IDENTIFYING STRUCTURE AND SEMANTICS IN BAYESIAN NETWORK INFERENCE

A Thesis

Submitted to the Faculty of Graduate Studies and Research

In Partial Fulfilment of the Requirements

For the Degree of

Doctor of Philosophy

in

Computer Science

University of Regina

By

Wen Yan

Regina, Saskatchewan

June, 2013

UNIVERSITY OF REGINA

FACULTY OF GRADUATE STUDIES AND RESEARCH

SUPERVISORY AND EXAMINING COMMITTEE

Wen Yan, candidate for the degree of Doctor of Philosophy in Computer Science**,** has presented a thesis titled, ***Identifying Structure and Semantics in Bayesian Network inference,*** in an oral examination held on May 28, 2013. The following committee members have found the thesis acceptable in form and content, and that the candidate demonstrated satisfactory knowledge of the subject material.

External Examiner:	Dr. Yang Xiang, University of Guelph
Supervisor:	Dr. Cortney J. Butz, Department of Computer Science
Committee Member:	Dr. Malek Mouhoub, Department of Computer Science
Committee Member:	*Dr. Yiyu Yao, Department of Computer Science
Committee Member:	Dr. Yang Zhao, Department of Mathematics & Statistics
Chair of Defense:	Dr. Dongyan Blachford, Faculty of Graduate Studies & Research

*Not present but submitted questions

Abstract

Bayesian networks are a semantic modeling tool for managing uncertainty in complex domains. While the numerous techniques for exact inference vary when they apply the multiplication and marginalization (addition) operators, they all center around eliminating variables. These inference algorithms start and end with clear structure and semantics, yet all intermediate distributions, whether normalized or unnormalized, are denoted as potentials.

In this thesis, we reveal the structure and semantics of the intermediate factors constructed during exact inference in discrete Bayesian networks. Our discussion is primarily based on Variable Elimination (VE), which is a standard approach to Bayesian network inference. We first show that, when evidence is not considered, every multiplication and every addition in VE yields a conditional probability table (CPT).

As for semantics, we present several techniques to establish whether or not the CPT generated by VE using the given Bayesian network CPTs is the same as if it was constructed by brute force from the joint probability distribution defined by the given Bayesian network. The culminating result in this thesis is the Semantics in Inference (SI) algorithm. Four important properties of SI are shown, including soundness, completeness, strong completeness, and polynomial

time complexity.

This work is important in several ways. First, we have removed potentials from any discussion on exact inference in discrete Bayesian network. Potentials do not have clear physical interpretation, as they are unnormalized probability distributions. In contrast, CPTs have clear semantic meaning.

In addition, we have revealed that d-separation plays an important role in understanding semantics inference. Pearl, the founder of Bayesian networks, emphasizes the importance of d-separation with respect to Bayesian network modeling. With respect to inference, Pearl only states that d-separation can determine the minimum information needed for answering a query posed to a Bayesian network. No claim has ever been made that d-separation can also provide semantics during Bayesian network inference.

The above results may serve as a pedagogical aid to newcomers to the field, since Bayesian network inference is regarded as being hard to understand. Practical benefits of our findings are introduced, but primarily remain for future work.

Acknowledgements

Here I would like to show my appreciation to my supervisor, Dr. Cory Butz. He provided me with a motivating, enthusiastic and critical research atmosphere. He has always been there to listen and give advice. He taught me how to get an idea and how to express it. His style of energetic and earnest work deeply influenced me and is so beneficial to my research and my work. It is a great pleasure for me to conduct this thesis under his supervision. I thank him for his patience and encouragement that helped me through difficult time. Without his guidance, I would never finish this research project.

I would like to thank Dr. Pawan Lingras for his role in helping to understand structure in VE. In addition, I am grateful to Dr. Anders L. Madsen for his help in determining which independency to test when deciding semantics in VE.

I would also like to thank the Faculty of Graduate Studies and Research, the Department of Computer Science and Dr. Cory Butz for the financial support and computer facilities provided.

Post Defense Acknowledgements

I am greatly thankful to my external examiner Dr. Yang Xiang, internal examiner Dr. Malek Mouhoub, Dr. Yiyu Yao and internal-external examiner Dr. Yang Zhao. Their valuable feedback helped me to improve this thesis.

Dedication

I dedicate this thesis to my family and friends. I sincerely appreciate my parents, my sister and my wife. Without their encouragement and love, I could not maintain a high level of confidence and enthusiasm in my research. I would also like to thank Uncle Jiahua's family. They are always there to help me get through the difficulties.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Bayesian networks (BNs) [10, 15, 17, 33] provide a rigorous foundation for uncertainty management by combining probability theory and graph theory, and have been successfully applied in practice to a wide variety of problem domains. A discrete BN consists of a *directed acyclic graph* (DAG) and a corresponding set of *conditional probability tables* (CPTs). Uncertainty may be managed by letting the random variables in a problem domain be represented as vertices in a DAG, direct relationships between variables are qualitatively modeled with directed arcs in the DAG, and the strength of these relationships quantified by CPTs. Given a DAG on a variable set U, the probabilistic conditional independencies [39] graphically encoded in the DAG ensure that the product of BN CPTs is a *joint probability distribution* $p(U)$. By providing clear semantics in a modeling, BNs have been successfully applied in a wide variety of settings. Exact inference is a fundamental topic discussed in all BN textbooks, regardless of whether emphasis seems to be placed on exact inference [15, 33, 36], approxi-

mate inference [18], hybrid BNs [10, 17], recursive conditioning [11], multiagent reasoning [40], or bioinformatics [29].

While the many techniques for exact inference vary in when they apply the multiplication and marginalization (addition) operators, they all center around eliminating variables from the networks. The *sum-out* (SO) algorithm removes a variable v as a two-step process. First, the product of all distributions involving v is taken. Second, v is marginalized out from the obtained product. Other variables can be eliminated in a recursive manner. The probabilistic reasoning literature has always denoted the probability distributions constructed during BN inference as *potentials*. However, this description is not as precise as it should be.

Koller and Friedman [18] state that it is interesting to consider the semantics of the potentials constructed during inference. They mention that sometimes the probabilities are defined with respect to the joint distribution, but not at other times. As no practical algorithm exists for deciding the semantics of inference, all inference algorithms denote the intermediate factors constructed during inference as potentials. Potentials are unnormalized probability distributions [40] and have no constraints [18] meaning they do not have clear physical interpretation [8].

In this manuscript, the structure and semantics of intermediate distributions built in the exact inference in discrete BN are illustrated. We first show that every multiplication operation and every marginalization operation involved in eliminating variables from a discrete BN yields a CPT. The concept of *expanded form* is introduced to define each potential constructed by BN inference in terms of a sequence of multiplication and marginalization operators on the given BN

CPTs. We then establish that each expanded form can be equivalently rewritten in *normal form*, that is, as the marginalization of a product of BN CPTs. By applying our key observation, it is established that every distribution constructed by BN inference is indeed a CPT $\psi(X|Y)$, although ψ might not be p.

Our main result is an algorithm, called *Semantics in Inference* (SI) [5], that clearly specifies the semantics of every intermediate distribution constructed during exact inference in discrete BNs. SI works by introducing the notion of evidence normal form to organize how each potential was constructed, which follows the structure of the intermediate potential $\psi(X|Y)$. Then, SI utilizes the transitive closure [9] of the BN DAG to compute the ancestors and descendants of XS, where S are those variables eliminated in the evidence normal form. Finally, SI decides semantics of the potential by performing d-separation to test a specific conditional independency.

Moreover, formal properties of the SI algorithm are obtained, namely, polynomial time complexity, soundness, completeness, and strong completeness. The time complexity shows that denoting semantics during BN inference costs $O(n^3)$ instead of $O(n!)$ [5], which is the time complexity of the method to checking a specific condition proposed in [3], where n is the number of nodes in BN DAG. Soundness of the SI algorithm guarantees that if SI denotes the semantics of a potential ψ during BN inference as p, then $\psi(X|Y) = p(X|Y)$ [6]. The completeness of the SI algorithm states that whenever SI indicates that an intermediate distribution ψ is not p, then we can always find at least one BN instance where $\psi(X|Y) \neq p(X|Y)$ [6]. The strong completeness result of SI is especially impor-

tant, since it means that for nearly all instances of CPTs for the BN DAG, the SI algorithm correctly denotes the semantics of potentials constructed during exact inference [5].

Our work shows the important role d-separation plays in semantics of inference. Our research reveals that conditional independency is not only the most fundamental factor behind the organization of probabilistic knowledge and facilitating inference as traditionally stated, but is also crucial to the semantics of BN inference. Another advantage of using SI to denote BN inference is improved clarity. Pearl [33], the founder of BNs, opens the chapter on BNs by emphasizing that probabilistic reasoning is not about numbers - it is about the structure of reasoning. The SI algorithm reveals structure and semantics of reasoning. We use two recent textbooks [11, 18] on BNs to show how SI improves the discussion for the BN community itself.

This thesis is organized as follows. Chapter 2 reviews the pertinent notions of the probability theory, Bayesian networks, and some related concepts. We present the current limited understanding of semantics of BN inference in Chapter 3. In Chapter 4, the claim that every intermediate distribution during BN inference algorithms has CPT structure is established. Chapter 5 illustrates how to denote the semantics of the intermediate distributions during BN inference algorithms by using a topological order constraint and the proposed SI algorithm. We then establish four theoretical foundations of the SI algorithm, namely, polynomial time complexity, soundness, completeness, and strong completeness in Chapter 6. In Chapter 7, an alternative SI algorithm is illustrated and a deep look on the

alternation of semantics is presented. Also, the discussion of the advantages of this thesis work are presented in this charpter. Conclusions and future work are made in Chapter 8.

Chapter 2

Background Knowledge

In this chapter, we briefly review the pertinent notions of the probability theory, Bayesian networks, and some related concepts.

2.1 Probability Theory

Let $U = \{v_1, v_2, \ldots, v_m\}$ be a finite set of variables. Each variable v_i has a finite domain, denoted $dom(v_i)$, representing the values that v_i can take on. For a subset $X = \{v_i, \ldots, v_j\}$ of U, we write $dom(X)$ for the Cartesian product of the domains of the individual variables in X, namely, $dom(X) = dom(v_i) \times \ldots \times dom(v_j)$. Each element $c \in dom(X)$ is called a *configuration* of X. If c is a configuration on X and $Y \subseteq X$, then $c.Y$ denotes the restriction of c onto Y. As done in relational databases [26], I assume that there is a value λ such that $c.\emptyset = \lambda$ for any configuration c.

Definition 2.1.1 [36] *A* potential *on* $dom(X)$ *is a function* ψ *on* $dom(X)$ *such*

Table 2.1: Potentials $\psi(a, b)$ and $\psi(b, c)$.

a	b	$\psi(a, b)$	b	c	$\psi(b, c)$
0	0	0.2	0	0	0.6
0	1	1.8	0	1	0.4
1	0	0.0	1	0	0.5
1	1	1.6	1	1	0.5

that the following two conditions both hold: (i) $\psi(x) \geq 0$, for each configuration $x \in dom(X)$, and (ii) $\psi(x) > 0$, for at least one configuration $u \in dom(X)$.

For brevity, we refer to ψ as a potential on X rather than $dom(X)$, and we call X, not $dom(X)$, its domain [36]. Also, for simplified notation, we use XY to denote $X \cup Y$ in this thesis.

For example, let a, b be two binary variables. Potentials $\psi(a, b)$ is shown in Table 2.1.

Definition 2.1.2 *Let $\psi_1(X)$ and $\psi_2(Y)$ be two potentials. The product of $\psi_1(X)$ and $\psi_2(Y)$, denoted ψ_3, is defined as: for each configuration c of XY,*

$$\psi_3(c) \;=\; \psi_1(c.X) \cdot \psi_2(c.Y),$$

where $c.X$ is the configuration of X obtained by deleting the values of the variables in $Y - X$, and $c.Y$ is the configuration of Y obtained by deleting the values of the variables in $X - Y$.

