



FRIEDRICH-SCHILLER-  
UNIVERSITÄT  
JENA

---

# Exploiting Markov Equivalence for Fast Inference

---

MASTER THESIS

Submitted for the degree of  
MASTER OF SCIENCE (M.Sc.)

FRIEDRICH SCHILLER UNIVERSITY JENA  
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE  
MATHEMATICS

*Author*

Jenette SELLIN

Born December 22, 1995  
in CALIFORNIA, USA

*Advisor*

Andreas GORAL

*Supervisor*

Prof. Dr. Joachim GIESEN

Jena, March 30, 2020



# Abstract

# **Zusammenfassung**

## Acknowledgements

## Notation and Abbreviations

---

DAG	Directed Acyclic Graph
iff	If and only if
$G = (V, E)$	A graph $G$ with vertices $V$ and edges $E$
$[G]$	The Markov equivalence class of $G$
MEC	Markov Equivalence Class
	essential graph of $G$
$G, G', G_1, G_2, \dots, G_n$	directed graphs
$H, H', H_1, H_2, \dots, H_n$	undirected and mixed graphs
$B, B', B_1, B_2, \dots, B_n$	Bayesian networks

# Contents

<b>Contents</b>	<b>V</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Background . . . . .	1
1.3 Goal . . . . .	3
1.4 Motivating Example . . . . .	3
1.5 Related work . . . . .	5
1.6 Outline . . . . .	6
<b>2 Preliminaries</b>	<b>7</b>
2.1 Foundations of Graph Theory . . . . .	7
2.2 Foundations of Bayesian Statistics . . . . .	9
<b>3 Querying</b>	<b>10</b>
3.1 Outline of the Task . . . . .	10
3.2 Structure of a Query . . . . .	10
<b>4 Markov Equivalence</b>	<b>12</b>
4.1 Goal . . . . .	12
4.2 Basic Properties of Markov Equivalence . . . . .	12
4.3 Effects of Edge Reversal . . . . .	16
4.4 Find Reversible Edges of a DAG Model . . . . .	19
<b>5 Exploiting Markov Equivalence using Greedy Strategy</b>	<b>24</b>
5.1 Motivation . . . . .	24
5.2 Implementation . . . . .	26
5.3 Cost . . . . .	26
<b>6 Comparison of methods</b>	<b>27</b>





# Chapter 1

## Introduction

### 1.1 Motivation

Many fields rely on prediction and classification of new information using known data, processes which require encoding statistical relationships into analyzable structures. Conditional dependencies between random variables in probability distributions can be effectively modeled using Bayesian networks, giving us a reliable way to do inference on our model.

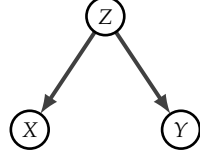
For instance, in the medical field, we may have information about a large number of patients (such as height, age, sex, blood type, etc.) and we may wish to use this information to predict whether a patient has a specific disease or not. In this scenario, we would need to model the known data and then query the model for the probability that, given these factors, a patient has the disease.

Naturally, it is in our best interest to create a model and a method for querying which allow us to do inference on our probabilistic models as efficiently as possible, in order to achieve results more quickly. This thesis explores a method for answering queries more efficiently in the context of graphical models.

### 1.2 Background

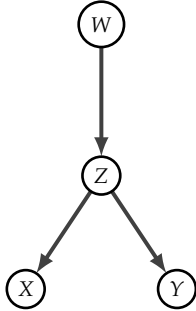
The family of models explored in this thesis are directed graphical models over discrete probability distributions. The purpose of graphical models is to encode information about vectors of random variables, namely the interdependencies satisfied by the vector's joint probability function. For example, consider a vector of three random binary variables  $(X, Y, Z)$ . Suppose that the values of

$X$  and  $Y$  depend on  $Z$ , but  $X$  and  $Y$  are independent from one another given  $Z$ . These dependencies can be encoded in the following graph:



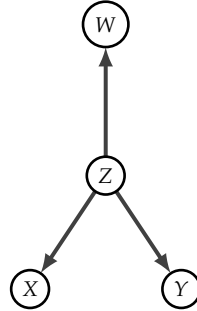
$$p(X, Y | Z) = p(X | Z)p(Y | Z)$$

In the case of directed models, which we explore in this thesis, there is not necessarily a unique graph which satisfies the dependency constraints of the joint probability function. A random vector and its constraints may be described by a variety of different graph structures which ultimately describe the same underlying probability distribution. This will be explored in more detail later. To understand how the complexity of answering a query depends on graph structure, consider the following two models:



**(A)**

$$p(X, Y, Z, W) = p(X | Z)p(Y | Z)p(Z | W)p(W)$$



**(B)**

$$p(X, Y, Z, W) = p(X | Z)p(Y | Z)p(W | Z)p(Z)$$

If we wish to answer a query about the random variable  $Y$  in graph **(A)**, for instance, we must also compute the probabilities of each vertex it depends on either directly or indirectly, that is, all of its **ancestors** (in this case  $Z$  and  $W$  in addition to  $Y$  itself). Alternatively, if we wish to answer a query about  $Y$  in graph **(B)**, we need only to compute  $Z$  and  $Y$ . The number of variables which must be sampled is reduced when we consider **(B)** instead of **(A)**.

In a context where answering queries about a single vertex may require us to

sample arbitrarily many other vertices in the graph, we seek to find a graphical representation of our probability distribution which minimizes the number of vertices that must be sampled when answering a query.

### 1.3 Goal

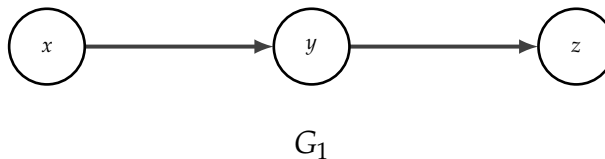
There are certain circumstances in which two Bayesian networks with different graphical structures can describe the same underlying probability distribution. This means that querying the graphs will produce statistically indistinguishable results. Such sets of graphs are called *Markov equivalent*.

The goal of this research is as follows: given a query and a Bayesian network  $B$  (which represents a single factorization of the probability distribution we wish to query), find a Markov equivalent Bayesian network  $B'$  which minimizes the number of variables that must be computed to answer the query. The purpose of the minimization is to reduce the cost of querying in order to achieve faster inference, as well as evaluate the gained speed-up versus the cost of transforming the graph structure.

