm `PCBN.prune` (10) to the inference problem presented in figure 5.6. This network consists of 16 nodes, with query $Q = \{U_3, U_{16}\}$ and evidence $E = \{U_5, U_8\}$.

We start by searching the connections of $U_3 \in Q$:

1. We initialize $V_0^{keep} = E_1^0 = \{U_3\}$ and $E_2^0 = E = \{U_5, U_8\}$.
2. The only evidence variable in E_2^0 that has an earliest common ancestor with U_3 is U_5, with that earlier ancestor being:

$$U_{3,5} = U_3$$

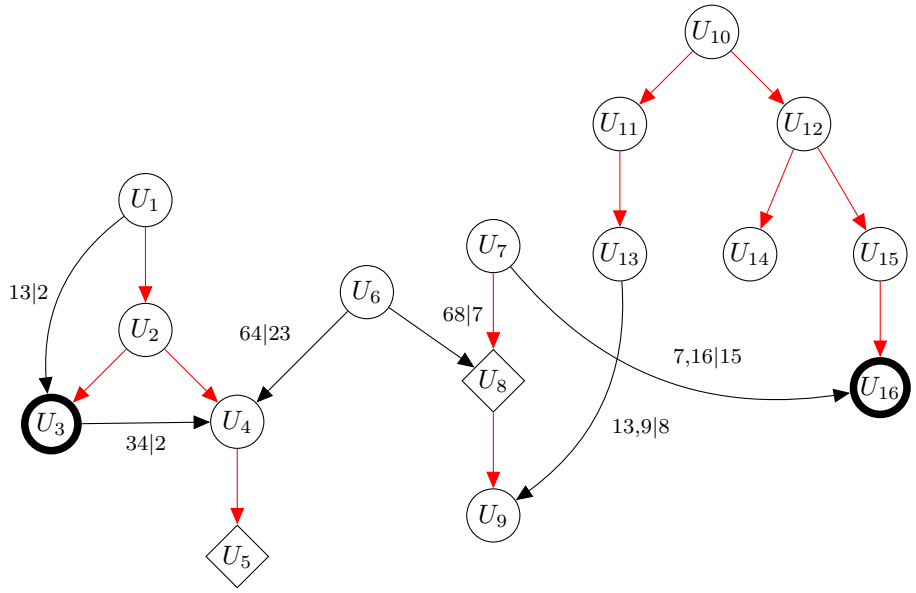


Figure 5.6: Example for PCBN pruning.

The minimal trail implied by this earliest common ancestor is:

$$U_3 \to U_4 \to U_5$$

The specified nodes are $V_{3,5} = \{U_3, U_4, U_5\}$ and the specified copulas are $\mathscr{C}_{3,5} = \{C_{23}, C_{34|2}, C_{45}\}$. Therefore, the nodes that are to be kept based on this trail are:

$$V^3_{3,5} = \{U_2, U_3, U_4, U_5\}$$

We set:

$$E^1_1 = \{U_5\} \text{ and } E^1_2 = \{U_8\}$$

3. E^1_2 only contains U_8 which has an earliest common ancestor with $U_5 \in E^1_1$, that being:

$$U_{5,8} = U_6$$

The implied trail is:

$$U_5 \leftarrow U_4 \leftarrow U_6 \to U_8$$

The specified nodes are $V_{5,8} = \{U_4, U_5, U_6, U_8\}$ and the specified copulas are $\mathscr{C}_{5,8} = \{C_{45}, C_{64|23}, C_{68|7}\}$. Therefore, the nodes that are to be kept based on this trail also include the conditioning sets of these copulas and:

$$V^3_{5,8} = \{U_2, U_3, U_4, U_5, U_6, U_7, U_8\}$$

Then we are left with $E^2_1 = \emptyset$ which finalizes the search for nodes connected to U_3.

Therefore, the nodes we must keep in order to reach the evidence from U_3 are:

$$V^{3,keep} = V^3_{3,5} \cup V^3_{5,8} = \{U_2, U_3, U_4, U_5, U_6, U_7, U_8\}$$

Next, we begin the search from the query node U_{16}.

1. We initialize $V_0^{keep} = E_1^0 = \{U_{16}\}$ and $E_2^0 = E = \{U_5, U_8\}$.
2. The only evidence variable in E_2^0 that has an earliest common ancestor with U_{16} is U_8. This common ancestor is:

   $$U_{16,8} = U_7.$$

   The implied trail is:

   $$U_16 \leftarrow U_7 \rightarrow U_8$$

   The specified nodes are $V_{16,8} = \{U_7, U_8, U_16\}$ and the specified copulas are $\mathscr{C}_{16,8} = \{C_{78}, C_{7,16|15}\}$. Therefore, the nodes that are to be kept due to this trail are:

   $$V_{16,8}^{16} = \{U_7, U_8, U_{15}, U_{16}\}.$$

   We set:

   $$E_1^1 = \{U_8\} \text{ and } E_2^1 = \{U_5\}.$$

3. E_2^1 only contains U_5 which has an earliest common ancestor with $U_8 \in E_1^1$, which is:

   $$U_{8,5} = U_6.$$

   The implied trail is:

   $$U_8 \leftarrow U_6 \rightarrow U_4 \rightarrow U_5.$$

   The specified nodes are $V_{8,5} = \{U_4, U_5, U_6, U_8\}$ and the specified copulas are $\mathscr{C}_{8,5} = \{C_{45}, C_{64|23}, C_{68|7}\}$. Therefore, the nodes that are to be kept because of this trail also include the conditional sets and:

   $$V_{8,5}^{16} = \{U_2, U_3, U_4, U_5, U_6, U_7, U_8\}.$$

   Then we are left with $E_1^2 = \emptyset$ which concludes the algorithm for node U_{16}.

Therefore, the nodes we must keep due to U_{16} are:

$$V^{16,keep} = V_{3,5}^3 \cup V_{5,8}^3 = \{U_2, U_3, U_4, U_5, U_6, U_7, U_8, U_{15}\}.$$

Finally, both obtained sets are joined and we get a set of nodes that cannot be pruned:

$$V^{keep} = V^{2,keep} \cup V^{16,keep} = \{U_2, U_3, U_4, U_5, U_6, U_7, U_8, U_{15}\}.$$

The pruned PCBN for this example is presented in figure 5.7.

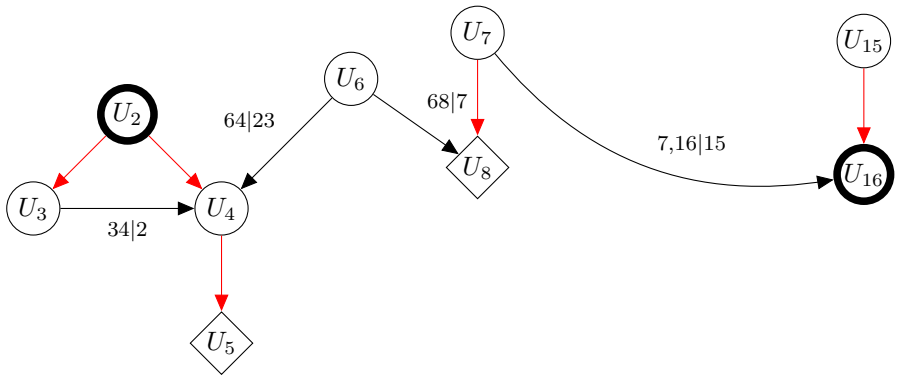


Figure 5.7: Pruned PCBN of example 5.6.

6

Simulation Study

In this chapter the results of the extensive simulation study are presented where the algorithms proposed in chapter 4 are tested.

The goals of the simulation study are:

- To test the effectiveness and efficiency of the node sampling algorithms.
- To find scenarios in which node sampling under-performs.
- To decide the choice of importance sample.
- To test the effectiveness and efficiency of the full PCBN inference through sample propagation.

The third goal is motivated by the comments made in subsection 4.1. There, we have discussed that the optimal choice of the importance sample is somewhat unclear, especially in Bilateral Sampling. We want to investigate which choice should be made and under what conditions. For the testing Bilateral Sampling, the importance sample was chosen according to our initial proposals made in chapter 4. This choice will be further discussed in section 6.7, where other types of importance samples will be considered.

To compare samples obtained through the PCBN inference algorithms with the distributions obtained through GBN inference, we opted to rely on the **overlap coefficient** (OVL). The OVL measures the area under two integrable PDFs f_1, f_2 of distributions with the same support $\mathscr{X}$:

$$OVL = \int_{\mathscr{X}} min(f_1(x), f_2(x))dx$$

In other words, the OVL can be seen as a measure of agreement between two distributions. It takes values between 0 and 1 and we will use it to see how 'close' the empirical conditional distribution, obtained through the proposed in this thesis method, is as compared to the theoretical one. A representation of the OVL for the distributions and $\mathcal{N}(1.5, 2)$ can be seen in figure 6.1.

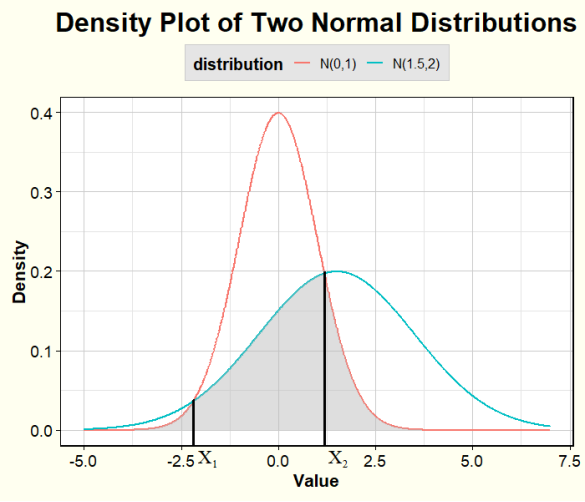


Figure 6.1: Illustration of the Overlapping Coefficient.

The points X_1, X_2 where the two densities are equal in value can be calculated by [38]:

$$X_1 = \frac{\mu_1\sigma_2^2 - \mu_2\sigma_1^2 - \sigma_1\sigma_2\sqrt{(\mu_1-\mu_2)^2 + (\sigma_2^2-\sigma_1^2)ln\left(\frac{\sigma_2^2}{\sigma_1^2}\right)}}{\sigma_2^2-\sigma_1^2}.$$

$$X_2 = \frac{\mu_1\sigma_2^2 - \mu_2\sigma_1^2 + \sigma_1\sigma_2\sqrt{(\mu_1-\mu_2)^2 + (\sigma_2^2-\sigma_1^2)ln\left(\frac{\sigma_2^2}{\sigma_1^2}\right)}}{\sigma_2^2-\sigma_1^2}.$$

Under the assumption $X_1 < X_2$, the closed form expression of the OVL between two normal distributions $\mathcal{N}(\mu_1, \sigma_1^2)$ and $\mathcal{N}(\mu_2, \sigma_2^2)$ for $\sigma_1^2 < \sigma_2^2$ has been shown by Inman and Bradley Jr [38] to be:

$$OVL = 1 + \Phi\left(\frac{X_1-\mu_1}{\sigma_1}\right) + \Phi\left(\frac{X_2-\mu_2}{\sigma_2}\right) - \Phi\left(\frac{X_1-\mu_2}{\sigma_2}\right) - \Phi\left(\frac{X_2-\mu_1}{\sigma_1}\right)$$

6.1. Simulation Setup

In this section we will provide details of the simulation process.

In order to be able to reach conclusions about the effectiveness and efficiency of the sampling algorithms we chose to work with PCBNs that feature only Gaussian copulas and Gaussian marginals. This family of PCBNs is equivalent to the GBN model, in which solving inference problems analytically is easy as shown in subsection 3.3.

Each simulation was made up of the following stages:

Stage 1 : Set-up of simulation parameters for the following stages:

- Number of Samples we wished to obtain, set to $50, 100, 500, 10^3, 10^4$.
- Gaussian Copula Parameter values, set to $0.5, 0.6, 0.7, 0.8, 0.9$.

- Number of Parents, type 1 children and type 2 children, according to the context of each sampling type.
- Number of Importance Samples, when SIR was required, set to 100, 500, 10^3, 10^4.

Stage 2 : Calculation 10^4 unconditional PCBN simulations, as seen in subsection 2.3.4.

Stage 3 : Estimation of the covariance matrix of the joint normal distribution from the samples obtained in Stage 2.

Stage 4 : Set the observed values of parents and children to those of a PCBN simulation from Stage 2. The following steps were taken for 100 different choices of PCBN simulation.

Stage 4.1 : Calculation of the likelihood of the observed values by integrating the rest of the variables from the PCBN density.

Stage 4.2 : Calculation of the conditional distribution of the node being sampled using GBN theory and the Gaussian-transformed observed values from Stage 4.

Stage 4.3 : Forward/Backward/Secondary Backward or Bilateral Sampling of a node given the observed values provided by Stage 4, and calculating how much time it took to complete.

Stage 4.4 : Calculation of the Overlapping Coefficient between the theoretical distribution and the empirical distribution of the sample.

Stages 4.1-4.4 were conducted for all unique combinations of parameters defined in Stage 1 along with the 100 different simulation choices made in Stage 4.

As seen in Stage 4, unconditional PCBN samples were used for the values of the parents and children. This was done in order to ensure the "realistic" likelihood of observing such values; for extraordinarily low likelihoods of observations we are faced with programming issues such as rounding and floating point errors. As an example, consider the inference problem presented in figure 6.2, where all copula parameters are equal to 0.95 and the observed values are 0.95 for U_1 and 0.05 for U_2. For Forward Sampling U_3 we would require the conditional marginal $u_{1|2}$, whose value is rounded to 1. This is highly problematic for many reasons, including the fact that the existing packages only allow values in $(0, 1)$. Furthermore, while this rounding error may seem negligible in the uniform world, when translated to the normal marginals the difference between the real value and $\Phi^{-1}(1) = +\infty$ is major.

Besides the accuracy of sampling algorithms which was measured by the OVL, of central importance was the efficiency of each sampling type. The focus was the time until completion of the sampling process and how the simulation parameters such as sample size and number of parents/children affected it.

We are interested in the relative performance for different simulation parameters, but for reference we note that the system that the simulation study was run on was an

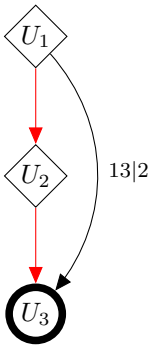


Figure 6.2: PCBN inference problem featuring extraordinarily unlikely evidence. ($r = 0.95$, $U_1 = 0.95, U_2 = 0.05$)

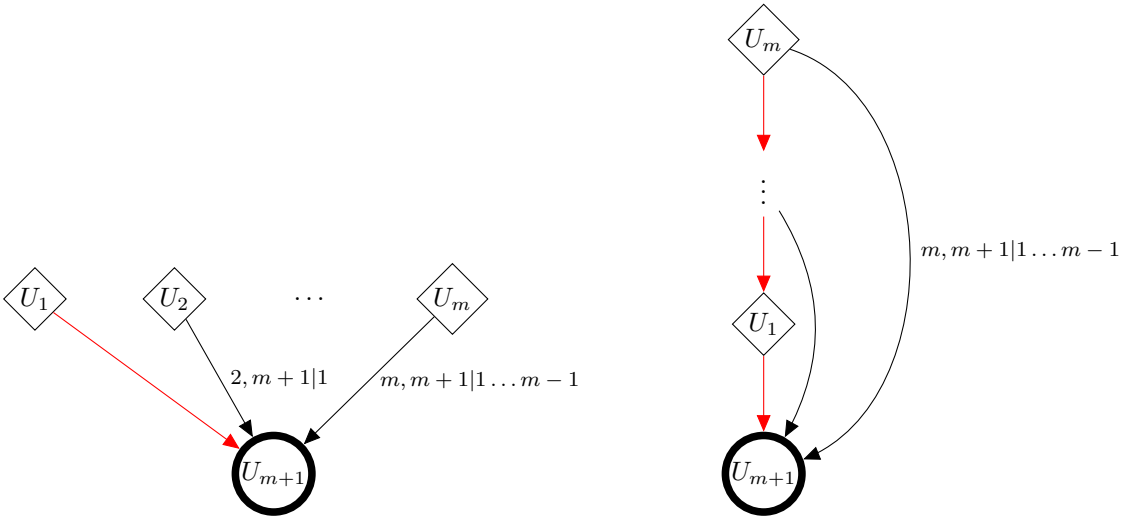
11th Gen Intel(R) Core(TM) i7-11800H processor with Base Frequency 2.30GHz, Boost Frequency 4.60 GHz, 8 Cores and 16 Logical Processors.

Multiprocessing was only used to run multiple (15) simulations at the same time. Each individual simulation however was conducted linearly, with the only parallelization being vectorization of commands when possible.

We tested the algorithms of Forward, Backward, Secondary Backward and Bilateral Sampling. Therefore, in order to test each sampling type, a different DAG structure was chosen. In the following sections we present these tested structures along with the number of parents/children used.

6.2. Forward Sampling Results

For testing the Forward Sampling algorithm, we divided the scope into two types of DAG structures. The first was when all parents are pairwise independent, and the second when they are connected through arcs. These two types of DAG structures are illustrated in figure 6.3.



(a) Network for independent parents.

(b) Network for dependent parents.

Figure 6.3: Networks used for testing the Forward Sampling algorithm, for $m = 1, \ldots, 6$.

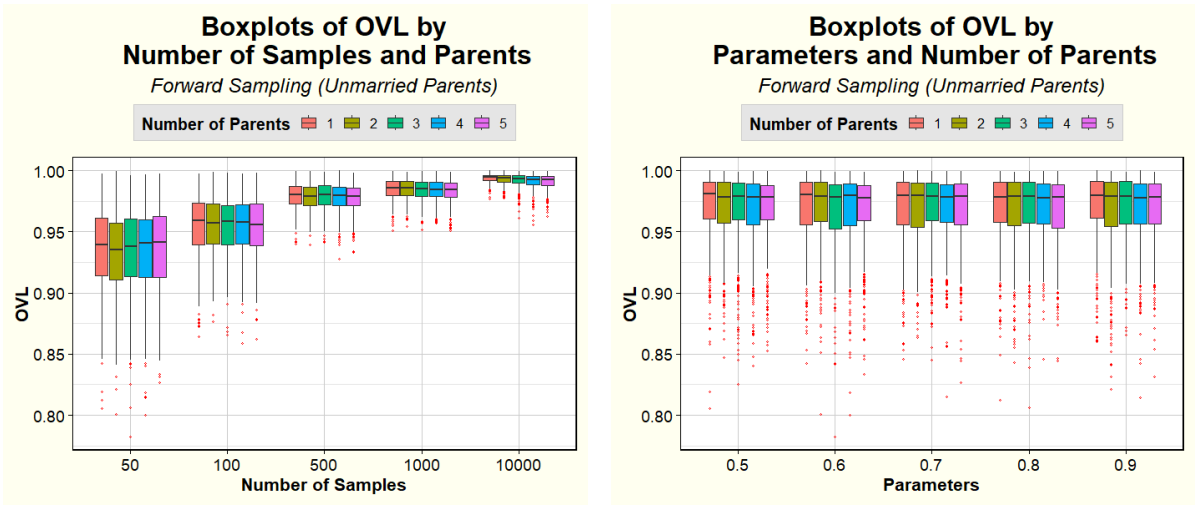
For Forward Sampling the number of parents took values $1, 2, \ldots, 6$.

Effectiveness The results between the two scenarios of Forward Sampling were identical.

Overall, we observe excellent performance for even low sample sizes, which is to be expected as SIR was not used and the Sampling Method sampled directly from the target distribution. The estimated average Overlapping Coefficient was 0.9687 with a $5 \cdot 10^{-4}$ margin of error at a 95% confidence level.

In figure 6.4a we can clearly see how OVL values become bigger for higher samples taken. On the other hand, the number of parents does not show any impact on the OVL values.

For the same exact reason, changing the parameters of the Gaussian Copulas did not show any change in performance, as illustrated in figure 6.4b.



(a) **(b)**

Figure 6.4: Forward Sampling OVL Box-Plots (unmarried parents).

Let us now consider how observation likelihood may affect the effectiveness of the algorithm. In the DAG structure of our first scenario of Forward Sampling, parents are unmarried and thus independent. This means that, we do not need to specify the likelihood of the observation, as this would be constantly equal to 1 for all observations.

On the other hand, likelihood weights may be relevant in the presence of arcs between parents. The Pearson coefficient between OVL values and weights which was approximately 0.006 indicated that no linear correlation was present. This was further supported by the scatter-plot between those two variables as illustrated in figure 6.5. we can conclude that likelihood of observations does not influence the performance of Forward Sampling.

Efficiency Unsurprisingly, Sample Size affected the run-time of the algorithm, as seen in figure 6.6a. Despite the noticeable increase in speed for higher Sample Sizes, the

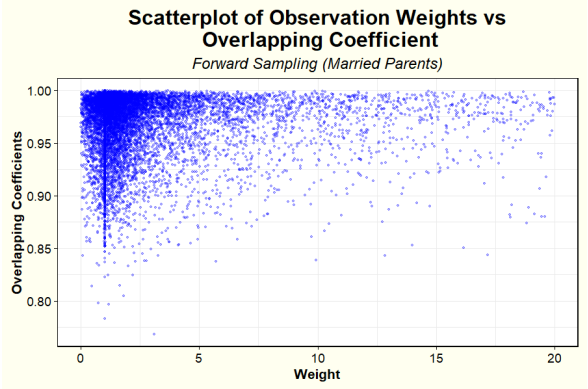
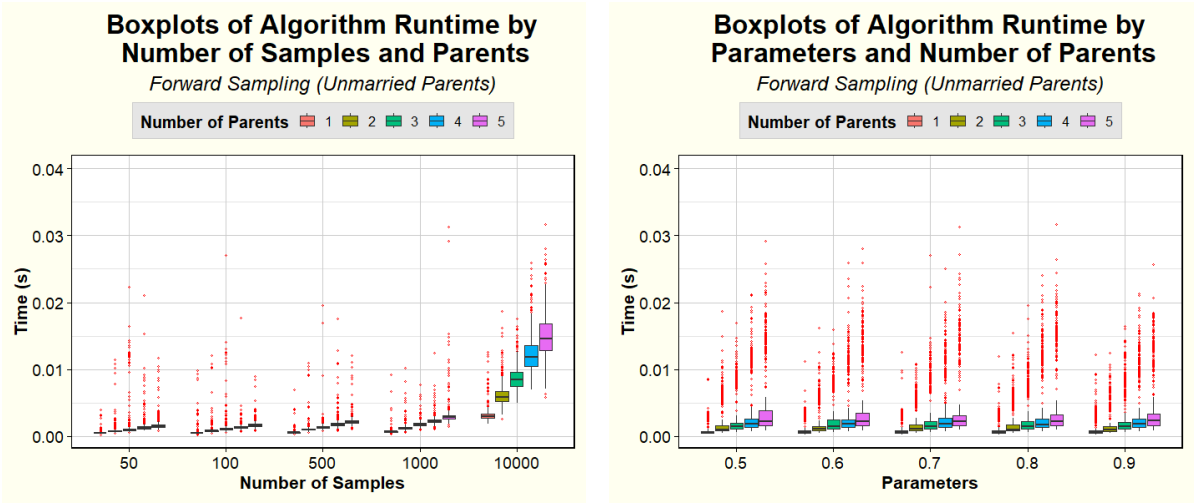


Figure 6.5

algorithm remained fast for all simulations averaging at 0.00303 seconds with a margin of error of $7 \cdot 10^{-5}(95\%)$ seconds.



(a) **(b)**

Figure 6.6: Forward Sampling time Box-Plots (unmarried parents).

In the same figure we can observe how the number of parents also caused an increase in algorithm runtime. In the same way as with the OVL, the parameter choices did not affect the speed of the algorithm as seen in figure 6.6b.

The results for the Forward Sampling scenario when arcs between parents are present in the DAG, are identical to those for unmarried parents, with the exception being a slightly higher run-time for the former case which was 0.0048 seconds with a margin of error of $1 \cdot 10^{-4}(95\%)$ seconds. For the case where parents are connected, the corresponding figures are in appendix A.

6.3. Type 1 Backward Sampling

We recall that type 1 Backward Sampling algorithm is used when there exists an outgoing arc with assigned copula $C_{ij|K_{ij}}$, such that all type 1 children belong to $\{U_j\} \cup K_{ij}$. An illustration of the DAG structure used for four children is presented in figure 6.7.

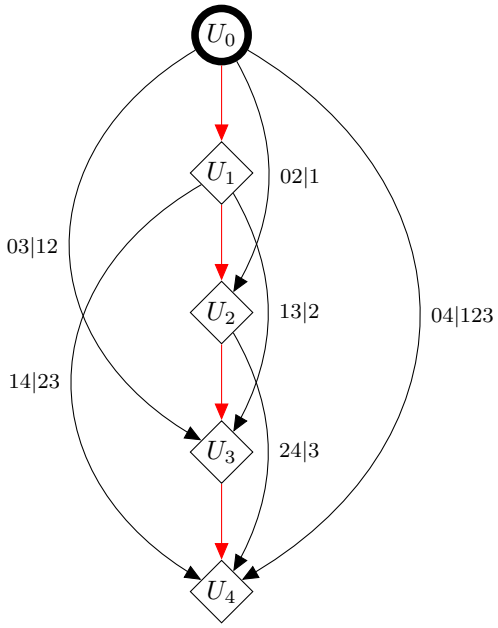
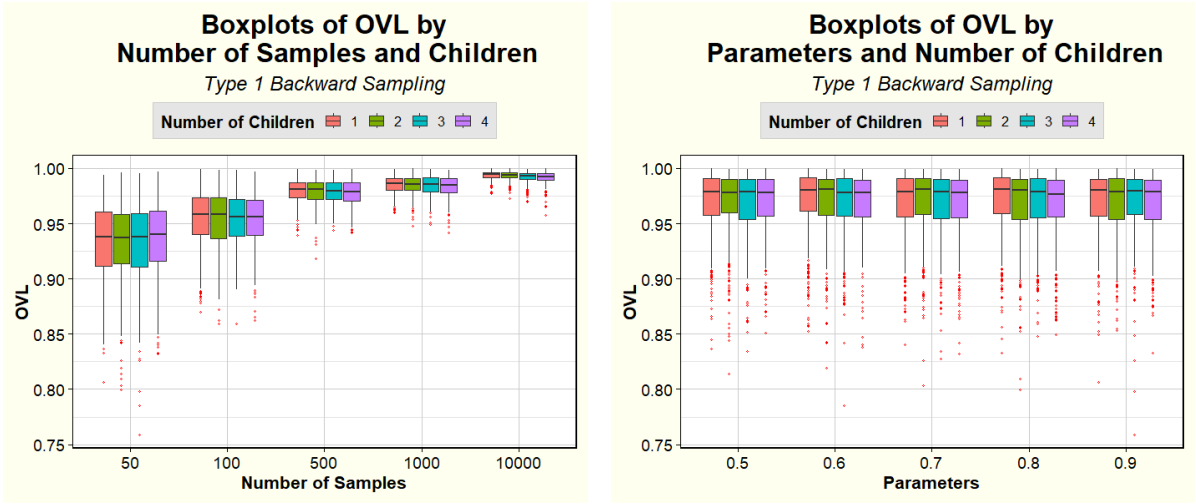


Figure 6.7: Example of DAG for simulating type 1 Backward Sampling with four children.

We considered $1, 2, 3$ and 4 children in the context of type 1 Backward Sampling.

Effectiveness Type 1 Backward Sampling greatly resembles Forward Sampling due to the sample being obtained directly through the h-function recursion. Hence we can expect that its performance be comparable to that of Forward Sampling.

Indeed, the behavior of OVL values was similar, as seen in figure 6.8. The total average was 0.9688 with a $5 \cdot 10^{-4}$ margin of error on a 95% confidence level; virtually identical to the results seen in section 6.2. For higher sample size OVL became more effective, while number of Parents or Gaussian Copula Parameter values did not seem to affect the OVL.



(a) **(b)**

Figure 6.8: Type 1 Backward Sampling OVL Box-Plots.

Additionally, the likelihood weights of the observed values did not seem to influence the efficiency either. Both the Pearson correlation coefficient being equal to 0.0003 and the graphical representation in figure 6.9 imply that in type 1 Backward Sampling OVL values are not affected by how likely or unlikely the children's values are.

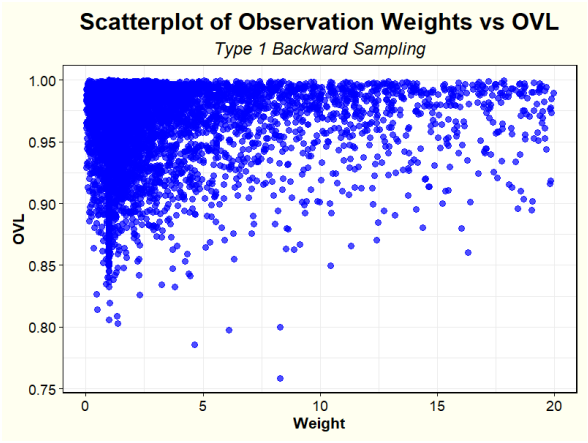
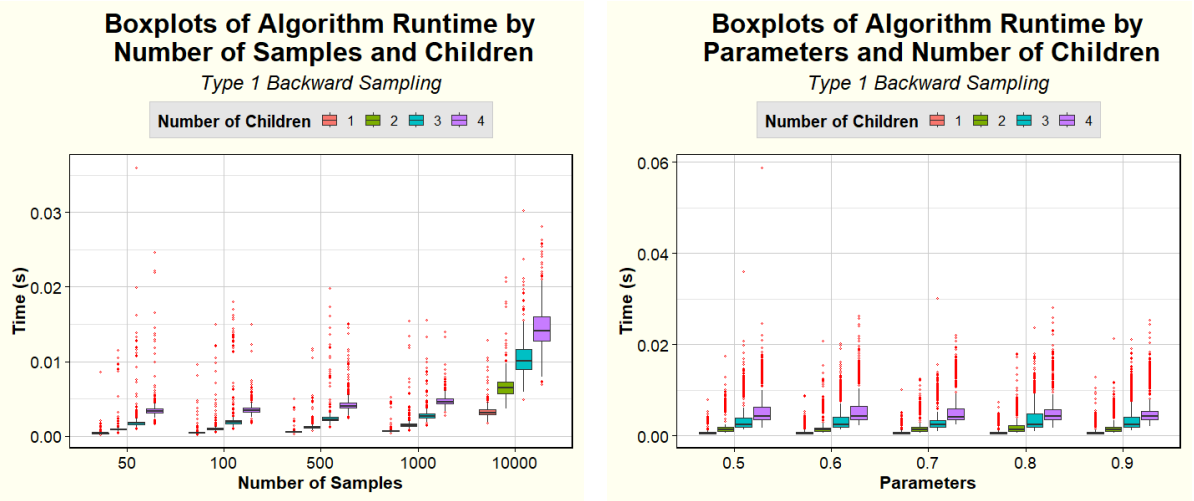


Figure 6.9

Efficiency In terms of the algorithm's execution time, type 1 Backward Sampling was once again comparable to Forward Sampling, averaging at 0.0035 seconds with a $7 \cdot 10^{-5}$ seconds margin of error on a 95% confidence level.