Table 2.2: (i) The product of $\psi(a, b)$ and $\psi(b, c)$ in Table 2.1 yields $\psi(a, b, c)$. (ii) The marginalization of $\psi(a, b, c)$ onto $\{a, c\}$ yields $\psi(a, c)$.

a	b	c	$\psi(a, b, c)$
0	0	0	0.12
0	0	1	0.08
0	1	0	0.90
0	1	1	0.90
1	0	0	0.00
1	0	1	0.00
1	1	0	0.80
1	1	1	0.80

a	c	$\psi(a, c)$
0	0	1.02
0	1	0.98
1	0	0.80
1	1	0.80

For example, multiplying $\psi(a, b)$ and $\psi(b, c)$ in Table 2.1 yields the potential $\psi(a, b, c)$ in Table 2.2 (i).

Definition 2.1.3 *Given a potential $\psi(Z)$, let $X \subseteq Z$ and $Y = Z - X$. The marginal of $\psi(Z)$ onto X, denoted $\psi(X)$, is defined as: for each configuration x of X,*

$$\psi(x) \;=\; \sum_{y \in dom(Y)} \psi(x, y),$$

where x, y is the configuration of Z obtained by combining x with the configuration y of Y.

For example, the marginalization of potential $\psi(c, d, e)$ in Table 2.2 (i) onto $\{c, e\}$ yields the potential $\psi(c, e)$ in Table 2.2 (ii).

Table 2.3: The unity-potential $1(a)$.

a	$1(a)$
0	1.0
1	1.0

Definition 2.1.4 [4] *The unity-potential* $1(v_i)$ *for a single variable* v_i *is a function* 1 *mapping every element of* $dom(v_i)$ *to one. More generally, the unity-potential* $1(X)$ *for a set* $X = \{v_1, v_2, \ldots, v_k\}$ *of variables is defined as follows:* $1(X) = 1(v_1) \cdot 1(v_2) \cdot \cdots \cdot 1(v_k)$. *That is,* $1(X)$ *is table on* X*, where the probability value is one for each row.*

For instance, the unity-potential $1(a)$ is shown in Table 2.3.

Note that $\phi(Y) = \phi(Y) \cdot 1(X)$, if $X \subseteq Y$. Thus, multiplying $\phi(a, b)$ in Table 2.1 with $1(a)$ in Table 2.3 still gives us the table $\phi(a, b)$ in Table 2.1.

Definition 2.1.5 [36] *A conditional probability table* (CPT) *on a set* X *of variables given a disjoint set* Y *of variables, denoted* $p(X|Y)$*, is a potential on the union of* X *and* Y *satisfying the property: for each configuration* $y \in dom(Y)$*,* $\sum_{x \in dom(X)} p(X = x|Y = y) = 1$.

For instance, let c, d, i, g, s, l, j, h be binary variables. Table 2.4 shows CPTs $p(c)$, $p(d|c), p(i), p(g|d, i), p(s|i), p(l|g), p(j|s, l)$ and $p(h|g, j)$.

Whenever $p(X|Y)$ is written with X and Y not disjoint, then we mean $p(X|Y - X)$ to satisfy the disjointness condition of CPTs. In [28], three special cases of CPTs are denoted as $p(X|\emptyset)$, $p(\emptyset|Y)$ and $p(\emptyset|\emptyset)$. However, in BN

Table 2.4: CPTs $p(c), p(d|c), p(i), p(g|d,i), p(s|i), p(l|g), p(j|s,l)$ and $p(h|g,j)$.

c	$p(c)$
0	0.20
1	0.80

c	d	$p(d\|c)$
0	0	0.40
0	1	0.60
1	0	0.70
1	1	0.30

g	l	$p(l\|g)$
0	0	0.30
0	1	0.70
1	0	0.60
1	1	0.40

d	i	g	$p(g\|d,i)$
0	0	0	0.90
0	0	1	0.10
0	1	0	0.20
0	1	1	0.80
1	0	0	0.50
1	0	1	0.50
1	1	0	0.40
1	1	1	0.60

i	$p(i)$
0	0.75
1	0.25

i	s	$p(s\|i)$
0	0	0.40
0	1	0.60
1	0	0.80
1	1	0.20

s	l	j	$p(j\|s,l)$
0	0	0	0.10
0	0	1	0.90
0	1	0	0.60
0	1	1	0.40
1	0	0	0.45
1	0	1	0.55
1	1	0	0.50
1	1	1	0.50

g	j	h	$p(h\|g,j)$
0	0	0	0.25
0	0	1	0.75
0	1	0	0.65
0	1	1	0.35
1	0	0	0.50
1	0	1	0.50
1	1	0	0.85
1	1	1	0.15

literature, they are more commonly written as $p(X)$, $1(Y)$ and 1, respectively. Also, note that a CPT is a special case of potentials. For instance, by definition, CPT $p(d|c)$ in Table 2.4 is a potential. On the contrary, potential $\phi(a, b)$ in Table 2.1 is not a CPT.

Definition 2.1.6 [36] *A* joint probability distribution *(jpd) on U is a function p on U such that the following two conditions both hold: (i) $0 \leq \phi(u) \leq 1$, for each configuration $u \in U$, and (ii) $\sum_{u \in U} \phi(u) = 1.0$.*

For instance, a jpd on $U = \{c, d, i, g, l, s, j, h\}$ is defined later in Equation (2.2).

2.2 Bayesian Networks

The key concept of a Bayesian network can now be given. Bayesian networks [33] (BNs) are a semantic modelling tool for managing uncertainty in complex domains.

2.2.1 Bayesian Network Representation

It is necessary to first introduce the concept probabilistic conditional independence.

Definition 2.2.1 [39] Let x, y and z denote arbitrary configurations of pairwise disjoint subsets X, Y, Z of U, respectively. We say X and Z are *conditionally independent* given Y under the joint probability distribution $p(U)$, denoted

$I_p(X, Y, Z)$, if

$$p(X = x|Y = y, Z = z) \quad = \quad p(X = x|Y = y),$$

whenever $p(Y = y, Z = z) > 0$.

Henceforth, we may write $W = w$ simply as W. The independence $I_p(X, Y, Z)$ in Definition 2.2.1 can be equivalently written as

$$p(X, Y, Z) \quad = \quad \frac{p(X, Y) \cdot p(Y, Z)}{p(Y)}. \tag{2.1}$$

It is not necessary to assume that X, Y and Z be pairwise disjoint, as $I_p(X, Y, Z)$ holds if and only if $I_p(X - Y, Y, Z - Y)$ holds [33]. For simplified notation, we will then write $I_p(X - Y, Y, Z - Y)$ as $I_p(X, Y, Z)$.

To define a BN, we need to first introduce the concept of I-map. Let B be any graph object associated with a set of independencies I_B. B is an I-map for a set of independencies I_p if $I_B \subseteq I_p$ [18].

Definition 2.2.2 [33] *A Bayesian network (BN) on* $U = \{v_1, v_2, \ldots, v_n\}$ *is a pair* (B, C). *B is a directed acyclic graph (DAG)* (V, E) *on* U, *which is a I-map of a probability function p over* V *for any three disjoint subsets of variables. C is a set of CPTs* $\{p(v_1|P(v_1)), p(v_2|P(v_2)), \ldots, p(v_n|P(v_n))\}$ *defined as follows: for each variable* $v_i \in B$, *there is a CPT* $p(v_i|P(v_i))$ *for* v_i *given its parents* $P(v_i)$, *which is the immediate predecessors of* v_i *in* B.

Example 1 [18] *Consider a scenario that a company is trying to hire a recent college graduate. The goal is to hire intelligent employees, which can not be*

tested directly. However, the company can access to the student's SAT scores and ask for a recommendation letter from the student's professor. The professor is absentminded and never remembers the name of her students. She can only look at his grade, and she writes her letter for him based on that information alone. The students's grade, in this case, depends not only on his intelligence but also on the difficulty of the course. This Extended Student Bayesian Network (ESBN) is represented by the DAG in Figure 2.1 on the set U of variables, along with the CPTs in Table 2.4. To simplify the discussion, in this thesis, we will use the characters in the brackets in Figure 2.1 instead of the real variable names and assume that only binary variables are used in examples.

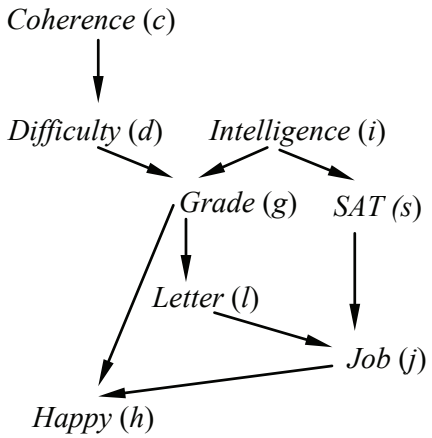

Figure 2.1: The DAG of the ESBN.

The DAG B in a BN graphically encodes probabilistic conditional independencies, which can be inferred from B using the *d-separation* approach, which is defined as follows.

The definition of d-separation involves the notion of active path in a BN.

Definition 2.2.3 [18] *Let v_1 and v_2 be two vertices in a DAG, and Z be a set of vertices from the same DAG. An* active path *from v_1 to v_2 given Z is a path such that: (1) every vertex on the path with converging arrows is in Z or has a descendent in Z, (2) every other vertex on the path is outside Z.*

We now define the formal definition of d-separation.

Definition 2.2.4 [33] *Let X, Y and Z represent three disjoint set of vertices in a DAG. Z* d-separates *X and Y if there is no active path between any $v_1 \in X$ and $v_2 \in Y$ given Z.*

For example, in ESBN BN in Figure 2.1, variable d d-separates c from $\{g, i\}$ in the DAG B, since there is no active path between $\{g, i\}$ and c given d.

In this thesis, an independence statement $I(X, Y, Z)$ holds in B identified by d-separation is denoted as $I_B(X, Y, Z)$. Note that the reason we differentiate the independence holding in the jpd and holding in the DAG by denoting them as $I_p(X, Y, Z)$ and $I_B(X, Y, Z)$, respectively, is because d-separation is sound but not complete.

Lemma 1 [11, 18, 38] *Given a BN (B, C) defining a joint distribution $p(U)$. Then $I(X, Y, Z)$ holds in B by d-separation, only if $I(X, Y, Z)$ holds in $p(U)$.*

Lemma 1 states that if the CI $I(X, Y, Z)$ is verified to be hold in B by d-separation, then it must hold in the jpd $p(U)$. For example, $I_B(c, d, gi)$ holds by

d-separation, it must hold in the jpd $p(U)$, no matter which instance of CPTs is given.

As it is not feasible to test every $I_p(X_1, \emptyset, Y)$ in $p(U)$, d-separation is relied on to test $I_B(X_1, \emptyset, Y)$ in B. Unfortunately, independencies in $p(U)$ can escape detection in B. However, it is well known that d-separation satisfies a weaker notion of completeness.

Lemma 2 [27] *Suppose that d-separation indicates that $I_B(X, Y, Z)$ does not hold in a discrete BN B on U. Then there exists a set C of CPTs for B defining a joint distribution $p(U)$ such that $I_p(X, Y, Z)$ does not hold.* [1]

Lemma 2 shows if the CI $I(X, Y, Z)$ does not hold by d-separation in B, then it is guaranteed that there exists a set of CPTs defining a $p(U)$ such that $I(X, Y, Z)$ does not hold in $p(U)$. For example, $I_B(g, \emptyset, ij)$ does not hold by d-separation in the ESBN B of Figure 2.1. As required by Lemma 2, there must exist a set C of CPTs, such as those in Table 2.4, defining a $p(U)$ such that $I_p(g, \emptyset, ij)$ does not hold.

Moreover, Lemma 2 can be made significantly stronger as [27] showed.

Lemma 3 [27] *Except for a measure zero set* [2] *in the space of all joint distributions $p(U)$ defined by all discrete BNs (B, C), the independencies satisfied by*

[1] Geiger and Pearl [13] showed the completeness of d-separation for Gaussian distributions.

[2] A set has measure zero if it is infinitesimally small relative to the overall space [18].

$p(U)$ are precisely those satisfied by d-separation in B.

Lemma 3 says that for nearly all choices C of CPTs for a BN B defining $p(U)$, d-separation perfectly characterizes the independencies in $p(U)$, i.e., for $X, Y, Z \subseteq U$,

$$I_p(X, Y, Z) \iff I_B(X, Y, Z).$$

For example, considering the CPTs in Table 2.4, testing CIs using d-separation is sound and complete.