To achieve this, we must first define the set of nodes which must be computed for a given query over a  $B$ . Then, we must search for a Markov equivalent Bayesian network  $B'$  such that the size of this set of necessary nodes is minimized for the same query over  $B'$ . Finally, we compare the costs of computing the query over  $B$  to the costs of computing the query over the minimized graph  $B'$  plus the costs of identifying  $B'$ .

### 1.4 Motivating Example

The following example demonstrates how answering a query can be sped up by querying a Markov equivalent graph with a different structure. Consider the following DAG:



with conditional probabilities given by

$$p(x = 1) = 0.2$$

$$p(y = 1|x = 0) = 0.3$$

$$p(z = 1|y = 0) = 0.1$$

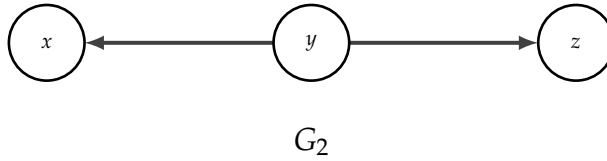
$$p(y = 1|x = 1) = 0.4$$

$$p(z = 1|y = 1) = 0.9$$

and the joint probability

$$p(x, y, z) = p(x) \cdot p(y|x) \cdot p(z|y)$$

Here, if we wish to answer the query  $p(z|y)$ , our computation relies on all three random variables, since  $z$  relies on  $y$  and  $y$  relies on  $x$ . However, we can find a Markov equivalent graph in which we can answer the same query while relying on fewer random variables. Consider the following DAG:



Here, we compute the conditional probabilities of  $G_2$  to show that the model can equivalently describe our joint probability distribution from  $G_1$ , and therefore be used to answer the query  $p(z|y)$ . We then observe how the structure of the new model affects our computation of the query. By applying Bayes' Theorem, which states that  $p(A|B) = \frac{p(B|A)p(A)}{p(B)}$ , we can compute

$$p(x|y) = \frac{p(y|x) \cdot p(x)}{p(y)} = \frac{p(y|x) \cdot p(x)}{\sum_x p(y|x) \cdot p(x)}$$

Using our pre-defined conditional probabilities,

$$\begin{aligned} p(y = 0) &= \sum_x p(y = 0|x) \cdot p(x) \\ &= 0.7 \cdot 0.8 + 0.6 \cdot 0.2 \\ &= 0.56 + 0.12 \\ &= 0.68 \end{aligned}$$

and

$$\begin{aligned}
p(y=1) &= \sum_x p(y=1|x) \cdot p(x) \\
&= 0.3 \cdot 0.8 + 0.4 \cdot 0.2 \\
&= 0.24 + 0.08 \\
&= 0.32.
\end{aligned}$$

which allow us to compute

$$p(x=1|y=0) = \frac{p(y=0|x=1) \cdot p(x=1)}{p(y=0)} = \frac{0.6 \cdot 0.2}{0.68} = \frac{3}{17}$$

and

$$p(x=1|y=1) = \frac{p(y=1|x=1) \cdot p(x=1)}{p(y=1)} = \frac{0.4 \cdot 0.2}{0.32} = \frac{1}{4}$$

Then, since  $x$  and  $y$  do not depend on  $z$ , the conditional probability of  $z$  remains unchanged. Therefore, model  $G_2$  is an equivalent model to  $G_1$ . If we answer the query  $p(z|y)$  on model  $G_2$ , we do not need to consider  $x$ , since  $y$  (and consequently  $z$ ) no longer depends on  $x$ .

Through this example, we see that it may be possible to find an equivalent graph to our original model, and to use the new graph structure to answer certain queries more efficiently. It is important to note that the efficiency of the new model for answering a query relies both on the conditional probabilities we are interested in, as well as the content of our query (as opposed to an arbitrary query).

## 1.5 Related work

Significant work has been done on individual aspects of the subject, though this thesis pioneers in using them together for fast inference. Verma and Pearl [1] give a framework for understanding how the same probability distribution can be encoded into two distinct graphs, as well as present a simplified criterion for when two graphs describe indistinguishable distributions. Flesch and Lucas [2] present a survey-style overview of Markov Equivalent graph structures, reduction of graph structures to simplified representations of equivalent distributions, as well as providing a strong background of relevant Graph Theory and Bayesian Statistics. Chickering [3] explores parameters of equivalent networks, presents a

procedure for altering graphs without changing the described distribution, and quantifies the complexity of several such manipulations. Andersson et. al. [4] make similar headway by exploring reductions of graphs to representative form, as well as describing procedures by which one can alter the structure.

## 1.6 Outline

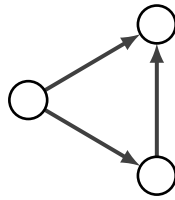
In Chapter 2, we introduce preliminary concepts from Graph Theory and Bayesian statistics to robustly define our models, and to aid our understanding of graph manipulations and the process of querying over a graph. Chapter 3 details the problem and proposed querying method, then aims to quantify the computational costs of answering an arbitrary query over a graph by this method. Chapter 4 builds context for understanding Markov equivalence between graphs. In Chapter 5, we examine the circumstances under which a Markov equivalence can be utilized to answer a query more efficiently, and then quantify the computational costs of finding a suitable Markov equivalent graph for faster inference. Finally, in Chapter 6, we compare the quantities explored in Chapters 3 and 5 to determine whether our proposed algorithm indeed increases inference speed, and if so, where it is effective.