(a) **(b)**

Figure 6.10: Type 1 Backward Sampling time Box-Plots (unmarried parents).

6.4. Type 2 Backward Sampling

We divided the scope of evaluating type 2 Backward Sampling by focusing on either the existence of arcs between children or lack thereof.

The DAG structure of the first scenario where children are not directly connected is presented in figure 6.11.

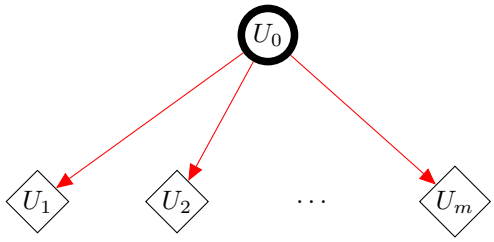


Figure 6.11: DAG structure of type 2 Backward Sampling without arcs between children.

The number of children in the context of the first scenario of case 2 Backward Sampling took values $2, 3, 4, 5$.

On the other hand, to consider the presence of arcs between nodes we follow a DAG structure presented in figure 6.12. The illustrated DAG includes six children of U_0. For DAGs with fewer children, we remove the necessary nodes in the reverse order of the node numbering.

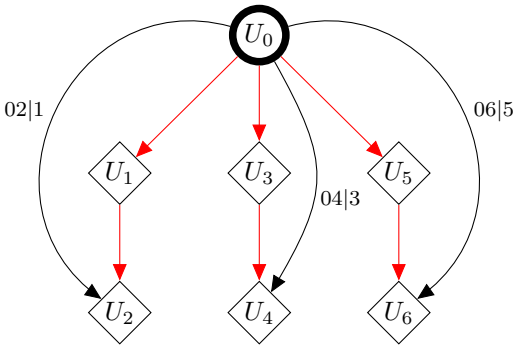


Figure 6.12: DAG structure of type 2 Backward Sampling with arcs between children.

The number of children in the context of this scenario of case 2 Backward Sampling took values $3, 4, 5, 6$. We will divide the results for Type 2 Backward Sampling based on the graph structure used to obtain them.

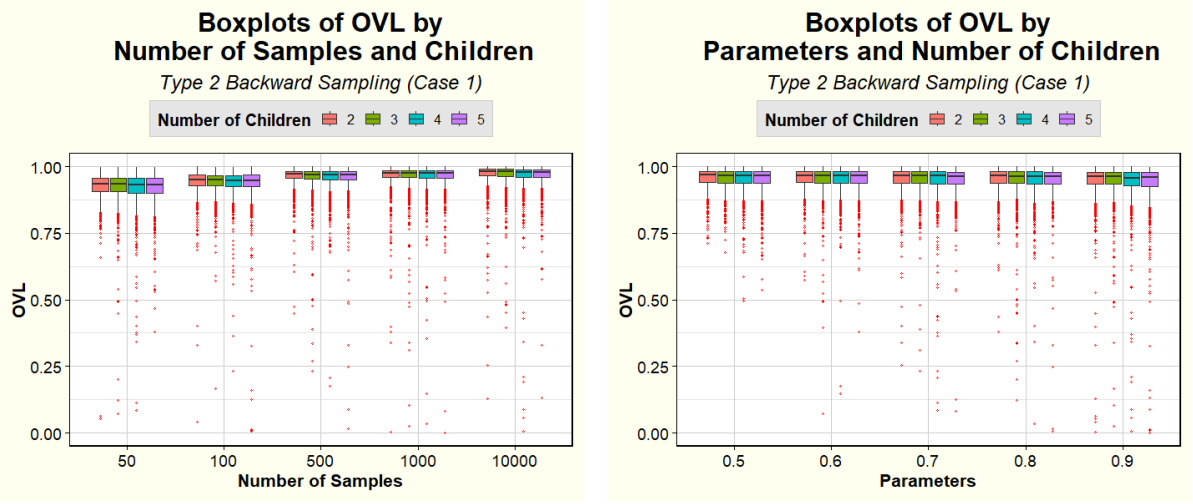
6.4.1. First Case

The first case covers the scenario where children have no arcs between them.

Effectiveness From the boxplots in figure 6.13a we can see that the majority of OVLs were very high, with a total average of 0.9518 with a $5 \cdot 10^{-4}$ margin of error on 95% confidence level.

In the same figure however, we can observe that despite the general increase of OVL with larger sample sizes, there was a significant number of low OVL values. The number of children, on the other hand, seems to have negligible effect on the OVL values. Figure 6.13b illustrates how higher Gaussian Copula Parameters lead to lower tails to the OVL distribution.

In order to assess the causes of the instances with poor performance let us explore the figure 6.14. This figure includes the Number of Importance Samples and we can clearly



(a) (b)

Figure 6.13: Type 2 Backward Sampling OVL Box-Plots (Case 1).

see that the low OVL outliers can be found for a combination of smaller Importance Sample Sizes, and larger Parameter values. For parameters $0.5, 0.6, 0.7$ the Simulations conducted with Importance Sample Size of 100 were the ones that feature a significant number of problematic results. For higher parameter values, even simulations with Size 500 Importance Sample were often inaccurate.

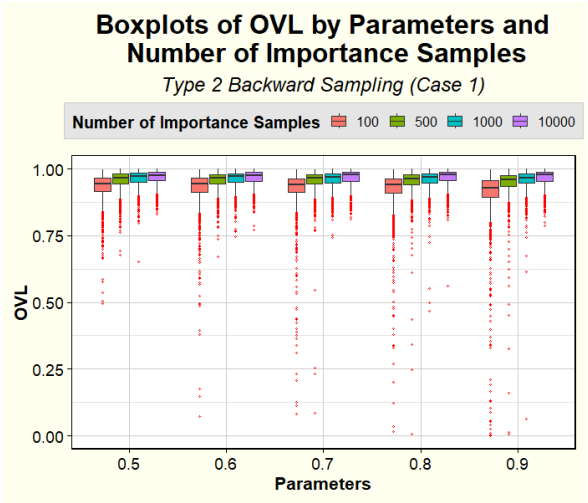


Figure 6.14

Furthermore, Likelihood of Observation was a factor that contributed slightly to the inaccurate results. In figure 6.15 we have a scatter-plot of the Observations Weights and OVL values. The distribution of weights had a very heavy tail and thus we have cut the x-axis at point $x = 20$. We can see that, for larger values of the weights, the distribution of OVL starts to shift to incorporate more outliers and showcases a larger variance. The difference may be weak and quite gradual with a Pearson coefficient approximately equal to -0.155, but it is also quite clear, as for weights smaller than 2 there were no significantly low values of OVL.

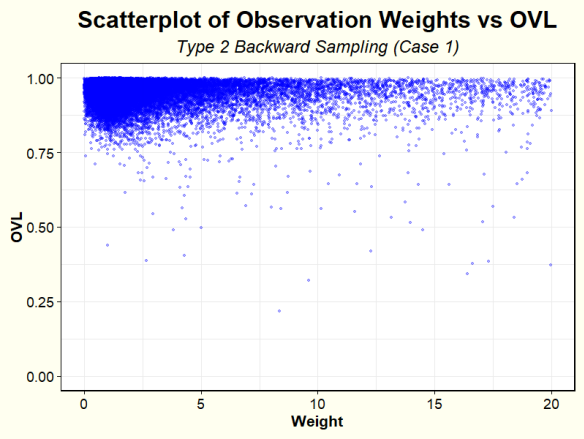
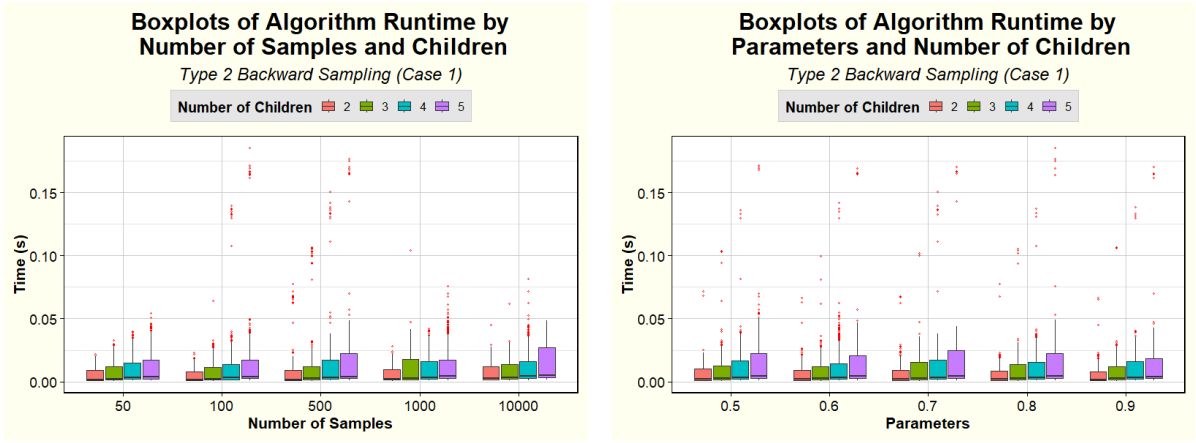


Figure 6.15

Efficiency Let us now have a look at the time efficiency of type 2 Backward Sampling for each of the first case of DAG structures.

In figure 6.16a it is illustrated that neither the number of children nor the sample size influences the algorithm's runtime significantly. The algorithm is generally fast but slower than the previous sampling types, which is to be expected given that now SIR plays a central role in our sampling procedure. The total average runtime of 0.0082 seconds and $1 \cdot 10^{-4}$ seconds of margin of error on a 95% confidence level. However, in the same figure we can see that there were some consistent outliers in which simulation took more than 2 seconds to execute.

The same exact behavior is observed in figure 6.16b, where we additionally see that copula parameters do not affect execution time.



(a) **(b)**

Figure 6.16: Type 2 Backward Sampling time Box-Plots (Case 1).

The reason for these elevated execution times in some cases was the factor of Importance Sample Size. In figure 6.17 one can observe how all of the simulations that took more than a second to complete belong to those that had 10^5 importance samples. On the other hand, the algorithm was quite fast for even 10^3 importance samples which, as

seen in subsection 6.4.1, was effective in reliably obtaining high OVL values for even the larger parameter values.

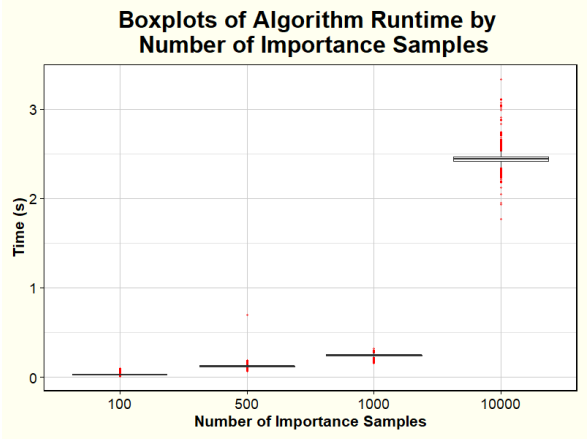


Figure 6.17

6.4.2. Second Case

Effectiveness The results for the second DAG structure were quite similar but with a relative improvement in effectiveness and smaller lower tails of the OVL distribution. The average of all OVL values was 0.9565 with a $3 \cdot 10^{-4}$ margin of error on a 95% confidence level. Just like in the first case, there was a relative increase of OVL values for larger sample sizes, while number of children did not seem to affect the accuracy of the algorithm.

Furthermore, there was a slight decrease of OVL for larger Gaussian Copula parameters, but once again, the main variable causing erroneous results was low Importance Sample Sizes. Both of the aforementioned impacts are illustrated in figure 6.18. For more figures the reader may refer to appendix A.

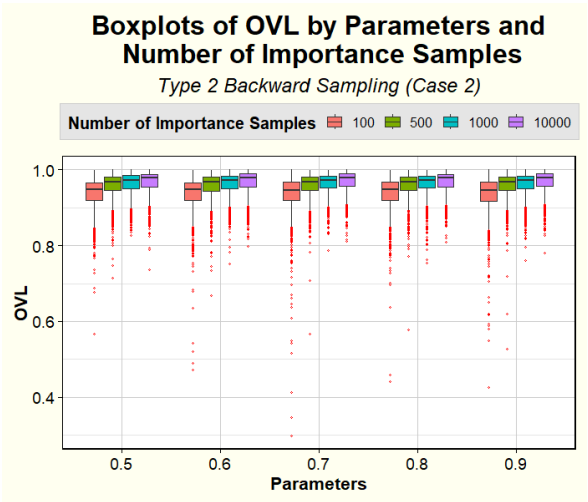
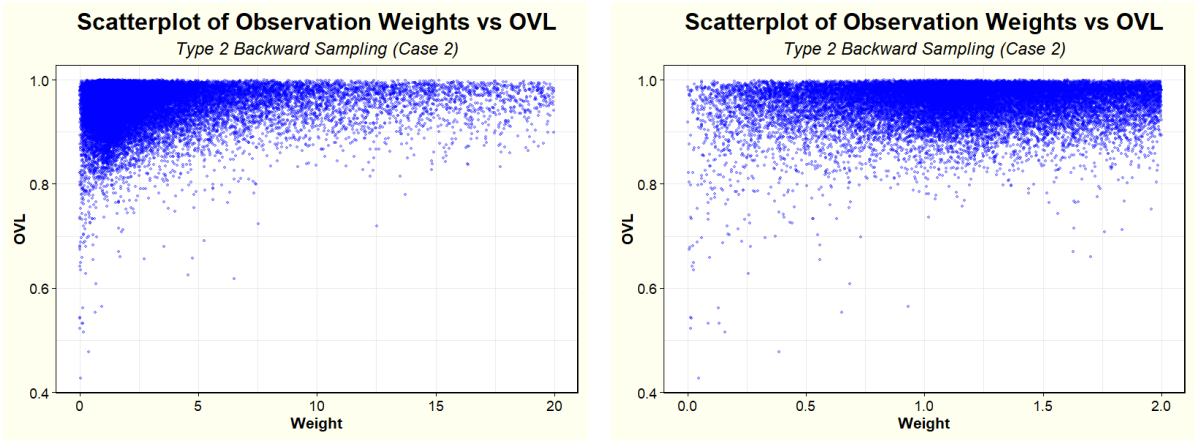


Figure 6.18

Weights in this scenario were also relevant. In figure 6.19a we have limited the weight-axis to (0,20). For larger values of Observation Weights, the values of OVL seem to

become more concentrated towards 1 and feature smaller lower tails. On the other hand, for smaller values of the weights, we have a larger variance of OVL and a higher frequency of miss-estimation. To highlight this, we additionally present figure 6.19b, where the weight-axis was cut to $(0, 2)$. In this second figure it is more clear how for observation with significantly low likelihood, SIR has a higher chance of under-performing.



(a) **(b)**

Figure 6.19: Type 2 Backward Sampling (Second Case) OVL - Weight Scatter Plots.

Efficiency Efficiency results of type 2 Backward Sampling for the second DAG structure were elevated, yet comparable to those corresponding to the first structure. The average of the algorithm's runtime was 0.0093 seconds with a margin of error of $1 \cdot 10^{-4}$ seconds on a 95% confidence level.

While neither the number of samples nor the Gaussian copula parameters had an effect on execution time, the number of children and the number of Importance samples had a clear influence. This effect is illustrated in figure 6.20.

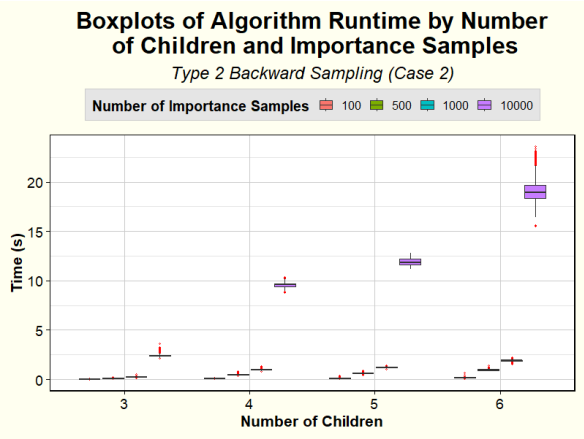


Figure 6.20

6.5. Secondary Backward Sampling Results

For testing Secondary Backward Sampling we follow a DAG structure presented in figure 6.21. The illustrated DAG includes six children of U_2, three of which are type 2 children. For DAGs with less type 1 or type 2 children we remove the necessary nodes in the reverse order of the node numbering.

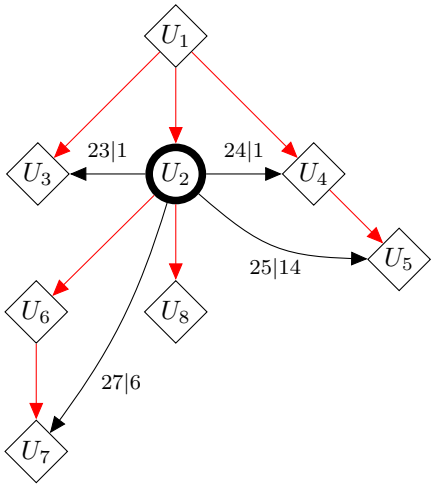


Figure 6.21: DAG structure of Secondary Backward Sampling.

The values used for the numbers of children were:

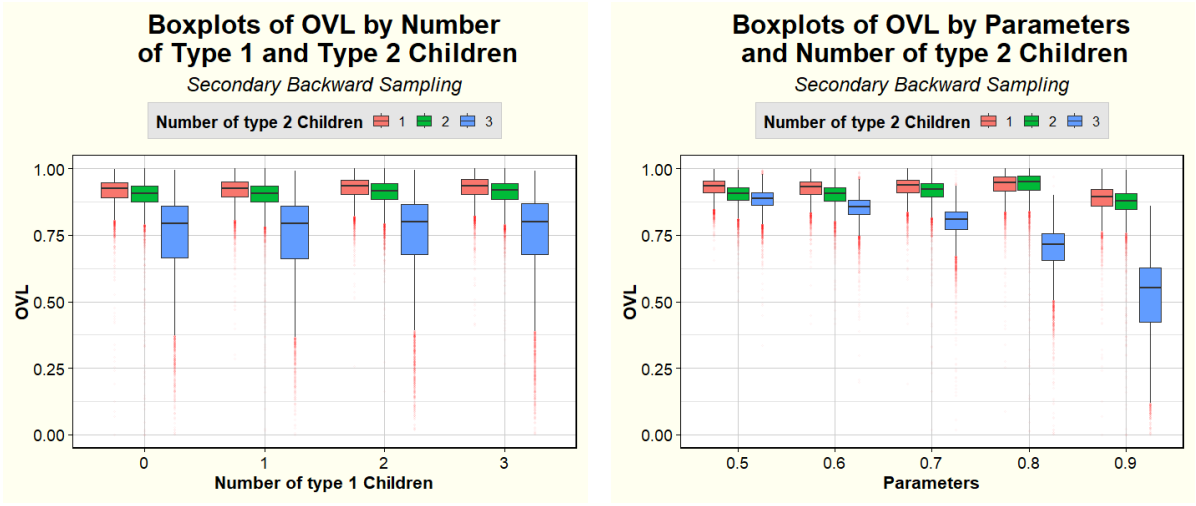
1. For type 1 children: $0, 1, 2, 3$.
2. For type 2 children: $1, 2, 3$.

Effectiveness Secondary Backward Sampling incorporates integration over the common arterial ancestor between the node we wish to sample and its type 2 children. This poses an additional layer of difficulty which we expect to affect our results.

Indeed, we observed a substantial drop in performance compared to the previous sampling types. The total average of OVL values through all of the simulations was 0.8571 with a $7 \cdot 10^{-4}$ margin of error on a 95% confidence level. Despite this lower performance, there are certain factors that contribute more than others to this, which in a realistic setting the user may want to consider.

Due to the incorporation of type 1 children in the importance sample and to the integration over the copula density factors corresponding to type 2 children, OVL values did not seem to be noticeably affected by the number of type 1 children. Type 2 children on the other hand showed a strong effect with a steep decrease in algorithmic performance being observed for three type 2 children. These results are illustrated in figure 6.22a.

Gaussian Copula parameter values also played a central role in the algorithm's performance. As seen in figure 6.22b, the effect of parameter values was negligible for one or two type 2 children, yet for three type 2 children, we observed a clear decrease in OVL values for larger copula parameters. This could be attributed to the fact that the third type 2 child belongs to a separate arterial branch than the other two. Consequently,



(a) (b)

Figure 6.22: Secondary Backward Sampling OVL Box-Plots.

there is no copula containing all three type 2 children in its marginals or conditioning set. Hence, the integration errors for the weights in importance sampling become more detrimental.

It should also be underlined that the weights were estimated through approximations by 10^3 Monte-Carlo estimates. It would be a valid assumption that increasing the number of Monte-Carlo samples would be more beneficial in a more realistic setting.

Furthermore, both the number of samples and the number of importance samples had a slight yet clear effect on the OVL values. Most prominent was that of the number of importance samples. These effects are illustrated in figure 6.23.

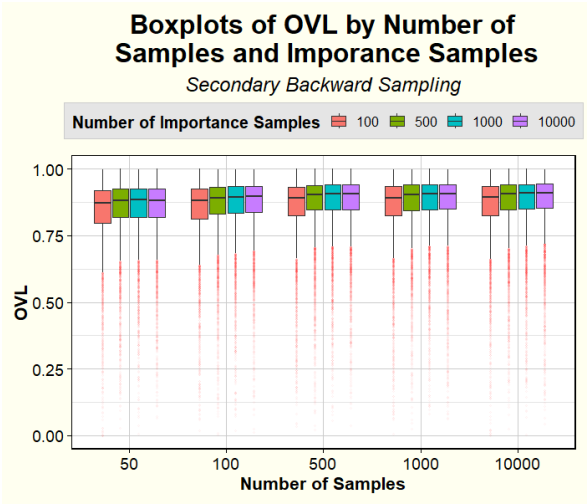


Figure 6.23

Let us now study how the likelihood weights of the observed values influenced the OVL. In figure 6.24 we can observe that the effect seems to be small yet similar to that of the second DAG case of type 2 Backward Sampling; for extraordinarily small likelihoods,

OVL values are more likely to be also smaller.

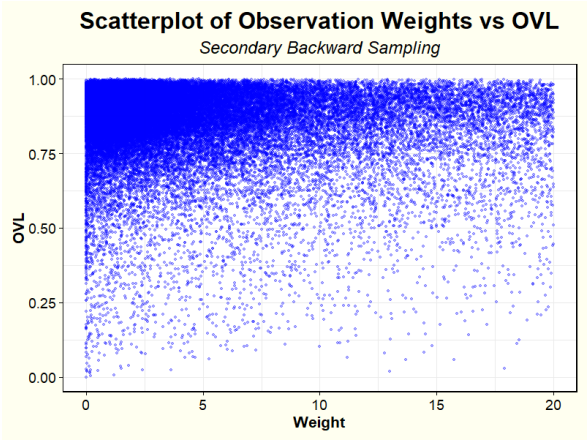
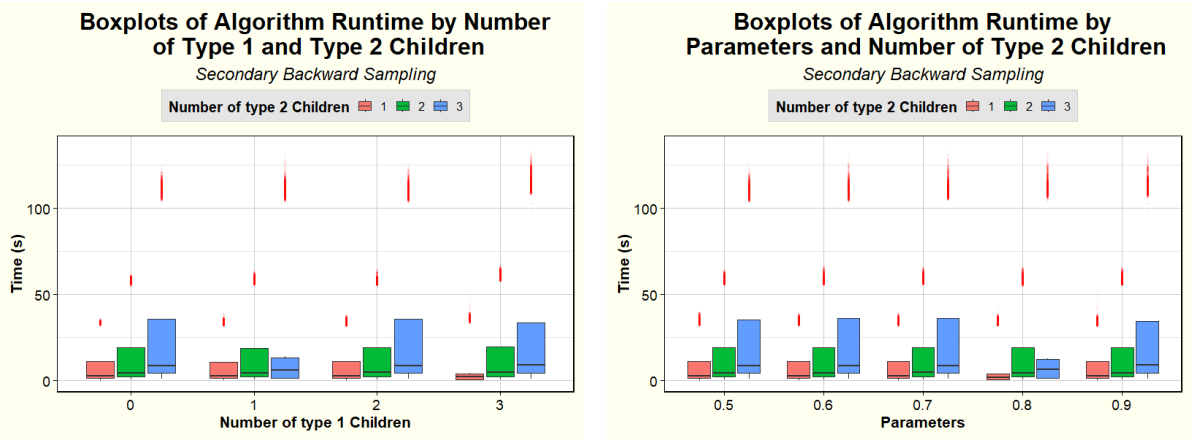


Figure 6.24

Efficiency We will now discuss the time-efficiency of the Secondary Backward Sampling Algorithm. Initially we expected Secondary Backward Sampling to be much slower than any other type of sampling due to the need to calculate each SIR weight independently through Monte-Carlo integration. Indeed the execution times were elevated with an average of 20.5 seconds with a 0.3 seconds margin of error on a 95% confidence level.



(a) **(b)**

Figure 6.25: Secondary Backward Sampling Time Box-Plots.

It should be noted that Secondary Backward Sampling was the only sampling type in which parallelization of SIR weight calculation was not possible through vectorized commands. For this reason, Secondary Backward Sampling execution time results cannot be compared to those of other sampling types.

In figure 6.25a we can see that the number of type 1 children did not strongly affect the execution time, while for higher number of type 2 children execution time became noticeably more elevated. On the other hand, in figure 6.25b, it is shown that the parameter values did not influence execution either, which is natural. In both figures, we

observe that there seem to be a lot of extraordinarily high outliers for every case. These can be attributed to the number of importance samples, as we will next discuss.

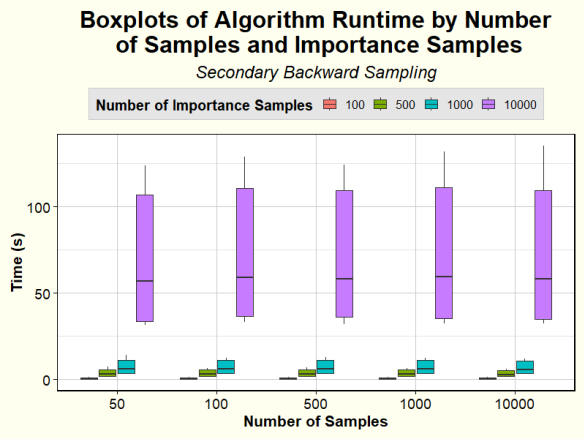


Figure 6.26

In figure 6.26, the strong effect of importance samples on execution time is clear. While for 10^3 or less samples the execution time was less than 15 seconds, for 10^4 importance samples a significant proportion of simulations took more than a minute to execute. Especially compared to the effect of the number of Importance Samples on the run-time, the number of final samples was not quite as influential.

6.6. Bilateral Sampling Results

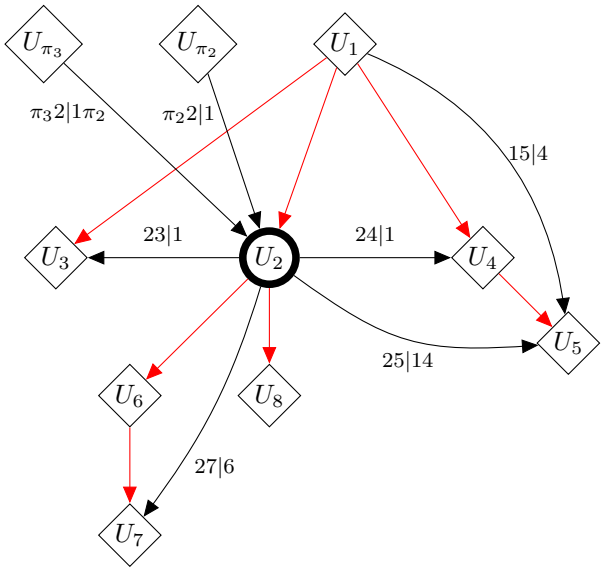


Figure 6.27: DAG structure of Bilateral Sampling.