One salient feature of BNs is that the *independencies* [39] encoded in the DAG of a BN indicate that the product of the CPTs is a unique joint probability distribution on the set U of variables. For instance, given the ESBN,

$$p(U) = p(c) \cdot p(d|c) \cdot p(i) \cdot p(g|d, i) \cdot p(l|g) \cdot p(s|i) \cdot p(j|l, s) \cdot p(h|g, j). \qquad (2.2)$$

Another important concept, called *initial segment* [36], which will be used a lot in our proof, is now introduced in.

Definition 2.2.5 [36] *A set $V \subseteq U$ in a directed acyclic graph is an* initial segment *if the parents of each v_i in V are also in V.*

For example, in the ESBN DAG in Figure 2.1, variable set $\{c, d, g, i\}$ is an initial segment.

Shafer (1996) showed that if V is an initial segment, then

$$p(V) = \prod_{v_i \in V} p(v_i|P(v_i)).$$

For instance, as $\{c, d, g, i\}$ is an initial segment, we know

$$p(c, d, g, i) \;=\; p(c) \cdot p(d|c) \cdot p(g|d, i) \cdot p(i).$$

The following concept *topological ordering* will play an important role through my thesis.

Definition 2.2.6 [18] *A topological ordering is an ordering $\prec$ of the variables in a BN B so that for every arc (v_i, v_j) in B, v_i precedes v_j in $\prec$.*

For example, $c \prec d \prec i \prec g \prec s \prec l \prec j \prec h$ is a topological ordering of the directed acyclic graph in Figure 2.1, but $d \prec c \prec i \prec g \prec h \prec l \prec j \prec s$ is not.

2.2.2 Bayesian Network Inference

Probabilistic inference, also known as query processing, typically means computing $p(X|E = e)$, which are useful for many reasoning patterns, including explanation, prediction, intercausal reasoning, and many more [18], where $X \cap E = \emptyset$ and $X, E \subseteq U$. The *evidence* in the query is that E is instantiated to configuration e, while X contains *target* variables. Probabilistic inference can be conducted directly in an entire BN or in parts of a BN [12, 21, 35, 41]. It can also be conducted in a join tree [14, 15, 19, 23, 25, 36]. Next, we review two fundamental BN inference algorithms, namely variable elimination and arc reversal, which are related to this thesis.

Variable Elimination

While the many techniques for exact inference vary in when they apply the multiplication and marginalization (addition) operators, they all center around eliminating variables from the networks.

Two important properties of marginalization are now discussed.

Lemma 4 [36] *If ϕ is a potential on U, and $X \subseteq Y \subseteq U$, then marginalizing ϕ onto Y and subsequently onto X is the same as marginalizing ϕ directly onto X.*

Lemma 4 indicates that a marginal can be obtained by a series of marginalizations in any order. For example,

$$\sum_{a,b} \psi(a,b,c) \quad = \quad \sum_{a} \; (\; \sum_{b} \psi(a,b,c)) \quad = \quad \sum_{b} \; (\; \sum_{a} \psi(a,b,c)).$$

Lemma 5 [36] *If ϕ is a potential on X and ψ is a potential on Y, then the marginalization of $\phi \cdot \psi$ onto X is the same as ϕ multiplied with the marginalization of ψ onto $X \cap Y$.*

For instance,

$$\sum_{c} \psi(a,b) \cdot \psi(b,c) \quad = \quad \psi(a,b) \cdot \sum_{c} \psi(b,c).$$

In this thesis, we focus on a basic discrete inference algorithm for computing $p(X|E = e)$, called *variable elimination* (VE), first put forth by [41]. BN Inference involves the elimination of variables. Algorithm 1, called *sum-out* (SO), eliminates a single variable v from a set Φ of *potentials* [18], and returns the

resulting set of potentials. The algorithm *collect-relevant* simply returns those potentials in Φ involving variable v.

Algorithm 1. SO(v,Φ)

1 Ψ = collect-relevant(v,Φ)

2 ψ = the product of all potentials in Ψ

3 $\tau = \sum_v \psi$

4 $\Phi' = (\Phi - \Psi) \cup \{\tau\}$

5 **return** Φ'

Algorithm 1. Sum-out (SO) Algorithm.

Example 2 [18] *Given the BN in Example 2.1, let d be the variable needs to be eliminated. As* $\Phi = \{p(c), p(d|c), p(g|d,i), p(i), p(l|g), p(s|i), p(j|l,s), p(h|g,j)\}$, $\Psi = \{p(d|c), p(g|d,i)\}$. *Then, d is eliminated as follows:*

$$\begin{aligned} \sum_d p(d|c) \cdot p(g|d,i) &= \sum_d \psi(d,g|c,i) \\ &= \psi(g|c,i). \end{aligned}$$

We then have $\Phi' = \{p(c), p(i), p(l|g), p(s|i), p(j|l,s), p(h|g,j), \psi(g|c,i)\}$.

To answer the query $p(X|E=e)$, BN inference algorithm multiplies the *evidence potential*.

Definition 2.2.7 *The* evidence potential *for* $E = e$, *denoted* $1(E = e)$, *assigns probability* 1 *to the single value* e *of* E *and probability* 0 *to all other values of* E.

Hence, for a variable v observed taking value λ and $v \in \{v_i\} \cup P(v_i)$, the product $p(v_i|P(v_i)) \cdot 1(v = \lambda)$ keeps only those configurations agreeing with $v = \lambda$.

It is well known that eliminating the same variable set using different elimination ordering during BN inference will cost different amount of calculation. There are many approaches for deciding a good elimination ordering. "Generally, Min-Fill and Weighted-Min-Fill tend to work better on more problems." [18]. Therefore, in this thesis, we choose the Weighted-Min-Fill (WMF) measurement [18] to determine the elimination ordering. The detail of WMF is shown in Appendix A. For instance, given the undirected graph in 2.9, one elimination ordering for variables $\{c, d, g, l, s\}$ using WMF algorithm is $\sigma = c, d, l, s, g$.

Algorithm 2, taken from [18], computes $p(X|E = e)$ from a discrete BN B. VE calls SO to eliminate variables one by one. More specifically, in Algorithm 2, Φ is the set C of CPTs for B, X is a list of query variables, E is a list of observed variables, e is the corresponding list of observed values, and σ is an elimination ordering for variables $U - XE$, where XE denotes $X \cup E$.

Algorithm 2. VE(Φ, X, E, e, σ)

1 Multiply evidence potentials with appropriate CPTs

2 **While** σ is not empty

3 Remove the first variable v from σ

4 Φ = sum-out(v, Φ)

5 $p(X, E = e)$ = the product of all potentials $\psi \in \Phi$

6 $p(E = e) = \sum_X p(X, E = e)$

7 **return** $p(X, E = e)/p(E = e)$

Algorithm 2. Variable Elimination (VE) Algorithm.

Example 3 *As in [18], suppose the observed evidence for the ESBN is* $i = 1$ *and* $h = 0$ *and the query is* $p(j|h = 0, i = 1)$. *The weighted-min-fill algorithm [18] can yield* $\sigma = c, d, l, s, g$. *VE first incorporates the evidence:*

$$\begin{aligned}
\psi(i = 1) &= p(i) \cdot 1(i = 1), \\
\psi(d, g, i = 1) &= p(g|d, i) \cdot 1(i = 1), \\
\psi(i = 1, s) &= p(s|i) \cdot 1(i = 1), \\
\psi(g, h = 0, j) &= p(h|g, j) \cdot 1(h = 0).
\end{aligned}$$

To eliminate c, the SO algorithm computes

$$\begin{aligned}
\sum_c p(c) \cdot p(d|c) &= \sum \psi(c, d) \\
&= \psi(d).
\end{aligned}$$

SO computes the following to eliminate d

$$\begin{aligned}\sum_d \psi(d) \cdot \psi(d, g, i = 1) &= \sum_d \psi(d, g, i = 1) \\ &= \psi(g, i = 1).\end{aligned}$$

To eliminate l,

$$\begin{aligned}\sum_l p(l|g) \cdot p(j|l, s) &= \sum_l \psi(g, j, l, s) \\ &= \psi(g, j, s).\end{aligned}$$

SO computes the following when eliminating s,

$$\begin{aligned}\sum_s \psi(i = 1, s) \cdot \psi(g, j, s) &= \sum_s \psi(g, i = 1, j, s) \\ &= \psi(g, i = 1, j). \qquad (2.3)\end{aligned}$$

For g, SO can compute:

$$\begin{aligned}&\sum_g \psi(g, i = 1, j) \cdot \psi(g, i = 1) \cdot \psi(g, h = 0, j) \\ = &\sum_g \psi(g, i = 1, j) \cdot \psi(g, h = 0, i = 1, j) \qquad (2.4) \\ = &\sum_g \psi(g, h = 0, i = 1, j) \\ = &\psi(h = 0, i = 1, j). \qquad (2.5)\end{aligned}$$

Next, VE multiplies all remaining potentials as

$$p(h = 0, i = 1, j) = \psi(i = 1) \cdot \psi(h = 0, i = 1, j).$$

According to VE, the following computation is performed

$$p(h = 0, i = 1) = \sum_j p(h = 0, i = 1, j).$$

Finally, VE answers the query by

$$p(j|h=0,i=1) \;=\; \frac{p(h=0,i=1,j)}{p(h=0,i=1)}.$$

Arc Reversal

Arc reversal (AR) [30,34] is another direct computation algorithm that eliminates a variable by making it barren (leaf) via a sequence of arc reversals. Given a query $p(X|E=e)$ and a BN, AR eliminates each variable $v_i \notin (X \cup E)$ in the BN one-by-one using the following steps. The following outline draws from [17,22,24,34].

AR eliminates a variable v_i by reversing the arcs (v_i, v_j) for each child v_j of v_i, where $j = 1, 2, \ldots, k$. Let v_i have parents $P(v_i) = P_1 \cup P_2$ and v_j have parents $P(v_j) = \{v_i\} \cup P_2 \cup P_3$, where P_1, P_2, P_3 are three pairwise disjoint variable sets. More specifically, P_1 are the parents specific for v_i, P_2 are the common parents, and P_3 are the parents specific for v_j, as seen in Figure 2.2.

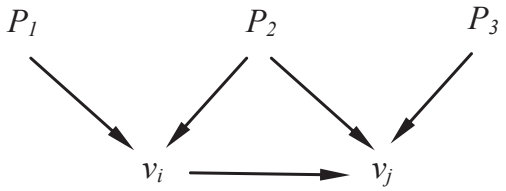

Figure 2.2: A DAG from which variable v_i is to be eliminated.

The reversal of arc (v_i, v_j) proceeds by setting $P(v_j) = P_1 \cup P_2 \cup P_3$ and $P(v_i) = P_1 \cup P_2 \cup P_3 \cup \{v_j\}$, as illustrated in Figure 2.3.

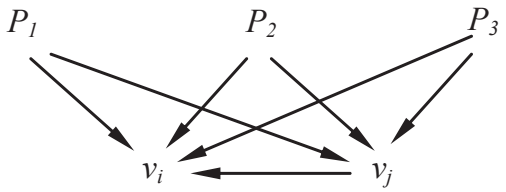

Figure 2.3: The modified DAG after reversing the arc (v_i, v_j).

The new CPT for v_j and v_i are physically computed as:

$$p(v_i, v_j|P_1, P_2, P_3) = p(v_i|P_1, P_2) \cdot p(v_j|v_i, P_2, P_3), \quad (2.6)$$

$$p(v_j|P_1, P_2, P_3) = \sum_{v_i} p(v_i, v_j|P_1, P_2, P_3), \quad (2.7)$$

$$p(v_i|v_j, P_1, P_2, P_3) = \frac{p(v_i, v_j|P_1, P_2, P_3)}{p(v_j|P_1, P_2, P_3)}. \quad (2.8)$$

Suppose the variable v_i to be removed has k children. Each arc from v_i to its child is reversed by recursively applying the above procedure until v_i becomes a leaf. Note that the distributions defined in Equations (2.6) - (2.8) are built for the first $k-1$ children. For the last child, however, only the distributions in Equations (2.6) and (2.7) are built. When considering the last child, there is no need to build the final distribution for v_i in Equation (2.8), since v_i will be removed as a leaf variable. Also, a DAG structure is maintained after eliminating a variable by applying arc reversal [22,24], since arc reversal uses a fixed ancestral numbering of the original BN to avoid creating directed cycles [2].