# Chapter 2

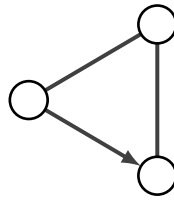
## Preliminaries

### 2.1 Foundations of Graph Theory

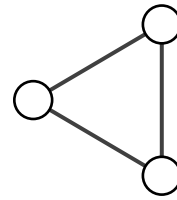
**Definition 2.1.1 (Graph (directed, undirected, mixed)).** A **graph** is defined as a pair  $G = (V, E)$  in which  $V$  is a finite set of vertices, and  $E \subset V \times V$  is a finite set of edges. Vertices (also sometimes called **nodes**) will generally be denoted by capital letters, i.e.  $X, Y, Z$ .  $G$  is called **undirected** if every edge in  $G$  is undirected, meaning that for each edge  $(X, Y) \in E$  (denoting an edge from vertex  $X$  to vertex  $Y$ ), the edge  $(Y, X)$  is also in  $E$ , for  $X \neq Y$ . Otherwise,  $G$  is called **directed**. We denote such undirected edges by  $\{X, Y\}$  or equivalently  $\{Y, X\}$ . A **mixed graph** (also called a hybrid graph) is a graph  $G$  which contains both directed and undirected edges.



directed



mixed

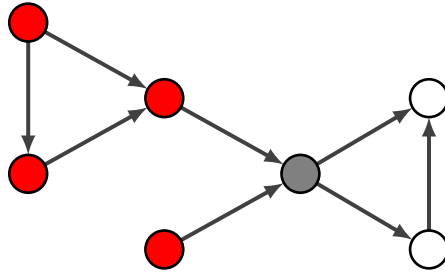


undirected

**Definition 2.1.2 (Route).** A **route** in a graph  $G = (V, E)$  is a sequence  $X_1, X_2, \dots, X_k$  of vertices in  $V$  such that there is an edge connecting  $X_i$  to  $X_{i+1}$  (independent of the direction of the edge), for  $i = 1, \dots, k - 1, k \geq 1$ . The integer  $k$  is the length of the route.

**Definition 2.1.3 (Path, directed cycle).** A (directed) **path** in a directed graph  $G = (V, E)$  is a route where vertices  $X_i$  and  $X_{i+1}$  are connected by a directed edge  $(X_i, X_{i+1})$ . A directed path which begins and ends at the same vertex is called a **directed cycle**.

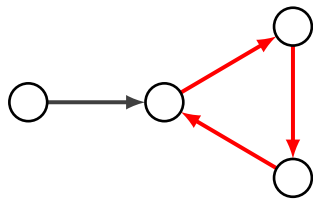
**Definition 2.1.4 (Parent, ancestor).** Given two vertices  $X, Y \in V$  in a directed graph  $G = (V, E)$ ,  $Y$  is called a **parent** of  $X$  if the edge  $(Y, X) \in E$ . The set of parents of a vertex  $X$  in a graph  $G = (V, E)$  is denoted  $\Pi_X^G$ . Likewise,  $X$  is called a **child** of  $Y$ . The set of ancestors of  $X$ , denoted  $\alpha(X)$ , is the set of all vertices  $Y$  such that there exists a directed path from  $Y$  to  $X$ , but no path from  $X$  to  $Y$ . In the context of this thesis, this is the set of vertices upon which a vertex  $X$  depends, either directly or indirectly. Likewise, if  $Y$  is an ancestor of  $X$ , then  $X$  is a **descendant** of  $Y$ .



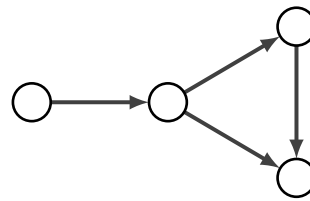
Ancestors of the gray vertex shown in red

**Remark.** Trivially,  $\Pi_X^G \subseteq \alpha(X)$ .

**Definition 2.1.5 (Chain graph, directed acyclic graph (DAG)).** A mixed graph  $G = (V, E)$  is called a **chain graph** if it contains no directed cycles. A **directed acyclic graph** is a chain graph which is directed. This is abbreviated as DAG. The directed cycle is shown below in red.



not a DAG



DAG

**Remark.** Every undirected graph is a chain graph.

**Definition 2.1.6 (Neighbor).** Given two vertices  $X, Y \in V$  in an undirected graph  $G = (V, E)$ ,  $X$  is called a **neighbor** of  $Y$  if  $\{X, Y\} \in E$ .



## 2.2 Foundations of Bayesian Statistics

**Definition 2.2.1 (Bayesian Network).** *A Bayesian network is a pair  $B = (G, P)$  where  $G = (V, E)$  is a DAG and  $P$  is a joint probability distribution defined on a set of random variables  $X$ .*

*Vertices in  $G$  have a one-to-one correspondence to the random variables  $X$  associated with  $P$ , meaning the set  $V$  indistinguishably represents both the vertices of  $G$  and the random variables of the joint probability distribution  $P$  of  $B$ .*

# Chapter 3

## Querying

### 3.1 Outline of the Task

Given a Bayesian network  $B$  and a query, our goal is to find a MEC equivalent graph  $B'$  such that the number of nodes which must be computed is minimized for the same query over  $B'$ . By minimizing the number of nodes which must be computed, we can speed up the querying process to achieve faster inference.

Therefore, the first task outlined by our goal is to quantify the number of nodes which must be considered for a given query over  $B$ . This will serve as our cost function; the minimization will therefore be to reduce the cost, that is, find  $B'$  such that this number of nodes is minimized.

Naturally, before we consider the minimization itself, we begin by identifying such a set of nodes. This chapter will focus on identifying this set of nodes which must be computed for a given query.

### 3.2 Structure of a Query

A query is an inference question which asks the probability of one or multiple variables given some observations. Queries can take advantage of multiple observations and have multiple targets. For example, one can ask for the probability of a variable  $X$  given that we have observed  $Y$ . Let  $x$  be a value attained by the random variable  $X$ , and let  $y$  be a value attained by the variable  $Y$ , and  $z$  a value attained by  $Z$ . Then, queries upon this data may have any of the following forms:

Scenario	Query	Interpretation	Example
Query a single target, single observation	$p(X = x Z = z)$	Probability that $X = x$ given that we have observed $Z = z$ .	Probability that a patient has lung cancer given that they are a tobacco smoker.
Query with multiple targets, single observation	$p(X = x, Y = y Z = z)$	Probability that $X = x$ and $Y = y$ given that we have observed $Z = z$ .	Probability that a patient has lung cancer and high blood pressure given that they are a tobacco smoker.
Query with a single target, multiple observations	$p(X = x W = w, Z = z)$	Probability that $X = x$ given that we have observed $W = w$ and $Z = z$ .	Probability that a patient has lung cancer given that they are over 50 years of age and a tobacco smoker

**Theorem 3.2.1.** *Computing a query  $p(X_0, X_1, \dots, X_n | Y_0, Y_1, \dots, Y_m)$  depends on the sets of ancestors  $\alpha(X_0), \alpha(X_1), \dots, \alpha(X_n)$  as well as all nodes on the paths between  $X_i$  and  $Y_j$  for  $i \in [0, n], j \in [0, m]$ .*

*Proof.*

□

Let  $G$  be a DAG and  $q$  a query. Define  $\delta(G, q)$  be the number of nodes involved in the computation of  $q$  on  $G$ . Then our goal is to find  $\arg \min_{G' \in [G]} (\Delta(G, q))$ .