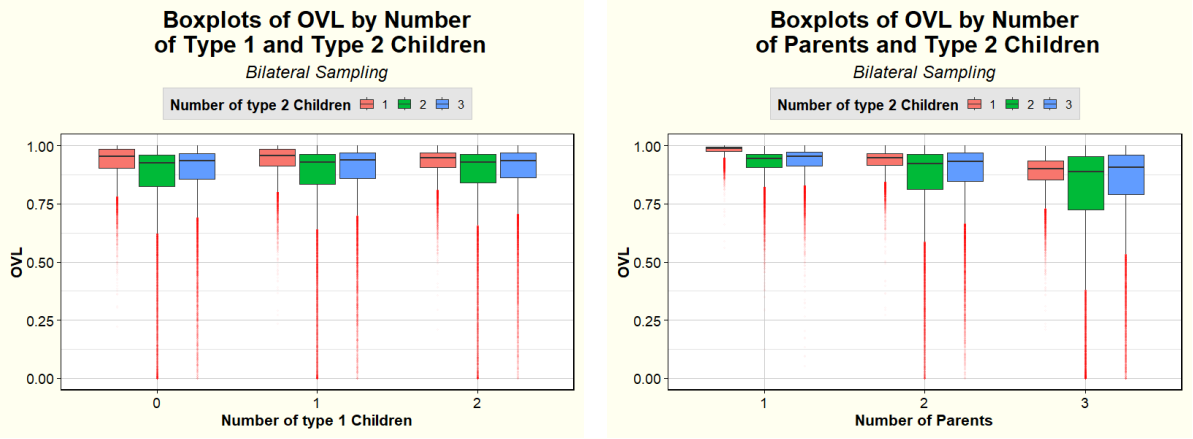
For testing Bilateral Sampling we follow a DAG structure presented in figure 6.27. This DAG includes three parents and five children of U_2, three of which are type 2 children which are type 1 children of U_1, the arterial parent of U_2. For DAGs with less type 1 or type 2 children or parents we remove the necessary nodes in the reverse order of the node numbering.

The simulation variables that were used for Bilateral Sampling were:

1. The size of the sample, taking values : $50, 100, 500, 10^3, 10^4$.
2. The size of the importance sample : $100, 500, 10^3, 10^4$.
3. The number of type 1 children, taking values : $0, 1, 2$.
4. The number of type 2 children, taking values : $1, 2, 3$.
5. The number of parents, taking values : $1, 2, 3$.
6. The Gaussian Copula parameter, taking values : $0.5, 0.6, 0.7, 0.8, 0.9$.

Effectiveness Let us now look at the results concerning the accuracy of bilateral sampling. In this case we expect the OVL values to be somewhat lower than other sampling types given that we generally had a larger sum of parents and children to incorporate in SIR. On the other hand, given that weights do not require integration over parental values (due to these being fixed by the observed values), we can expect Bilateral Sampling to perform somewhat better than Secondary Backward Sampling.

These expectations where met, with the total average of OVL values averaging at 0.8872 with a $5 \cdot 10^{-4}$ margin of error on a 95% confidence level.



(a) **(b)**

Figure 6.28: Bilateral Sampling OVL Box-Plots.

In figure 6.28a we can see the number of type 1 children affected very slightly the OVL distribution while the number of type 2 children was much more clear. Unsurprisingly, the best performance was seen for only a single type 2 child. However, the performance for two type 2 children was comparable or even worse than that for three type 2 children. While at first glance this may seem counter-intuitive, the reason behind this phenomenon is once again the arterial branches that include type 2 children. In both cases we have two arterial branches but in the second one, the second and the third child are part of the same branch and belong to the same copula. In the network in figure 6.27 we can see that this copula is $C_{15|4}$. The existence of this copula "stabilizes" the weights in SIR and shifts them more strongly towards the true value.

Figure 6.28b illustrates the expected effect of the number of parents on the performance of Bilateral Sampling. For more parents the OVL values become smaller. Despite the

inclusion of all parental values in the importance sample, the number of parents is still relevant for the performance due to the fact that parents U_{π_2}, U_{π_3}, as showcased in figure 6.27, are dependent on the type 2 children. Hence, due to the SIR weights not containing the values of U_{π_2}, U_{π_3}, SIR fails to incorporate fully the relationship between them and the type 2 children.

Overall, neither the number of samples nor the number of importance samples made an enormous change to the OVL distribution. As illustrated in figure 6.29, while there was a small increase in OVL for both larger number of samples and larger number of importance samples, the effects were less prominent than those of other simulation variables.

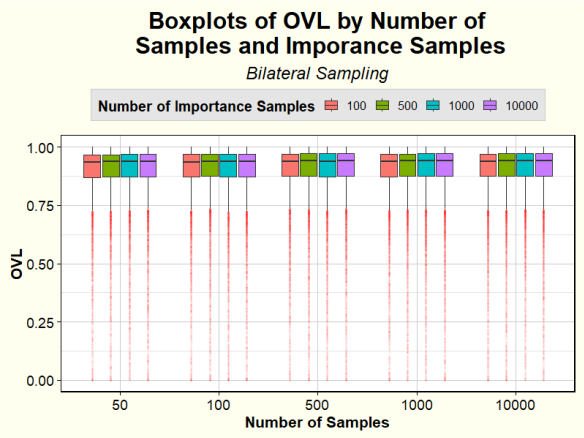
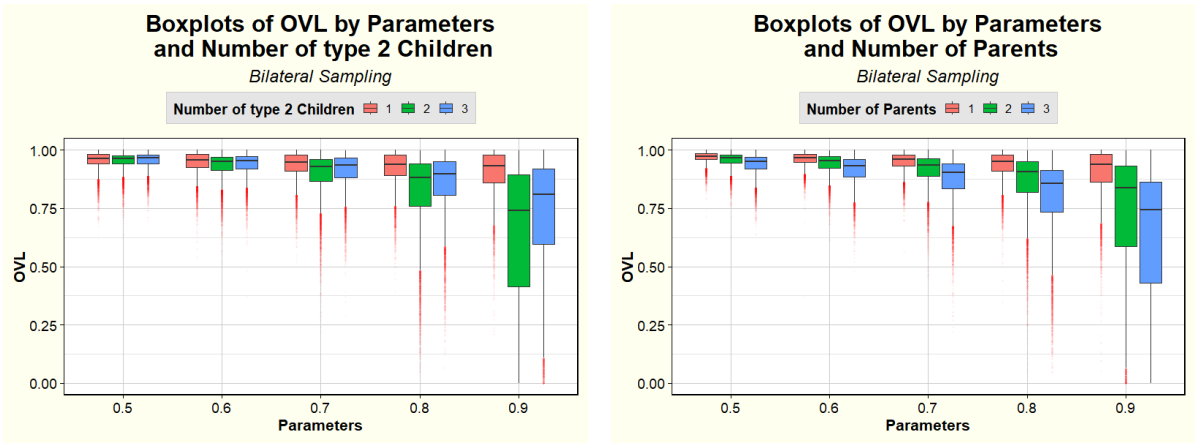


Figure 6.29

The most influential of the variables to the performance of Bilateral Sampling was the value of the Gaussian Copula Parameters. In figures 6.30a and 6.30b one can observe how the sub-optimal performance becomes more significant for higher parameter values, along with larger numbers of parents and type 2 children.



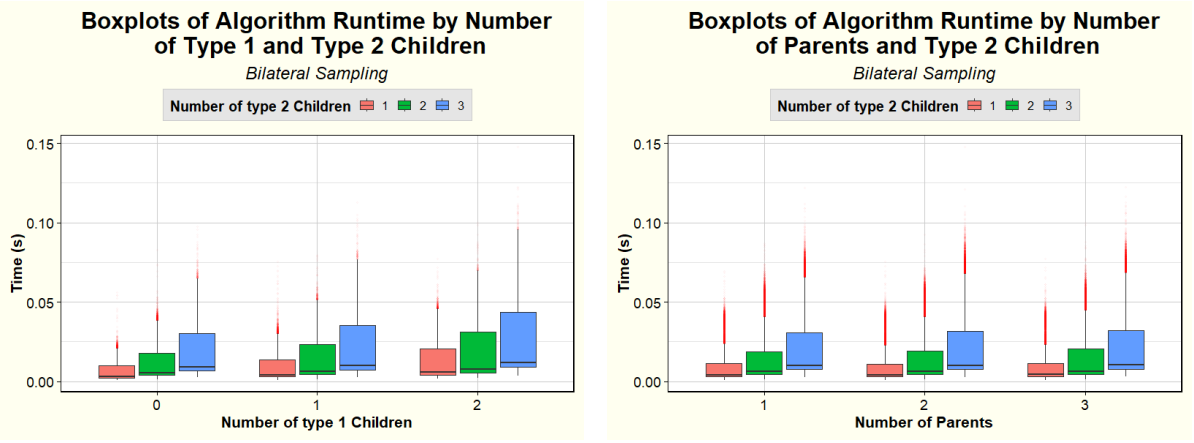
(a) **(b)**

Figure 6.30: Bilateral Sampling OVL Box-Plots based on Gaussian Copula Parameter Values.

Efficiency Let us now discuss the efficiency of the Bilateral Sampling Algorithm. The execution times were quite low with an average of 0.0156 seconds with a $7 \cdot 10^{-4}$ seconds

margin of error on a 95% confidence level.

In figure 6.31a one can observe how both the number of type 1 and type 2 children contributed in longer execution times. The effect however was slightly more clear for type 2 children. In figure 6.31b, it is also illustrated how the number of parents did not affect the execution time of the Bilateral Sampling algorithm.



(a) (b)

Figure 6.31: Bilateral Sampling time Box-Plots.

Figure 6.32 portrays the relationship between number of samples, number of importance samples and execution times. The algorithm's runtime did not display substantial changes for the different sample sizes. On the other hand, the effect of the number of importance samples was major, especially for the case of 10^4 importance samples. Despite this relationship, all execution times remained under 0.15 seconds.

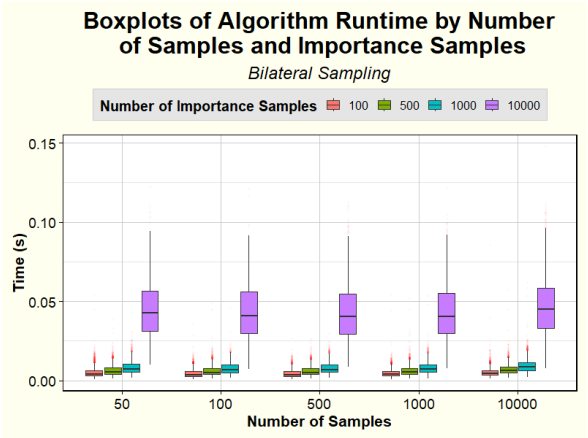


Figure 6.32

6.7. Choice of Importance Sample

Of central interest to our work is the choice of Importance Sample for SIR, when such choice is needed. We tested the performance of our methodology for different Importance Sample choices in the scope of Bilateral Sampling because it provides us with the the maximum number of such choices, namely:

Choice 1 : Choosing the Forward Sample as an Importance Sample.

Choice 2 : Choosing a Backward Sample on the maximum possible number of type 1 children as an Importance Sample.

Choice 3 : Choosing a uniformly distributed sample as an Importance Sample.

The first choice is the one already discussed in our theory of subsection 4.1.3, and the one that corresponds to the previously discussed results for Bilateral Sampling.

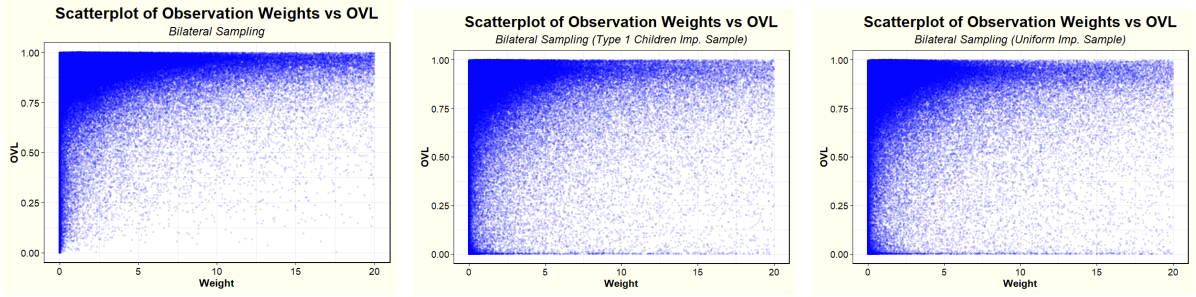
We tested the performance for the other two cases and in what follows we will discuss the results.

Overall, the first choice of importance sample was also the one that performed the best. The other two were significantly worse, with the choice of type 1 children being better than the uniform sample one. In table 6.1 we can see the total sample averages for each Importance Sample Choice.

Choice	OVL	95% Confidence Interval
Forward	0.8872	$\pm 5 \cdot 10^{-4}$
Backward	0.79	$\pm 1 \cdot 10^{-3}$
Uniform	0.775	$\pm 1 \cdot 10^{-4}$

Table 6.1: Comparison of OVL averages between the three Importance Sample choices for Bilateral Sampling

In figure 6.33 we see a comparison of the scatter plots of OVL and likelihood weights for each importance sample choice. We observe how both for the uniform or the Backward Sample, we have a significant number of low OVL values throughout the range of weights.

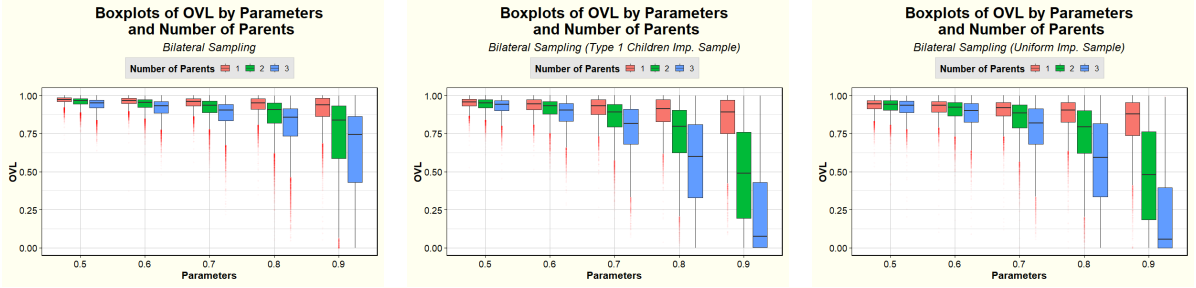


(a) **(b)** **(c)**

Figure 6.33: Bilateral Sampling time Box-Plots.

The culprit behind these cases of severe miss-estimation was a combination of elevated Gaussian Parameter values and large number of Parents. This effect is illustrated in figure 6.34. We find that for high parameter values, even the case of two parents underpeforms severely for the last two Importance Sample Choices. For three parents, the algorithm fails completely to capture the correct distribution for either choice.

The effect of Parameter values falls in line with what we have observed in section 6.6. There, for the choice of the Forward Sample as the Importance Sample, Parameter Values was the most influential variable for the algorithm's performance.



(a) (b) (c)

Figure 6.34: Bilateral Sampling time Box-Plots.

The effect of the number of parents is less intuitive. In section 6.6 we discussed the relevance of parental values on the SIR weights. In that scenario parents were included in the Importance Sample. When choosing a uniform or Backward sample as our Importance Sample we exclude parental values this is no longer the case.

Hence the negative effect of larger numbers of parents is exacerbated in our failure to incorporate their values in the importance sample on top of not being able to include their effect on type 2 children in the corresponding part of the weights.

We conclude that choosing the Forward Sample as the Importance Sample for SIR is the optimal choice for Bilateral Sampling.

6.8. Discussion

In this section we will discuss in detail the general results presented in the previous sections.

Choice	OVL	95% Confidence Interval
Forward Sampling	0.9687	$\pm 5 \cdot 10^{-4}$
Type 1 Backward Sampling	0.9686	$\pm 5 \cdot 10^{-4}$
Type 2 Backward Sampling (Case 1)	0.9518	$\pm 5 \cdot 10^{-4}$
Type 2 Backward Sampling (Case 2)	0.9565	$\pm 3 \cdot 10^{-4}$
Secondary Backward Sampling	0.8571	$\pm 7 \cdot 10^{-4}$
Bilateral Sampling	0.8872	$\pm 5 \cdot 10^{-4}$

Table 6.2: Comparison of OVL averages between the different sampling algorithms.

Overall we have seen the OVL averages presented in table 6.2. It is clear that the best performing algorithms are Forward Sampling and type 1 Backward Sampling. This is clearly because of how samples in both algorithms are sampled directly and not through the use of SIR.

Next performance-wise were the two cases for type 2 Backward Sampling followed by Bilateral Sampling.

An important remark is that Case 2 of type 2 Backward Sampling performed better than the first case, albeit slightly. A reason behind this is the inclusion of an additional type 1 child to the importance sample. Additionally, the presence of multiple type 1 children

on the same arterial branches, and in extension the existence of conditional copulas, makes SIR weights more biased towards the ”true” weight. We have touched upon this relevance of arterial branches when discussing the results in Bilateral Sampling, where we saw a slightly elevated performance for three type 2 children than two.

Lastly, Secondary Backward Sampling produced the worst results out of all sampling types. This was due to the integration over the possible values of their parents, for which an estimate was used causing deviations from the true weights.

In a larger scale of PCBN inference problems, the worse performance of Secondary Backward Sampling does not pose a significant issue to our problem. This is due to the following reasons:

- In a ”reasonably” connected artery, there is a limited number of nodes that have type 2 children. One can expect that for most applications with 20 node PCBNs that the number of nodes having type 2 children be between 0 and 5.
- In a ”reasonably” connected artery, there is a limited number of type 2 children a node can have. Each node in a 20 node PCBN that has type 2 children, can be expected to have one or sometimes two type 2 children.
- In a ”reasonably” connected artery, there is a limited number of nodes with type 2 children that need to be Secondary Backward Sampled. In order for a Secondary sample to be required, condition (B2) needs to be met, which implies the existence of an incoming arc specifying a copula conditioned on an arterial node. This is a very specific condition which very often is not met.

For all three specified reasons, a ”reasonable” connectivity of the artery refers to a sensibly sparse enough DAG structure. In practice, the application of PCBNs can be expected to be for less dense DAGs, both due to realistic assumptions on the conditional dependency structures of BNs, but also the existence of model more suited to dense graphs, such as Vine Copulas.

Thus, for practical applications, the somewhat lower performance of Secondary Backward Sampling has a smaller effect on the final output of sample propagation.

Let us now comment on the time-efficiency of the different sampling types. In figure 6.3 the different averages are presented. It is clear how Forward Sampling and both types of Backward Sampling are the most efficient ones, all of which requiring on average less than 0.01 seconds. Bilateral Sampling, albeit somewhat less efficient, still produces results with excellent speed.

On the other hand, the calculated execution times for Secondary Backward Sampling are not comparable to the other sampling types. This is because of the aforementioned inability to use vectorized commands for the calculation of the SIR weights. If vectorization were possible, the average of the algorithm’s runtime would be much closer to the averages of other sampling types.

Considering that Secondary Backward Sampling is not needed often in sample propagation, the elevated time is not an issue. In a practical setting of a PCBN inference problem for a DAG with 50 nodes and 4 nodes that require to be Secondary Backward Sampled, using the same system specifications used for our testing, inference can be

Choice	OVL	95% Confidence Interval
Forward Sampling	0.00303	$\pm 7 \cdot 10^{-5}$
Type 1 Backward Sampling	0.0035	$\pm 7 \cdot 10^{-5}$
Type 2 Backward Sampling (Case 1)	0.0082	$\pm 1 \cdot 10^{-4}$
Type 2 Backward Sampling (Case 2)	0.0093	$\pm 1 \cdot 10^{-4}$
Secondary Backward Sampling	20.5	$\pm 3 \cdot 10^{-1}$
Bilateral Sampling	0.0156	$\pm 7 \cdot 10^{-4}$

Table 6.3: Comparison of Run-time averages between the different sampling algorithms (seconds).

expected to take less than four minutes. We could expect the time to take close to four minutes if we choose that both the number of samples and number of importance samples be 10^4.

A more sensible choice would be to limit the number of importance samples to 10^3. We have seen for both Backward, Secondary Backward and Bilateral Sampling, that for 10^3 importance samples, the effectiveness of sampling remains excellent, while the expected execution time in Secondary Backward Sampling plummets to 10 seconds. In this scenario the same 50 node PCBN inference problem would take less than a minute to solve though Sample Propagation.

The user can choose the most sensible number of importance samples to balance acceptable algorithmic complexity with required accuracy levels. However, given that the number of samples does not significantly affect Secondary Backward Sampling execution times, a larger number is recommended.

7

General PCBN Inference

In this chapter we will discuss the ability of extending our proposed methodology for single-arterial PCBN inference to problems applied on the general PCBNs, that is, in PCBNs whose graph structures feature multiple arteries.

We recall from chapter 4 that the single-artery inference methodology involves the propagation of Forward, Backward, Secondary Backward and Bilateral Samples through the artery during the Forward and Backward Propagation Phases. This propagation follows closely the arterial paths.

For a general DAG structure, which includes multiple arteries, we wish to propagate samples through the trails that connect query and evidence nodes that possibly lie in different arteries. We need to consider two important points:

- The arcs joining arteries can only be assigned with conditioned copulas.
- There can be multiple distinct connections between arteries.

Due to the first point we will need to consider trails that are not arterial. The connecting arcs from one artery to another are always assigned conditional copulas.

We illustrate this first point with the example in figure 7.1. For arteries $\mathscr{A}_1$ and $\mathscr{A}_2$ to be connected we must have an arc from a node in $V_{\mathscr{A}_1}$ to a node in $V_{\mathscr{A}_2}$ or other way around. In the figure we can see the arc $U_2 \to U_3$, where U_3 belongs to $\mathscr{A}_1$, has a parent in this artery and this parent is the first in the parental ordering of U_3. The copula assigned to the arc $U_2 \to U_3$ has to be the conditional copula.

The second point of interest is the fact that there can be multiple connections between arteries. In figure 7.2, we can see an example of a PCBN in which arteries $\mathscr{A}_1$ and $\mathscr{A}_2$ are connected through two arcs, namely $U_2 \to U_4$ and $U_6 \to U_9$. The DAG of this PCBN is restricted, and if we wish to conduct sample propagation from one artery to the other, it is clear that both connections between these arteries must be taken into account.

Note that each of the connections between two arteries $\mathscr{A}_1$ and $\mathscr{A}_2$ are v-structures. One of the parents belongs to one artery while the other parent and the node with converging connection belongs to the other artery. We call such v-structures arterial links.

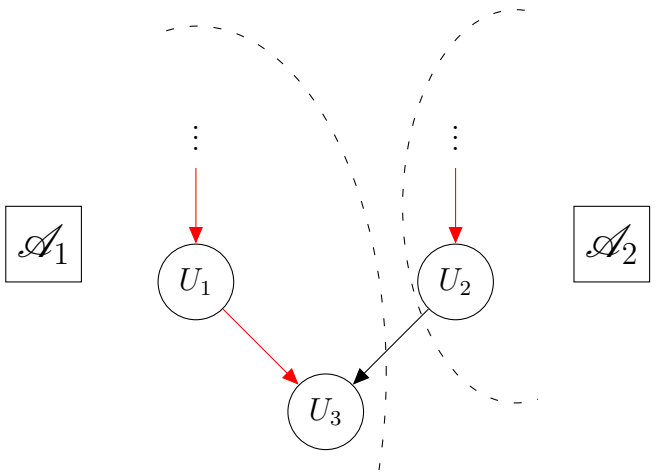


Figure 7.1: Illustration of the type of connections between two arteries.

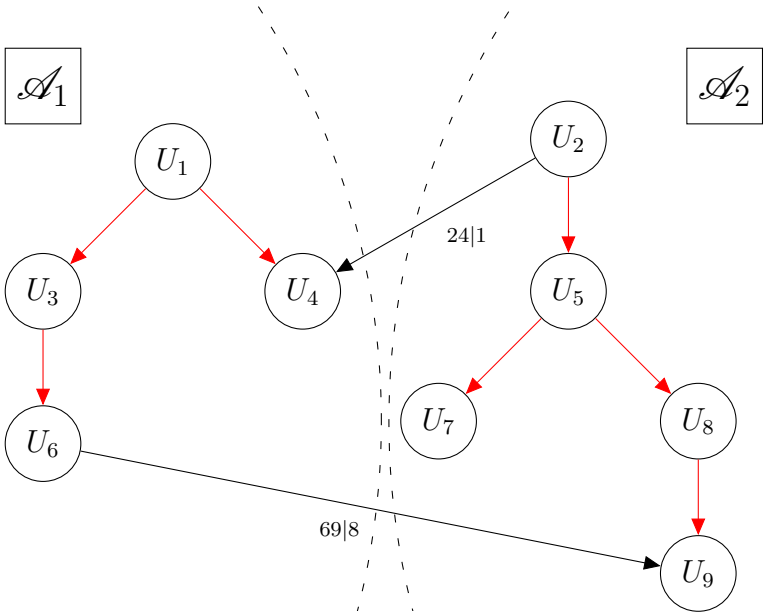


Figure 7.2: Illustration a PCBN featuring three different connections between the same arteries.

Definition 7.0.1 (Arterial Links)**.** Let $(\mathscr{G}, \mathscr{O}, \mathscr{F}, \mathscr{C})$ be a PCBN with and $\mathscr{A}_1, \mathscr{A}_2$ two arteries in $\mathscr{G}$. A converging connection of the form $U_1 \to U_2 \leftarrow U_3$ is called an **arterial link (between $\mathscr{A}_1$ and $\mathscr{A}_2$)** if for $U_2 \in V_{\mathscr{A}_1}$ either of the following is true:

- $U_1 \in pa_{\mathscr{A}_1}(U_2)$ and $U_3 \in V_{\mathscr{A}_2}$
- $U_2 \in pa_{\mathscr{A}_1}(U_2)$ and $U_1 \in V_{\mathscr{A}_2}$

By definition, one of the two parents in an arterial link is the arterial parent of the converging node and thus belongs to the same artery. On the other hand, the other parent must belong to a different artery that U_2. The connecting arc between two arteries is the most crucial for propagating the sample from one artery to another, hence we focus on the converging node and the parent from the other artery.

Definition 7.0.2 (Link Nodes)**.** Let $(\mathscr{G}, \mathscr{O}, \mathscr{F}, \mathscr{C})$ be a PCBN with and $\mathscr{A}_1, \mathscr{A}_2$ two arteries in $\mathscr{G}$. Assume that $U_1 \to U_2 \leftarrow U_3$ is an arterial link with $U_2 \in V_{\mathscr{A}_1}$, $U_1 \in pa_{\mathscr{A}_1}(U_2)$ and $U_3 \in V_{\mathscr{A}_2}$.

Then:

- U_1 is called a **transmitter (link) node**,
- U_2 is called a **receiver (link) node**
- U_1 and U_2 will be referred to as **link nodes**.

For example, in the PCBN of figure 7.2 the two connections are defined by the following arterial links:

$$U_1 \to U_4 \leftarrow U_2,$$

$$U_6 \to U_9 \leftarrow U_8.$$

For the first arterial link, the receiver node U_4 belong to the first artery, while U_1 is its arterial parent and the transmitter node U_2 belongs to the second artery. In the second arterial link, the receiver link U_9 and its arterial parent U_8 belong to the second artery, while the transmitter node U_6 to the first.

In our proposed procedure for general PCBN inference, propagation from one artery to another would be done by initially conducting arterial PCBN inference on the first artery. For the next step, a procedure similar to arterial PCBN sample propagation would be conducted for the second artery, with the inclusion of all arterial links.

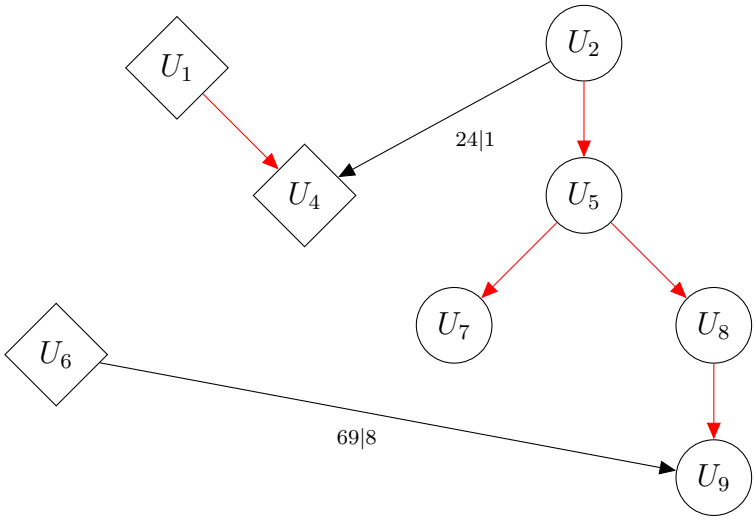


Figure 7.3: Illustration of the inclusion of v-structures for cross-arterial sample propagation.

In figure 7.3 an example of the additional stage related to the propagation from $\mathscr{A}_1 \to \mathscr{A}_2$ in the PCBN of figure 7.2 is shown. The process requires propagation in the first artery for obtaining the samples of $\mathscr{A}_1$'s nodes that belong to arterial links, conditioned on the evidences in $\mathscr{A}_1$. These are included in the next stage, when performing inference in the second artery using single artery propagation for $\mathscr{A}_2$.

Remark. We note that a receiver link node may have multiple extended arterial parents. As the focus lay in the link nodes, this does not influence our approach for multi-arterial sample propagation. We can instead treat all extended arterial parents the same way we treat the receiver's arterial parent. Hence we will include them in the modified sample propagation procedure of the second artery.

7.1. Sampling Modifications

As we will see in the next section, the Modified Arterial Sample Propagation Procedure differs only when sampling nodes in the artery that are arterial links.

In this section we will examine ways we can modify the sampling algorithms from chapter 4 to be applied to such nodes.

Let us start with a simple example in figure 7.4. We assume that the final arterial propagation samples of the nodes U_1 and U_3 have been obtained. Now we wish to sample U_2. This is the simplest transmitter node sampling problem.