Example 4 *Recall Example 3. Let us use AR to eliminate c, d, l, s, g. To eliminate c, for its child d, we compute*

$$p(c, d) = p(c) \cdot p(d|c) \quad (2.9)$$

and

$$p(d) \;=\; \sum_{c} p(c,d). \tag{2.10}$$

The DAGs after reversing arcs and eliminating c are shown in Figure 2.4 (i) and (ii). As d is the only child of c in the DAG in Figure 2.1, the calculation of

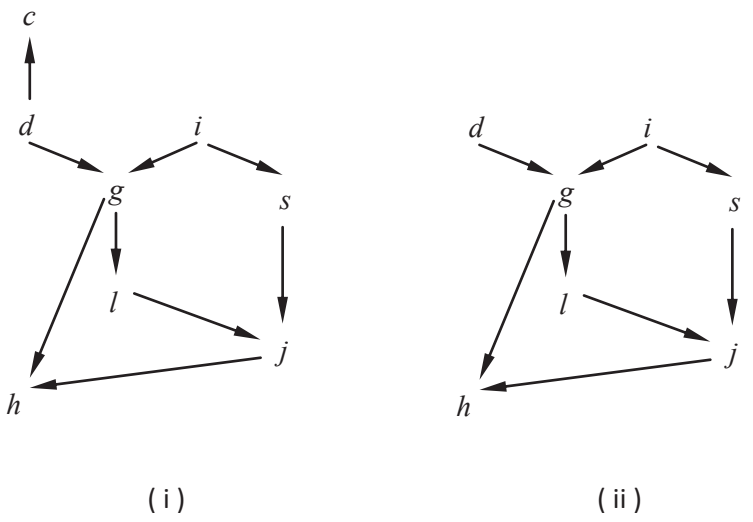

Figure 2.4: To eliminate c, (i) the DAG after reversing arcs; (ii) the DAG after eliminating c.

Equation 2.8 can be saved.

Next, let us eliminate d. Similarly, as it has one child g in Figure 2.4 (ii), we only need to compute

$$p(d,g|i) \;=\; p(d)\cdot p(g|d,i) \tag{2.11}$$

and

$$p(g|i) \;=\; \sum_{d} p(d,g|i). \tag{2.12}$$

The DAGs after reversing arcs and eliminating d are shown in Figure 2.5 (i) and (ii), respectively.

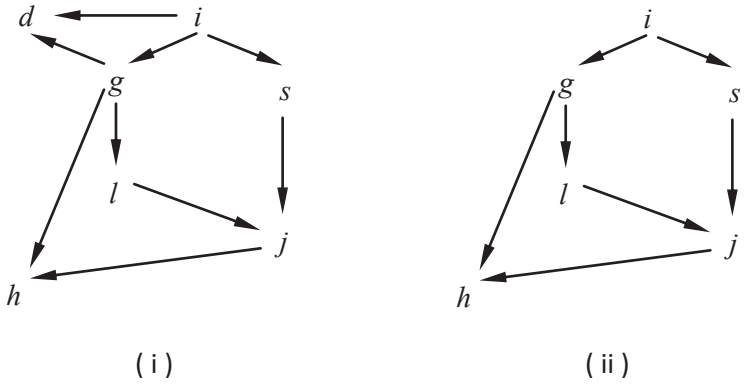

Figure 2.5: To eliminate d, (i) the DAG after reversing arcs; (ii) the DAG after eliminating d.

To eliminate l, for its only child j in Figure 2.5 (ii), AR computes

$$p(j, l|g, s) \;=\; p(l|g) \cdot p(j|l, s)$$

and

$$p(j|g, s) \;=\; \sum_{l} p(j, l|g, s).$$

The DAGs after reversing arcs and eliminating l are shown in Figure 2.6 (i) and (ii), respectively.

To eliminate s, for its only child j in Figure 2.6 (ii), AR computes

$$p(j, s|g, i) \;=\; p(s|i) \cdot p(j|g, s)$$

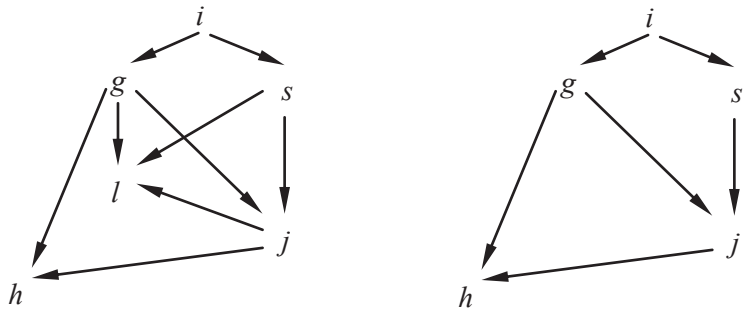

Figure 2.6: To eliminate l, (i) the DAG after reversing arcs; (ii) the DAG after eliminating l.

and

$$p(j|g,i) \;=\; \sum_{s} p(j,s|g,i).$$

The DAGs after reversing arcs and eliminating s are shown in Figure 2.7 (i) and (ii), respectively.

To eliminate g, note that there are two children of g in Figure 2.7 (ii), namely j and h. For j, AR computes

$$\begin{aligned} p(g,j|i) &= p(g|i)\cdot p(j|g,i) \\ p(j|i) &= \sum_{g} p(g,j|i) \end{aligned}$$

and

$$p(g|i,j) \;=\; \frac{p(g,j|i)}{p(j|i)}.$$

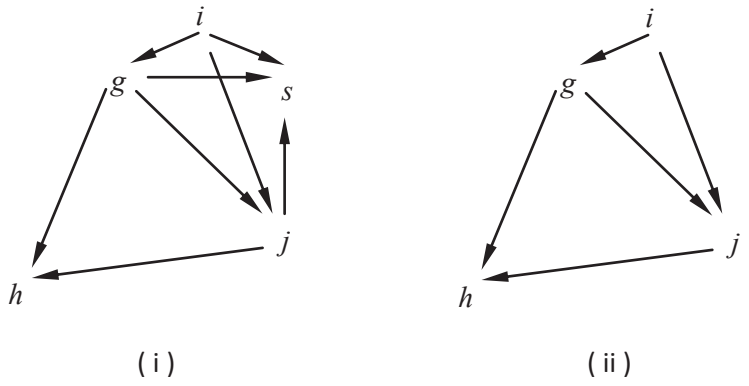

Figure 2.7: To eliminate s, (i) the DAG after reversing arcs; (ii) the DAG after eliminating s.

For the child h, we have

$$p(g, h|i, j) \;=\; p(g|i, j) \cdot p(h|g, j)$$

and

$$p(h|i, j) \;=\; \sum_{g} p(g, h|i, j).$$

The DAGs after reversing arcs and eliminating g are shown in Figure 2.8 (i) and (ii), respectively.

Lazy Join Tree Propagation

There are many different join tree propagation methods. The classical join tree propagation algorithms were proposed by Lauritzen and Spiegelhalter [20], Shafer and Shenoy [37], and Jensen et al. [14]. In this thesis, the join tree propagation

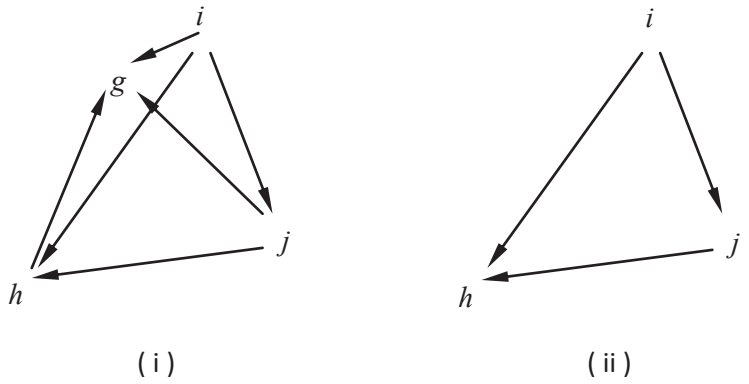

Figure 2.8: To eliminate g, (i) the DAG after reversing arcs; (ii) the DAG after eliminating g.

algorithm we discussed is called Lazy Propagation, which was suggested by Madsen and Jensen [25]. Lazy propagation empirically demonstrated a significant improvement in efficiency over the classical join tree propagation methods. Before we outline the join tree propagation methods, we need to first introduce some notions related to a join tree.

The task of transforming a DAG into a join tree has been extensively studied in the probabilistic reasoning literature [16]. A DAG can be converted into a join tree via the moralization and triangulation procedures. If necessary, undirected edges are added to the moralization B_m to create a *triangulated* [8] graph B_t.

Definition 2.2.8 [8] *The moralization of a DAG B is the undirected graph B_m obtained by connecting vertices in D that have a common child, and then making all edges in the graph undirected.*

For example, to obtain the moralization of the DAG in Figure 2.1, undirected edges (d, i), (g, j) and (l, s) are added, and then the direction of all edges is dropped in the graph, as shown in Figure 2.9.

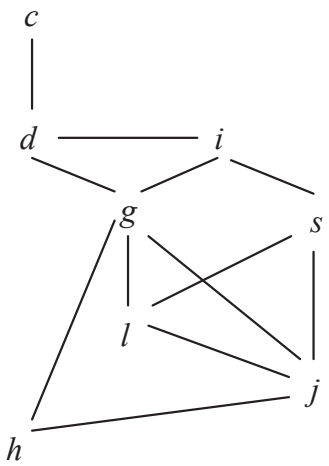

Figure 2.9: The moralization of the DAG in Figure 2.1.

Definition 2.2.9 [8] *An undirected graph is triangulated (chordal) if each cycle of length four or more possesses an edge (v_i, v_j) between two non-adjacent variables v_i and v_j in the cycle.*

For instance, a triangulation of the moralized graph in Figure 2.9 is shown in 2.10.

After triangulation, each maximal clique [8] of the triangulated graph is represented by a node in a join tree. Then the conditional of each variable v_i in the given BN is assigned to precisely one join tree node containing v_i and its parents P_i in the BN.

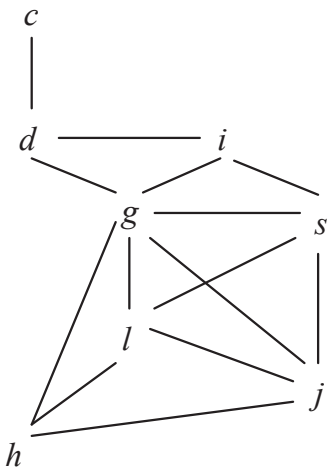

Figure 2.10: The triangulation of the moralized graph in Figure 2.9.

Definition 2.2.10 [33,36] *A join tree is a tree having sets of variables as nodes, with the property that any variable in two nodes is also in any node on the path between the two. The separator S between any two neighbouring nodes N_i and N_j is $S = N_i \cap N_j$. A join tree node with only one neighbour is called a leaf; otherwise it is a non-leaf node.*

Example 5 *Consider the triangulated graph in Figure 2.10. The maximal cliques of this triangulated graph are $\{cd, dgi, gis, gjls, ghjl\}$, which gave the nodes of the join tree in Figure 2.11. In this join tree, the separators are d, gi, gs and gjl. The CPT $p(g|d,i)$, for instance, is assigned to join tree node dgi.*

The name of a join tree node corresponds to the variables in the join tree node. For instance, for the join tree in Figure 2.11, node *cd* means the join tree node consists of variables c and d.

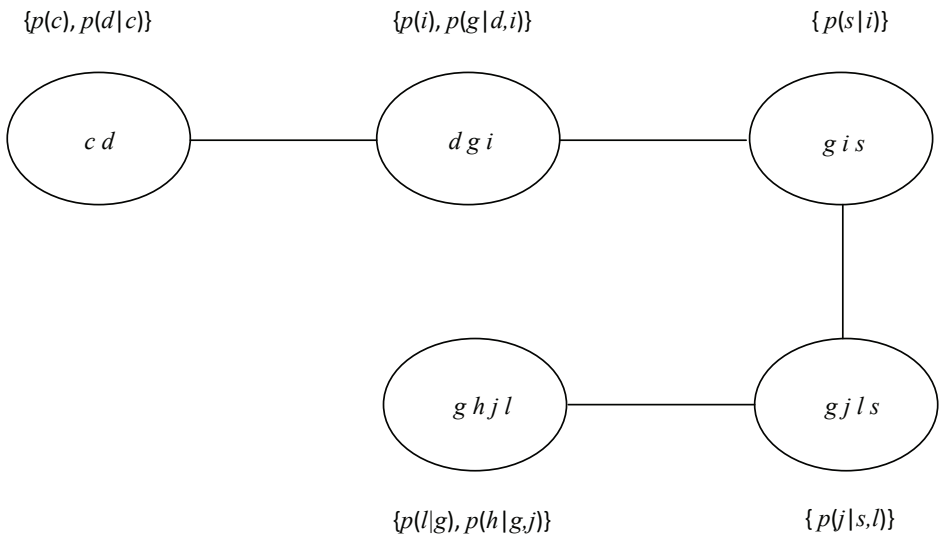

Figure 2.11: The join tree of the ESBN DAG in Figure 2.1.