# Chapter 4

## Markov Equivalence

### 4.1 Goal

In order to reduce the cost of a query on a graph  $G$ , we will exploit the observation that two graphs with distinguishable structures can model the same probability distribution. Such graphs are called *Markov equivalent*. Given a query on a graph  $G$ , we search for a Markov equivalent graph  $G'$  such that the cost of the query is minimized when considered on  $G'$  instead of  $G$ .

To do so, we must develop a robust understanding of Markov equivalence. This includes: how to identify whether two graphs are Markov equivalent, how a graph can be altered while remaining in the set of Markov equivalent graphs (called the *Markov equivalence class*), how computations are altered when the graph structure is altered, and how to search for minimizing graphs. In this chapter, we present definitions, theorems, and examples to establish a sufficient understanding of Markov equivalence for this purpose.

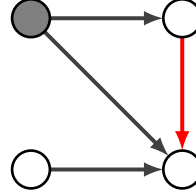
### 4.2 Basic Properties of Markov Equivalence

**Definition 4.2.1 (Markov equivalence [1]).** Two directed acyclic graphs  $G$  and  $G'$  are *Markov equivalent* if for every Bayesian network  $B = (G, \theta_G)$ , there exists a Bayesian network  $B' = (G', \theta_{G'})$  such that  $B$  and  $B'$  describe the same probability distribution, and vice versa. This is denoted  $G \sim_M G'$ .

**Definition 4.2.2 (Markov equivalence class [1]).** The set of all DAGs which are Markov equivalent to a DAG  $G$  is called the *Markov equivalence class* (MEC) of  $G$ ,

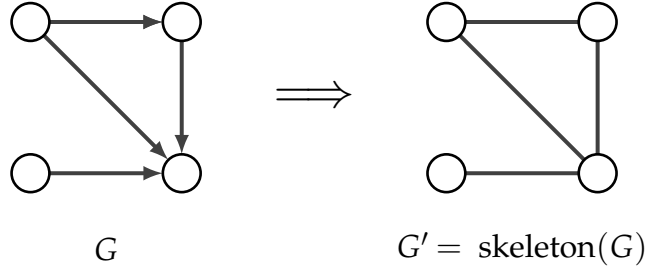
denoted  $[G]$ .

**Definition 4.2.3 (Covered Edge [3]).** Given a DAG  $G = (V, E)$ , an edge  $e = (X, Y) \in E$  is called **covered** in  $G$  iff  $\Pi_X^G = \Pi_Y^G \cup Y$ . That is,  $(X, Y)$  is covered in  $G$  iff  $X$  and  $Y$  have identical parents in  $G$  with the exception that  $X$  is not a parent of itself.

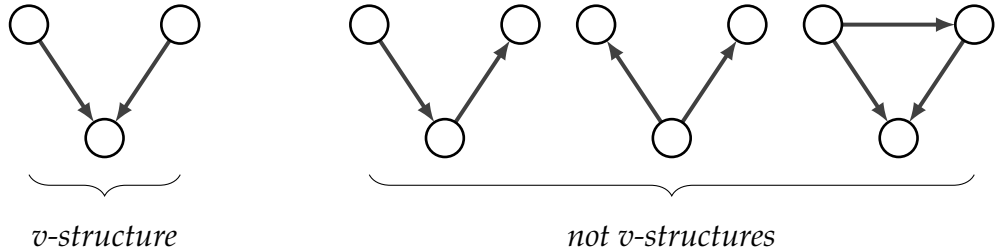


Here, for example, the red edge is the only covered edge, since the nodes it connects have a shared set of parents, namely, the single gray node.

**Definition 4.2.4 (Skeleton).** Let  $G = (V, E)$  be a directed graph. The undirected graph  $G' = (V, E')$  which is constructed by removing the orientation of all edges in  $E$  is called the **skeleton** of  $G$ .



**Definition 4.2.5 (v-structure).** A set of three vertices  $X, Y, Z$  of a graph  $G = (V, E)$  is called a **v-structure** (or *immorality*) iff  $(X, Y) \in E$  and  $(Z, Y) \in E$  but  $X$  and  $Z$  are not adjacent.



**Definition 4.2.6 (Irreversible edge [4]).** Let  $G$  be a DAG. A directed edge  $(X, Y) \in G$  is called **irreversible** in  $G$  if changing  $(X, Y)$  to  $(Y, X)$  either creates or destroys a v-structure, or creates a directed cycle. An edge which is not irreversible is called **reversible**.

**Lemma 4.2.1 (Criterion for MEC-equivalence [5]).** *Two graphs are equivalent iff they have the same skeleton and v-structures.*

*Proof.* The intuition for this Actually this should be cited to the 1989 paper by Pearl et al, (Verma, Geiger)?OR wait, Olmsted? Looks like the real proof is in Smith 1987, Influence Diagrams for Statistical Modelling.  $\square$

**Lemma 4.2.2 (Covered edges are exactly reversible edges [3]).** *Let  $G = (V, E)$  be a DAG, let  $X, Y \in V$  and let  $e = (X, Y) \in E$ . Let  $G' = (V, E')$  be the DAG constructed from  $G$  by reversing the edge  $(X, Y)$  to  $(Y, X)$ . Then  $G$  and  $G'$  are equivalent iff  $e$  is a covered edge in  $G$ .*

*Proof.* Let  $G = (V, E)$  be a DAG. A sketch of the proof is as follows:

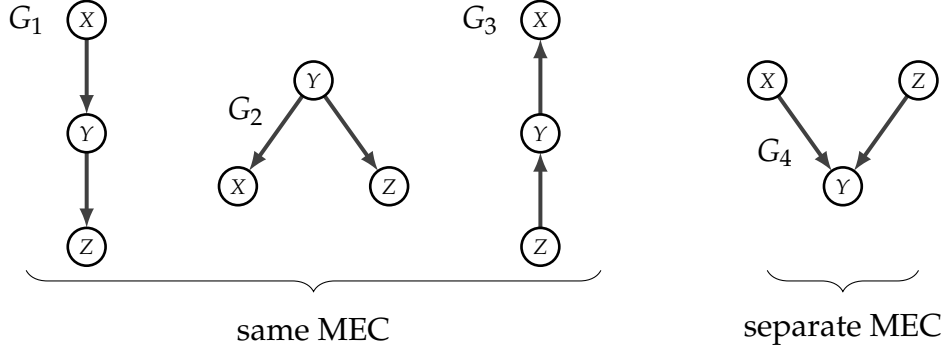
(if) Suppose  $e = (X, Y)$  is a covered edge in  $G$ . Let  $G'$  be equivalent to  $G$ , except that  $e$  is reversed to  $e' = (Y, X)$ . Then:

- $G'$  is also a DAG. To show this, suppose that  $G'$  is not a DAG. Since  $G$  and  $G'$  only differ by  $e' = (Y, X)$ , there must be a directed cycle including  $e'$  in  $G'$ . Then there is a path in  $G$  from  $Y$  to  $X$ . By assumption,  $Y$  and  $X$  have a shared set of parents. However, one can conclude from this that  $G$  also contains a cycle, and therefore is not a DAG, a contradiction. By this contradiction, we conclude that  $G'$  must be a DAG.
- $G' \sim_M G$ . Firstly,  $G$  and  $G'$  trivially have the same skeleton, since they only differ by the orientation of edge  $e$ . Then, if they were *not* Markov equivalent graphs, they would differ by a v-structure. If  $G'$  has a v-structure not present in  $G$ , it must include the edge  $(Y, X)$ . However, this would imply that  $X$  has a parent which is not a neighbor to  $Y$ , contradicting the assumption that  $e$  is covered. A similar argument holds for the scenario where  $G$  has a v-structure not present in  $G'$ .

(only if) We now show that if  $e = (X, Y)$  is not a covered edge, we are in one of two cases: either  $G'$  contains a directed cycle, or  $G'$  is a DAG which is not equivalent to  $G$ . Both of these cases require that one of the nodes  $X$  or  $Y$  has a parent that the other does not. If there is a node  $Z$  which is a parent to  $Y$  but not  $X$ , we can quickly derive the presence of a directed cycle in  $G'$ . By similar argument, if  $Z$  is a parent to  $X$  but not  $Y$ , then  $G$  would contain a directed cycle. By contradiction, neither case is possible.

$\square$

**Example 4.2.1 (Members of Markov equivalence class [1]).** Consider the following four DAGs:



The fact that  $G_1, G_2$ , and  $G_3$  are in the same MEC while  $G_4$  is not can be seen by considering the joint probability distribution  $p(X, Y, Z)$  of each of the above graphs.

$$G_1 \quad p(X, Y, Z) = p(X) \cdot p(Y|X) \cdot p(Z|Y)$$

$$G_2 \quad p(X, Y, Z) = p(Y) \cdot p(X|Y) \cdot p(Z|Y)$$

$$G_3 \quad p(X, Y, Z) = p(Z) \cdot p(Y|Z) \cdot p(X|Y)$$

By Bayes' theorem, all of the values of  $G_2$  can be completely determined from values from  $G_1$ :

$$p(X)p(Y|X) = p(X, Y) = p(Y)p(X|Y)$$

and  $p(Z|Y)$  is unchanged. A similar application of Bayes' theorem shows that  $G_3$  is completely determined by  $G_1$  and  $G_2$ :

$$p(z) \cdot p(Y|Z) = p(Z, Y) = p(Y) \cdot p(Z|Y)$$

where  $p(Y)$  can be attained directly from  $G_2$  and  $p(Z|Y)$  can be attained from  $G_1$ . Therefore, we can conclude that since all three describe the same distribution, they lie in the same equivalence class. In contrast, the joint probability for  $G_4$  is given by

$$p(X, Y, Z) = p(X) \cdot p(Z) \cdot p(Y|X, Z)$$

which can not be determined from the values of  $G_1, G_2$  and  $G_3$ . This is because in  $[G_1]$  (but not in  $G_2$  and  $G_3$ ),  $X$  is conditionally independent from  $Y$  and  $Z$ , that is,  $X$  is only independent in the circumstance that we are given values for  $Y$  and  $Z$ . Meanwhile, in  $G_4$ ,  $X$  is marginally independent of  $Z$ , that is, independent in the circumstance that we ignore  $Y$ . Conditional independence does not

provide information about marginal independence, and conversely, marginal independence does not imply conditional independence. Therefore,  $G_4$  describes a different distribution, and does not lie in  $[G_1]$ .

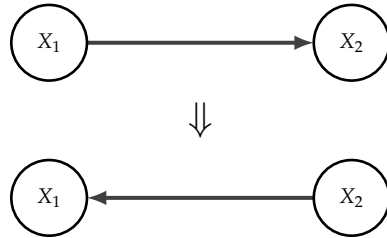
**Corollary 1.** *Lemma 4.2.2 and Lemma 4.2.1 together imply that the graph  $G'$  which results from only reversing the orientation of a covered edge  $(X, Y) \in G$  is in the MEC of  $[G]$ .*

*Proof.* This follows from the fact that edge reversal does not change the skeleton of the graph  $G$  (since no edges are added nor removed) and reversing covered edges does not alter the v-structures of  $G$  ( Lemma 4.2.2), thus the v-structures and skeleton are unchanged.  $\square$

### 4.3 Effects of Edge Reversal

**Lemma 4.3.1.** *Given a single directed edge between two parentless nodes in which  $p(X_2) = p(X_1) \cdot p(X_2|X_1)$ , the joint probability after switching edge orientation is given by*

$$p(X_1) = p(X_2) \frac{p(X_2|X_1) \cdot p(X_1)}{\sum_{X_1} p(X_2|X_1) \cdot p(X_1)}.$$