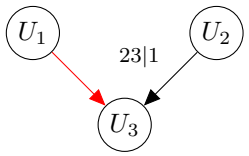


Figure 7.4: Elementary example of link node sampling.

In this simple example the required sample can be obtained through the h-function recursion of copula $C_{24|1}$. That is:

$$\hat{u}_2 = h^{-1}_{1\underline{2}}(u_1, h^{-1}_{\underline{23}|1}(U, u_{3|1})) = h^{-1}_{\underline{23}|1}(U, u_{3|1})$$

where $U \sim \mathcal{U}(0,1)$. The second equality is due to the independence between U_1 and U_2.

If instead we have a final sample of U_2 and wish to propagate towards node U_3 we have a receiver node sampling problem. In this simple example, we can obtain the required sample of U_3 through Forward Sampling using a uniform sample of U_1.

Next we will explore how sampling algorithms are modified for nodes that belong to arterial links. The general approach to including nodes from other arteries falls in line with the sampling theory introduced in chapter 4. The main difference is the inclusion of all nodes from the different artery with which the node is directly connected, i.e. it belongs to the node's Markov Boundary, which is defined in definition 2.2.3.

7.1.1. Link Node Sampling

Consider U_1 being a link node from artery $\mathscr{A}_2$ and assume that we need to sample U_1 using one of the sampling algorithms from chapter 4 while incorporating the information arriving to U_1 from the other arteries it is connected to.

An example that we will use in this section is presented in figure 7.5. In this network U_4 is both a receiver and a transmitter node.

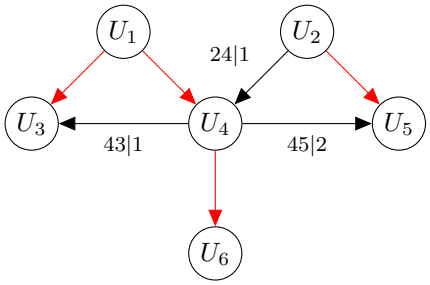


Figure 7.5: Reference example for link node sampling.

To modify the sampling algorithms for multiple arteries, the inclusion of U_1's Markov Boundary nodes that belong to different arteries is needed. These need to be included additionally to the nodes that are considered in chapter 4.

The modification in Case 2 Backward, Secondary Backward and Bilateral sampling, be done by multiplying the SIR weights by the product of all copula densities assigned to incoming and outgoing arterial link arcs. When sampling link node U_1 belonging to artery $\mathscr{A}_2$, such product will be of the form:

$$\prod_{U_i \in pa_{\mathscr{A}_1}(U_1)} c_{i1|K_{i1}} \prod_{U_i \in ch_{\mathscr{A}_1}(U_1)} c_{1i|K_{1i}}. \tag{7.1}$$

In the example presented in figure 7.5, this product is equal to:

$$c_{24|1} c_{45|2}$$

as these copulas correspond to links between two arteries.

To modify the Case 1 Backward Sampling, we will need to obtain the samples through SIR instead of the h-function recursion. As the importance sample the original Forward or Case 1 Backward Samples are used and then the assigned unnormalized weights will be equal to the product 7.1.

The SIR sampling will not be needed in Forward Sampling of receiver nodes that are not transmitter nodes. In this case, we can conduct Forward Sampling as usual, by using the h-function recursion of the copula with maximum conditioning set. This can be observed in the example shown in figure 7.5. If U_5 was not present in this DAG then U_4 would be a receiver node and not a transmitter, hence its Forward Sample would be done through the h-function recursion of copula $C_{24|1}$.

In all cases, if the conditioning set K_{ij} of a copula $C_{ij|K_{ij}}$ includes nodes from arteries besides $\mathscr{A}_1$ or $\mathscr{A}_2$, then integration of the weights must be conducted over the values of those nodes, in the same way that Secondary Sampling integrates the weights over the values of the arterial diverging ancestor.

Additionally, if the conditioning set K_{ij} of a copula $C_{ij|K_{ij}}$ includes extended arterial parents of U_1, while U_1 is being neither Forward nor Bilaterally Sampled, then U_1's extended arterial parents must also be integrated out.

To showcase the previous cases that require integration consider as an example of the network from figure 7.6. Here the copula $C_{24|01}$ is conditioned on variable U_0, which

belongs to a different artery, as well as the extended arterial parent U_1 of U_4. In this example, to conduct (Case 1) Backward Sampling of U_4 we must use SIR. The importance sample will be the original arterial Case 1 Backward Sample obtained through copula C_{46} and the respective SIR unnormalized weights equal to:

$$w = c_{45|2} \int_{[0,1]^2} c_{24|01} du_0 du_1.$$

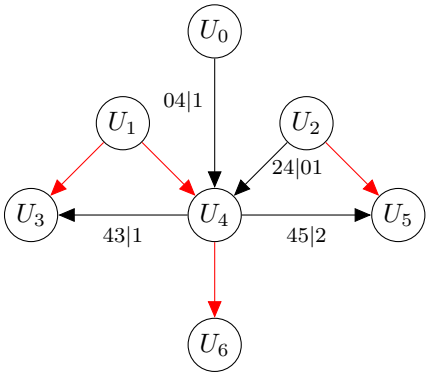


Figure 7.6: Example for integration over nodes from other arteries during modified sampling.

7.1.2. Non-Link Node Sampling

During the modified Arterial Propagation from $\mathscr{A}_1$ to $\mathscr{A}_2$, nodes of $\mathscr{A}_2$ whose Markov Boundary includes nodes in $\mathscr{A}_1$, must include the latter in their sampling. This does not only happen for the link nodes of $\mathscr{A}_2$ but also for extended arterial parents of receiver link nodes of $\mathscr{A}_2$, as the respective transmitter nodes belong to their Markov Boundaries.

An example that we will use in this segment is presented in figure 7.7. In this example, U_3 is part of the arterial link $U_3 \to U_6 \leftarrow U_4$ and has U_4 in its Markov Blanket. Hence, we must include U_4 in its sampling

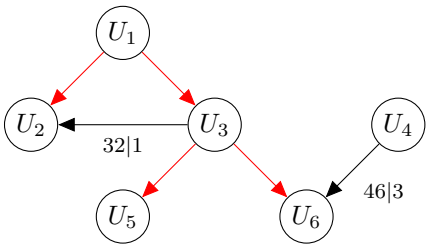


Figure 7.7: Reference example for non-link node sampling.

Assume that we wish to sample U_1, which has a receiver node $U_2 \in ch_{\mathscr{G}_{\mathscr{A}}}(U_1)$ as its extended arterial child. Because transmitters are part of the Markov Boundary (they are parents of common child receivers) their inclusion to sampling is only relevant when the the receiver is also required for sampling. This is because a transmitter will be dependent on U_1 only conditioned on the converging connection formed by the receiver.

Therefore, in the following three scenarios the transmitter is not included:

- Forward Sampling of non-link nodes will never include transmitters.
- Backward Sampling of non-link nodes will not include transmitters if the receiver common child is a type 2 child.
- Bilateral Sampling of non-link nodes will not include transmitters if the receiver common child is excluded from Bilateral Sampling.

In each of these three cases, the receiver node U_2 is not included in the sampling of U_1. If the common child between U_1 and a transmitter is excluded from the sampling procedure, we cannot add the connected transmitter as the converging connection of U_2.

In figure 7.8 we can see an example of a network in which both non-link nodes U_1 and U_3 have the transmitter node U_2 in their Markov Boundary. Let us illustrate the aforementioned exceptions to U_2's exclusion from the sampling of either U_1 or U_3.

Forward Sampling of either U_1 or U_3 excludes U_2. U_4 is a type 2 child of U_3, hence neither U_2 nor U_4 are included in U_3's Backward Sampling. Lastly, the Bilateral Sample $b^{bil}_{1\to 4}$ of U_1 excludes U_4, hence we will also not include U_2.

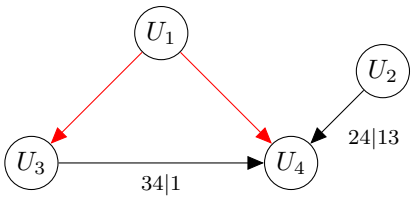


Figure 7.8: Example for non-link node sampling that illustrates exceptions to the inclusion of transmitter nodes.

As in case of Link Node sampling for Case 2 Backward, Secondary Backward and Bilateral Sampling we incorporate each transmitter U_3 by including the copula density $c_{32|K_{32}}$ in the product of SIR unnormalized weights.

Once again, for Case 1 Backward Sampling, inclusion of transmitters must be done through SIR and using the product of copulas assigned to link arcs as unnormalized SIR weights.

As in Link Node Sampling, integration over variables in the conditioning set K_{32} not belonging to in either $\mathscr{A}_1$ or $\mathscr{A}_2$ is required. Additionally, when performing Backward or Secondary Backward Sampling in case when K_{32} contains parents of U_1, those must also be integrated out, since they should not to be included in U_1's sampling.

7.2. Multiple Artery Propagation

In this section we will discuss the road-map of Sample Propagation between different arteries.

We have already mentioned in the introduction of this chapter the outline of Propagation between two arteries $\mathscr{A}_1$ and $\mathscr{A}_2$ with corresponding sets of evidences E_1 and E_2. We wish to perform inference in artery $\mathscr{A}_2$ that includes all evidences.

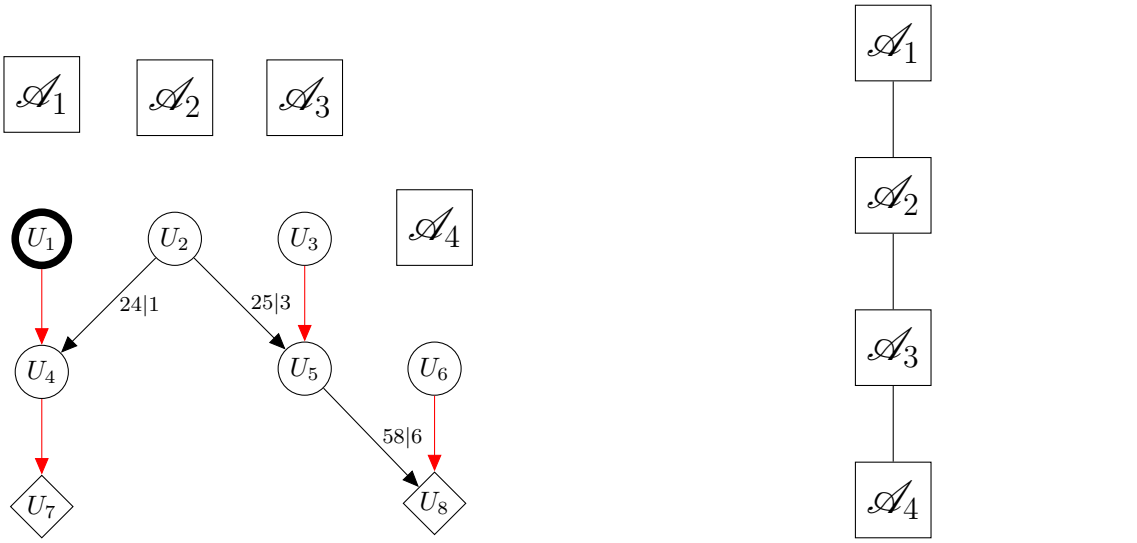
First the single arterial sample propagation in $\mathscr{G}_{\mathscr{A}_1}$ as presented in chapter 4 is performed. Then we proceed to do the same for $\mathscr{G}_{\mathscr{A}_2}$ with the inclusion of arterial links.

Propagation in $\mathscr{G}_{\mathscr{A}_2}$ remains the same as described in chapter 4, with the only differences being the modification of sampling algorithms proposed in the previous section. The resulting final samples will be the inference samples that include both E_1 and E_2.

7.2.1. Full Propagation

So far we showed how to conduct sample propagation from one artery to another. Now we wish to conduct inference through multiple arteries. In the general scenario of a PCBN, the model will have a number of arteries α, $\mathscr{A}_1 \dots \mathscr{A}_\alpha$. The connectivity between these arteries in a PCBN will be represented by an undirected graph where each node is an artery and each arc represents the existence of active arterial link.

Example 7.2.1. In figure 7.9a we can see an example of a PCBN with four arteries, corresponding to roots U_1, U_2, U_3 and U_6 of the PCBN. The PCBN also features three arterial links, and figure 7.9b contains the visual representation of the connections between arteries.



(a) Multi-arterial PCBN.

(b) Arterial connections in PCBN of figure 7.9a.

Figure 7.9: Example of the graphical representation of Arterial connections.

In the simple example of figure 7.9, it is straightforward to perform inference, given the ability to propagate samples from all arteries to $\mathscr{A}_1$.

Let us denote the final propagation samples in artery $\mathscr{A}_1$ conditioned on the evidences $E_1, E_2, \dots, E_n$ as $\boldsymbol{S}^{\mathscr{A}_1}_{\mathbf{2,\dots,n}}$, where $E_1, E_2, \dots, E_n$ refers to the evidence variables that belong to the arteries $\mathscr{A}_1, \mathscr{A}_2, \dots, \mathscr{A}_n$, respectively. This notation may also be used to refer to the inference problem $f_{Q_1|E_1,\dots,E_n}$ directly, where Q_1 is the set of query variables that belong to artery $\mathscr{A}_1$.

With the set notation we can consider the inference problem $S^{\mathscr{A}_1}_{1,2,3,4}$ in the PCBN in figure 7.9. $\mathscr{A}_1$ is only connected to artery $\mathscr{A}_2$, hence the final samples that $\mathscr{A}_2$ must send to $\mathscr{A}_1$ are $S^{\mathscr{A}_2}_{2,3,4}$, which contain the information of evidences E_2, E_3, E_4.

To calculate $S^{\mathscr{A}_2}_{2,3,4}$, $\mathscr{A}_2$ must receive samples conditioned on E_3, E_4 from its connected arteries. $\mathscr{A}_1$ cannot send such samples to $\mathscr{A}_2$, hence only $\mathscr{A}_3$ sends the samples $S^{\mathscr{A}_3}_{3,4}$.

Finally, the samples $S^{\mathscr{A}_3}_{3,4}$ are calculated by the single inter-arterial propagation from $\mathscr{A}_4$ to $\mathscr{A}_3$. In this case $\mathscr{A}_4$ sends its single-arterial final samples $S^{\mathscr{A}_4}_4$.

The Multi-Arterial Propagation procedure is presented in the compact form below:

$$S^{\mathscr{A}_4}_4 \rightsquigarrow S^{\mathscr{A}_3}_{3,4} \rightsquigarrow S^{\mathscr{A}_2}_{2,3,4} \rightsquigarrow S^{\mathscr{A}_1}_{1,2,3,4}$$

In figure 7.10 we can see a graphical representation of the sample propagation.

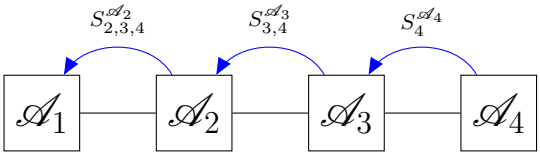


Figure 7.10: Graphical representation of the inter-arterial propagations required for calculating $S^{\mathscr{A}_1}_{2,3,4}$.

The previous example was a very simple one because the undirected graph corresponding to the connectivity between arteries was serial. The same approach could be used for PCBNs whose arterial undirected graph representations are polytrees (no loops).

In a general PCBN this may not be the case, and this poses a main problem in building a comprehensive method for sample propagation in general PCBNs that feature multiple arteries.

Therefore, more research is needed to successfully describe this problem.

8

Trail Properties of Arteries

In this section we will discuss some crucial results about the Properties of trails in arteries and extended arteries. The focus will lay on how the existence of certain arcs in the DAG imply the existence of other arcs, as well as the ability to restrict the possible parental orderings.

This work is mostly relevant for the inference stage of PCBNs. The results in this chapter are closely related to the ones presented in Horsman [36], where methods to find parental orderings which do not lead to the need of integrations in restricted DAGs are introduced.

The following proposition is a simple observation about the extended arterial density.

Proposition 8.0.1. *Let $(\mathscr{G}, \mathscr{O}, \mathscr{F}, \mathscr{C})$ be a restricted PCBN with an artery $\mathscr{A}$.*

For every $U_1, U_2, U_3 \in V_{\mathscr{A}}$ such that $C_{12;3} \in \mathscr{C}$, then either $C_{31} \in \mathscr{C}$ or $C_{13} \in \mathscr{C}$.

Proof. We have $U_1 \to U_2$ due to $C_{12;3} \in \mathscr{C}$. This implies that $U_3 \in pa(U_2)$ and $U_3 <_2 U_1$. Therefore:

$$C_{32} \in \mathscr{C}.$$

Since both U_1, U_3 belong to the same artery corresponding to root U, then

$$\exists\, U_1^1, \dots U_1^{n_1-1}, U_3^1, \dots, U_3^{n_3-1} \in V$$

such that:

$$U \to U_1^1 \to \cdots \to U_1^{n_1-1} \to U_1^{n_1} \equiv U_1$$
$$U \to U_3^1 \to \cdots \to U_3^{n_3-1} \to U_3^{n_3} \equiv U_3$$

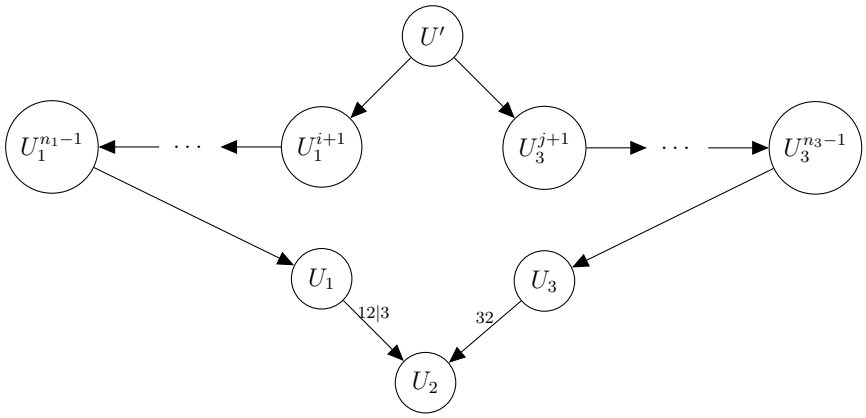
We can assume that these are the shortest such trails, hence they have no chords.

Let $U' \equiv U_1^i = U_3^j$ be the earliest common ancestor of U_1, U_3. Indeed the earliest common ancestor of U_1 and U_3 exists because the arterial root U is a common ancestor.

Assume that $U_1, U_3 \neq U'$, hence neither $U_1 \in pa(U_3)$ nor $U_3 \in pa(U_1)$ hold. Then there exists the trail:

$$U_2 \leftarrow U_1 \leftarrow \cdots \leftarrow U_1^{i+1} \leftarrow U' \to U_3^{j+1} \to \cdots \to U_3 \to U_2$$

which consists only of serial connections and the diverging connection $U_1^{i+1} \leftarrow U' \rightarrow U_3^{j+1}$. This trail is visualized in the following graph. Because $U_1, U_3 \neq U'$, the length of the trail is strictly larger than three and because U' is the earliest common ancestor of U_1 and U_3, there are no chords in the trail.



Thus, this trail is an active cycle in $\mathscr{G}$ if U_1 and U_3 are not connected. Therefore, either $U_1 \in pa(U_3)$ or $U_3 \in pa(U_1)$. □

Proposition 8.0.1 implies that all v-structures created by the first two parents, according to the parental ordering of a node, are coupled. Another way to gain an intuitive understanding behind this, is to notice that the existence of copula $C_{12|3}$ implies the necessity of being able to calculate the conditional margin: $u_{1|3}$. Due to U_1 and U_3 belonging to the same artery, they are not independent of each other as they have a common ancestor, the root of the artery. Therefore, we need the specification of copula C_{13} or copula C_{31} to be able to calculate this conditional margin.

This result cannot be extended to any two parents of U_2, not even when the assumption that they are consecutive parents in ordering of parents of U_2 is made. For a counterexample we can see the PCBN in figure 8.1, in which U_2 and U_3 are consecutive parents of U_4 according to its parental ordering, however the v-structure formed by the incoming arcs $U_2 \rightarrow U_4$ and $U_3 \rightarrow U_4$ is not coupled, despite the graph being a restricted DAG.

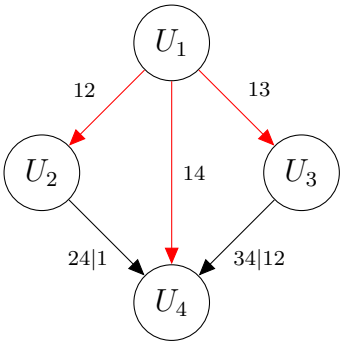


Figure 8.1: Counterexample for the extension of proposition 8.0.1.

Next, we take a closer look at some properties that are implied by the existence of arcs between two nodes of which one is a type 1 child of the other.

Lemma 8.0.2. *Let $(\mathcal{G}, \mathcal{O}, \mathcal{F}, \mathcal{C})$ be a restricted PCBN with an artery $\mathcal{A}$ and let $U_i \in V_{\mathcal{A}}$, U_0 a type 1 child of U_i. Then, for every $U_j \in an_{\mathcal{A}}(U_0) \cap de_{\mathcal{A}}(U_i)$, we have:*

$$U_j \in pa_{\mathcal{G}_{\mathcal{A}}}(U_0) \text{ with } U_j <_0 U_i.$$

Proof. Note that U_0 being a type 1 child of U_i implies that $U_i \in an_{\mathcal{A}}(U_0)$.

First, we will show that in order for such a U_j to exist, U_i cannot be the arterial parent of U_0.

- Assume that U_i is an arterial parent of U_0, hence $U_i \in pa_{\mathcal{A}}(U_0)$. For $U_j \in an_{\mathcal{A}}(U_0)$ we will show that $U_j \notin de_{\mathcal{A}}(U_i)$.

  Since U_0 is the type 1 child of U_i then U_i is the only arterial parent of U_0. Since $U_j \in an_{\mathcal{A}}(U_0)$ then U_j is either equal to U_i or it is an arterial ancestor of U_i. Therefore, U_j cannot be an arterial descendant of U_i.

We will prove this lemma by induction on the order of U_i in the parental order of U_0.

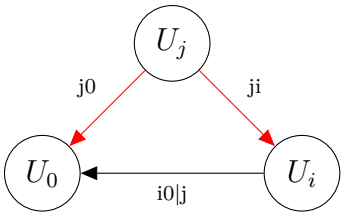
- We showed that U_i cannot be first in the parental order of U_0. Let us assume it is the second. We will show that any node $U_j \in an_{\mathcal{A}}(U_0) \cap de_{\mathcal{A}}(U_i)$ satisfies:

  $$U_j \in pa_{\mathcal{G}_{\mathcal{A}}}(U_0) \text{ with } U_j <_0 U_i.$$

  U_i is the second node in U_0's parental order, hence the specified copula of the arc $U_i \to U_0$ is of the form $C_{i0|j}$. We see that $U_j \in pa_{\mathcal{A}}(U_0)$ with specified copula C_{j0}. U_j satisfies $U_j \in pa_{\mathcal{G}_{\mathcal{A}}}(U_0)$.

  Since U_i is the second in the parental order of U_0 then by proposition 8.0.1, either $C_{ij} \in \mathcal{C}$ or $C_{ji} \in \mathcal{C}$.

  If $C_{ji} \in \mathcal{C}$, then $U_j \to U_i$ and we have the following DAG structure:

  Therefore $U_i \notin an_{\mathcal{A}}(U_0)$, which contradicts the initial assumptions for U_i.

  Hence, $C_{ij} \in \mathcal{C}$ which entails that $U_j \in de_{\mathcal{A}}(U_i)$. Furthermore, given that U_j is the arterial parent of U_0, $U_j \in an_{\mathcal{A}}(U_0)$. Therefore $U_j \in an_{\mathcal{A}}(U_0) \cap de_{\mathcal{A}}(U_i)$.

  In this case $U_j \in pa_{\mathcal{G}_{\mathcal{A}}}(U_0)$ with $U_j <_0 U_i$.

  There are no other variables U_l that meet the conditions because U_j is the only node in $an_{\mathcal{A}}(U_0) \cap de_{\mathcal{A}}(U_i)$.

- Let us assume that the lemma holds for all nodes U_i up to the n'th in the parental order of U_0.

We will prove the lemma holds for the n+1'th, denoted as U_i, element in the parental order of U_0. We will show that if U_0 is a type 1 child of U_i, any node $U_j \in an_{\mathscr{A}}(U_0) \cap de_{\mathscr{A}}(U_i)$ satisfies:

$$U_j \in pa_{\mathscr{G}_{\mathscr{A}}}(U_0) \text{ with } U_j <_0 U_i.$$

Due to $U_i \in an_{\mathscr{A}}(U_0)$ we have a path between U_i and U_0 that consists of only serial connections, whose arcs are assigned with unconditional copulas, i.e.:

$$U_i \to U_{i_1} \to \cdots \to U_{i_m} \to U_0 \tag{8.1}$$

with $U_{i_j} \in pa_{\mathscr{A}}(U_{i_{j+1}})$.

- We will show that since $U_i \to U_0$ then also $U_{i_1} \to U_0$.

If it was not the case then the following trail would be present:

$$U_0 \leftarrow U_i \to U_{i_1} \to \cdots \to U_{i_m} \to U_0,$$

such that the sub-trail $U_{i_1} \to \cdots \to U_0$ is the shortest such sub-trail (hence has no chords). If U_{i_1} is not a parent of U_0, this trail's length is at least five so it would be an active cycle. Thus $U_{i_1} \to U_0$.

- We will now show that $U_{i_1} <_0 U_i$.

Assume that the above is not the case, hence $U_i <_0 U_{i_1}$. Then the copula $C_{i_10|K_{i_10}}$, assigned to arc $U_{i_1} \to U_0$, contains U_i in its conditioning set K_{i_10}.

Since the PCBN is assumed to be restricted then it must be that copulas required to calculate the following conditional margin

$$u_{i_1|K_{i_10}}$$

are specified. We will show that computation of such conditional margin is not possible if $U_i <_0 U_{i_1}$.

Due to $U_{i_m} \in pa_{\mathscr{A}}(U_0)$, $U_{i_m} \in K_{i_10}$ and U_{i_m} is the first in the parental order of U_0. This means that at least one of the nodes $U_{i_2}, \ldots, U_{i_m}$ is part of K_{i_10}. In the calculation of the conditional margin $u_{i_1|K_{i_10}}$ some of these nodes may be omitted from K_{i_10} due to conditional independencies, but at least one such node will be conditionally dependent of U_{i_1} given the rest of the nodes. Let these nodes be:

$$U_{\iota_1}, \ldots, U_{\iota_k}$$

Due to U_i being the arterial parent of U_{i_1}, U_{i_1} is also dependent of U_i conditional on the rest of the nodes in K_{i_10}. This means that neither U_i nor U_{ι_j} nodes can be omitted from the calculation of the conditional margin.

However, the arc $U_i \to U_{i_1}$ is assigned with the unconditional copula C_{ii_1}, meaning that the calculation of $u_{i_1|K_{i_10}}$ cannot be done. This violates the assumption of a restricted PCBN. Meaning that $U_{i_1} <_0 U_i$.

By repeated use of the inductive assumption on U_{i_1}, and for all nodes in the arterial path between U_{i_1} and U_0, we have that $U_l \in pa_{\mathscr{G}_{\mathscr{A}}}(U_0)$ and $U_l <_0< U_{i_1} <_0 U_i$, for $l = 2 \dots m$.

□

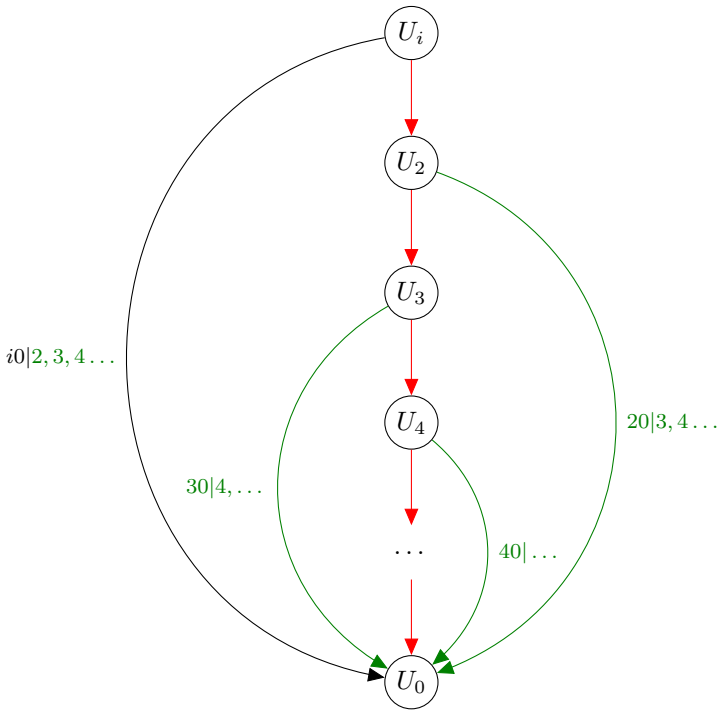


Figure 8.2: Visual representation of lemma 8.0.2. Green arcs and specified copulas are the ones implied by the arc $U_i \to U_0$.

We can observe the justification of lemma 8.0.2 in PCBN in figure 8.2. The existence of arc $U_i \to U_0$ implies the existence of all arcs $U_j \to U_0$ for U_j being nodes that are arterial descendants of U_i and arterial ancestors of U_0. Furthermore, the lemma provides us with the relative positions of these nodes in the parental order of U_0.

Next we will show a similar results for type 2 children. Both the statement and proof are similar to those corresponding to lemma 8.0.2, however they are treated separately due to differences caused by the structural distinction between type 1 and type 2 children.