Next, we will outline the Lazy propagation as follows. Lazy propagation utilize a classical join tree architecture, namely Shafer-Shenoy [37], which is conceived as a parallel algorithm [37]. In this architecture, two storage registers are allocated in each separator, one for a message sent in each direction. Each node sends messages to all its neighbours, according to the following two rules:

1. Each node waits to send its message to a particular neighbour until it has received messages from all its other neighbours.

2. When a node N_i is ready to send its message to a particular neighbour N_j, it computes the message $\phi(N_i \cap N_j)$ by collecting all its messages from other neighbours, multiplying its potential $\phi(N_i)$ by these messages, and marginalizing the product to $N_i \cap N_j$.

When a join tree node N is ready to send its messages to a particular neighbour N', Lazy propagation computes the messages from node N to N' using the following three steps:

i) Collect all messages from N's other neighbours.

ii) Identify the relevant and irrelevant variables.

iii) Call the specific algorithm to iteratively eliminate every variable in node N not appearing in the neighbour N'.

According to the mechanism to physically compute the messages passed from a join tree node to a neighbour in Step iii), Lazy propagation is categorized into Lazy VE and Lazy AR propagation. As indicated by name, Lazy VE utilize VE approach to compute the messages and Lazy AR utilize AR approach.

Example 6 *Let us apply Lazy VE on the join tree in Figure 2.11. The message from node cd to dgi is calculated as*

$$\begin{aligned} \sum_c p(c) \cdot p(d|c) &= \sum_c \psi(c,d) \\ &= \psi(d). \end{aligned}$$

After obtain the message $\psi(d)$, node dgi computes

$$\begin{aligned} \sum_d \psi(d) \cdot p(g|d,i) &= \sum_d \psi(g,d,i) \\ &= \psi(g,i). \end{aligned}$$

Then, node dgi pass messages

$$p(i) \quad \textit{and} \quad \psi(g,i)$$

to node gis. Now consider the message from node gis to gjls. To eliminate i, Lazy VE performs

$$\begin{aligned}\sum_i \psi(g,i)\cdot p(i)\cdot p(s|i) &= \sum_i \psi(g,i,s)\\ &= \psi(g,s).\end{aligned}$$

Therefore, the message from gis to gjls is

$$\psi(g,s).$$

The calculation for the rest messages is omitted for simplicity.

Now let us apply Lazy AR on the join tree in Figure 2.11. Note that it has been shown that each node in join tree keeps a sub-DAG [7]. For instance, given the join tree in Figure 2.11, the sub-DAG each node keeps is shown in 2.12.

Example 7 *For the message from node cd to dgi, Lazy AR computes exactly the same as Equations (2.9) and (2.10) at node cd to reverse edge from c to d and then remove c as shown in Figure 2.13.*

Thus, the message from cd to dgi is

$$p(d).$$

After obtaining the message p(d), node dgi performs Equations (2.11) and (2.12), as shown in Figure 2.14.

Then, the message from dgi to gis is

$$p(i) \quad \textit{and} \quad p(g|i).$$

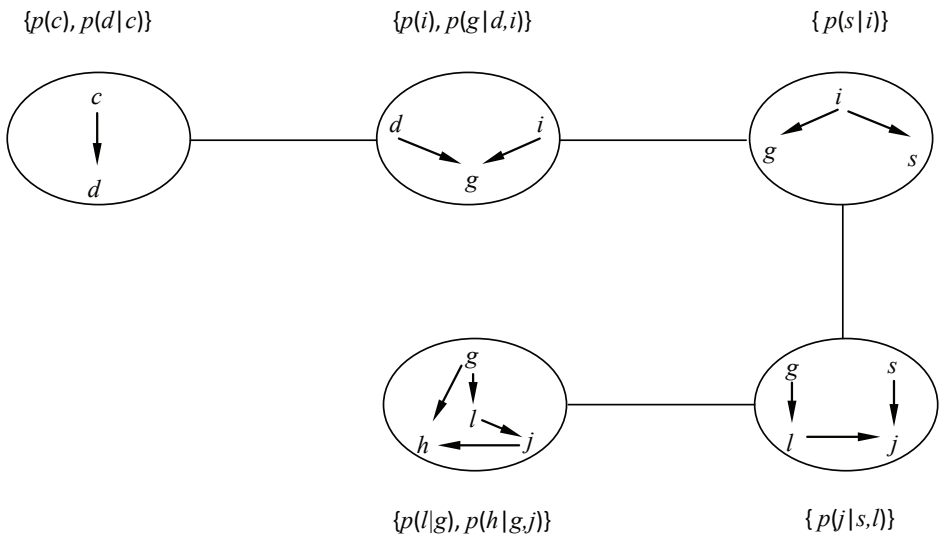

Figure 2.12: The join tree node keeps a sub-DAG.

Now let us consider to compute the message from gis to gjls using AR. By join tree propagation construction, i needs to be eliminated as shown in Figure 2.15.

For the child g, AR computes

$$\begin{aligned} p(g,i) &= p(i) \cdot p(g|i) \\ p(g) &= \sum_{g} p(g,i) \end{aligned}$$

and

$$p(i|g) = \frac{p(g,i)}{p(g)}.$$

For the child s, we have

$$p(i,s|g) = p(i|g) \cdot p(s|i)$$

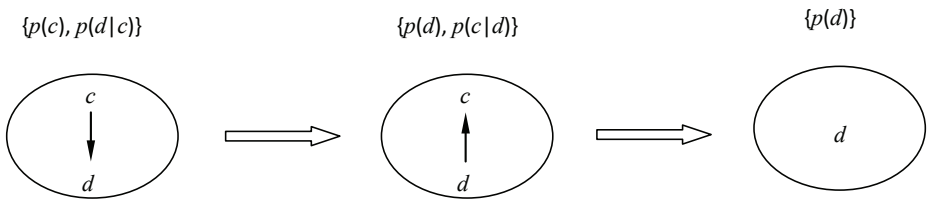

Figure 2.13: Lazy AR reverses edge from c to d and removes c.

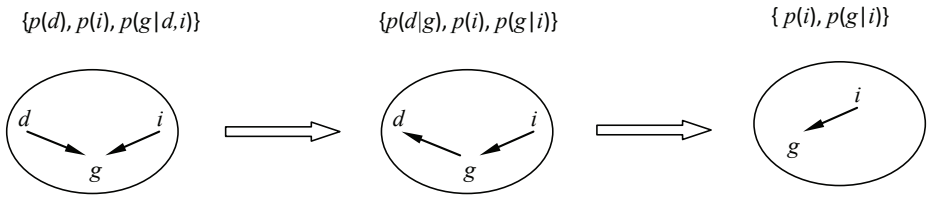

Figure 2.14: Lazy AR reverses edge from d to g and removes d.

and

$$p(s|g) \;=\; \sum_i p(i, s|g).$$

Thus, the message sent from node gis to gjls is

$$p(g) \quad and \quad p(s|g).$$

The calculation for the rest nodes is omitted for simplicity.

Now let us do a comparison between the amount of calculation performed by Lazy VE and Lazy AR. Suppose all the variables in the ESBN have binary

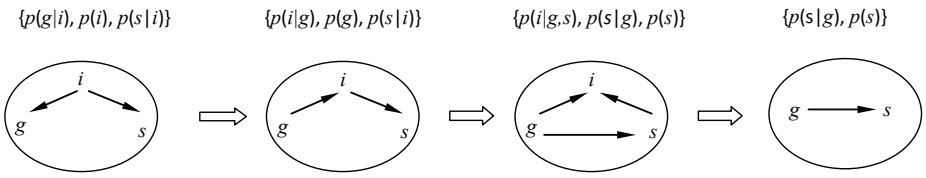

Figure 2.15: Lazy AR reverses edges from i to g and i to s, and then removes i.

value. Then, it can be verified that for computing message from node cd to dgi, Lazy VE requires 4 multiplications and 2 additions and Lazy AR requires 4 multiplications and 2 additions. For the message from node dgi to gis, Lazy VE requires 8 multiplications and 4 additions and Lazy AR requires the same amount of computation. However, for the message from node gis to $gjls$, Lazy VE requires 12 multiplications and 4 additions while Lazy AR requires 12 multiplications 6 additions and 4 divisions. It has already been proved that eliminating the same variables by VE will not cost more calculations than using AR [1].

However, sometime Lazy AR requires less calculations than Lazy VE. This is because it maintains a multiplicative factorization of potentials at each join tree node and each join tree separator. Maintaining a decomposition of potentials allows the receiving join tree node to exploit barren variables and independencies induced by evidence, such that we can reduce the amount of work needed to physically compute its outgoing messages. Modelling structure in this manner leads to significant computational savings [7, 25]. The following example supports this claim very well.

Example 8 *Consider the BN in Figure 2.16 (i). Its join tree with the assignment of CPTs is illustrated in Figure 2.16 (ii).*

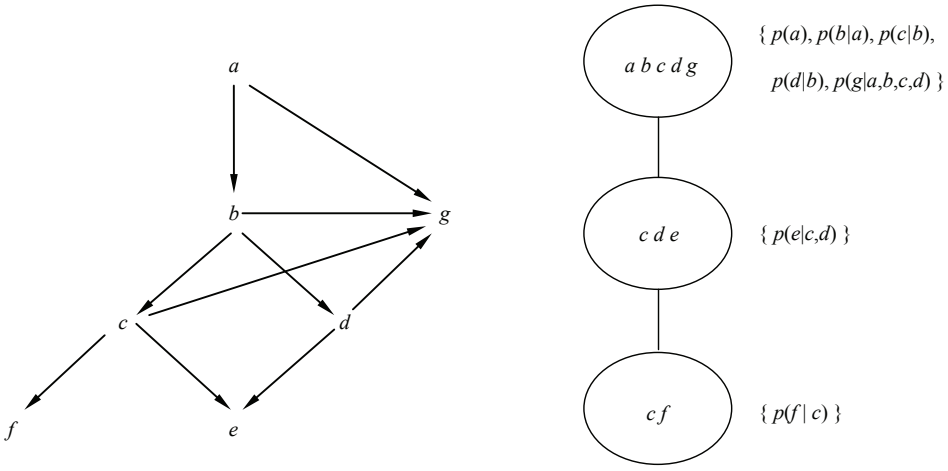

Figure 2.16: (i) A BN; (ii) its join tree.

Let us focus on computing the message from cde to cf. It can be verified that Lazy VE requires 2 additions at least, while Lazy AR requires no computation. The reason AR saves work is because Lazy VE send message from abcdg to cde in the form $\psi(c, d)$, while Lazy AR keeps the factorization $p(c)$ and $p(d|c)$ in message from abcdg to cde. It leads that d is exploited to be a barren variable and $p(c)$ is directly forwarded to cf with involving any calculation.

As we can see, for the two different kinds of join trees such as the ones in Figure 2.11 and Figure 2.16 (ii), the performance of Lazy VE and Lazy AR are different. It naturally follows the question "How should we choose VE or AR to eliminate variables in the join tree node". There had already been some research focusing on it, such as [2, 23].

The measurement proposed by [2] is if the message is one single distribution, Lazy VE should be applied. Otherwise, choose Lazy AR. For instance, the message from *cd* to *dgi* in Figure 2.11 should be computed by VE. On the contrary, the message from *dgi* to *gis* should be calculated by AR. Note that all this measurements ask join tree algorithms either exclusively apply VE or arc reversal (AR) [30] at all join tree nodes [23], or pick whether to apply VE or AR at each node [2].

Chapter 3

Understanding Semantics

In this chapter, we introduce the concept of semantics and then review the current limited understanding of semantics in inference.