*Proof.* This follows directly from Bayes' theorem, which gives us

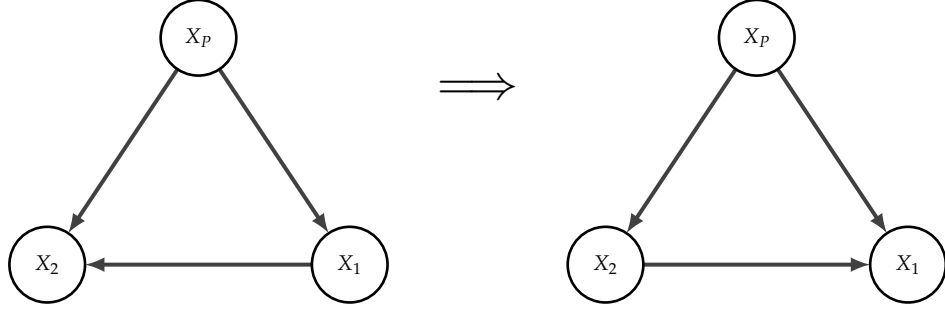
$$\begin{aligned}
 p(X_1) &= p(X_2) \cdot p(X_1|X_2) \\
 &= p(X_2) \frac{p(X_2|X_1) \cdot p(X_2)}{p(X_2)} \\
 &= p(X_2) \frac{p(X_2|X_2) \cdot p(X_2)}{\sum_{X_1} p(X_2|X_1) \cdot p(X_1)}.
 \end{aligned}$$

$\square$

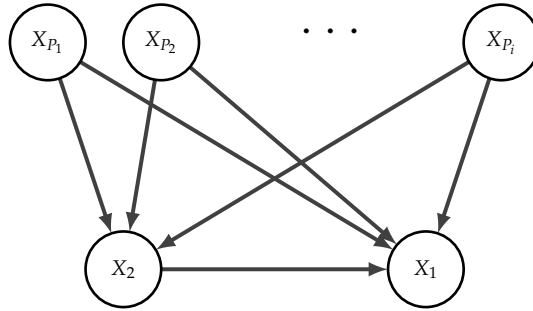


**Lemma 4.3.2.** Given a directed edge between two nodes  $X_1$  and  $X_2$  with a shared parent node  $X_P$ , the joint probability after switching edge  $(X_1, X_2)$  to  $(X_2, X_1)$  is given by

$$p(X_1, X_2, X_P) = p(X_P) \cdot p(X_2|X_P) \cdot p(X_1|X_2, X_P) = \dots$$



**Remark.** Furthermore, because this result does not depend on  $p(X_P)$ , we can conclude that for a set of shared parents  $P = \{X_{P_1}, X_{P_2}, \dots, X_{P_i}\}$  of two nodes  $X_1$  and  $X_2$ , the same equality holds. Additionally, note that any reversal of an edge with a set of shared parents is MEC-invariant, since all such edges are covered (corollary 1).



*Proof.* From before the edge reversal, we have access to the values  $p(X_P)$ ,  $p(X_1|X_P)$ , and  $p(X_2|X_1, X_P)$ . Our goal is to determine the new joint probability distribution  $p(X_1, X_2, X_P)$  using this information. Therefore, we must find  $p(X_P)$ ,  $p(X_2|X_P)$ , and  $p(X_1|X_2, X_P)$ . First,  $p(X_P)$  is already trivially available. Then, to compute  $p(X_1|X_1, X_P)$ ,

$$\begin{aligned}
p(X_1|X_2, X_p) &= \frac{p(X_1, X_2, X_p)}{p(X_2, X_p)} \\
&= \frac{p(X_2|X_1, X_p) \cdot p(X_1, X_p)}{p(X_2, X_p)} \\
&= \frac{p(X_2|X_1, X_p) \cdot p(X_1, X_p)}{p(X_2, X_p)} \\
&= \frac{p(X_2|X_1, X_p) \cdot p(X_1|X_p) \cdot p(X_p)}{p(X_2|X_1, X_p)} \\
&= \frac{p(X_2|X_1, X_p) \cdot p(X_1|X_p)}{p(X_2|X_p)} \\
&= \frac{p(X_2|X_1, X_p) \cdot p(X_1|X_p)}{p(X_2|X_p)}
\end{aligned}$$

which depends only on known quantities once we compute

$$\begin{aligned}
p(X_2|X_p) &= \sum_{X_1} p(X_1|X_p) \cdot p(X_2|X_1, X_p) \\
&= \frac{1}{p(X_p)} \sum_{X_1} p(X_1, X_p) \frac{p(X_2, X_1, X_p)}{p(X_1, X_p)} \\
&= \frac{1}{p(X_p)} \sum_{X_1} p(X_2, X_1, X_p) \\
&= \sum_{X_1} p(X_2, X_p)
\end{aligned}$$

TO BE CONT.

□

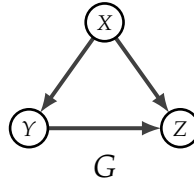
## 4.4 Find Reversible Edges of a DAG Model

**Definition 4.4.1 (Essential edge [4] [2] ).** An *essential edge* (also called a *compelled edge* [3]) of a graph  $G = (V, E)$  is an edge  $(X, Y) = e \in E$  such that for every graph  $G' = (V, E')$  in  $[G]$ , the orientation of  $e \in E$  is unchanged. That is, there is no graph  $G' = (V, E')$  in  $[G]$  such that  $(Y, X) \in E'$ .

**Definition 4.4.2 (Essential graph [4]).** The *essential graph* of a Markov equivalence class  $[G]$  is a mixed graph  $G_*$  such that the only directed edges in  $G_*$  are essential edges of  $[G]$ . That is,

- The directed edge  $e = (X, Y) \in G_*$  iff  $e \in G$  for every graph  $G \in [G]$ ,
- The undirected edge  $X \sim Y \in G_*$  iff there are graphs  $G_1$  and  $G_2 \in [G]$  such that  $(u, v)$  in  $G_1$  and  $(Y, X)$  in  $G_2$ .

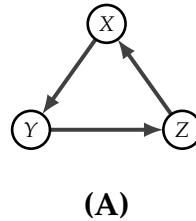
**Example 4.4.1.** In this example, we identify the essential edges of a Markov equivalence class and construct the essential graph. Consider the following DAG  $G = (V, E)$ :



We use the following realizations to show that the graph has no essential edges. If we switch any edge in  $G$ , the resulting graph  $G'$  becomes one of two cases:

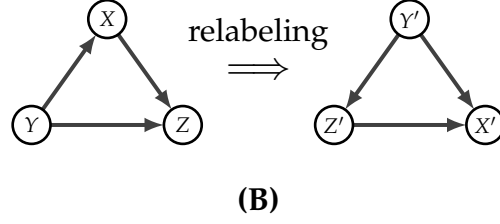
- $G'$  is cyclic, and therefore no longer a DAG, meaning this edge was irreversible.

Example: reversing edge  $(X, Z)$  as in **(A)** creates a cycle.