Lemma 8.0.3. *Let $(\mathscr{G}, \mathscr{O}, \mathscr{F}, \mathscr{C})$ be a restricted PCBN with an artery $\mathscr{A}$ and let $U_i \in V_{\mathscr{A}}$, U_0 a type 2 child of U_i. Then, for $U_j \in pa_{\mathscr{A}}(U_i)$, we have:*

$$U_j \in pa_{\mathscr{G}_{\mathscr{A}}}(U_0) \text{ with } U_j <_0 U_i.$$

Proof. U_0 being a type 2 child of U_i, by definition implies that $U_i \notin an_{\mathscr{A}}(U_0)$.

In order for U_0 to be a type 2 child of U_i, U_i cannot be the arterial parent of U_0, hence U_0 is not first in the parental order.

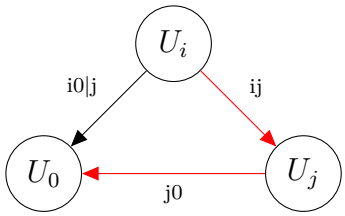
We will prove this lemma by induction on the position of U_i in the parental order of U_0.

- First assume U_i is the second in the parental order of U_0.

  This means that the copula specified to the arc $U_i \to U_0$ is of the form $C_{i0|j}$, which implies that $U_j \in pa_{\mathscr{A}}(U_0)$ and copula C_{j0} is specified.

  By proposition 8.0.1, either $C_{ij} \in \mathscr{C}$ or $C_{ji} \in \mathscr{C}$.

  If $C_{ij} \in \mathscr{C}$, then $U_i \to U_j$ and we have the following DAG structure:



  Therefore $U_i \in an_{\mathscr{A}}(U_0)$, which is not possible as U_0 is a type 2 child of U_i.

  This means that $C_{ji} \in \mathscr{C}$ which entails that $U_j \in pa_{\mathscr{A}}(U_i)$. Furthermore, given that U_j is the arterial parent of U_0, $U_j \in pa_{\mathscr{G}_{\mathscr{A}}}(U_0)$ with $U_j <_0 U_i$.

- Let the lemma hold for U_i being nth or earlier in the parental order of U_0.
- We will prove that this lemma holds for U_i which is $n+1$th element of the parental order of U_0.

  Due to $U_i \notin an_{\mathscr{A}}(U_0)$ we have a path between U_i and U_0 composed of only serial connections and a single diverging connection. All arcs are assigned with unconditional copulas, i.e.:

$$U_i \equiv U_{i_0} \leftarrow U_{i_1} \leftarrow \cdots \rightarrow U_{i_m} \rightarrow U_{i_{m+1}} \equiv U_0$$

  with a diverging connection being present in the trail.

  - We will show that $U_{i_1} \to U_0$.

  If that is not true we have a following loop without chords or converging connections:

$$U_0 \leftarrow U_i \leftarrow U_{i_1} \leftrightharpoons U_{i_2} \rightarrow U_{i_m} \ldots U_0$$

  If the edge $U_{i_1} \to U_0$ is not present in the DAG, the previous trail contains no chords and its length is greater or equal to five, making it an active circle. However we need a restricted DAG, hence $U_{i_1} \to U_0$.

  - We will now show that $U_{i_1} <_0 U_i$.

  Assume that $U_i <_0 U_{i_1}$. Then the copula $C_{i_1 0|K_{i_1 0}}$ contains U_i in its conditioning set $K_{i_1 0}$.

  In order to have a restricted PCBN we must be able to use the specified copulas to calculate the following conditional margin

$$u_{i_1|K_{i_1 0}}.$$

  We will show that this is not possible for $U_i <_0 U_{i_1}$.

Due to $U_{i_m} \in pa_{\mathscr{A}}(U_0)$, $U_{i_m} \in K_{i_1 0}$. This means that at least one of the nodes $U_{i_2}, \ldots, U_{i_m}$ is part of $K_{i_1 0}$. In the calculation of the conditional margin $u_{i_1|K_{i_1 0}}$ some of these nodes may be omitted from $K_{i_1 0}$ due to conditional independencies, but at least one such node will be conditionally dependent of U_1 given the rest of the nodes. Let these nodes be:

$$U_{\iota_1}, \ldots, U_{\iota_k}$$

Due to U_i being the arterial child of U_{i_1}, U_{i_1} is also dependent of U_1 conditional on the rest of the nodes in $K_{i_1 0}$. This means that neither U_i nor U_{ι_j} nodes can be omitted from the calculation of the conditional margin.

However, the arc $U_i \to U_{i_1}$ is assigned with the unconditional copula C_{ii_1}, meaning that the calculation of $u_{i_1|K_{i_1 0}}$ cannot be done. This violates the assumption of a restricted PCBN. Meaning that $U_{i_1} <_0 U_i$.

Let $U_m \in pa_{\mathscr{A}}(U_0)$ for which $U_m \neq U_{i_1}$ holds; U_m is also part of the same conditioning set $K_{i_1 0}$. Furthermore, U_{i_1} is dependent on at least one of the nodes $U_{i_2}, \ldots, U_{i_m}$ present in $K_{i_1 0}$ given U_i, as there exists a trail not blocked by U_i. The specified copula between U_{i_1} and U_i is $C_{i_1 i}$, and not conditioned on any other node. Hence it is impossible to calculate the conditional marginal using the specified copulas and thus U_{i_1} has to come before U_i in the parental order of U_0.

Therefore we have proven that the arterial parent $U_j \equiv U_{i_1}$ of U_i satisfies:

$$U_{i_1} \in pa_{\mathscr{G}_{\mathscr{A}}}(U_0) \; with \; U_{i_1} <_0 U_i$$

□

Let us illustrate graphically the lemma 8.0.3. In figure 8.3 we can see how the existence of arc $U_i \to U_0$ implies the existence of the arc $U_j \to U_0$ when U_j being the arterial parent of U_i. Furthermore, the lemma provides us with information on the relative positions of U_i and U_j in the parental order of U_0.

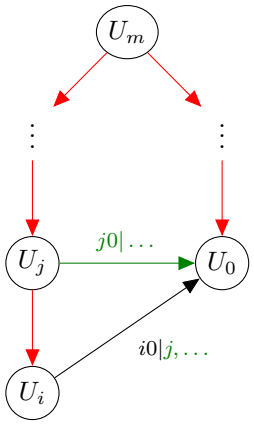


Figure 8.3: Visual representation of lemma 8.0.3. The green arc and specified copulas are the ones implied by the arc $U_i \to U_0$.

We may immediately extend lemma 8.0.3 to include all arterial ancestors of U_i that are arterial descendants of the arterial diverging node U_m. The result follows by recursive application of lemma 8.0.3 to each arterial parent until U_m is reached. Moreover, we can prove a stronger result which is the basis of our proposed methodology of parental order specification.

Theorem 8.0.4. *Let $\mathscr{G}_{\mathscr{A}}$ be an extended artery and $U_1 \in V_{\mathscr{A}}$, $U_2 \in ch_{\mathscr{G}_{\mathscr{A}}}(U_1)$. Let the unique arterial trail from U_1 to U_2 be:*

$$U_1 = U_{i_0} \leftrightharpoons U_{i_1} \rightleftharpoons \cdots \rightleftharpoons U_{i_{m-1}} \rightleftharpoons U_{i_m} = U_2$$

where all connections are either serial or diverging.

Then:

- $U_{i_k} \in pa_{\mathscr{G}_{\mathscr{A}}}(U_1)\ \forall k = 1, \ldots, m-1$ *and*
- $U_{i_k} <_1 U_{i_{k+1}}\ \forall k = 1, \ldots, m-2$.

Proof. Due to the trail being the unique arterial trail joining U_1 and U_2, and the fact that arteries are trees, we can deduce that the trail is composed of the most of one diverging connection and serial connections. This trail does not contain converging connections.

First we prove the theorem in the case when all connections are serial. Then the trail is of the form

$$U_1 \to U_{i_1} \to \cdots \to U_{i_{m-1}} \to U_2.$$

Note that the opposite direction is not possible as U_2 is a child of U_1.

Due to the trail being arterial, U_1 is an arterial ancestor of U_2, hence U_2 is a type 1 child of U_1. Therefore, due to lemma 8.0.2,

- $U_{i_k} \in pa_{\mathscr{G}_{\mathscr{A}}}(U_1)\ \forall k = 1, \ldots, m-1$ and
- $U_{i_k} <_1 U_{i_{k+1}}\ \forall k = 1, \ldots, m-1$.

Let us now consider the case where a single diverging connection is present in the trail. Then, the trail takes the following form, where U_{i_l} is the arterial diverging node of the trail.

$$U_1 \leftarrow U_{i_1} \cdots \leftarrow U_{i_{l-1}} \leftarrow U_{i_l} \to U_{i_{l+1}} \cdots \to U_{i_{m-1}} \to U_2$$

Because each arc in this trail is arterial, it clearly follows that $U_1 \notin an_{\mathscr{A}}(U_2)$, hence U_2 is a type 2 child of U_1. Now applying lemma 8.0.3, we deduce that $U_{i_1} \in pa_{\mathscr{G}_{\mathscr{A}}}(U_2)$ and $U_{i_1} <_2 U_1$.

If $U_{i_1} \neq U_{i_l}$, then U_2 is also a type 2 child of U_{i_1} and again by application of lemma 8.0.3 we get that $U_{i_2} \in pa_{\mathscr{G}_{\mathscr{A}}}(U_2)$ and $U_{i_2} <_2 U_{i_1} <_2 U_1$.

Recursive applications of lemma 8.0.3 leads to:

$$\{U_1, U_{i_1}, \ldots, U_{i_{l-1}}, U_{i_l}\} \subseteq pa_{\mathscr{G}_{\mathscr{A}}}(U_2), \tag{8.2}$$

$$U_{i_l} <_2 U_{i_{l-1}} <_2 \cdots <_2 U_{i_1} <_2 U_1. \tag{8.3}$$

If $U_{i_l} = U_{i_{m-1}}$ the theorem's statement has been shown. Otherwise, we notice that U_{i_l} being the only diverging node in the trail implies that $U_{i_l} \in an_{\mathscr{A}}(U_2)$. Therefore U_2 is a type 1 child of U_{i_l} and lemma 8.0.2 can be applied, proving that:

$$\{U_{i_{l+1}}, \dots, U_{i_{m-1}}\} \subseteq pa_{\mathscr{G}_{\mathscr{A}}}(U_2), \tag{8.4}$$

$$U_{i_{m-1}} <_2 U_{i_{l-1}} <_2 \cdots <_2 U_{i_{l+1}} <_2 U_{i_l}. \tag{8.5}$$

Using results 8.2 and 8.4 we arrive at the theorem's statement:

- $U_{i_k} \in pa_{\mathscr{G}_{\mathscr{A}}}(U_1)\ \forall k = 1, \dots, m-1$ and
- $U_{i_k} <_1 U_{i_{k+1}}\ \forall k = 1, \dots, m-1$.

□

Before discussing the impact of this theorem a graphical representation in a small example is shown. Let us consider the network in figure 8.4. The arterial arcs are in red. The blue arcs to U_9 from U_1 and U_8 imply the existence of all green arcs. The arc $U_2 \to U_9$ follows from the existence of $U_1 \to U_9$. The arcs from U_4 and U_6 to U_9 are consequences of the arc $U_8 \to U_9$. Lastly, all reminding arcs are implied by existence of both blue arcs. Additionally, the theorem provides us with the following orderings:

$$U_7 <_9 U_5 <_9 U_3 <_9 U_4 <_9 U_6 <_9 U_8$$

$$U_7 <_9 U_5 <_9 U_3 <_9 U_2 <_9 U_1$$

We deduce that the first three elements in the parental ordering of U_9 must be U_7, U_5 and U_3, implying that $K_{59} = \{U_7\}$ and $K_{39} = \{U_7, U_5\}$. Next in the parental order of U_9 could have either U_2 or U_4. Then the next candidates would be either U_1 and U_4 if we had chosen U_2 previously, or U_2 and U_6 if we had chosen U_4.

Theorem 8.0.4 provides with a fundamentally different approach to understanding the structure of restricted DAGs and available parental orderings. The approach of Horsman [36] relied on the definitions of B-sets. Theorem 8.0.4 uses instead on a "valid" artery choice and the trail structure of that artery. In this context, "valid" choice we refer to a choice of the fist element in the parental order of every node in the network, for which proper collection of parental orders $\mathscr{O}$ exists. If it does exist, then theorem 8.0.4 provides us with a fast way of calculating all possible parental orders in the network.

Studying methods of finding such "valid" artery choices in a given restricted DAG would be needed to facilitate the implementation of theorem 8.0.4 for parental order calculation. This research topic remains open and is left as future research.

Corollary 8.0.5. *Let $\mathscr{G}_{\mathscr{A}}$ be an extended artery and $U_1, U_2 \in V_{\mathscr{A}}$ such that U_2 be a type 2 child of U_1 with specified copula $C_{12|K_{12}}$. Assume that U_i is the arterial diverging node in the trail between U_1 and U_2. Then:*

$$U_j \in K_{12}\ \forall U_j \in (an_{\mathscr{A}}(U_1) \cap de_{\mathscr{A}}(U_i)) \cup \{U_i\}$$

Proof. The corollary follows immediately from theorem 8.0.4 upon the observation that all of the nodes in $(an_{\mathscr{A}}(U_1) \cap de_{\mathscr{A}}(U_i)) \cup \{U_i\}$ are part of the arterial trail between U_1 and U_2. □

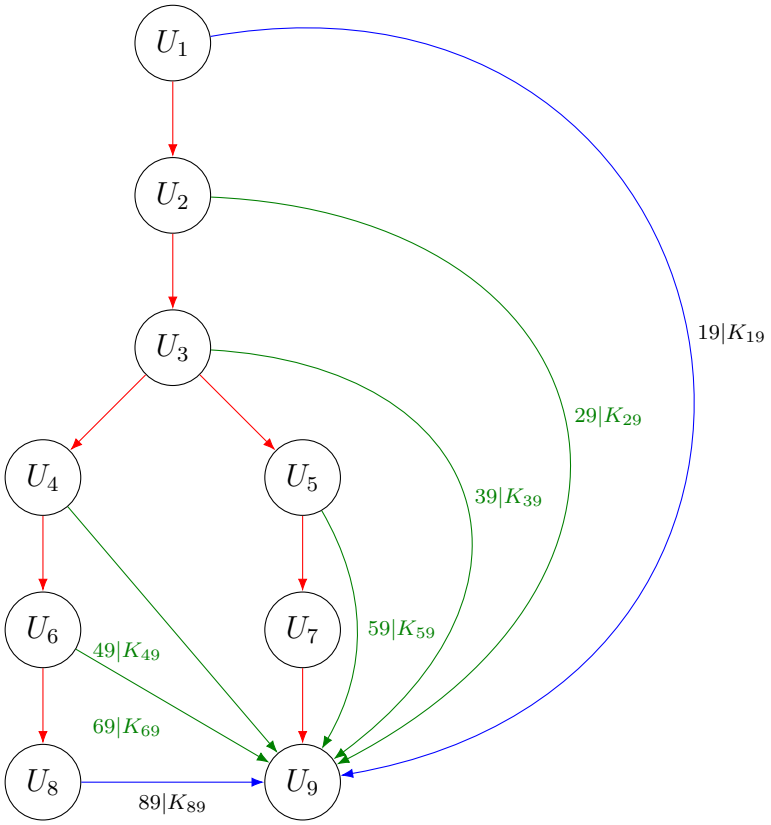


Figure 8.4: Illustration of theorem 8.0.4.

This corollary is central to motivate exclusion of type 2 children from Backward Sampling. Each of the nodes in $(an_{\mathscr{A}}(U_1) \cap de_{\mathscr{A}}(U_i)) \cup \{U_i\}$ is part of the conditional set of copula $C_{12|K_{12}}$. For U_2's inclusion in Backward Sampling of U_1, the Backward Samples of nodes in $(an_{\mathscr{A}}(U_1) \cap de_{\mathscr{A}}(U_i)) \cup \{U_i\}$ would be required, which do not abide by the order of Backward Sampling.

9

Conclusions and Future Research

Our work draws to a close and we summarize the key findings, their implications, and the contributions to the field. This thesis includes an exhaustive study of inference problems in Pair Copula Bayesian Networks, a pivotal topic in the evolution of Bayesian Network Inference theory and dependence modeling. Building upon fundamental to the field concepts such as Variable Elimination and Belief Propagation, we expand the mathematical scope of Bayesian Network inference approaches. By introducing several open research questions, we additionally pave the way for future exploration in this domain.

The goals of this thesis were to construct a robust mathematical foundation for PCBN inference and to develop algorithms that perform inference on PCBN sub-structures called arteries. Additionally, we aimed to offer comprehensive insights on simplification of inference problems through pruning, as well as a possible methodology for conducting inference in general PCBN models.

In chapter 4 we introduced the concepts of arteries and constructed the Sample Propagation methodology for single-arterial PCBN inference. Through sequential calculation of Forward, Backward, Secondary Backward and Bilateral Samples, the information about evidence nodes can be propagated to the query nodes to compute their requested conditional distributions. The outlined methodology constitutes a sample-based extension of Pearl's Belief Propagation algorithm, implemented in PCBNs with the incorporation of Variable Elimination theory.

A core characteristic of our proposal is the improved efficiency through the ability to exclude nodes that are not required in the Sample Propagation processes, both through exclusion of UBI and USBI nodes and through the proposed PCBN pruning, explored in chapter 5.

Pruning led to the simplification of the inference problem through the removal of nodes and arcs from the network, without disturbing the conditional distributions of query variables given evidence variables. In comparison to root pruning in the general case of Bayesian Networks, our pruning algorithm exploits key features of PCBNs to remove more nodes and obtain a smaller network, subsequently leading to faster and easier

implementations of Sample Propagation.

The core attributes of our single-arterial Sample Propagation methodology include:

- Granularity: Due to the sequential aspect of Belief/Sample Propagation our algorithm's scope has a high level of detail, finely breaking down propagation into sampling of individual nodes. This also means that the algorithm is easily scalable in parallel environments, boosting its efficiency
- Scalability: Regardless of the number of query variables, the algorithm only performs sample propagation once, making it more efficient for finding multiple conditional distributions given the same evidence.
- Minimal number of integrations: Compared to Pearl's Algorithm, our methodology avoids computation of messages that are unnecessary to obtain the final result.
- Ability to have distributional evidence: Instead of constant values, evidence can take the form of probability distributions that pose a shift from the marginal distribution in the PCBN. Sample Propagation can easily incorporate this type of evidence by using samples from the evidence distribution.

We conducted a thorough test of our methodology by performing a simulation study. We assessed the effectiveness and efficiency of our sampling algorithms.This simulation study is discussed in detail in chapter 6. The results were very positive, illustrating that for relatively low expected execution times, the algorithms manage to sample effectively from the theoretical distribution.

We conclude that in systems similar to ours, this algorithm can be expected to handle inference problems on 50 node PCBNs in less than four minutes, under the most demanding tested simulation settings. With some sensible concessions to those settings, the time efficiency can be increased by a factor of four, with negligible losses in accuracy.

In chapter 7 we present a generalization of the single-arterial Sample Propagation for its application in multiple-artery PCBNs. While not conclusive, this work examines all core aspects necessary for the future study of multi-arterial PCBN inference through Sample Propagation. The scope of the chapter extends for inference problems in PCBNs whose inter-arterial connectivity can be represented by a polytree. When this condition is not met, certain theoretical problems arise which are left open for future research.

Lastly, this thesis includes a supplementary chapter on trail properties of arterial structures, culminating in the proof of Theorem 8.0.4. This theorem ensures the existence of certain arcs in restricted DAGs, as well as rules that the parental orders must abide by. This Theorem plays a central role to the motivation behind the distinction between type 1 and type 2 children in chapter 4, but it also extends to a comprehensive and intuitive approach for finding the possible parental orders in a PCBN, given an arterial structure.

Let us now include several open research questions introduced by our work which hold particular significance for the field. These questions are listed for further exploration, marking potential directions for future research.

1. Further testing of arterial propagation.

This thesis provided a detailed testing phase in which the performance of sampling algorithms was compared for different sampling parameters. However, the timeframe of this thesis limited the testing that could be done to this algorithm. More specifically, the following are topics which could prove interesting for future study:

- Finding and proving convergence rates of sampling algorithms based on Sample Size and Importance Sample Size with incorporation of parameters such as number of children-parents and copula parameter values.
- Additional research on the choice of the Importance Sample in Secondary Backward and Bilateral Sampling will also be essential for efficient implementations. Despite the Forward Sample choice performing best on average, for specific DAG structures or copula parameters/evidence this might not be true. A heuristic could be instead considered for such cases.

2. Testing the methodology for multiple arteries.

   Despite having proposed an approach for multiple arterial PCBNs in chapter 7, the methodology remains untested. Some additional Parameters that may be relevant when assessing performance in multiple arteries could be

   - Number of arterial links between arteries.
   - Number of arteries in the PCBN
   - Number of arteries that contain evidence in the PCBN.

3. Investigating ways to test non Gaussian dependency structures.

   To compare the results of our methodology with the true distribution, knowing the latter is essential. Gaussian Bayesian Networks are the only continuous BN family for which an analytical solution to inference problem is known.

   Therefore, Gaussian copulas posed a clear restriction that we needed to abide by for the simulation study of this thesis. Gaussianity has several properties such as symmetry and less extreme tail dependencies which might influence positively the effectiveness of our algorithms and testing them using different copulas can provide deeper insights.

   Hence, a deeper understanding of how to compare these more complex dependency structures is required to assess the performance in such cases.

4. Investigating the use of distributional evidence.

   We have mentioned previously that the choice of a sample-based approach allows non-deterministic evidence to be part of the inference problem. This can be done by sampling from the evidence distribution and using those samples as Forward, Backward Secondary Backward and Bilateral Samples to be used in the Sample Propagation procedure.

   During our testing only constant evidences were considered and the research on the effect of stochastic evidence remains open for future investigation.

5. Researching the generalization of Multi-Arterial Sample Propagation in arterial structures that are not polytrees.

There are clear issues that arise under the presence of active loops in the arterial connections. Loops are easily dealt with in the scope of arteries, however the same problems present in Pearl´s Algorithm are relevant when the active loops are formed using different arteries.

6. Incorporation of Theorem 8.0.4 in parental ordering finding. The current methodology calculates the full parental order of each node in the PCBN before moving on to other nodes. In order to use Theorem 8.0.4 however, it is required to have an arterial structure meaning that we need to have the first parent of each parental order for all nodes.

   The problem of finding the different arterial structures deviates significantly from the scope of the algorithm proposed in Horsman [36]. If an efficient solution is found for this problem however, application of Theorem 8.0.4 could prove a more effective way to provide all possible parental orderings in the DAG.

7. Implementation of Sample Propagation algorithms.

   Integration of our proposed algorithms in PCBN inference in programming packages would simplify the implementation process, encourage wider adoption, and contribute to advancements in Bayesian Network inference across various fields of research.

References

[1] Michelle Baker and Terrance Boult. "Pruning Bayesian networks for efficient computation". In: Jan. 1990, pp. 225–232.

[2] D Barber. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012. ISBN: 9780521518147.

[3] Alexander Bauer and Claudia Czado. "Pair-Copula Bayesian Networks". In: *Journal of Computational and Graphical Statistics* 25.4 (Oct. 2016), pp. 1248–1271. ISSN: 1061-8600, 1537-2715. DOI: 10.1080/10618600.2015.1086355. URL: https://www.tandfonline.com/doi/full/10.1080/10618600.2015.1086355 (visited on 12/26/2023).

[4] Alexander Bauer, Claudia Czado, and Thomas Klein. "Pair-copula constructions for non-Gaussian DAG models". In: *Canadian Journal of Statistics* 40.1 (Mar. 2012), pp. 86–109. ISSN: 0319-5724, 1708-945X. DOI: 10.1002/cjs.10131. URL: https://onlinelibrary.wiley.com/doi/10.1002/cjs.10131 (visited on 01/23/2024).

[5] Tim Bedford and Roger M Cooke. "Probability density decomposition for conditionally dependent random variables modeled by vines". In: *Annals of Mathematics and Artificial Intelligence* 32 (2001), pp. 245–268.

[6] Tim Bedford and Roger M Cooke. "Vines—A new graphical model for dependent random variables". In: *The Annals of Statistics* 30.4 (2002), pp. 1031–1068.

[7] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. 1st ed. 2006. Corr. 2nd printing. Information science and statistics. Springer, 2006. ISBN: 9780387310732.

[8] Béla Bollobás. *Modern Graph Theory*. 1st ed. Graduate Texts in Mathematics 184. Springer-Verlag New York, 1998. ISBN: 978-1-4612-0619-4.

[9] John Adrian Bondy, Uppaluri Siva Ramachandra Murty, et al. *Graph theory with applications*. Vol. 290. Macmillan London, 1976.

[10] Susanne Gammelgaard Bøttcher. "Learning conditional Gaussian networks". In: *Aalborg Universitetsforlag* (2005).

[11] Jian Cheng and Marek J Druzdzel. "AIS-BN: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks". In: *Journal of Artificial Intelligence Research* 13 (2000), pp. 155–188.

[12] Umberto Cherubini et al. *Dynamic Copula Methods in Finance*. 1st ed. Wiley, Nov. 2011. ISBN: 978-1-118-46740-4. DOI: 10.1002/9781118467404. URL: https://onlinelibrary.wiley.com/doi/book/10.1002/9781118467404 (visited on 12/01/2023).

[13] A Christofides et al. "The optimal discretization of probability density functions". In: *Computational statistics & data analysis* 31.4 (1999), pp. 475–486.

[14] Gregory F. Cooper. "The computational complexity of probabilistic inference using bayesian belief networks". In: *Artificial Intelligence* 42.2 (Mar. 1990), pp. 393–405. ISSN: 00043702. DOI: 10.1016/0004-3702(90)90060-D. URL: https://linkinghub.elsevier.com/retrieve/pii/000437029090060D (visited on 01/30/2024).

[15] Robert G Cowell et al. *Probabilistic networks and expert systems: Exact computational methods for Bayesian networks*. Springer Science & Business Media, 2007.

[16] Claudia Czado. *Analyzing Dependent Data with Vine Copulas: A Practical Guide With R*. Vol. 222. Lecture Notes in Statistics. Cham: Springer International Publishing, 2019. ISBN: 978-3-030-13785-4. DOI: 10.1007/978-3-030-13785-4. URL: http://link.springer.com/10.1007/978-3-030-13785-4 (visited on 12/01/2023).

[17] Paul Dagum and Michael Luby. "Approximating probabilistic inference in Bayesian belief networks is NP-hard". In: *Artificial Intelligence* 60.1 (Mar. 1993), pp. 141–153. ISSN: 00043702. DOI: 10.1016/0004-3702(93)90036-B. URL: https://linkinghub.elsevier.com/retrieve/pii/000437029390036B (visited on 01/30/2024).

[18] Nir Friedman Daphne Koller. *Probabilistic Graphical Models: Principles and Techniques*. 1st ed. Adaptive Computation and Machine Learning series. The MIT Press, 2009. ISBN: 0262013193.

[19] CJ van Dongen et al. "Bonaparte: Application of new software for missing persons program". In: *Forensic Science International: Genetics Supplement Series* 3.1 (2011), e119–e120.

[20] James Dougherty, Ron Kohavi, and Mehran Sahami. "Supervised and unsupervised discretization of continuous features". In: *Machine learning proceedings 1995*. Elsevier, 1995, pp. 194–202.

[21] Denise L Draper and Steve Hanks. "Localized partial evaluation of belief networks". In: *Uncertainty Proceedings 1994*. Elsevier, 1994, pp. 170–177.

[22] Usama M Fayyad and Keki B Irani. "Multi-interval discretization of continuous-valued attributes for classification learning". In: *Ijcai*. Vol. 93. 2. Citeseer. 1993, pp. 1022–1029.

[23] G David Forney. "The viterbi algorithm". In: *Proceedings of the IEEE* 61.3 (1973), pp. 268–278.

[24] Nir Friedman, Moises Goldszmidt, et al. "Discretizing continuous attributes while learning Bayesian networks". In: *ICML*. 1996, pp. 157–165.

[25] Robert Fung and Kuo-Chu Chang. "Weighing and integrating evidence for stochastic simulation in Bayesian networks". In: *Machine intelligence and pattern recognition*. Vol. 10. Elsevier, 1990, pp. 209–219.

[26] Robert Fung and Brendan Del Favero. "Backward simulation in Bayesian networks". In: *Uncertainty Proceedings 1994*. Elsevier, 1994, pp. 227–234.

[27] A. Gelman et al. *Bayesian Data Analysis*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press, 2013. ISBN: 9781439898208.

[28] Corrado Gini. "The measurement of the differences between two quantity groups and in particular between the characteristics of two populations". In: *Acta Genetica et Statistica Medica* (1953), pp. 175–191.

[29] P. Glasserman. *Monte Carlo Methods in Financial Engineering*. Applications of mathematics : stochastic modelling and applied probability. Springer, 2004. ISBN: 9780387004518. URL: https://books.google.nl/books?id=e9GWUsQkPNMC.

[30] Anca Hanea and Dorota Kurowicka. "Mixed non-parametric continuous and discrete Bayesian belief nets". In: *Advances in mathematical modeling for reliability* 1 (2008), pp. 9–16.

[31] Anca M Hanea, Dorota Kurowicka, and Roger M Cooke. "Hybrid method for quantifying and analyzing Bayesian belief nets". In: *Quality and Reliability Engineering International* 22.6 (2006), pp. 709–729.

[32] Anca M Hanea et al. "Mining and visualising ordinal data with non-parametric continuous BBNs". In: *Computational Statistics & Data Analysis* 54.3 (2010), pp. 668–687.

[33] David Heckerman and Dan Geiger. "Learning Bayesian networks: a unification for discrete and Gaussian domains". In: *arXiv preprint arXiv:1302.4957* (2013).