3.1 Semantics in Bayesian Network Inference

The premise of discussing semantics of the intermediate distribution is that we understand the structure of them. Thus, let me first make a brief claim here that every intermediate distribution during VE inference is a CPT, when evidence is not considered. The formal proof will be left to next Chapter.

By semantics, we mean that a CPT $\psi(X|Y)$ constructed by VE's manipulation of Bayesian network CPTs is not necessarily equal to the CPT $p(X|Y)$ obtained from the defined joint probability distribution $p(U)$. For instance, it can be

verified that in the ESBN,

$$p(h|g,j) \cdot \sum_{d} p(g|d,i) \cdot \sum_{c} p(c) \cdot p(d|c) \tag{3.1}$$

produces the CPT $\psi(g,h|i,j)$ shown in Table 3.1 (left). In contrast, the CPT $p(g,h|i,j)$ built from the joint distribution $p(U)$ in (2.2) is shown in Table 3.1 (right).

The following is the formal definition of semantics.

Definition 3.1.1 *Given a BN (B,C) over U. Let $\psi(X_1, X_2 = x_2|Y_1, Y_2 = x_2)$ be any CPT built during the process of performing VE to answer a query, where x_2 and y_2 are observed values of variables X_1 and X_2, respectively. The semantics of $\psi(X_1, X_2 = x_2|Y_1, Y_2 = x_2)$ means if*

$$\psi(X_1, X_2 = x_2|Y_1, Y_2 = x_2) = p(X_1, X_2 = x_2|Y_1, Y_2 = x_2)$$

where $p(X_1, X_2 = x_2|Y_1, Y_2 = x_2)$ is the CPT defined with respect to the joint probability distribution $p(U)$.

Definition 3.1.1 simply asks one question "when an intermediate distribution $\psi(X_1, X_2 = x_2|Y_1, Y_2 = x_2)$ constructed by VE is equal to the CPT $p(X_1, X_2 = x_2|Y_1, Y_2 = x_2)$ obtained from the defined joint probability distribution $p(U)$?" If the answer is yes, we denote the distribution as $p(X_1, X_2 = x_2|Y_1, Y_2 = x_2)$. Otherwise, it will be denoted as $\phi(X_1, X_2 = x_2|Y_1, Y_2 = x_2)$. One goal of this thesis is to answer this question.

3.2 Remarks on Current Understanding of Semantics

Kjaerulff and Madsen [17] suggest that in working with probabilistic networks it is convenient to denote distributions as potentials. In fact, the use of potentials is built into the standard inference algorithms (see the SO and VE algorithms, for instance).

For example, suppose query $p(j)$ is posed to the ESBN [18]. Even without evidence being considered, the initial step of VE is to regard CPTs as potentials, i.e., $p(U)$ now is factorized as

$$\begin{aligned} & p(U) \\ = \; & \psi(c) \cdot \psi(c,d) \cdot \psi(i) \cdot \psi(d,g,i) \cdot \psi(g,l) \cdot \psi(i,s) \cdot \psi(j,s,l) \cdot \psi(g,h,j). \end{aligned} \quad (3.2)$$

By comparing (2.2) and (3.2), it is clear that semantics are destroyed even before the CPTs in computer memory have been modified. The notation used for potentials does not convey the semantic meaning of the probabilities comprising the potential.

The same vagueness also appears when evidence is considered. For instance, $\phi(g, h = 0|i = 1, j)$ in (2.4) is represented as $\psi(g, h = 0, i = 1, j)$ according to [17, 18]. The latter notation does not indicate conditional probabilities, nor that their semantics are not defined with respect to the joint distribution.

There are a few recent research on the semantics of the intermediate distributions during BN inference.

Darwiche [11] ascribes meaning during inference by representing each potential by what we will call evidence expanded form, except that products involving evidence potentials are taken.

Definition 3.2.1 *Let ψ be any potential constructed by VE. The* evidence expanded form *of ψ, denoted $F(\psi)$, is the unique expression defining how ψ was built using the multiplication and marginalization operators on the Bayesian network CPTs together with any appropriate evidence potentials.*

Example 9 *Consider potential $\psi(g, i = 1, j)$ in (2.3). $F(\psi(g, i = 1, j))$, the evidence expanded form, can be easily obtained in a recursive manner as follows:*

$$\begin{aligned} & \sum_s \psi(i = 1, s) \cdot \psi(g, j, s) \\ = & \sum_s \psi(i = 1, s) \cdot (\sum_l (p(l|g) \cdot p(j|l, s))) \\ = & \sum_s ((p(s|i) \cdot 1(i = 1)) \cdot (\sum_l (p(l|g) \cdot p(j|l, s)))). \end{aligned} \tag{3.3}$$

Henceforth, parentheses in the evidence expanded form are understood and may not be shown.

While it certainly can be argued that the evidence expanded form in Example 9 provides more meaning that simply writing the intermediate distribution as potential $\psi(g, i = 1, j)$, this approach is not safe from constructive criticism. First, using the expanded form becomes unwieldy as the expression becomes larger. For instance, if we want to denote the semantics of $\psi(h = 0, i = 1, j)$ in Example 3, then the expanded form of the constructed intermediate distribution

would be:

$$\sum_g p(h=0|g,j) \cdot \sum_s p(s|i=1) \cdot \sum_l p(l|g) \cdot p(j|s,l) \cdot \sum_d p(g|d,i=1)$$
$$\cdot \sum_c p(c) \cdot p(d|c),$$

which is somewhat cumbersome. Second, and more importantly, while the expanded form clearly indicates how the distribution was built, it does not explicitly make clear the CPT structure nor the semantics of these conditional probabilities.

Another research trying to answering semantics is made by Koller and Friedman [18]. In their comprehensive and highly recommended text, Koller and Friedman [18] consider the semantics of potential $\psi(b,c,d)$ built by eliminating variable a from the Bayesian network B in Figure 3.1:

$$\psi(b,c,d) \quad = \quad \sum_a p(a) \cdot p(b|a) \cdot p(d|a,c). \tag{3.4}$$

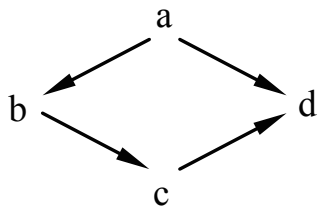

Figure 3.1: A Bayesian networks B.

Koller and Friedman [18] incorrectly state that

$$p(b,d|c) \quad \neq \quad \psi(b,c,d). \tag{3.5}$$

While this claim is almost always true, there are a few exceptions to refute it. For one counter-example, eliminating variable a using the CPTs in Table 3.2 yields:

$$p(b, d|c) \quad = \quad \psi(b, c, d). \tag{3.6}$$

Koller and Friedman [18] also state it must necessarily be the case that

$$p'(b, d|c) \quad = \quad \psi(b, c, d), \tag{3.7}$$

where $p'(U)$ is defined by a *different* Bayesian network B' - the one given in Figure 3.2.

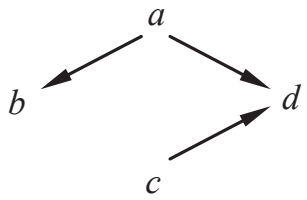

Figure 3.2: A Bayesian networks B'.

However, it makes more sense if we can stipulate semantics in the *current* Bayesian network B - the one on which inference is being conducted other than a different one. This is one objective of this thesis.

Table 3.1: (left) The CPT $\psi(g,h|i,j)$ built by (3.1). (right) The CPT $p(g,h|i,j)$ built from $p(U)$ in (2.2).

i	j	g	h	$\psi(g,h\|i,j)$	i	j	g	h	$p(g,h\|i,j)$
0	0	0	0	0.1890	0	0	0	0	0.1960
0	0	0	1	0.5670	0	0	0	1	0.5880
0	0	1	0	0.1220	0	0	1	0	0.1080
0	0	1	1	0.1220	0	0	1	1	0.1080
0	1	0	0	0.4914	0	1	0	0	0.4762
0	1	0	1	0.2646	0	1	0	1	0.2564
0	1	1	0	0.2074	0	1	1	0	0.2272
0	1	1	1	0.0366	0	1	1	1	0.0402
1	0	0	0	0.0680	1	0	0	0	0.0846
1	0	0	1	0.2040	1	0	0	1	0.2537
1	0	1	0	0.3640	1	0	1	0	0.3309
1	0	1	1	0.3640	1	0	1	1	0.3308
1	1	0	0	0.1768	1	1	0	0	0.1518
1	1	0	1	0.0952	1	1	0	1	0.0817
1	1	1	0	0.6188	1	1	1	0	0.6515
1	1	1	1	0.1092	1	1	1	1	0.1150

Table 3.2: Exceptional CPTs for B in Figure 3.1.

a	$p(a)$
0	0.2
1	0.8

a	b	$p(b\|a)$
0	0	0.4
0	1	0.6
1	0	0.9
1	1	0.1

b	c	$p(c\|b)$
0	0	0.5
0	1	0.5
1	0	0.5
1	1	0.5

a	c	d	$p(d\|a,c)$
0	0	0	0.5
0	0	1	0.5
0	1	0	0.5
0	1	1	0.5
1	0	0	0.5
1	0	1	0.5
1	1	0	0.5
1	1	1	0.5

Chapter 4

CPT Structure

Before showing the semantics of VE inference, it is necessary to present the CPT structure of BN inference. In this chapter, we will show that every multiplication operation and every marginalization operation involved in eliminating variables from a discrete Bayesian network yields a CPT, if evidence is not presented.

4.1 Structure of a Product of Given Bayesian Network CPTs

A key observation is that the product of any non-empty subset of Bayesian network CPTs is itself a CPT. Our claim is then shown by rewriting the factorization to exploit our key observation.

As shown in Chapter 2, one salient feature of Bayesian networks is shown, namely, that the product of the given CPTs is a joint probability distribution.

For example, in the ESBN Bayesian network, we have:

$$p(c,d,i,g,l,s,j,h) \;=\; p(c)\cdot p(d|c)\cdot p(i)\cdot p(g|d,i)\cdot p(l|g)\cdot p(s|i)\cdot p(j|l,s)\cdot p(h|g,j). \tag{4.1}$$

Equation (4.1) can be established by showing that

$$\sum_{c,d,i,g,l,s,j,h} p(c)\cdot p(d|c)\cdot p(i)\cdot p(g|d,i)\cdot p(l|g)\cdot p(s|i)\cdot p(j|l,s)\cdot p(h|g,j) \;=\; 1. \tag{4.2}$$

Let $\prec$ be a topological ordering [18] of the variables $\{c,d,i,g,l,s,j,h\}$ in the ESBN DAG, say $c \prec d \prec i \prec g \prec s \prec l \prec j \prec h$. By marginalizing the variables in reverse order of $\prec$, the variable v_i being marginalized only appears in one CPT $p(v_i|P_i)$. By the definition of CPT, $\sum_{v_i} p(v_i|P_i) = 1(P_i)$. Then, the claim follows.

For example, here, h will be chosen to eliminated first. Then, by Lemma 4, the left side of Equation (4.2) can be rewritten as

$$\sum_{c,d,i,g,l,s,j}\sum_{h} p(c)\cdot p(d|c)\cdot p(i)\cdot p(g|d,i)\cdot p(l|g)\cdot p(s|i)\cdot p(j|l,s)\cdot p(h|g,j)$$

As h only appears in $p(h|g,j)$, by Lemma 5, Equation (4.3) can be equivalently rewritten as

$$\sum_{c,d,i,g,l,s,j} p(c)\cdot p(d|c)\cdot p(i)\cdot p(g|d,i)\cdot p(l|g)\cdot p(s|i)\cdot p(j|l,s)\cdot \sum_{h} p(h|g,j)$$

By definition of CPT, we have

$$\sum_{c,d,i,g,l,s,j} p(c)\cdot p(d|c)\cdot p(i)\cdot p(g|d,i)\cdot p(l|g)\cdot p(s|i)\cdot p(j|l,s)\cdot 1(g,j). \tag{4.3}$$

By definition of the unity potential,

$$1(g,j) \;=\; 1(g)\cdot 1(j). \tag{4.4}$$

Substituting Equation (4.4) into (4.3) yields

$$\sum_{c,d,i,g,l,s,j} p(c) \cdot p(d|c) \cdot p(i) \cdot p(g|d,i) \cdot p(l|g) \cdot p(s|i) \cdot p(j|l,s) \cdot 1(g) \cdot 1(j). \quad (4.5)$$

By the property of unity potential shown in Chapter 2, we have

$$p(g|d,i) \cdot 1(g) \;=\; p(g|d,i) \quad (4.6)$$

and

$$p(j|l,s) \cdot 1(j) \;=\; p(j|l,s). \quad (4.7)$$

By substituting Equations (4.6) and (4.7) into (4.5),

$$\sum_{c,d,i,g,l,s,j} p(c) \cdot p(d|c) \cdot p(i) \cdot p(g|d,i) \cdot p(l|g) \cdot p(s|i) \cdot p(j|l,s).$$

The remaining variables can be removed similarly, thereby establishing that the product of all CPTs in Table 2.4 is a joint probability distribution. This well-known proof is a special case of a more general result.