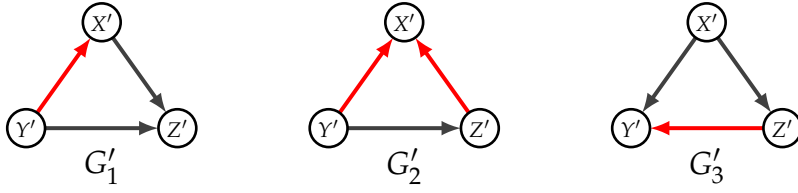


- $G'$  is equivalent to  $G$  up to relabeling, and therefore in the same MEC.

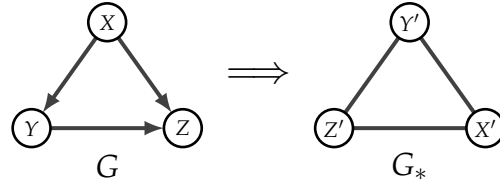
Example: reversing  $(X, Y)$  and changing labels according to  $X \rightarrow Y', Y \rightarrow Z'$ , and  $Z \rightarrow X'$ , as in **(B)**.



So the following graphs are included in  $[G]$ , though they are not the entire MEC. The red edges in each are those with orientation which differs from edges in  $G$ . Notice, however, that some relabeling was required, meaning these edges do not necessarily correspond (under this new labeling) to covered edges.



Therefore, under some circumstance, for every edge  $(U, V) = e \in G$ , there is a DAG  $G' \in [G]$  such that  $(V, U) \in G'$ . This means that  $G$  has no essential edges. We construct the essential graph  $G_*$  of  $G$  by removing the orientation of all non-essential edges in  $G$ :



**Remark.** Andersson et. al. [4] demonstrate that essential graphs are consistent across a MEC and unique to it, meaning an essential graph can be used as a representative for a given MEC. Furthermore, essential graphs are not necessarily DAGs, and therefore do not generally belong to the MEC that they represent.

This can be seen by the fact that the distinction between two MECs lies entirely in either differences in their skeletons or in the orientations of their compelled edges.

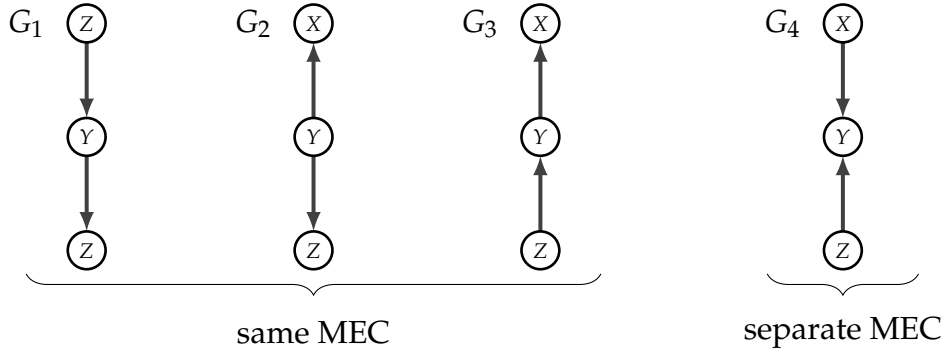
*“It is only in the heart that one can see rightly; what is essential is invisible to the eye.”*  
*-Antoine de Saint-Exupéry, The Little Prince.*

**Remark.** One should take care to note the distinction between implications about non-essential edges versus covered edges:

- For  $G = (V, E)$ , an edge  $e \in E$  is non-essential iff there exists a graph  $G'$  in  $[G]$  such that  $e$  is reversed,
- An edge  $e$  in  $G$  is covered iff the graph  $G'$  derived from *only* reversing  $e$ , is in the MEC of  $G$ . (Follows from corollary 1.)

That is, reversing  $e$  *exclusively* while remaining in the MEC of  $G$  requires  $e$  to be covered. Meanwhile, a non-essential edge  $e \in G$  may require other edges in  $G$  to be reversed as well to remain in the same MEC. This means that not all non-essential edges can be reversed at a given time. The following example demonstrates that some configurations of non-essential edge reversals are not possible if we wish to remain in the MEC of  $G$ .

**Example 4.4.2.** Again consider members of the Markov equivalence class seen in example 4.2.1. The positions of the nodes are visually rearranged for clarity, but no properties are changed.



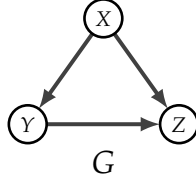
Notice that edge  $(Y, Z) \in G_1$  is non-essential, since there exists another graph (namely  $G_3$ ) in  $[G_1]$  such that  $(Z, Y) \in G_3$ . Therefore, we know that there exists some combination of edge reversals such that  $(Y, Z)$  can be reversed without leaving  $[G_1]$ . However, if we choose to switch only  $(Y, Z)$  to  $(Z, Y)$ , we have exactly graph  $G_4$ , which is no longer in the same MEC. Therefore, the fact that an edge is non-essential does not provide information about the circumstances in which the edge can be reversed without altering the MEC; only that in some context, it can be. This realization leads us to corollary 2.

**Corollary 2.** Let  $E_c$  be the set of covered edges in  $G$ , and let  $E_e$  be the set of essential edges in  $G$ . Then  $E_c \subseteq \bar{E}_e$ , the set of non-essential edges in  $G$ .

*Proof.* This follows directly from the realization that non-essential edges can be switched while remaining in the same MEC under the correct conditions, while

covered edges can be switched immediately, without regard to other edge conditions (the conditions for their reversal are always met).  $\square$

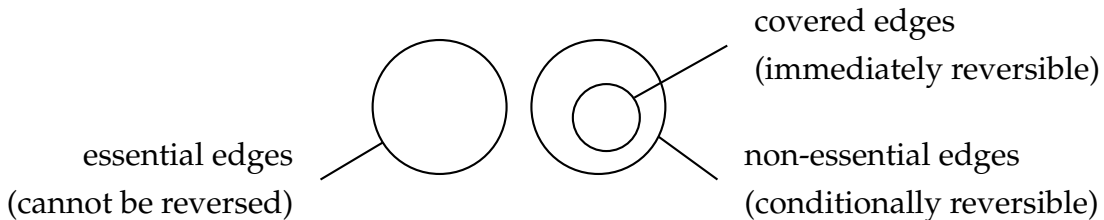
**Example 4.4.3.** We use the graph  $G$  from example 4.4.1 to demonstrate corollary 2. Of course, this is not sufficient for a proof, but only helps us contextualize the lemma.