[34] Max Henrion. "Propagating uncertainty in Bayesian networks by probabilistic logic sampling". In: *Machine intelligence and pattern recognition*. Vol. 5. Elsevier, 1988, pp. 149–163.

[35] Dawn E. Holmes and Lakhmi C. Jain, eds. *Innovations in Bayesian Networks*. Vol. 156. Studies in Computational Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. ISBN: 978-3-540-85066-3. DOI: 10.1007/978-3-540-85066-3. URL: http://link.springer.com/10.1007/978-3-540-85066-3 (visited on 12/01/2023).

[36] Niels Horsman. *On the restrictions of Pair-Copula Bayesian Networks for integration-free computations*. MSc Thesis. 2023.

[37] E. Horvitz, H.J. Suermondt, and G.F. Cooper. "Bounded Conditioning: Flexible Inference for Decisions Under Scarce Resources". In: *Proceedings of Conference on Uncertainty in Artificial Intelligence*. Association for Uncertainty in Artificial Intelligence. Mountain View, CA: Association for Uncertainty in Artificial Intelligence, Aug. 1989, pp. 182–193.

[38] Henry F Inman and Edwin L Bradley Jr. "The overlapping coefficient as a measure of agreement between probability distributions and point estimation of the overlap of two normal densities". In: *Communications in Statistics-theory and Methods* 18.10 (1989), pp. 3851–3874.

[39] Finn V Jensen and Thomas Dyhre Nielsen. *Bayesian networks and decision graphs*. Vol. 2. Springer, 2007. ISBN: 9780387682815.

[40] Harry Joe. *Dependence modeling with copulas*. Monographs on statistics and applied probability 134. Boca Raton, Fla.: CRC Press, 2015. 462 pp. ISBN: 978-1-4665-8322-1.

[41] 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 Ludger Rüschendorf, Berthold Schweizer, and Michael D. Taylor. Lecture Notes—Monograph Series 28. Hayward, California: Institute of Mathematical Statistics, 1996, pp. 120–141.

[42] Harry Joe. *Multivariate Models and Dependence Concepts*. Monographs on Statistics and Applied Probability 73. Springer US, 1997. ISBN: 9780412073311.

[43] Willian Darwin Júnior. *Modeling copulas with Bayesian networks*. Doutorado em Sistemas Dinâmicos. São Carlos, Feb. 23, 2021. DOI: 10.11606/T.18.2021.tde-23032021-200921. URL: https://www.teses.usp.br/teses/disponiveis/18/18153/tde-23032021-200921/ (visited on 12/01/2023).

[44] Daphne Koller, Uri Lerner, and Dragomir Anguelov. "A General Algorithm for Approximate Inference and Its Application to Hybrid Bayes Nets". In: *Conference on Uncertainty in Artificial Intelligence*. 1999. URL: https://api.semanticscholar.org/CorpusID:7824624.

[45] Alexander V Kozlov and Daphne Koller. "Nonuniform dynamic discretization in hybrid networks". In: *arXiv preprint arXiv:1302.1555* (2013).

[46] Dorota Kurowicka and Roger Cooke. "Distribution-Free Continuous Bayesian Belief Nets". In: *Modern Statistical and Mathematical Methods in Reliability*. World Scientific, 2005, pp. 309–322. DOI: 10.1142/9789812703378_0022. URL: https://www.worldscientific.com/doi/abs/10.1142/9789812703378_0022.

[47] Dorota Kurowicka and Roger Cooke. *Uncertainty analysis with high dimensional dependence modelling*. Wiley series in probability and statistics. Chichester: Wiley, 2006. 284 pp. ISBN: 978-0-470-86306-0.

[48] Dorota Kurowicka and Harry Joe, eds. *Dependence modeling: vine copula handbook*. Singapore: World Scientific, 2011. 360 pp. ISBN: 978-981-4299-87-9.

[49] Helge Langseth et al. "Inference in hybrid Bayesian networks". In: *Reliability Engineering & System Safety* 94.10 (2009), pp. 1499–1509.

[50] Steffen L Lauritzen and David J Spiegelhalter. "Local computations with probabilities on graphical structures and their application to expert systems". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 50.2 (1988), pp. 157–194.

[51] Michael J. McGeachie, Hsun-Hsien Chang, and Scott T. Weiss. "CGBayesNets: Conditional Gaussian Bayesian Network Learning and Inference with Mixed Discrete and Continuous Data". In: *PLoS Computational Biology* 10.6 (June 2014). Ed. by Robert F. Murphy, e1003676. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1003676. URL: https://dx.plos.org/10.1371/journal.pcbi.1003676 (visited on 12/02/2023).

[52] Serafín Moral, Rafael Rumi, and Antonio Salmerón. "Mixtures of Truncated Exponentials in Hybrid Bayesian Networks". In: *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*. Vol. 2143. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 156–167. ISBN: 978-3-540-44652-1. DOI: 10.1007/3-540-44652-4_15. URL: http://link.springer.com/10.1007/3-540-44652-4_15 (visited on 12/01/2023).

[53] K. Murphy, Y. Weiss, and M. Jordan. "Loopy Belief Propagation for Approximate Inference: An Empirical Study". In: *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence*. UAI-99. San Francisco, CA: Morgan Kaufmann Publishers, 1999, pp. 467–475.

[54] Richard E. Neapolitan. *Learning Bayesian Networks*. illustrated edition. Prentice Hall series in artificial intelligence. Pearson Prentice Hall, 2019. ISBN: 9780130125347.

[55] Michael J Pazzani. "An iterative improvement approach for the discretization of numeric attributes in Bayesian classifiers." In: *KDD*. Vol. 95. 1995, pp. 228–233.

[56] J. Pearl. "Fusion, Propagation, and Structuring in Belief Networks". In: *Artificial Intelligence* 29 (1986), pp. 241–288. DOI: 10.1016/0004-3702(86)90072-X.

[57] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan kaufmann, 1988.

[58] Darío Ramos-López et al. "MAP inference in dynamic hybrid Bayesian networks". In: *Progress in Artificial Intelligence* 6.2 (June 2017), pp. 133–144. ISSN: 2192-6352, 2192-6360. DOI: 10.1007/s13748-017-0115-7. URL: http://link.springer.com/10.1007/s13748-017-0115-7 (visited on 12/01/2023).

[59] Darío Ramos-López et al. "Scalable importance sampling estimation of Gaussian mixture posteriors in Bayesian networks". In: *International Journal of Approximate Reasoning* 100 (Sept. 2018), pp. 115–134. ISSN: 0888613X. DOI: 10.1016/j.ijar.2018.06.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S0888613X18300276 (visited on 12/01/2023).

[60] Christian P Robert, George Casella, and George Casella. *Introducing monte carlo methods with r*. Vol. 18. Springer, 2010. ISBN: 9781441915764.

[61] Christian P Robert, George Casella, and George Casella. *Monte Carlo statistical methods*. Vol. 2. Springer, 1999. ISBN: 9780387212395.

[62] Ross D Shachter and Mark A Peot. "Simulation approaches to general probabilistic inference on belief networks". In: *Machine intelligence and pattern recognition*. Vol. 10. Elsevier, 1990, pp. 221–231.

[63] Ross D. Shachter and M. Peat. "Simulation approaches to probabilistic inference for general probabilistic inference on belief networks". In: *Fifth Workshop on Uncertainty in Artificial Intelligence*. Detroit, Michigan, 1989.

[64] JH Sigurdsson, LA Walls, and JL Quigley. "Bayesian belief nets for managing expert judgement and modelling reliability". In: *Quality and reliability engineering international* 17.3 (2001), pp. 181–190.

[65] Wei Sun and K. C. Chang. "Convergence study of message passing in arbitrary continuous Bayesian networks". In: *Signal Processing, Sensor Fusion, and Target Recognition XVII*. Ed. by Ivan Kadar. Vol. 6968. Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series. Apr. 2008, 69680Z, 69680Z. DOI: 10.1117/12.780626.

[66] Bjørnar Tessem. *Extending the A/R algorithm for interval probability propagation*. Tech. rep. 42. University of Bergen, Dec. 1989.

[67] Bjørnar Tessem. “Interval probability propagation”. In: *International Journal of Approximate Reasoning* 7 (1992), pp. 95–120.

[68] John W Tilton. “The measurement of overlapping.” In: *Journal of Educational Psychology* 28.9 (1937), p. 656.

[69] Douglas B. West. *Introduction to Graph Theory*. 2nd ed. Prentice Hall, Sept. 2000. ISBN: 0130144002.

[70] Alyson Wilson et al. *Modern Statistical and Mathematical Methods in Reliability*. Vol. 10. Series on Quality, Reliability and Engineering Statistics. WORLD SCIENTIFIC, Oct. 2005. ISBN: 978-981-256-356-9. DOI: 10.1142/5844. URL: http://www.worldscientific.com/worldscibooks/10.1142/5844 (visited on 01/05/2024).

[71] Changhe Yuan and Marek J. Druzdzel. “Importance sampling algorithms for Bayesian networks: Principles and performance”. In: *Mathematical and Computer Modelling* 43.9 (May 2006), pp. 1189–1207. ISSN: 08957177. DOI: 10.1016/j.mcm.2005.05.020. URL: https://linkinghub.elsevier.com/retrieve/pii/S0895717705005443 (visited on 12/01/2023).

[72] N. L. Zhang and D. Poole. “Exploiting Causal Independence in Bayesian Network Inference”. In: *Journal of Artificial Intelligence Research* 5 (Dec. 1996), pp. 301–328. ISSN: 1076-9757. DOI: 10.1613/jair.305. URL: https://jair.org/index.php/jair/article/view/10178 (visited on 12/01/2023).

[73] N.L. Zhang and D. Poole. “A simple approach to Bayesian network computations”. In: *Proc. of the Tenth Canadian Conference on Artificial Intelligence*. 1994, pp. 171–178.

[74] Kailun Zhu, Dorota Kurowicka, and Gabriela F. Nane. “Simplified R-vine based forward regression”. In: *Computational Statistics & Data Analysis* 155 (Mar. 2021), p. 107091. ISSN: 01679473. DOI: 10.1016/j.csda.2020.107091. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167947320301821 (visited on 12/01/2023).

A

Effectiveness Results

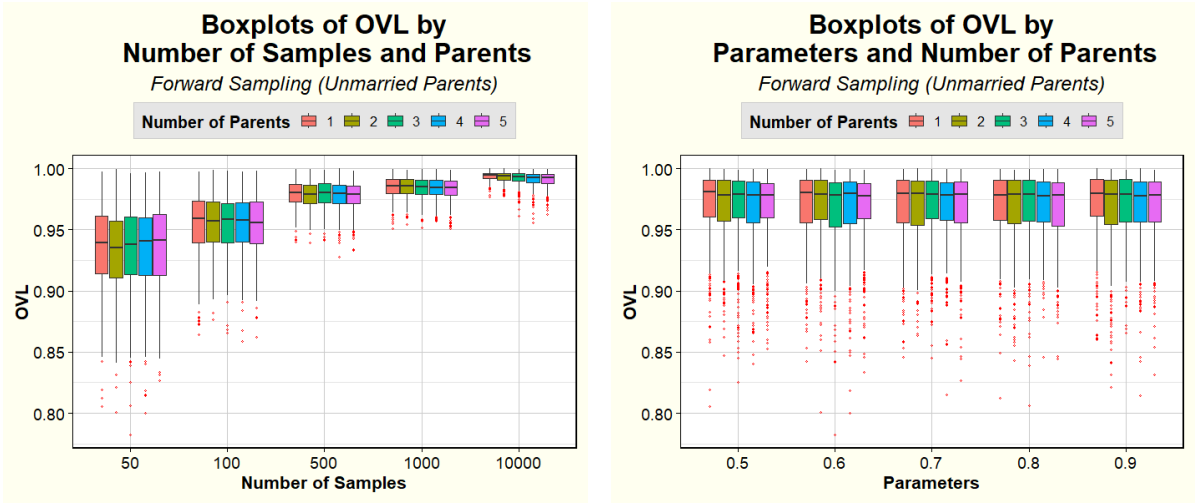
In this appendix we summarize the effectiveness results of the simulation study of chapter 6, focusing on the distribution of the Overlapping Coefficient (OVL) values and how it is affected by other parameters such as number of samples or Gaussian Parameter values.

We will present the figures along with the respective sample averages, for each sampling type.

A.1. Forward Sampling

Unmarried Parents

In this segment, the results of Forward Sampling are collected, for the DAG structure exhibiting parents that do not contain arcs between them. This DAG structure is illustrated in figure 6.3a.



(a) (b)

Figure A.1: Forward Sampling OVL plots (Unmarried Parents).

Sample Size	OVL	95% CI
50	0.936	(0.935, 0.937)
100	0.954	(0.953, 0.955)
500	0.978	(0.978, 0.979)
1000	0.984	(0.984, 0.985)
10000	0.992	(0.992, 0.992)

Table A.1: Table of average OVL by Sample Size in Forward Sampling (Unmarried Parents).

Param.	OVL	95% CI
0.5	0.970	(0.968, 0.971)
0.6	0.970	(0.969, 0.971)
0.7	0.968	(0.967, 0.970)
0.8	0.969	(0.968, 0.970)
0.9	0.968	(0.967, 0.969)

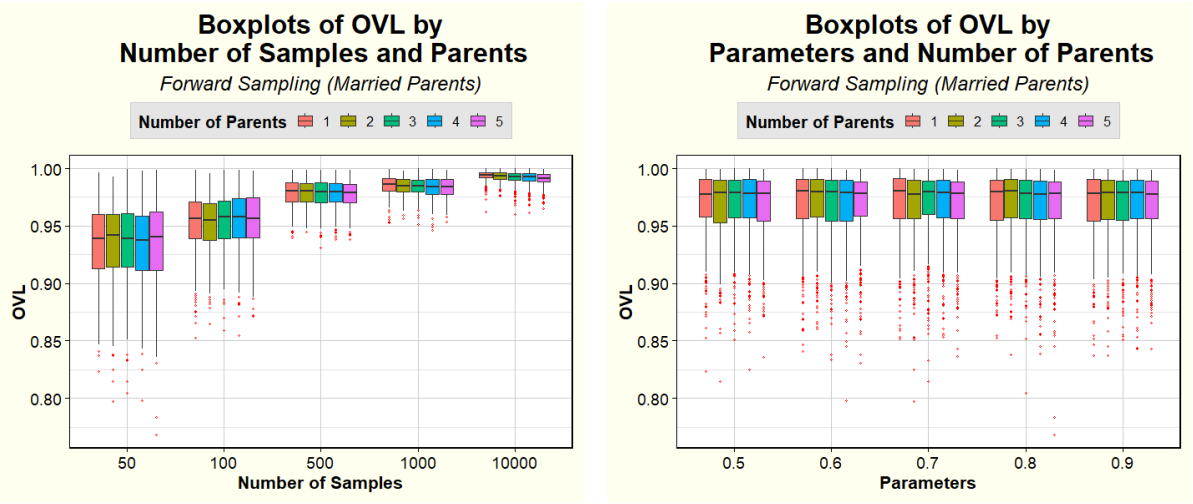
Table A.2: Table of average OVL by Parameters in Forward Sampling (Unmarried Parents).

Parents	OVL	95% CI
1	0.970	(0.969, 0.971)
2	0.969	(0.968, 0.970)
3	0.969	(0.968, 0.970)
4	0.968	(0.967, 0.969)
5	0.969	(0.968, 0.970)

Table A.3: Table of average OVL by number of Parents in Forward Sampling (Unmarried Parents).

Married Parents

Here we collect the result for the second DAG structure in Forward Sampling, in which parents are connected by arcs. This DAG structure is illustrated in figure 6.3b.



(a) **(b)**

Figure A.2: Forward Sampling OVL plots (Married Parents).

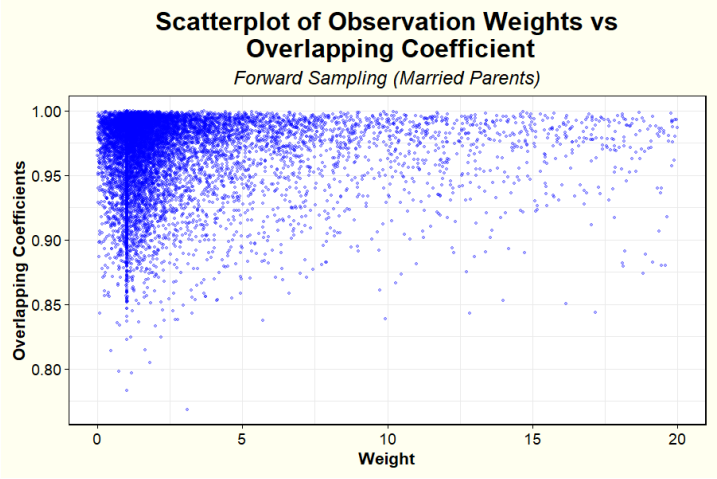


Figure A.3

Sample Size	OVL	95% CI
50	0.936	(0.934, 0.937)
100	0.953	(0.952, 0.954)
500	0.979	(0.978, 0.979)
1000	0.984	(0.984, 0.984)
10000	0.992	(0.992, 0.993)

Table A.4: Table of average OVL by Sample Size in Forward Sampling (Married Parents).

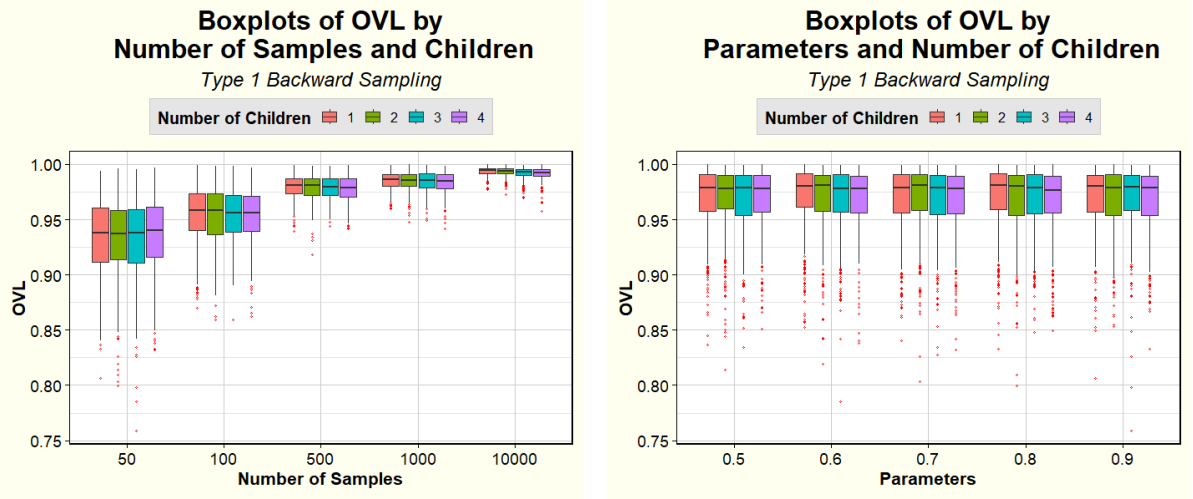
Param.	OVL	95% CI
0.5	0.969	(0.968, 0.970)
0.6	0.969	(0.968, 0.970)
0.7	0.969	(0.967, 0.970)
0.8	0.969	(0.968, 0.971)
0.9	0.968	(0.967, 0.969)

Table A.5: Table of average OVL by Parameters in Forward Sampling (Married Parents).

Parents	OVL	95% CI
1	0.969	(0.968, 0.970)
2	0.969	(0.968, 0.970)
3	0.969	(0.968, 0.970)
4	0.969	(0.967, 0.970)
5	0.968	(0.967, 0.969)

Table A.6: Table of average OVL by number of Parents in Forward Sampling (Married Parents).

A.2. Type 1 Backward Sampling



(a) (b)

Figure A.4: Type 1 Backaward Sampling OVL plots.

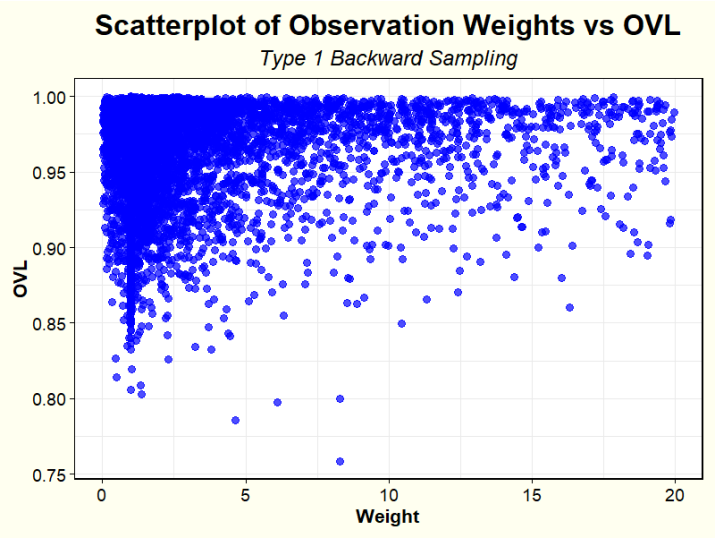


Figure A.5

Sample Size	OVL	95% CI
50	0.934	(0.933, 0.936)
100	0.953	(0.952, 0.954)
500	0.979	(0.978, 0.979)
1000	0.985	(0.984, 0.985)
10000	0.993	(0.992, 0.993)

Table A.7: Table of average OVL by Sample Size in Type 1 Backward Sampling.

Param.	OVL	95% CI
0.5	0.968	(0.967, 0.970)
0.6	0.970	(0.968, 0.971)
0.7	0.968	(0.967, 0.969)
0.8	0.969	(0.968, 0.970)
0.9	0.968	(0.967, 0.970)

Table A.8: Table of average OVL by Parameters in Type 1 Backward Sampling.

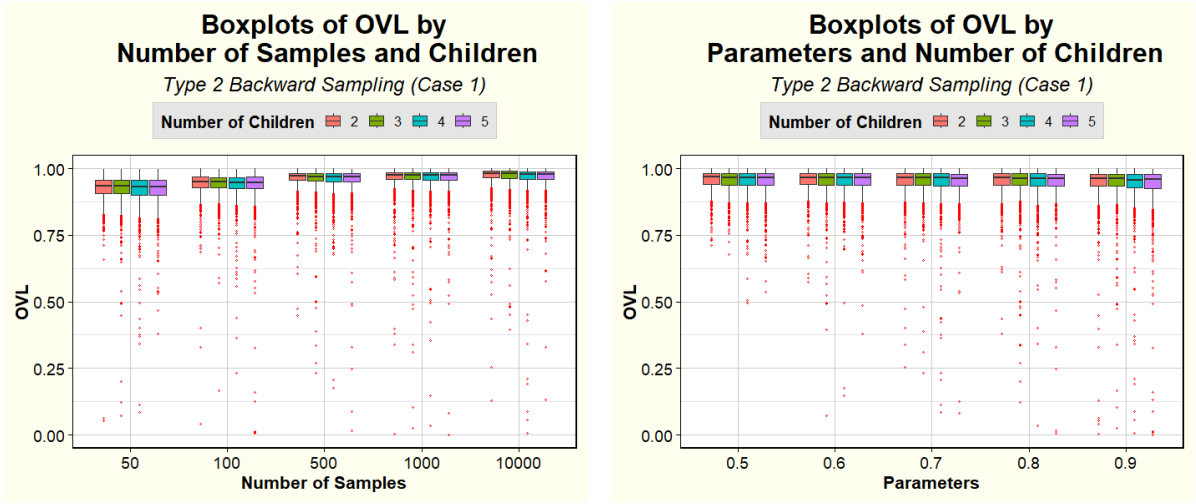
Children	OVL	95% CI
1	0.969	(0.968, 0.970)
2	0.969	(0.967, 0.970)
3	0.969	(0.968, 0.970)
4	0.968	(0.967, 0.969)

Table A.9: Table of average OVL by number of children in Type 1 Backward Sampling.

A.3. Type 2 Backward Sampling

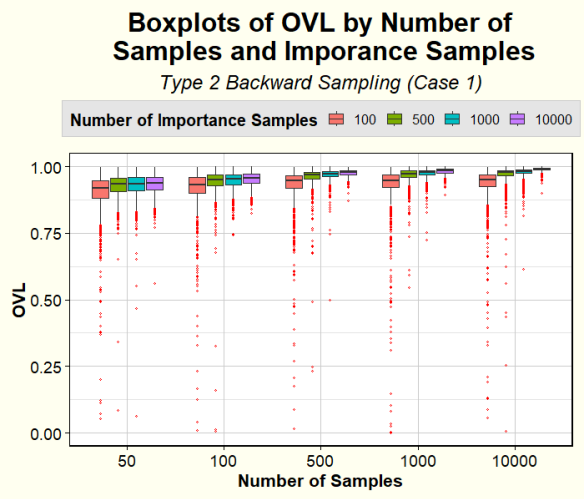
Uncoupled Children

In this segment, the results of Type 2 Backward Sampling are collected, for the DAG structure exhibiting children that do not contain arcs between them. This DAG structure is illustrated in figure 6.11.

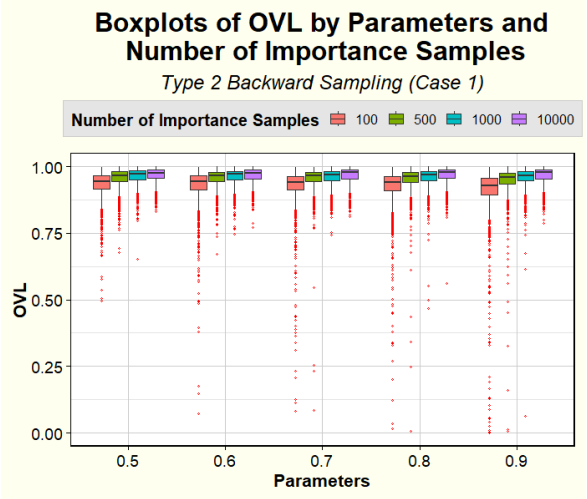


(a) (b)

Figure A.6: Type 2 Backward Sampling OVL plots by number of children (Uncoupled Children).



(a)



(b)

Figure A.7: Type 2 Backward Sampling OVL plots by number of importance samples (Uncoupled Children).

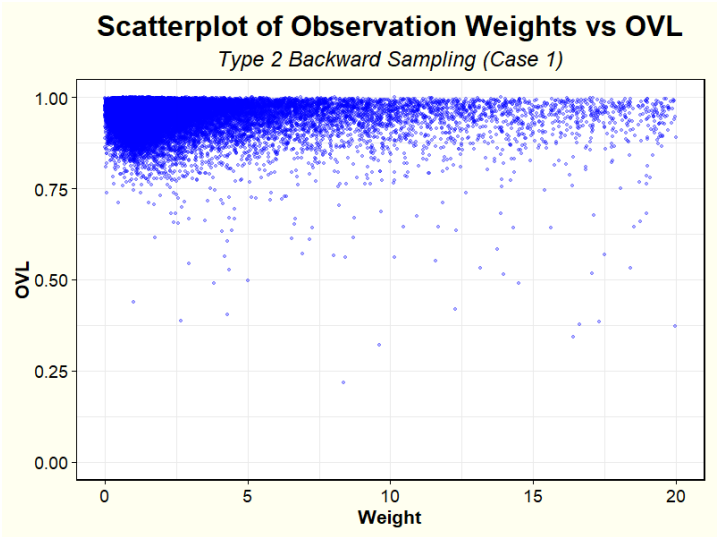


Figure A.8

Sample Size	OVL	95% CI
50	0.925	(0.924, 0.926)
100	0.942	(0.941, 0.943)
500	0.961	(0.960, 0.962)
1000	0.965	(0.964, 0.966)
10000	0.969	(0.969, 0.970)

Table A.10: Table of average OVL by Sample Size in Type 2 Backward Sampling (Uncoupled Children).

Param.	OVL	95% CI
0.5	0.956	(0.955, 0.957)
0.6	0.955	(0.954, 0.956)
0.7	0.954	(0.953, 0.955)
0.8	0.952	(0.951, 0.953)
0.9	0.946	(0.944, 0.947)

Table A.11: Table of average OVL by Parameters in Type 2 Backward Sampling (Uncoupled Children).

Children	OVL	95% CI
1	0.954	(0.953, 0.955)
2	0.954	(0.953, 0.955)
3	0.951	(0.950, 0.953)
4	0.951	(0.950, 0.952)

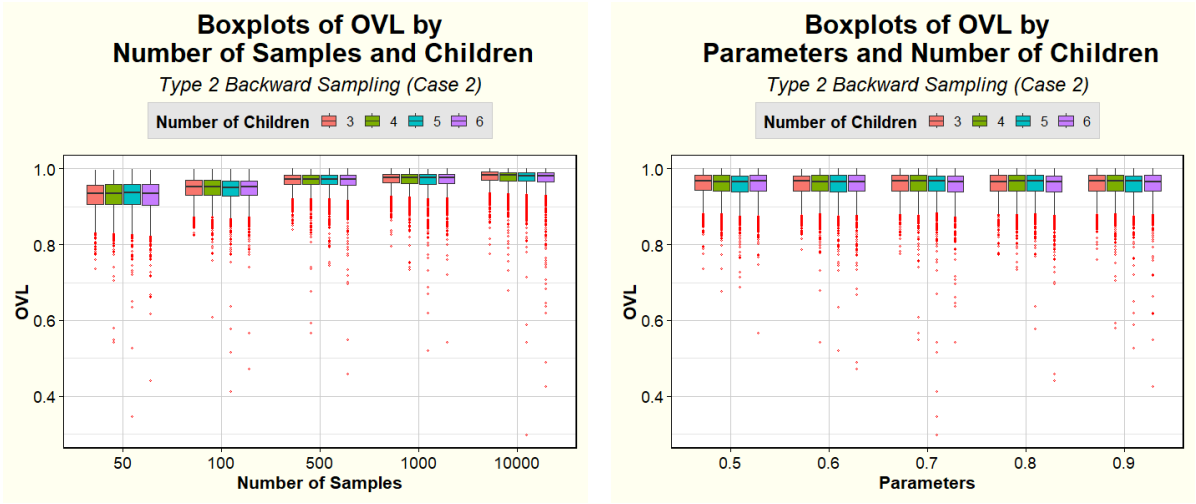
Table A.12: Table of average OVL by number of children in Type 2 Backward Sampling (Uncoupled Children).