Lemma 6 [4] *Consider a Bayesian network on variables* $U = \{v_1, v_2, \ldots, v_n\}$ *with DAG* B *and CPTs* $C = \{p(v_1|P(v_1)),\ p(v_2|P(v_2)), \ldots, p(v_n|P(v_n))\}$. *Let* $C' = \{p(v_i|P(v_i)),\ p(v_j|P(v_j)), \ldots, p(v_l|P(v_l)), p(v_m|P(v_m))\}$ *be any non-empty subset of* C. *The product of the CPTs in* C' *is a CPT of the variables* X *given* Y, *where* $X = \{v_i, v_j, \ldots, v_l, v_m\}$ *and* $Y = P(v_i, v_j, \ldots, v_l, v_m)$.

Proof. Let $C' = \{p(v_i|P(v_i)), p(v_j|P(v_j)), \ldots, p(v_l|P(v_l)), p(v_m|P(v_m))\}$. Similar to the proof that the product of all Bayesian network CPTs is a joint probability distribution, we show that the product of the Bayesian network CPTs in

C' is a CPT by establishing that:

$$\sum_{X} p(v_i|P(v_i)) \cdot p(v_j|P(v_j)) \cdot \cdots \cdot p(v_l|P(v_l)) \cdot p(v_m|P(v_m)) \;=\; 1(Y). \qquad (4.8)$$

Let $\prec$ denote a topological ordering of the variables in B. Without loss of generality, let $v_i \prec v_j \prec \cdots \prec v_l \prec v_m$. This $\prec$ and the fact that B is a DAG mean that v_m can only appear in one CPT of C', namely, $p(v_m|P(v_m))$. Thereby, let us choose v_m as the first variable to eliminate.

By Lemma 4, the left side of Equation (4.8) can be rewritten as

$$\sum_{X-v_m} \sum_{v_m} p(v_i|P(v_i)) \cdot p(v_j|P(v_j)) \cdot \cdots \cdot p(v_l|P(v_l)) \cdot p(v_m|P(v_m)).$$

As v_m only appear in one CPT of C', By Lemma 5, we have

$$\sum_{X-v_m} p(v_i|P(v_i)) \cdot p(v_j|P(v_j)) \cdot \cdots \cdot p(v_l|P(v_l)) \cdot \sum_{v_m} p(v_m|P(v_m))$$

By the definition of CPT, it equals to

$$\sum_{X-v_m} p(v_i|P(v_i)) \cdot p(v_j|P(v_j)) \cdot \cdots \cdot p(v_l|P(v_l)) \cdot 1(P(v_m)). \qquad (4.9)$$

Now let us categorize $P(v_m)$ into two sets, namely $P(v_m) \cap X$ and $P(v_m) - X$. Then, Equation (4.9) can be rewritten as

$$\sum_{X-v_m} p(v_i|P(v_i)) \cdot p(v_j|P(v_j)) \cdot \cdots \cdot p(v_l|P(v_l)) \cdot 1(P(v_m) - X) \cdot 1(P(v_m) \cap X).$$

By Lemma 5,

$$1(P(v_m)-X) \cdot \sum_{X-v_m} p(v_i|P(v_i)) \cdot p(v_j|P(v_j)) \cdot \cdots \cdot p(v_l|P(v_l)) \cdot 1(P(v_m) \cap X). \qquad (4.10)$$

By definition, $(P(v_m) \cap X) \subseteq X$. Since $v_m \notin P(v_m)$,

$$(P(v_m) \cap X) \subseteq X - v_m.$$

That is, $(P(v_m) \cap X) \subseteq \{v_i, v_j, \ldots, v_l\}$. By definition of unity-potential, $1(P(v_m) \cap X)$ can be factorized into unity-potentials $1(v)$ on its singleton variables v, and each $1(v)$ can be multiplied with the CPT $p(v|P(v))$, where $v \in \{v_i, v_j, \ldots, v_l\}$. By the property of unity-potential, $p(v|P(v)) \cdot 1(v)$ gives $p(v|P(v))$. It follows that Equation (4.10) can be rewritten as

$$1(P(v_m) - X) \cdot \sum_{X - v_m} p(v_i|P(v_i)) \cdot p(v_j|P(v_j)) \cdot \cdots \cdot p(v_l|P(v_l)).$$

By a similar argument for variables $v_l, \ldots, v_j, v_i$, we have

$$\begin{aligned}
& \sum_{X} p(v_i|P(v_i)) \cdot p(v_j|P(v_j)) \cdot \cdots \cdot p(v_l|P(v_l)) \cdot p(v_m|P(v_m)) \\
= \; & 1(P(v_i) - X) \cdot 1(P(v_j) - X) \cdot \cdots \cdot 1(P(v_l) - X) \cdot 1(P(v_m) - X) \\
= \; & 1(P(v_i) \cup P(v_j) \cup P(v_l) \cup P(v_m) - X) \\
= \; & 1(P(v_i, v_j, \ldots, v_l, v_m) \\
= \; & 1(Y)
\end{aligned}$$

We obtain our desired result:

$$\sum_{X} p(v_i|P(v_i)) \cdot p(v_j|P(v_j)) \cdot \cdots \cdot p(v_l|P(v_l)) \cdot p(v_m|P(v_m)) = 1(Y).$$

By definition of CPT, the claim is proved.

Lemma 6 establishes that the product of any subset of CPTs from a Bayesian network is a CPT. Note that it is not guaranteed that the product is $p(X|Y)$, namely, a CPT defined with respect to the joint distribution $p(U)$. Therefore, we denote it as $\psi(X|Y)$ for now.

Example 10 *Consider the CPTs* $\{p(g|d,i), p(s|i), p(h|g,j)\}$, *which are a subset*

of the ESBN Bayesian network of Figure 2.1. By Lemma 6,

$$\psi(g, h, s|d, i, j) \;=\; p(g|d, i) \cdot p(s|i) \cdot p(h|g, j).$$

4.2 Structure of any Intermediate Distribution Constructed by VE

The notion of *expanded form* is introduced to express potentials built by VE equivalently in terms of multiplication and marginalization operators on a subset of Bayesian network CPTs.

Definition 4.2.1 [4] *Let ψ be any potential constructed during VE. When evidence is not observed, the expanded form of ψ is the unique expression defining how VE built ψ using the multiplication and marginalization operators on the Bayesian network CPTs in C.*

Example 11 *As evidence is assumed not be considered, the process of answering $p(j|h, i)$ is very similar to Example 3 except multiplying the evidence potentials.*

By VE, to answer the query, $p(h, i, j)$ needs to be calculated first as follows:

$$\begin{aligned}
& p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \sum_l p(l|g) \cdot p(j|l,s) \cdot \sum_d p(g|d,i) \cdot \sum_c p(c) \cdot p(d|c) \\
= \; & p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \sum_l p(l|g) \cdot p(j|l,s) \cdot \sum_d p(g|d,i) \cdot \sum_c \psi(c,d) \qquad (4.11) \\
= \; & p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \sum_l p(l|g) \cdot p(j|l,s) \cdot \sum_d p(g|d,i) \cdot \psi(d) \qquad (4.12) \\
= \; & p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \sum_l p(l|g) \cdot p(j|l,s) \cdot \sum_d \psi(d,g,i) \qquad (4.13) \\
= \; & p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \sum_l p(l|g) \cdot p(j|l,s) \cdot \psi(g,i) \qquad (4.14) \\
= \; & p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \psi(g,i) \cdot \sum_l \psi(g,j,l,s) \qquad (4.15) \\
= \; & p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \psi(g,i) \cdot \psi(g,j,s) \qquad (4.16) \\
= \; & p(i) \cdot \sum_g p(h|g,j) \cdot \psi(g,i) \cdot \sum_s \psi(g,i,j,s) \qquad (4.17) \\
= \; & p(i) \cdot \sum_g p(h|g,j) \cdot \psi(g,i) \cdot \psi(g,i,j) \qquad (4.18) \\
= \; & p(i) \cdot \sum_g \psi(g,h,i,j) \cdot \psi(g,i,j) \qquad (4.19) \\
= \; & p(i) \cdot \sum_g \psi(g,h,i,j) \qquad (4.20) \\
= \; & p(i) \cdot \psi(h,i,j) \qquad (4.21) \\
= \; & p(h,i,j). \qquad (4.22)
\end{aligned}$$

The rest steps are omitted, as it is not important to our claim. The expanded form of potential $\psi(g, h, i, j)$ in Equation (4.19) is:

$$\psi(g,h,i,j) \;=\; p(h|g,j) \cdot \sum_d p(g|d,i) \cdot \sum_c p(c) \cdot p(d|c),$$

which is determined recursively as follows:

$$\begin{aligned}
\psi(g,h,i,j) &= p(h|g,j)\cdot\psi(g,i)\\
&= p(h|g,j)\cdot\left(\sum_d \psi(d,g,i)\right)\\
&= p(h|g,j)\cdot\left(\sum_d (p(g|d,i)\cdot\psi(d))\right)\\
&= p(h|g,j)\cdot\left(\sum_d \left(p(g|d,i)\cdot\left(\sum_c psi(c,d)\right)\right)\right)\\
&= p(h|g,j)\cdot\left(\sum_d \left(p(g|d,i)\cdot\left(\sum_c (p(c)\cdot p(d|c))\right)\right)\right).
\end{aligned}$$

Note that the expanded form can be considered as evidence expanded form without evidence potential multiplied.

Definition 4.2.2 [4] *The expanded form of a potential ψ constructed during VE is said to be in normal form, if all marginalizations take place on the product of all CPTs used to build ψ.*

Example 12 *The expanded form of potential $\psi(g,h,i,j)$ in Example 11 is not in normal form, since, for instance, the marginalization of variable c takes place on a product not involving the CPTs $p(g|d,i)$ and $p(h|g,j)$.*

The next result is critical to applying our key observation.

Lemma 7 [4] *The expanded form of any potential ψ constructed during VE can always be equivalently rewritten in normal form.*

Proof. Let $\sum_{v_i}$ be any marginalization operator in the expanded form. There are two cases to consider.

First, let us consider the case that there is another marginalization $\sum_{v_j}$ to the immediate left of $\sum_{v_i}$. By Lemma 4, we can equivalently rewrite $\sum_{v_j} \sum_{v_i}$ as

$$\sum_{v_i} \sum_{v_j} .$$

Next, consider the second case when an multiplication operator appears to the immediate left of $\sum_{v_i}$, say $\psi_1 \cdot \sum_{v_i} \psi_2$. By construction of VE, all distributions involving v_i are multiplied together as ψ_2 before v_i is marginalized away. This means that v_i does not appear in ψ_1. By Lemma 5, $\psi_1 \cdot \sum_{v_i} \psi_2$ can be equivalently rewritten as

$$\sum_{v_i} \psi_1 \cdot \psi_2.$$

By repeated argument, $\sum_{v_i}$ can be pulled to the left of all multiplication operators in the expanded form.