In example 4.4.1 we determined that all edges  $e_i \in E$  of  $G = (V, E)$  are non-essential, so the set of non-essential edges  $E_G^N = \{(X, Y), (Y, Z), (X, Z)\}$ . We can quickly see that the only covered edge of  $G$  is  $(Y, Z)$  since  $Y$  and  $Z$  share a single parent  $X$ , not including  $Z$  itself. Then, indeed,  $\{(Y, Z)\} \subset E_G^N$ .

**Remark.** There are two scenarios we consider for finding reversible edges, that is, edges which can be reversed without altering the MEC of the graph:

- Firstly, we may wish to find the set of edges which can be immediately reversed in one step, without regard to other edges in the graph. This coincides with the set of covered edges.
- Secondly, we may wish to find the set of edges which can ever be reversed within a graph, under the correct conditions of other edges being reversed beforehand/simultaneously. This coincides with the set of non-essential edges.



**Theorem 4.4.1 (Edge reversal sequence. [3]).** *Given two Markov equivalent DAGs  $G$  and  $G'$ , there exists a sequence of distinct edge reversals in  $G$  with the following properties:*

- Each edge reversed in  $G$  is a covered edge,
- After each reversal,  $G$  is a DAG and  $G \sim_M G'$
- After all reversals,  $G = G'$ .

That is, we can transform  $G$  into  $G'$  via a finite sequence of covered edge reversals, such that all of the intermittently resulting graphs  $G_1, G_2, \dots, G_n \sim_M G, G'$ .

*Proof.* Chickering's [3] proof defines the procedure **Procedure Find-Edge** which takes a DAG  $G$  as input. At each step, the procedure identifies a next edge to reverse. To be reversible, such an edge must be covered, by Lemma 4.2.2. The proof demonstrates that such an edge is identifiable when it exists, and each reversal reduces the number of edges in total which must be reversed, eventually transforming  $G$  into  $G'$ .  $\square$

# Chapter 5

## Exploiting Markov Equivalence using Greedy Strategy

### 5.1 Motivation

Now that we have established a cost function

One complication of considering Markov equivalence is that Markov equivalence classes can be quite large. None of the graphs in the MEC is necessarily a more natural representation of the probabilistic dependencies than the others.

We therefore begin by exploring a *greedy* strategy. The idea behind this strategy takes advantage of Theorem 4.4.1:

*Given two arbitrary graphs  $G$  and  $G'$  in a family of Markov equivalent graphs,  $G$  can be transformed into  $G'$  through a finite sequence of transformations such that 1) each step in the sequence involves reversing a single directed edge, and 2) after each edge reversal, the resulting graph is contained within the same Markov equivalence class.*

This result allows us to develop the greedy strategy, an algorithm in which each step entails searching for a single edge that can be reversed without leaving the original Markov equivalence class, and simultaneously reduces the number of variables which must be sampled to answer our original query.

There are two aspects we wish to consider while searching for a reversible edge which reduces the cost of answering a query:



- Does reversing the edge alter existing v-structures in the graph, either by creating a new v-structure or destroying an existing one?
- Does switching the edge indeed benefit our query, ie, reduce the number of variables that must be considered while computing the probabilities?

((((A FEW CHANGES to make here, namely, decide which node to begin the search from? Or I can change the algorithm to simply iterate through edges. In that case, it is harder to explicitly state the method by which I want to traverse the graph. If I choose to do that, (which I likely will) I will probably have to use a topological sort, and introduce what that is. Likely useful anyway.))))

---

**Algorithm 1:** COVERED EDGES returns a set of covered edges in a DAG

---

**Input:** A DAG  $G$

**Output:** The set of covered edges in  $G$

```

1  $X = X_0$ 
2  $queue = [X]$ 
3  $visited = [X]$ 
4  $covered = []$ 
5 while length of the queue  $\neq 0$  do
6   For  $p_i \in \prod_X^G$ 
7     if  $\prod_{p_i}^G \setminus X = \prod_X^G$  then
8        $covered.append((X, p_i))$ 
9       iterate
10    else
11      iterate
12 end
13 return  $covered$ 

```

---

where iteration entails

---

**Algorithm 2:** BREADTH-FIRST ITERATION

---

```

1  $append\ queue[0]\ to\ visited$ 
2  $X = queue[0]$ 
3  $remove\ queue[0]\ from\ queue$ 

```

---

## **5.2 Implementation**

## **5.3 Cost**

# Chapter 6

## Comparison of methods

A comparison of the speed of normal querying VS the speed of the new query + time required to exploit ME.

Ultimately trying to answer the questions:

- Can we do faster inference?
- Under what circumstances is it possible/effective?
- How much faster is it?
- Maybe include a section on future work in this chapter.

# Bibliography

- [1] T. Verma and J. Pearl, "Equivalence and synthesis of causal models," *Proceeding of the Sixth Annual Conference on Unverstainty in Artificial Intelligence, UAI '90*, 2004. DOI: 10.1142/S0218488504002576.
- [2] I. Flesch and P. J. Lucas, "Markov equivalence in bayesian-network structures," *J. Mach. Learn. Res.*, 2, 2002.
- [3] D. M. Chickering, "A transformational characterization of equivalent bayesian network structures," *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence UAI'95*, 1995. Morgan Kaufmann Publishers Inc.
- [4] S. A. D. Madigan and M. Perlman, "A characterization of markov equivalence classes for acyclic digraphs," *Annals of Statistics* 25, 1997.
- [5] J. P. D. Geiger and T. Verma, "The logic of influence diagrams," *Influence Diagrams, Belief Netowkrs and Decision Analysis*, 1989. John Wiley and Sons, Ltd., sussex, England, 1990.

## Declaration of Authorship

I hereby declare that I have completed this work independently and using only the sources and tools specified. The author has no objection to making the present Master's thesis available for public use in the university archive.

Ich erkläre, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe. Seitens des Verfassers bestehen keine Einwände die vorliegende Masterarbeit für die öffentliche Benutzung im Universitätsarchiv zur Verfügung zu stellen.

Ort, Abgabedatum  
(Place, date)

Unterschrift der Verfasserin  
(Signature of the Author)