L	OVL	95% CI
100	0.926	(0.924, 0.927)
500	0.955	(0.955, 0.956)
1000	0.961	(0.961, 0.962)
10000	0.967	(0.967, 0.968)

Table A.13: Table of average OVL by Importance Sample Size (L) in Type 2 Backward Sampling (Uncoupled Children).

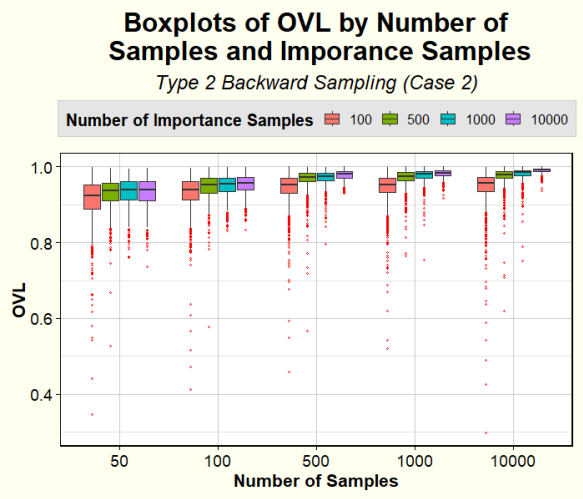
Coupled Children

In this segment, the results of Type 2 Backward Sampling are collected, for the DAG structure exhibiting children that contain arcs between them. This DAG structure is illustrated in figure 6.12.

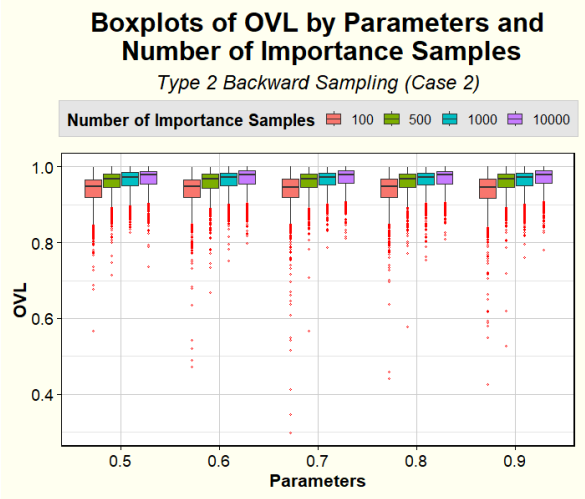


(a) **(b)**

Figure A.9: Type 2 Backward Sampling OVL plots by number of children (Coupled Children).



(a)



(b)

Figure A.10: Type 2 Backward Sampling OVL plots by number of importance samples (Coupled Children).

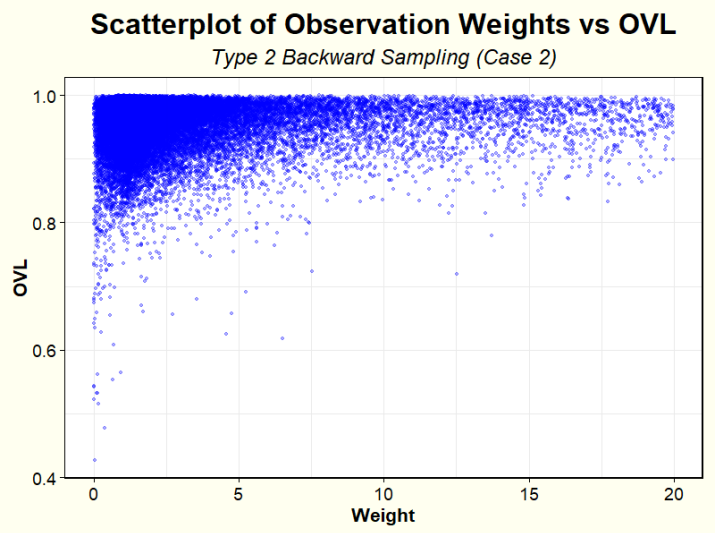


Figure A.11

Sample Size	OVL	95% CI
50	0.928	(0.927, 0.929)
100	0.945	(0.945, 0.946)
500	0.965	(0.965, 0.966)
1000	0.969	(0.969, 0.970)
10000	0.974	(0.974, 0.975)

Table A.14: Table of average OVL by Sample Size in Type 2 Backward Sampling (Coupled Children).

Param.	OVL	95% CI
0.5	0.957	(0.956, 0.957)
0.6	0.957	(0.956, 0.957)
0.7	0.957	(0.956, 0.957)
0.8	0.957	(0.956, 0.957)
0.9	0.956	(0.955, 0.957)

Table A.15: Table of average OVL by Parameters in Type 2 Backward Sampling (Coupled Children).

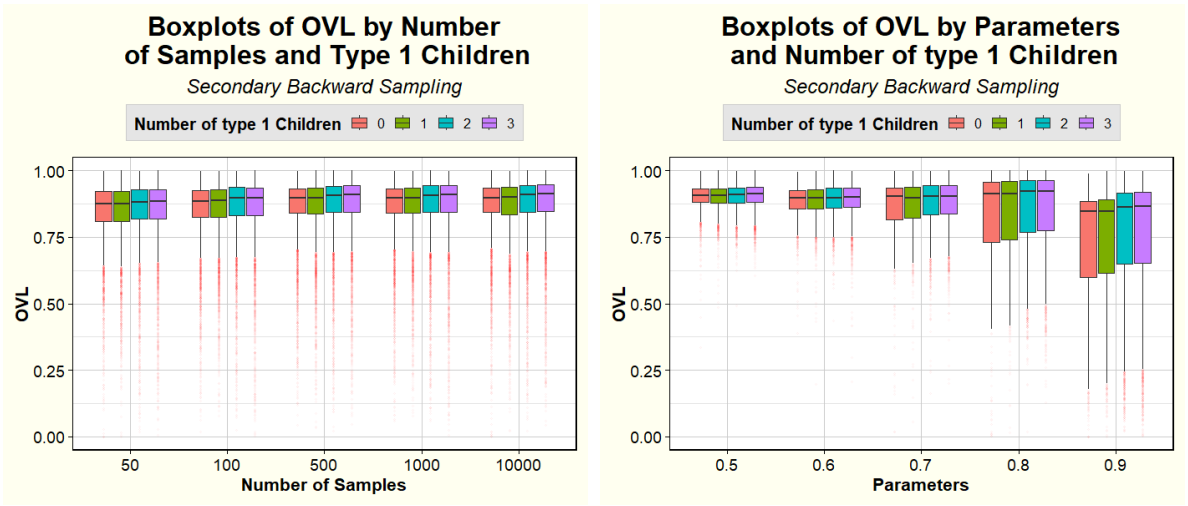
Children	OVL	95% CI
1	0.958	(0.957, 0.959)
2	0.957	(0.956, 0.958)
3	0.956	(0.955, 0.956)
4	0.955	(0.954, 0.956)

Table A.16: Table of average OVL by number of children in Type 2 Backward Sampling (Coupled Children).

L	OVL	95% CI
100	0.937	(0.936, 0.938)
500	0.959	(0.958, 0.959)
1000	0.963	(0.963, 0.964)
10000	0.967	(0.967, 0.968)

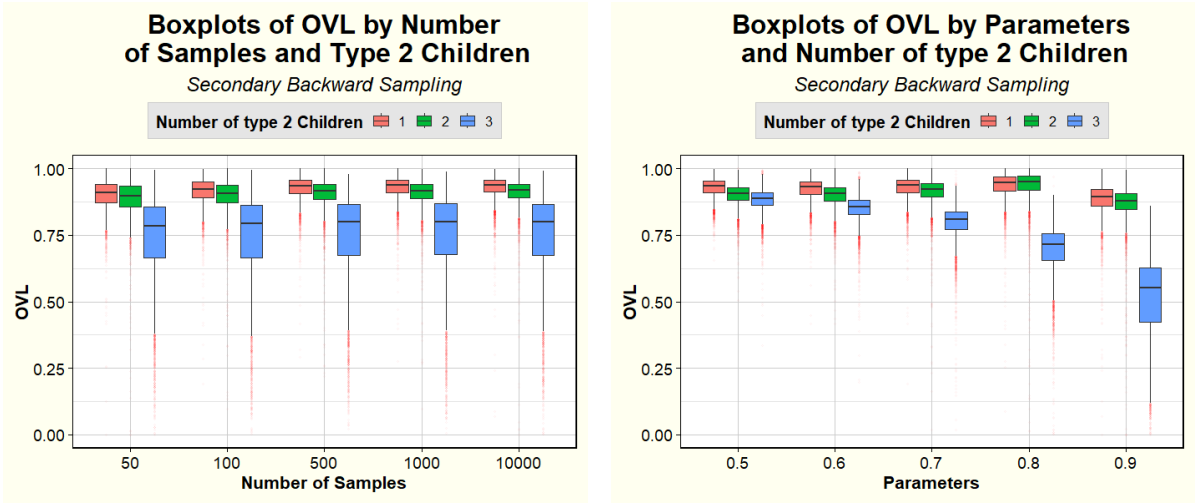
Table A.17: Table of average OVL by Importance Sample Size (L) in Type 2 Backward Sampling (Coupled Children).

A.4. Secondary Backward Sampling



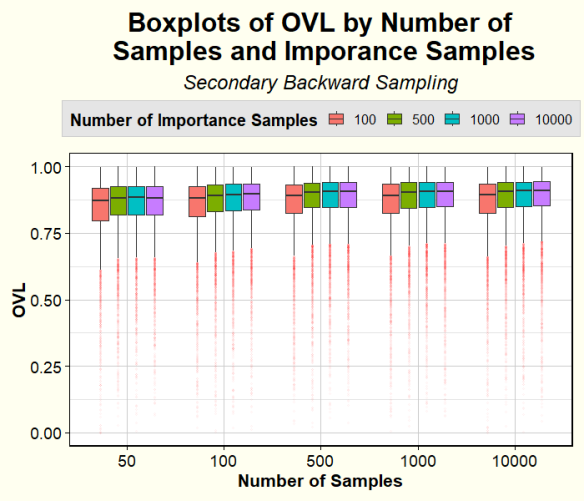
(a) (b)

Figure A.12: Secondary Backward Sampling OVL plots by number of type 1 children.

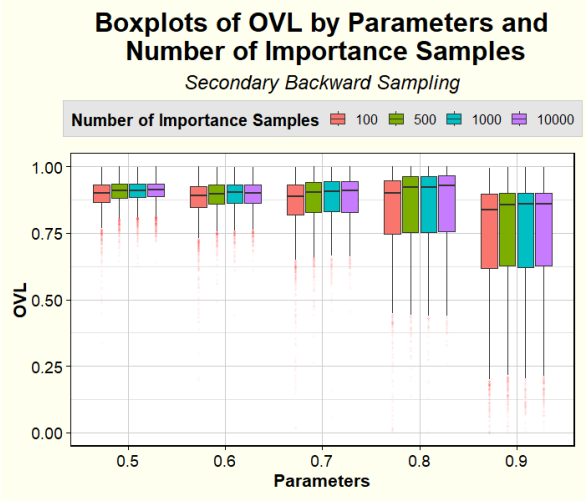


(a) (b)

Figure A.13: Secondary Backward Sampling OVL plots by number of type 2 children.



(a)



(b)

Figure A.14: Secondary Backward Sampling OVL plots by number of Importance Samples.

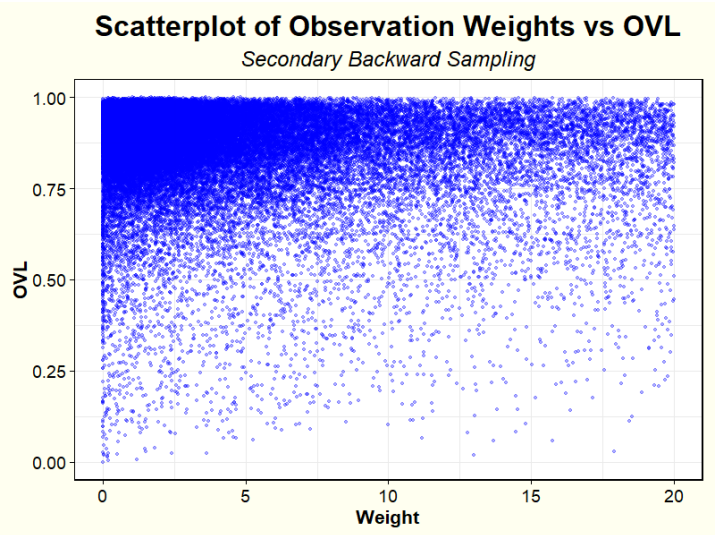


Figure A.15

Sample Size	OVL	95% CI
50	0.844	(0.843, 0.846)
100	0.853	(0.852, 0.855)
500	0.862	(0.860, 0.863)
1000	0.862	(0.861, 0.864)
10000	0.864	(0.862, 0.866)

Table A.18: Table of average OVL by Sample Size in Secondary Backward Sampling.

Param.	OVL	95% CI
0.5	0.903	(0.903, 0.904)
0.6	0.890	(0.889, 0.891)
0.7	0.879	(0.878, 0.880)
0.8	0.856	(0.854, 0.858)
0.9	0.757	(0.754, 0.759)

Table A.19: Table of average OVL by Parameters in Secondary Backward Sampling.

Ch. #1	OVL	95% CI
0	0.853	(0.851, 0.854)
1	0.854	(0.853, 0.856)
2	0.860	(0.859, 0.862)
3	0.861	(0.860, 0.863)

Table A.20: Table of average OVL by number of type 1 children in Secondary Backward Sampling.

Ch. #2	OVL	95% CI
1	0.920	(0.920, 0.921)
2	0.904	(0.904, 0.905)
3	0.747	(0.745, 0.748)

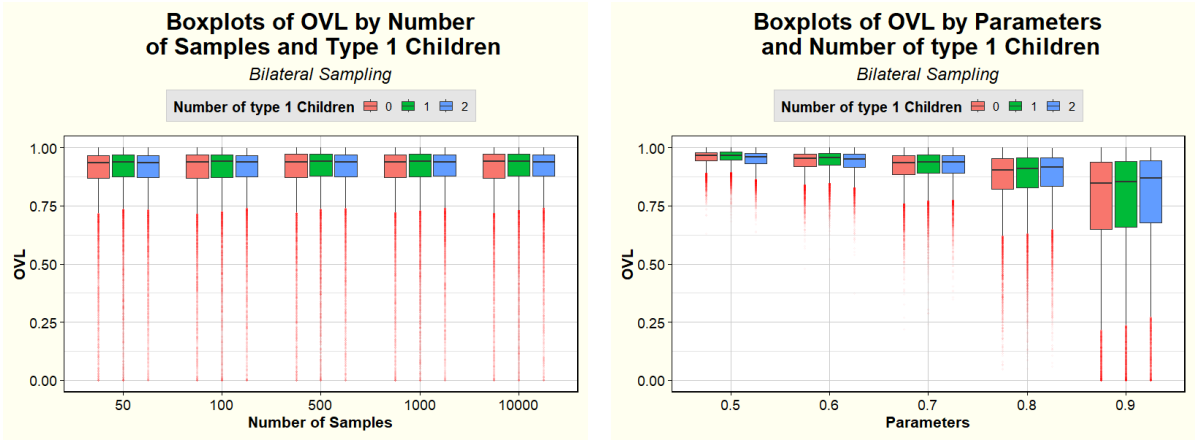
Table A.21: Table of average OVL by number of type 2 children in Secondary Backward Sampling.

L	OVL	95% CI
100	0.846	(0.844, 0.847)
500	0.859	(0.858, 0.860)
1000	0.861	(0.860, 0.863)
10000	0.862	(0.861, 0.864)

Table A.22: Table of average OVL by Importance Sample Size (L) in Secondary Backward Sampling.

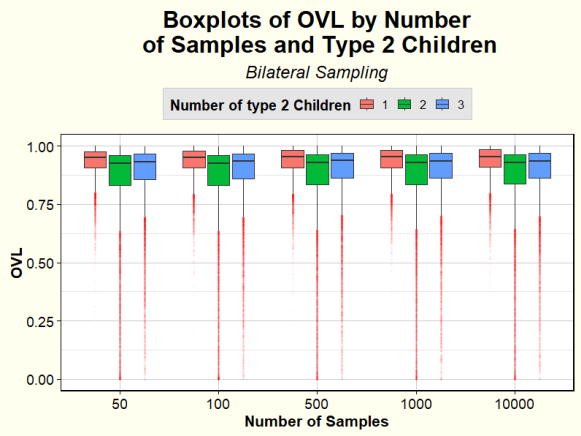
A.5. Bilateral Sampling

In this segment we attach the figures related to our simulation study of Bilateral Sampling. We note that these figure refer to Bilateral Sampling with the Forward Sample serving as the Importance Sample.

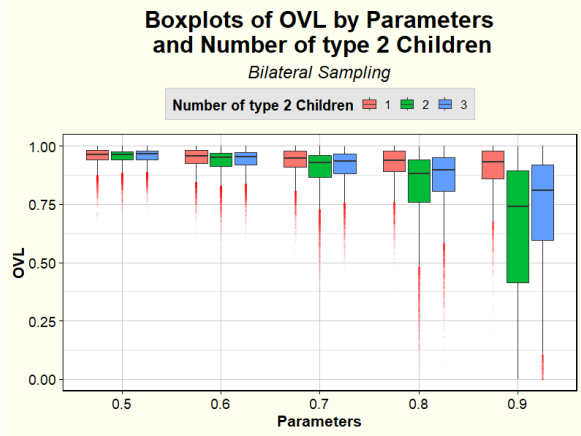


(a) **(b)**

Figure A.16: Bilateral Sampling OVL plots by number of type 1 children.

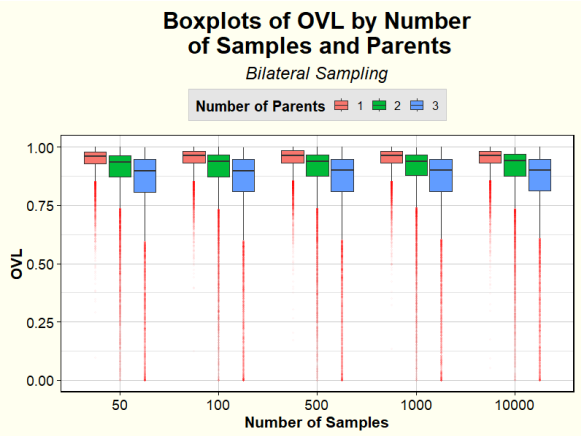


(a)

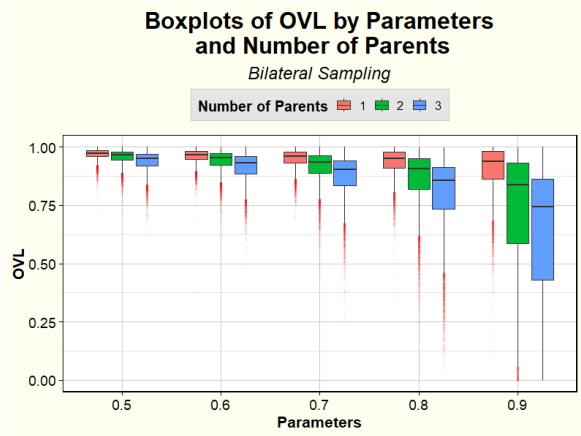


(b)

Figure A.17: Bilateral Sampling OVL plots by number of type 2 children.

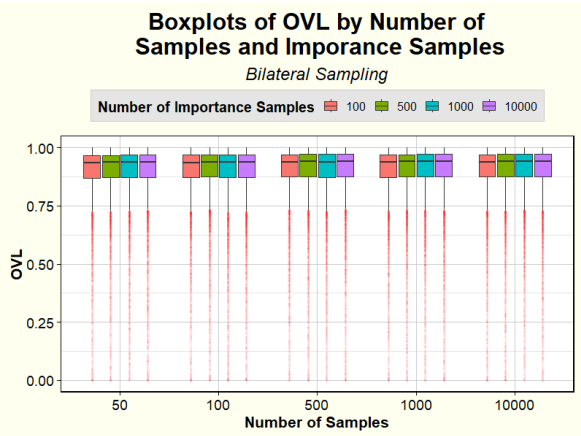


(a)

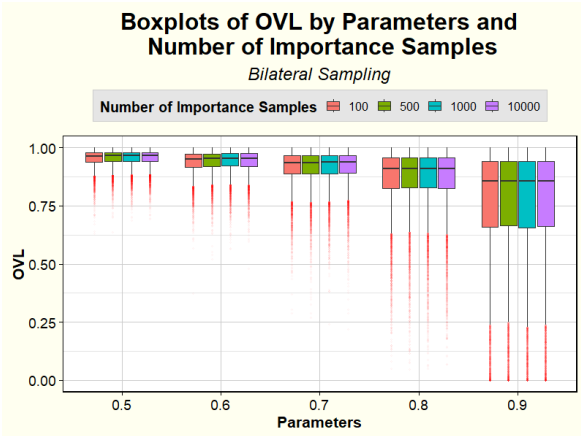


(b)

Figure A.18: Bilateral Sampling OVL plots by number of parents.



(a)



(b)

Figure A.19: Bilateral Sampling OVL plots by number of importance samples.

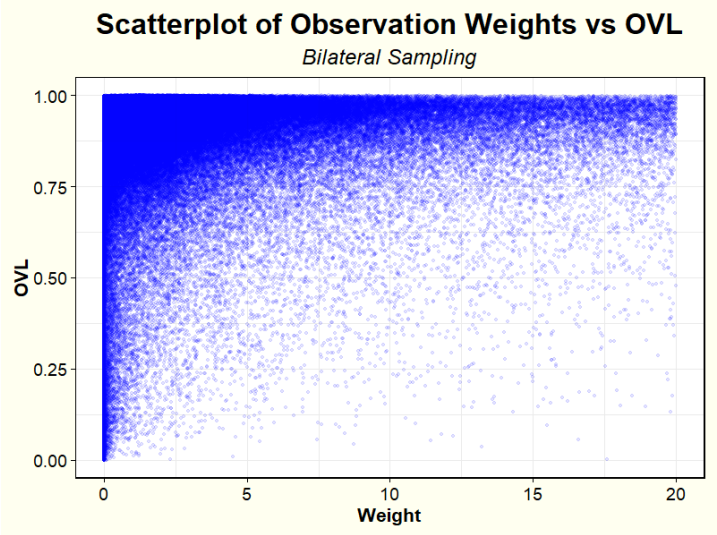


Figure A.20

Sample Size	OVL	95% CI
50	0.884	(0.883, 0.886)
100	0.886	(0.885, 0.888)
500	0.888	(0.887, 0.889)
1000	0.888	(0.887, 0.890)
10000	0.889	(0.888, 0.890)

Table A.23: Table of average OVL by Sample Size in Bilateral Sampling.

Param.	OVL	95% CI
0.5	0.955	(0.955, 0.956)
0.6	0.940	(0.940, 0.840)
0.7	0.916	(0.915, 0.916)
0.8	0.868	(0.867, 0.869)
0.9	0.757	(0.755, 0.759)

Table A.24: Table of average OVL by Parameters in Bilateral Sampling.

Ch. #1	OVL	95% CI
0	0.884	(0.883, 0.885)
1	0.889	(0.888, 0.890)
2	0.889	(0.888, 0.890)

Table A.25: Table of average OVL by number of type 1 children in Bilateral Sampling.

Ch. #2	OVL	95% CI
1	0.933	(0.933, 0.934)
2	0.850	(0.848, 0.851)
3	0.879	(0.878, 0.880)

Table A.26: Table of average OVL by number of type 2 children in Bilateral Sampling.

Parents	OVL	95% CI
1	0.944	(0.944, 0.945)
2	0.884	(0.883, 0.885)
3	0.834	(0.832, 0.835)

Table A.27: Table of average OVL by number of parents in Bilateral Sampling.

L	OVL	95% CI
100	0.886	(0.885, 0.887)
500	0.888	(0.887, 0.889)
1000	0.888	(0.886, 0.889)
10000	0.887	(0.886, 0.889)

Table A.28: Table of average OVL by Importance Sample Size (L) in Bilateral Sampling.

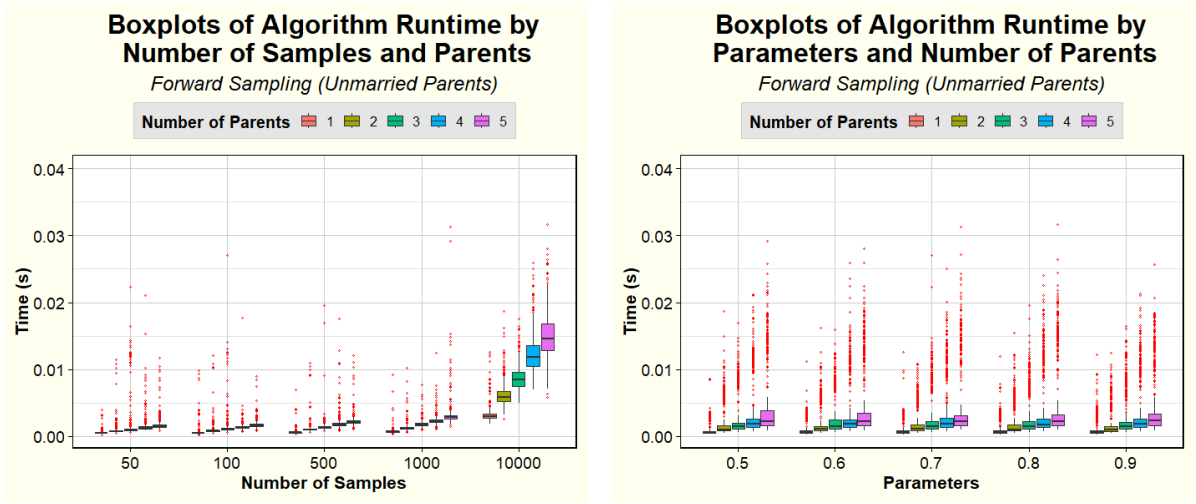
B

Efficiency Results

This appendix is the collection of the effectiveness results in chapter 6. We focus on the execution time of each algorithm and how different simulation parameters affect its distribution.

B.1. Forward Sampling

Unmarried Parents



(a) (b)

Figure B.1: Forward Sampling plots of execution time (Unmarried Parents).

Sample Size	Time (s)	95% CI
50	0.0044	(0.0042, 0.0045)
100	0.0052	(0.0050, 0.0.0055)
500	0.0064	(0.0062, 0.0066)
1000	0.0084	(0.0081, 0.0086)
10000	0.0430	(0.0420, 0.0440)

Table B.1: Table of average execution time by Sample Size in Forward Sampling (Unmarried Parents).

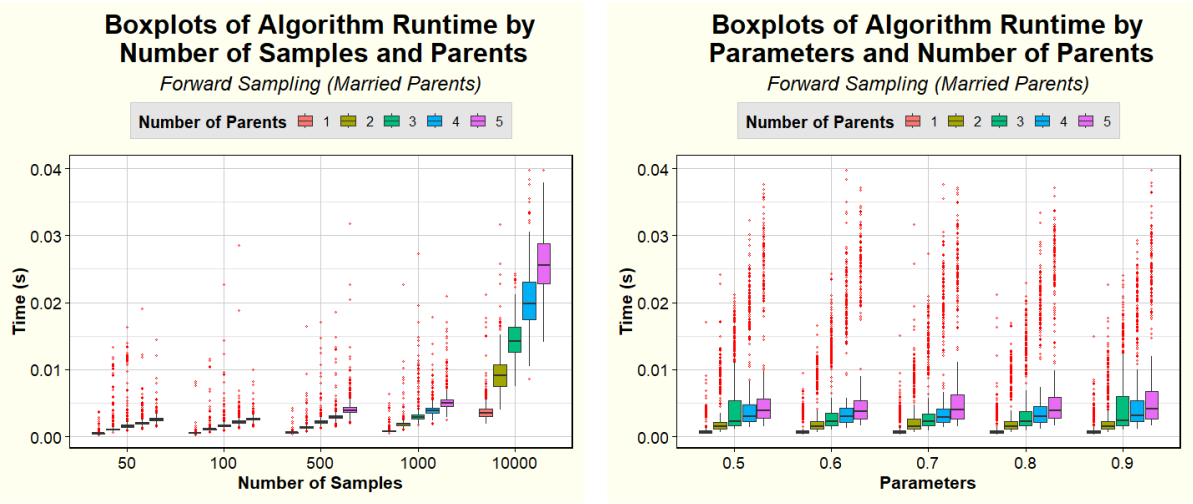
Param.	Time (s)	95% CI
0.5	0.014	(0.013, 0.015)
0.6	0.014	(0.013, 0.014)
0.7	0.014	(0.013, 0.014)
0.8	0.014	(0.013, 0.014)
0.9	0.013	(0.013, 0.014)

Table B.2: Table of average execution time by Parameters in Forward Sampling (Unmarried Parents).

Parents	Time (s)	95% CI
1	0.0052	(0.0049, 0.0055)
2	0.0095	(0.0090, 0.0099)
3	0.0140	(0.0130, 0.0140)
4	0.0180	(0.0170, 0.0190)
5	0.0210	(0.0200, 0.0220)

Table B.3: Table of average execution time by number of parents in Forward Sampling (Unmarried Parents).

Married Parents



(a) **(b)**

Figure B.2: Forward Sampling plots of execution time (Married Parents).

Sample Size	Time (s)	95% CI
50	0.0077	(0.0075, 0.0080)
100	0.0078	(0.0075, 0.0081)
500	0.0096	(0.0094, 0.0098)
1000	0.0120	(0.0120, 0.0130)
10000	0.0640	(0.0630, 0.0660)

Table B.4: Table of average execution time by Sample Size in Forward Sampling (Married Parents).

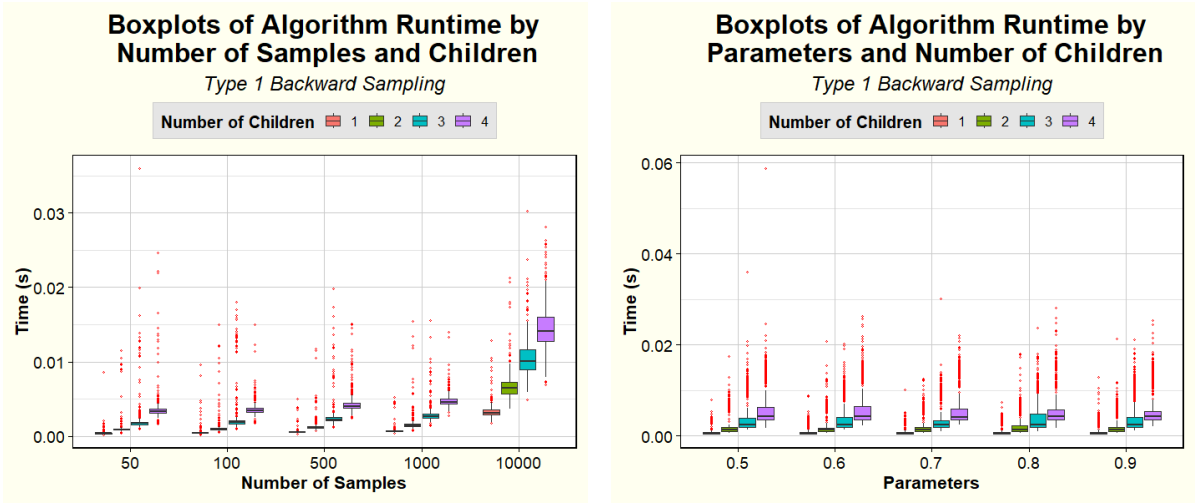
Param.	Time (s)	95% CI
0.5	0.020	(0.019, 0.021)
0.6	0.020	(0.019, 0.021)
0.7	0.020	(0.019, 0.021)
0.8	0.020	(0.019, 0.021)
0.9	0.021	(0.019, 0.022)

Table B.5: Table of average execution time by Parameters in Forward Sampling (Married Parents).

Parents	Time (s)	95% CI
1	0.0051	(0.0048, 0.0053)
2	0.0130	(0.0120, 0.0130)
3	0.0210	(0.0200, 0.0220)
4	0.0280	(0.0270, 0.0290)
5	0.0350	(0.0330, 0.0360)

Table B.6: Table of average execution time by number of Parents in Forward Sampling (Married Parents).

B.2. Type 1 Backward Sampling



(a) **(b)**

Figure B.3: Type 1 Backaward Sampling plots of execution time.

Sample Size	Time (s)	95% CI
50	0.0017	(0.0016, 0.0018)
100	0.0017	(0.0017, 0.0018)
500	0.0020	(0.0019, 0.0021)
1000	0.0024	(0.0023, 0.0025)
10000	0.0083	(0.0081, 0.0085)

Table B.7: Table of average execution time by Sample Size in Type 1 Backward Sampling.

Param.	Time (s)	95% CI
0.5	0.0032	(0.0031, 0.0034)
0.6	0.0032	(0.0030, 0.0033)
0.7	0.0032	(0.0031, 0.0035)
0.8	0.0032	(0.0031, 0.0034)
0.9	0.0032	(0.0030, 0.0033)

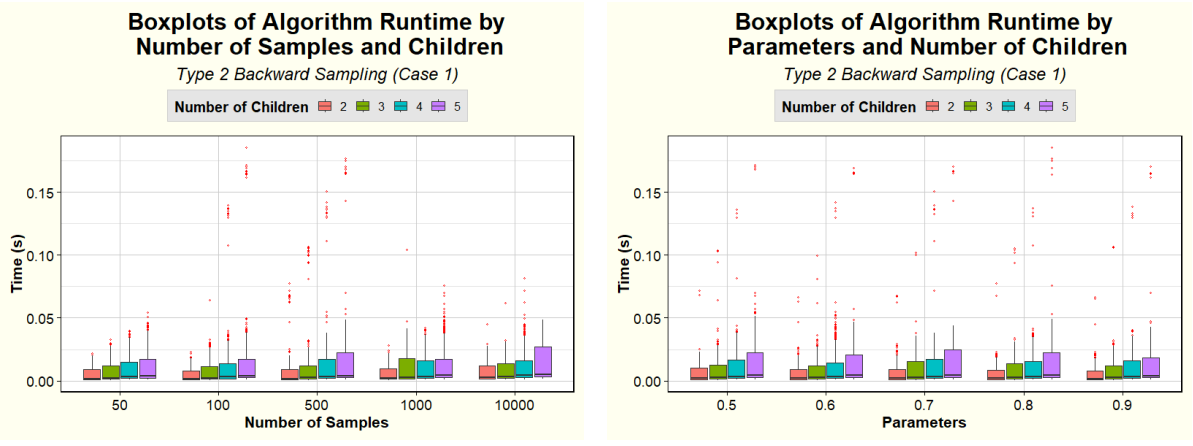
Table B.8: Table of average execution time by Parameters in Type 1 Backward Sampling.

Children	Time (s)	95% CI
1	0.0010	(0.00097, 0.0011)
2	0.0023	(0.00220, 0.0024)
3	0.0038	(0.00370, 0.0040)
4	0.0058	(0.00560, 0.0060)

Table B.9: Table of average execution time by number of children in Type 1 Backward Sampling.

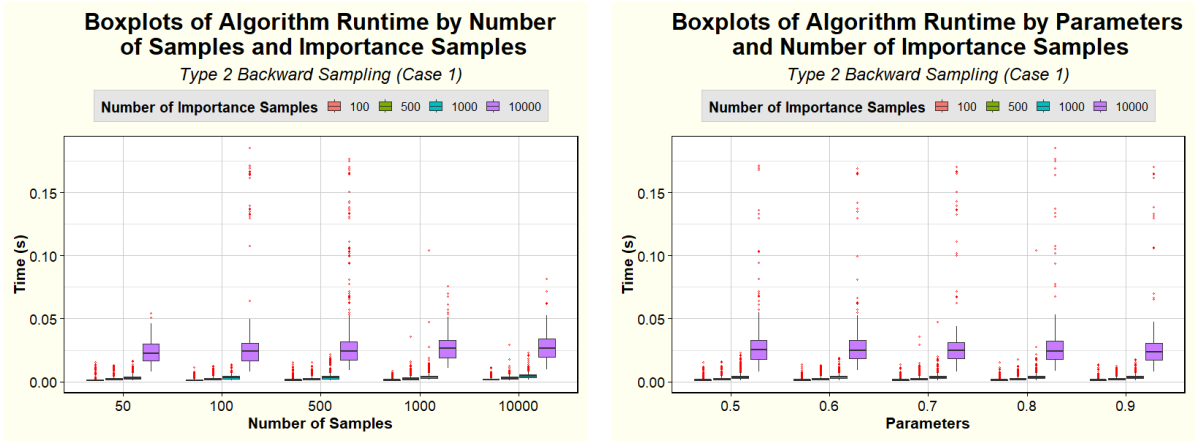
B.3. Type 2 Backward Sampling

Uncoupled Children



(a) **(b)**

Figure B.4: Type 2 Backward Sampling plots of execution time by number of children (Uncoupled Children).



(a) **(b)**

Figure B.5: Type 2 Backward Sampling plots of execution time by number of importance samples (Uncoupled Children).

Sample Size	Time (s)	95% CI
50	0.0073	(0.0071, 0.0076)
100	0.0079	(0.0076, 0.0082)
500	0.0084	(0.0081, 0.0087)
1000	0.0084	(0.0081, 0.0086)
10000	0.0089	(0.0087, 0.0092)

Table B.10: Table of average execution time by Sample Size in Type 2 Backward Sampling (Uncoupled Children).

Param.	Time (s)	95% CI
0.5	0.0084	(0.0081, 0.0086)
0.6	0.0082	(0.0080, 0.0085)
0.7	0.0082	(0.0079, 0.0084)
0.8	0.0082	(0.0079, 0.0085)
0.9	0.0079	(0.0077, 0.0082)

Table B.11: Table of average execution time by Parameters in Type 2 Backward Sampling (Uncoupled Children).

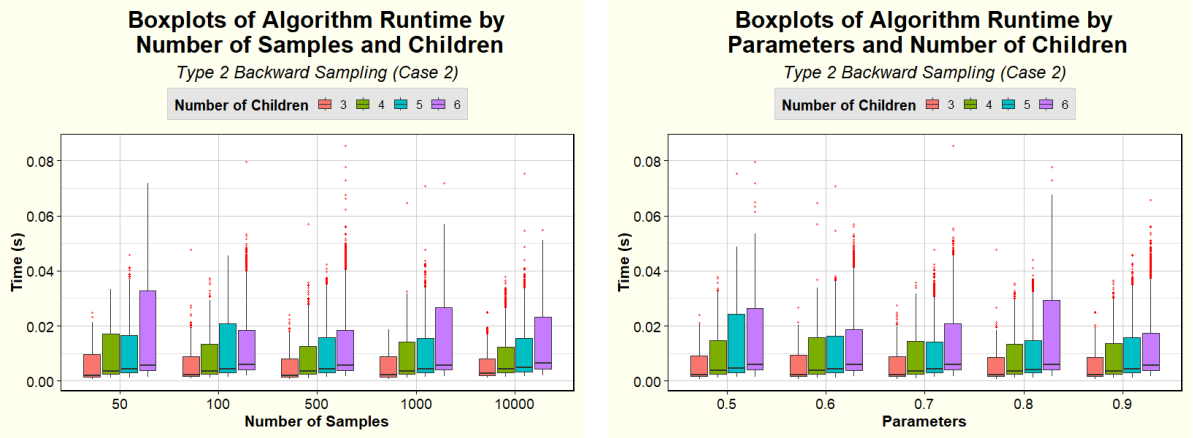
Children	Time (s)	95% CI
1	0.0048	(0.0047, 0.0049)
2	0.0070	(0.0068, 0.0072)
3	0.0094	(0.0091, 0.0096)
4	0.0120	(0.0110, 0.0120)

Table B.12: Table of average execution time by number of children in Type 2 Backward Sampling (Uncoupled Children).

L	Time (s)	95% CI
100	0.0014	(0.0014, 0.0015)
500	0.0025	(0.0025, 0.0025)
1000	0.0037	(0.0036, 0.0037)
10000	0.0250	(0.0250, 0.0250)

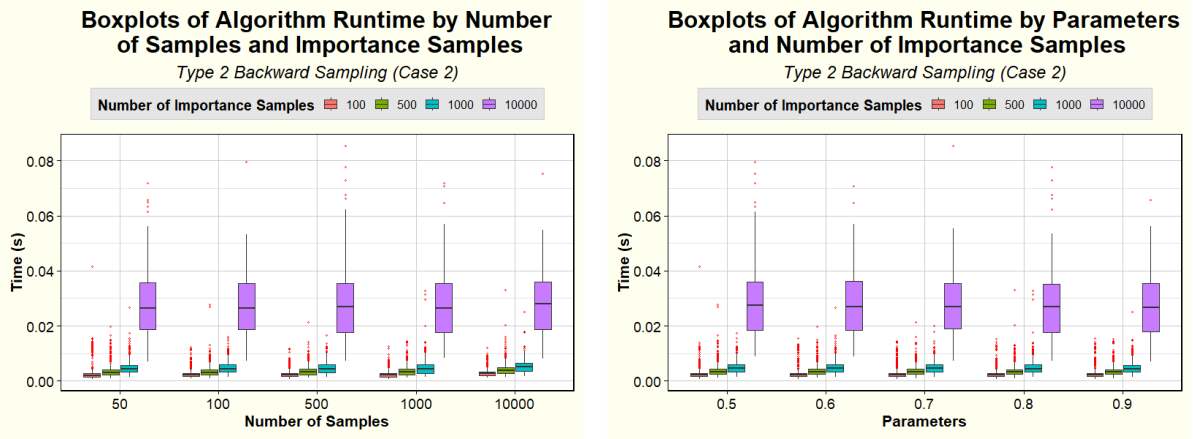
Table B.13: Table of average execution time by Importance Sample Size (L) in Type 2 Backward Sampling (Uncoupled Children).

Coupled Children



(a) **(b)**

Figure B.6: Type 2 Backward Sampling plots of execution time by number of children (Coupled Children).



(a) **(b)**

Figure B.7: Type 2 Backward Sampling plots of execution time by number of importance samples (Coupled Children).

Sample Size	Time (s)	95% CI
50	0.0091	(0.0089, 0.0094)
100	0.0091	(0.0089, 0.0094)
500	0.0092	(0.0089, 0.0095)
1000	0.0092	(0.0089, 0.0094)
10000	0.0099	(0.0096, 0.0100)

Table B.14: Table of average execution time by Sample Size in Type 2 Backward Sampling (Coupled Children).

Param.	Time (s)	95% CI
0.5	0.0094	(0.0091, 0.0097)
0.6	0.0094	(0.0091, 0.0096)
0.7	0.0093	(0.0091, 0.0096)
0.8	0.0092	(0.0090, 0.0095)
0.9	0.0092	(0.0090, 0.0095)

Table B.15: Table of average execution time by Parameters in Type 2 Backward Sampling (Coupled Children).

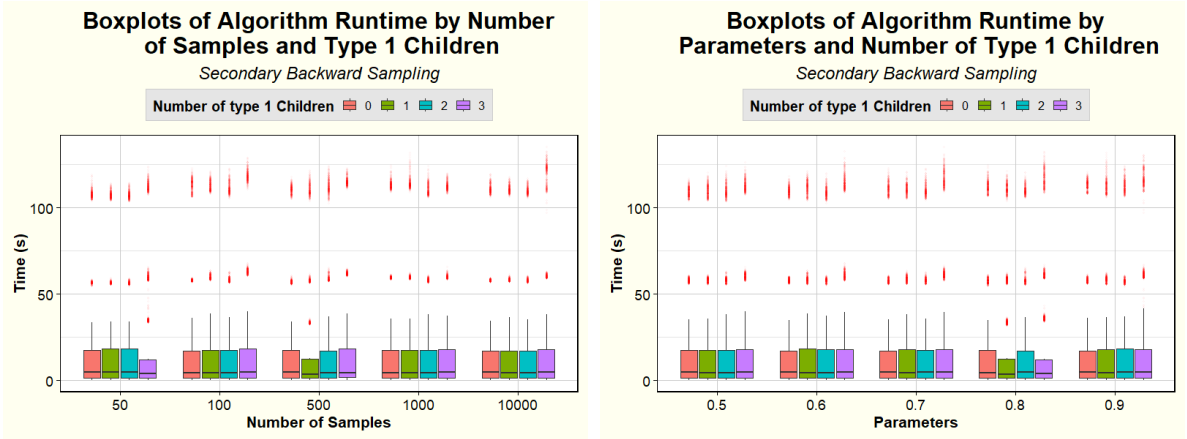
Children	Time (s)	95% CI
1	0.0048	(0.0047, 0.0049)
2	0.0082	(0.0081, 0.0084)
3	0.0100	(0.0100, 0.0100)
4	0.0140	(0.0140, 0.0140)

Table B.16: Table of average execution time by number of children in Type 2 Backward Sampling (Coupled Children).

L	Time (s)	95% CI
100	0.0024	(0.0024, 0.0024)
500	0.0034	(0.0034, 0.0034)
1000	0.0046	(0.0046, 0.0046)
10000	0.0270	(0.0270, 0.0270)

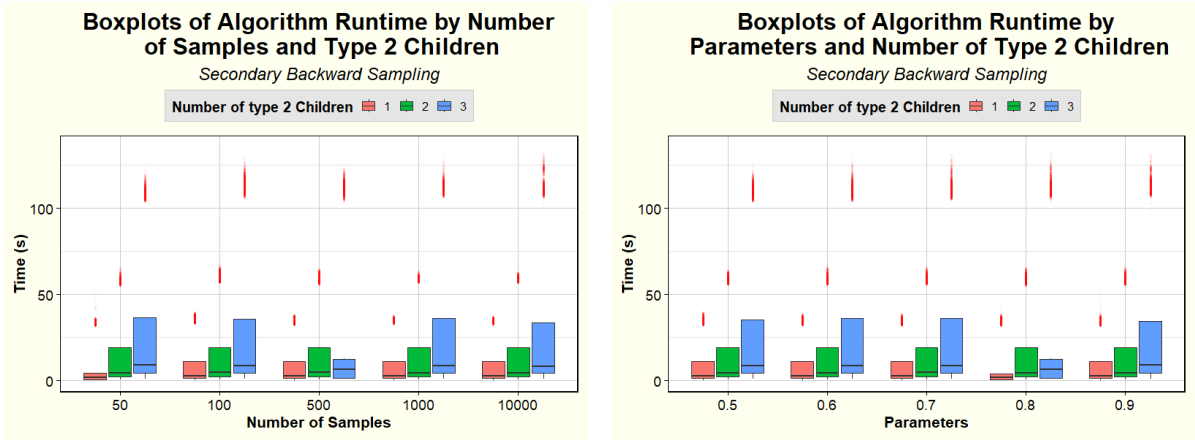
Table B.17: Table of average execution time by Importance Sample Size (L) in Type 2 Backward Sampling (Coupled Children).

B.4. Secondary Backward Sampling



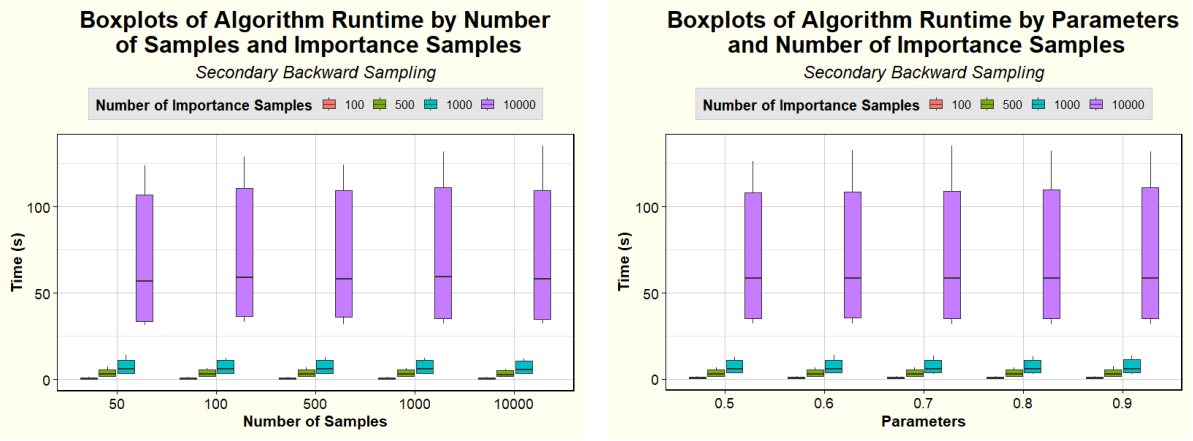
(a) (b)

Figure B.8: Secondary Backward Sampling plots of execution time by number of type 1 children.



(a) (b)

Figure B.9: Secondary Backward Sampling plots of execution time by number of type 2 children.



(a) **(b)**

Figure B.10: Secondary Backward Sampling plots of execution time by number of Importance Samples.

Sample Size	Time (s)	95% CI
50	21	(20, 22)
100	20	(20, 21)
500	22	(21, 22)
1000	20	(20, 20)
10000	20	(19, 20)

Table B.18: Table of average execution time by Sample Size in Secondary Backward Sampling.

Param.	Time (s)	95% CI
0.5	20	(19, 20)
0.6	20	(19, 20)
0.7	20	(20, 20)
0.8	23	(22, 23)
0.9	20	(20, 20)

Table B.19: Table of average execution time by Parameters in Secondary Backward Sampling.

Ch. #1	Time (s)	95% CI
0	20	(19, 20)
1	21	(20, 22)
2	20	(19, 20)
3	22	(21, 23)

Table B.20: Table of average execution time by number of type 1 children in Secondary Backward Sampling.

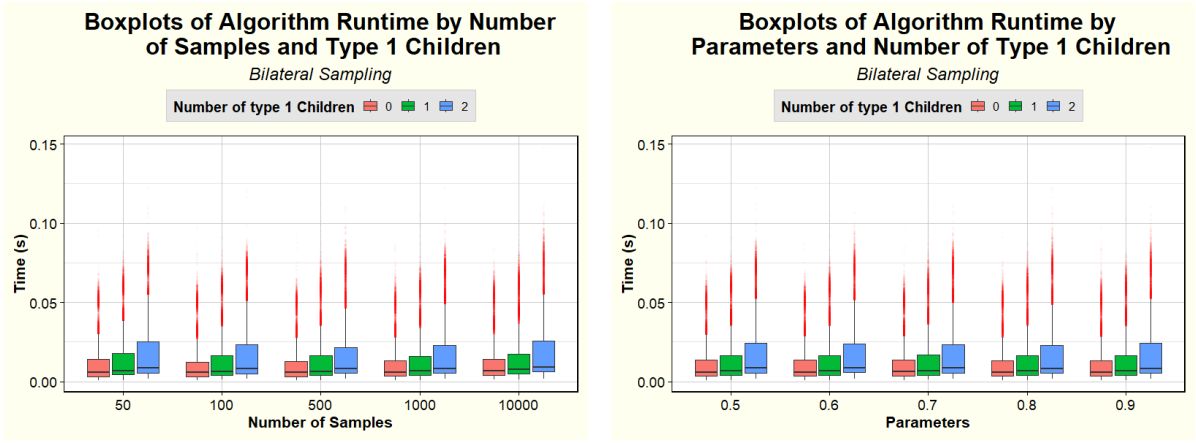
Ch. #2	Time (s)	95% CI
1	11	(10, 11)
2	17	(17, 17)
3	34	(33, 34)

Table B.21: Table of average execution time by number of type 2 children in Secondary Backward Sampling.

L	Time (s)	95% CI
100	0.7	(0.7, 0.71)
500	3.5	(3.5, 3.5)
1000	7	(6.9, 7)
10000	71	(70, 72)

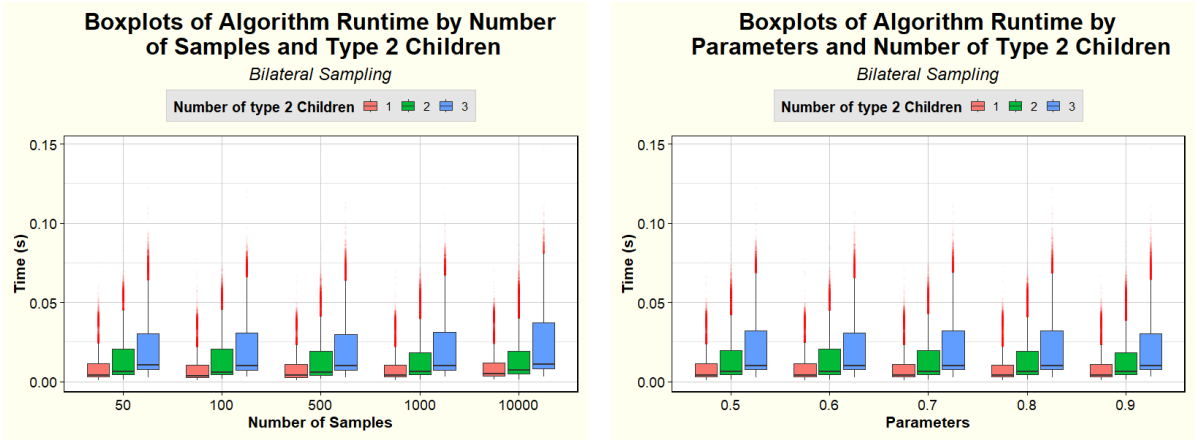
Table B.22: Table of average execution time by Importance Sample Size (L) in Secondary Backward Sampling.

B.5. Bilateral Sampling



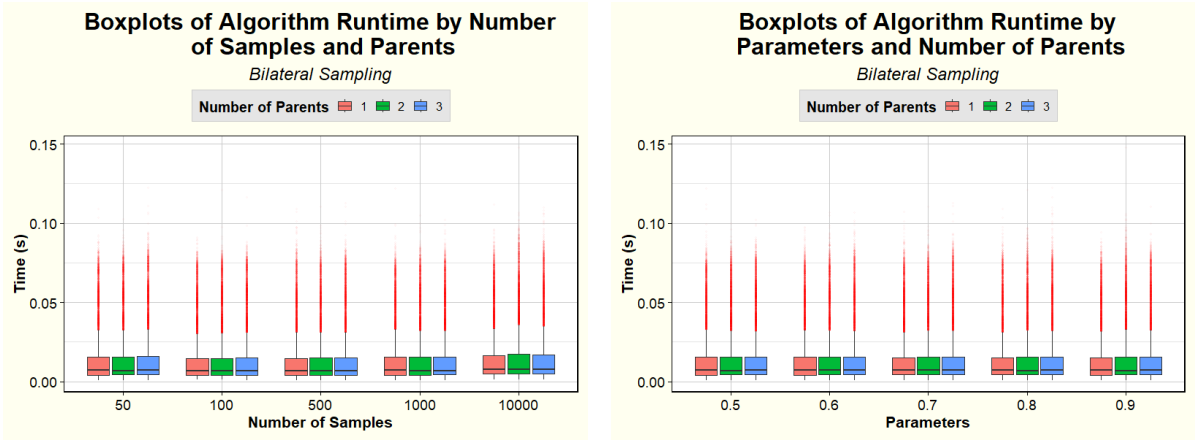
(a) (b)

Figure B.11: Bilateral Sampling plots of execution time by number of type 1 children.



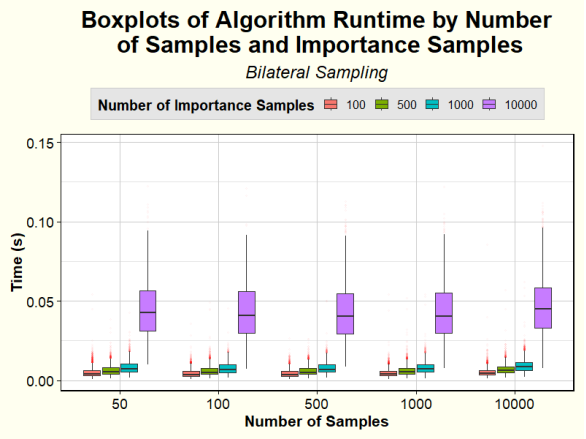
(a) (b)

Figure B.12: Bilateral Sampling plots of execution time by number of type 2 children.

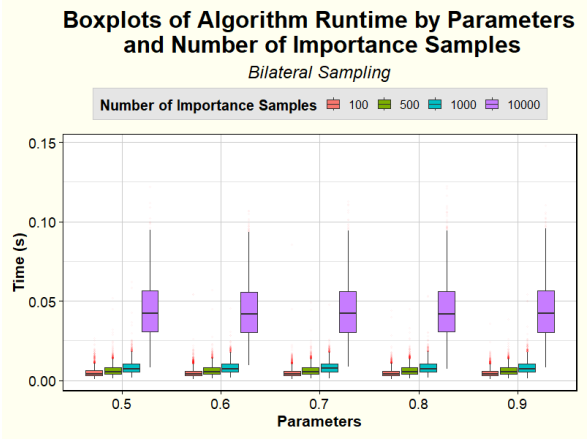


(a) (b)

Figure B.13: Bilateral Sampling plots of execution time by number of parents.



(a)



(b)

Figure B.14: Bilateral Sampling plots of execution time by number of importance samples.

Sample Size	Time (s)	95% CI
50	0.016	(0.016, 0.016)
100	0.015	(0.015, 0.015)
500	0.015	(0.015, 0.015)
1000	0.015	(0.015, 0.015)
10000	0.017	(0.017, 0.017)

Table B.23: Table of average execution time by Sample Size in Bilateral Sampling.

Param.	Time (s)	95% CI
0.5	0.016	(0.015, 0.017)
0.6	0.016	(0.015, 0.017)
0.7	0.016	(0.015, 0.017)
0.8	0.016	(0.015, 0.017)
0.9	0.016	(0.015, 0.017)

Table B.24: Table of average execution time by Parameters in Bilateral Sampling.

Ch. #1	Time (s)	95% CI
0	0.012	(0.012, 0.012)
1	0.015	(0.015, 0.015)
2	0.019	(0.019, 0.019)

Table B.25: Table of average execution time by number of type 1 children in Bilateral Sampling.

Ch. #2	Time (s)	95% CI
1	0.0096	(0.0095, 0.0097)
2	0.0150	(0.0150, 0.0150)
3	0.0220	(0.0220, 0.0220)

Table B.26: Table of average execution time by number of type 2 children in Bilateral Sampling.

Parents	Time (s)	95% CI
1	0.015	(0.015, 0.015)
2	0.016	(0.015, 0.016)
3	0.016	(0.016, 0.016)

Table B.27: Table of average execution time by number of parents in Bilateral Sampling.

L	Time (s)	95% CI
100	0.0045	(0.0045, 0.0045)
500	0.0061	(0.0061, 0.0061)
1000	0.0080	(0.0080, 0.0081)
10000	0.0440	(0.0440, 0.0440)

Table B.28: Table of average execution time by Importance Sample Size (L) in Bilateral Sampling.