This argument holds for all other marginalization signs. By definition, the expanded form is equivalently rewritten into normal form. □

Example 13 *In Example 11, the expanded form of potential* $\psi_4(b, f)$ *can be*

equivalently rewritten in normal form as follows:

$$
\begin{aligned}
& p(h|g,j) \cdot \left(\sum_d \left(p(g|d,i) \cdot \left(\sum_c (p(c) \cdot p(d|c)) \right) \right) \right) \\
= \; & p(h|g,j) \cdot \left(\sum_d \left(\sum_c (p(g|d,i) \cdot (p(c) \cdot p(d|c))) \right) \right) \\
= \; & p(h|g,j) \cdot \left(\sum_c \left(\sum_d (p(g|d,i) \cdot (p(c) \cdot p(d|c))) \right) \right) \\
= \; & \sum_c \left(p(h|g,j) \cdot \left(\sum_d (p(g|d,i) \cdot (p(c) \cdot p(d|c))) \right) \right) \\
= \; & \sum_c \left(\sum_d (p(h|g,j) \cdot (p(g|d,i) \cdot (p(c) \cdot p(d|c)))) \right) \\
= \; & \sum_{c,d} p(h|g,j) \cdot p(g|d,i) \cdot p(c) \cdot p(d|c).
\end{aligned}
$$

The main result of this manuscript is given next.

Theorem 1 [4] *Given a BN (B,C). During the process of performing VE to answer a query, every multiplication step and every marginalization step yields a CPT.*

Proof. Let ψ be any potential built during VE. By Definition 4.2.1, the expanded form of ψ is the unique expression defining how VE built ψ using the multiplication and marginalization operators on the Bayesian network CPTs in C.

By Lemma 7, the expanded form of ψ can be equivalently rewritten in normal form, say:

$$\sum_{X'} p(v_i|P(v_i)) \cdot p(v_j|P(v_j)) \cdots p(v_l|P(v_l)) \cdot p(v_m|P(v_m)).$$

Lemma 6 establishes that

$$\begin{aligned} & \psi(v_i v_j \ldots v_l v_m | P(v_i, v_j, \ldots, v_l, v_m)) \\ = \; & p(v_i|P(v_i)) \cdot p(v_j|P(v_j)) \cdots p(v_l|P(v_l)) \cdot p(v_m|P(v_m)). \end{aligned}$$

To eliminate any $v \in X'$, the VE algorithm requires that all distributions involving the variable v must be multiplied together before v is marginalized away. In other words, for any variable $v \in X'$, $p(v|P(v)) \in \{p(v_i|P(v_i)), p(v_j|P(v_j)), \ldots, p(v_l|P(v_l)), p(v_m|P(v_m))\}$. It follows that X' is a subset of $\{v_i, v_j, \ldots, v_l, v_m\}$. Thus, by definition of CPT,

$$\sum_{X'} \psi(v_i v_j \ldots v_l v_m | P(v_i, v_j, \ldots, v_l, v_m))$$

yields a CPT. Therefore, any potential built by the VE algorithm is a CPT. □

Example 14 *Recall the steps to answer query $p(j|h, i)$ in Example 11. By Theorem 1, we now have:*

$$
\begin{aligned}
& p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \sum_l p(l|g) \cdot p(j|l,s) \cdot \sum_d p(g|d,i) \cdot \sum_c p(c) \cdot p(d|c) \\
= \;& p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \sum_l p(l|g) \cdot p(j|l,s) \cdot \sum_d p(g|d,i) \cdot \sum_c \psi(c,d) \qquad (4.23) \\
= \;& p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \sum_l p(l|g) \cdot p(j|l,s) \cdot \sum_d p(g|d,i) \cdot \psi(d) \qquad (4.24) \\
= \;& p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \sum_l p(l|g) \cdot p(j|l,s) \cdot \sum_d \psi(d,g|i) \\
= \;& p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \sum_l p(l|g) \cdot p(j|l,s) \cdot \psi(g|i) \\
= \;& p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \psi(g|i) \cdot \sum_l \psi(j,l|g,s) \\
= \;& p(i) \cdot \sum_g p(h|g,j) \cdot \sum_s p(s|i) \cdot \psi(g|i) \cdot \psi(j|g,s) \\
= \;& p(i) \cdot \sum_g p(h|g,j) \cdot \psi(g|i) \cdot \sum_s \psi(j,s|g,i) \\
= \;& p(i) \cdot \sum_g p(h|g,j) \cdot \psi(g|i) \cdot \psi(j|g,i) \\
= \;& p(i) \cdot \sum_g \psi(g,h|i,j) \cdot \psi(j|g,i) \\
= \;& p(i) \cdot \sum_g \psi(g,h,j|i) \\
= \;& p(i) \cdot \psi(h,j|i) \\
= \;& p(h,i,j).
\end{aligned}
$$

The significance of Theorem 1 is that the potentials $\psi(c,d)$, $\psi(d)$, $\psi(d,g,i)$, $\psi(g,i)$, $\psi(g,j,l,s)$, $\psi(g,j,s)$, $\psi(g,i,j,s)$, $\psi(g,i,j)$, $\psi(g,h,i,j)$, $\psi(g,h,i,j)$ *and* $\psi(h,i,j)$ *constructed in Equations (4.11) - (4.21) of Example 11 are actually the CPTs* $\psi(c,d)$, $\psi(d)$, $\psi(d,g|i)$, $\psi(g|i)$, $\psi(j,l|g,s)$, $\psi(j|g,s)$, $\psi(j,s|g,i)$, $\psi(j|g,i)$, $\psi(g,h|i,j)$, $\psi(g,h,j|i)$ *and* $\psi(h,j|i)$, *respectively.*

It should be emphasized here that the notation $\psi(c, d)$ in Equations (4.11) and (4.23) are totally different. $\psi(c, d)$ in Equation (4.11) represent a potential with no pattern implied in the distribution. On the contrary, $\psi(c, d)$ in Equation (4.23) is actually a CPT, which should be formally denoted as $\psi(c, d|\emptyset)$. Similar case for $\psi(d)$ in Equations (4.12) and (4.24).

Chapter 5

Denoting Semantics

In this chapter, we propose an algorithm, called Semantics in Inference (SI), to denote the semantics of intermediate distribution during VE algorithm.

5.1 Denoting Semantics with a Topological Constraint

In this section, we gave a condition based on topological ordering for denoting the intermediate distribution with respect to $p(U)$ in Bayesian inference.

To obtain our claim, the notion evidence normal form needs to be introduced here as follows:

Definition 5.1.1 *The evidence expanded form $F(\psi)$ of any potential ψ constructed by VE is in evidence normal form, if $F(\psi)$ is written as*

$$\gamma \cdot N,$$

where γ is the product of 1 *and all evidence potentials in $F(\psi)$, and N is the same factorization as $F(\psi)$ except without products involving evidence potentials.*

Recall $\psi(g, h = 0, i = 1, j)$ in (2.4). Then $F(\psi)$ is

$$p(h = 0|g, j) \cdot \sum_d p(g|d, i = 1) \cdot \sum_c p(c) \cdot p(d|c), \tag{5.1}$$

except that products involving evidence potentials are shown as having been taken due to lack of space. The corresponding evidence normal form $\gamma \cdot N$ is

$$\gamma \quad = \quad 1(h = 0, i = 1)$$

and N is

$$p(h|g, j) \cdot \sum_d p(g|d, i) \cdot \sum_c p(c) \cdot p(d|c).$$

Note that unlike expanded form and evidence expanded form, normal form introduced in Chapter 4 and evidence normal form here are very different. Evidence normal form requires that the product of evidence potentials is pulled to the left instead of the marginalization operators.

Lemma 8 *The evidence expanded form $F(\psi)$ of any potential ψ constructed by VE always can be equivalently written into evidence normal form, i.e., $F(\psi) = \gamma \cdot N$.*

Proof Since evidence variables can never be marginalized in VE, the claim follows from Lemma 5.

Observe that, by Theorem 1, N in evidence normal form is a CPT. In this thesis, we may denote evidence normal form $\gamma \cdot N$ simply as N with evidence γ

understood, since γ only serves to select configurations of N agreeing with the evidence.

Theorem 2 [3] *In a Bayesian network (B, C) defining a joint distribution $p(U)$, suppose VE computes a potential $\psi(X|Y)$ whose evidence normal form is $\gamma \cdot N$. Then,*

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

if there is a topological ordering $\prec$ of B in which the variables in XS appear consecutively, where S are the the variables eliminated in N.

Proof. Let $W = XS$ for simplicity. Suppose there exists a topological ordering $\prec$ of the variables in B in which the variables in W appear consecutively. Let V be the set of all variables appearing in $\prec$ before W. By definition, V and VW are both initial segments. It means that

$$p(V) = \prod_{v_i \in V} p(v_i|P(v_i)) \tag{5.2}$$

and

$$p(VW) = \prod_{v_i \in V} p(v_i|P(v_i)) \cdot \prod_{v_i \in W} p(v_i|P(v_i)). \tag{5.3}$$

By substituting Equation (5.2) into (5.3),

$$p(VW) = p(V) \cdot \prod_{v_i \in W} p(v_i|P(v_i)). \tag{5.4}$$

According to $\prec$, $Y \subseteq V$. Let $Z = V - Y$. So $V = YZ$. Thus, Equation (5.4) can be rewritten as

$$p(WYZ) = p(YZ) \cdot \prod_{v_i \in W} p(v_i|P(v_i)).$$

By proof of Theorem 1, $\prod_{v_i \in W} p(v_i|P(v_i)) = \psi(W|Y)$. Thus, by Lemma 5, marginalizing away Z yields

$$p(WY) = p(Y) \cdot \prod_{v_i \in W} p(v_i|P(v_i)).$$

By rearrangement, we have

$$p(W|Y) = \prod_{v_i \in W} p(v_i|P(v_i)). \tag{5.5}$$

Applying Lemma 5 on N in evidence normal form $\gamma \cdot N$, gives

$$N \;=\; \sum_{S} \prod_{v_i \in W} p(v_i|P(v_i)). \tag{5.6}$$

Substituting Equation (5.5) into (5.6) we obtain our desired result

$$N \;=\; p(X|Y).$$

Example 15 *Recall $\psi(g, i = 1, j)$ in (2.3). The evidence expanded form is*

$$\sum_{s}((p(s|i) \cdot 1(i = 1)) \cdot (\sum_{l}(p(l|g) \cdot p(j|l, s)))). \tag{5.7}$$

Its normal form $\gamma \cdot N$ is

$$\gamma \;\;=\;\; 1(i = 1) \tag{5.8}$$

and

$$N = \sum_{s} p(s|i) \cdot \sum_{l} p(l|g) \cdot p(j|l, s). \tag{5.9}$$

Thus, we have

$$X = \{j\} \quad \textit{and} \quad S = \{l, s\}$$

By testing topological ordering, we can see there exists a topological order that $\{j, l, s\}$ *appear consecutively, say* $c \prec d \prec i \prec g \prec l \prec s \prec j \prec h$. *According to Theorems 1 and 2, we have*

$$N \quad = \quad p(j|g, i).$$

Thus, $\psi(g, i = 1, j)$ *in (2.3) is* $p(j|g, i = 1)$.

The condition provided in this section seemingly did not utilize d-separation. Instead, semantics in this approach were based on testing for the existence of a particular topological order of a Bayesian network B and had $O(n!)$ time complexity, where n is the number of variables in the Bayesian network B under consideration. That means the method is not practical in reality.

5.2 The Semantics in Inference (SI) Algorithm

In this section, we will show a practical way to denote the semantics during VE.

To understand when $N = p(X|Y)$ in evidence normal form, some terminology is taken from [30].

A *path* from v_1 to v_n is a sequence $v_1, v_2, \ldots, v_n$ with arcs (v_i, v_{i+1}) in B, $i = 1, \ldots, n-1$. With respect to a variable v_i, three sets are defined as follows:

(i) the ancestors of v_i, denoted $A(v_i)$, are those variables having a path to v_i;

(ii) the descendants of v_i, denoted $D(v_i)$, are those variables to which v_i has a path;

(iii) the children of v_i are those variables v_j such that arc (v_i, v_j) is in B.

Then, the ancestors of a set $X \subseteq U$ are defined as $A(X) = (\cup_{v_i \in X} A(v_i)) - X$. The descendants $D(X)$ are defined similarly.

In practice, the ancestors and descendants of the variables in a DAG can be determined from the *transitive closure* [9]. Given a directed graph $G = (V, E)$ with vertex set $V = \{v_1, v_2, \ldots, v_n\}$, one concept called transitive closure is defined as follows:

Definition 5.2.1 *[9] The transitive closure of G is defined as the graph $G^* = (V, E^*)$, where*

$$E^* = \{(v_i, v_j) | \textit{ there is a path from vertex } v_i \textit{ to } v_j \textit{ in } G\}.$$

To obtain a transitive closure, an *adjacency matrix* [9] of the DAG must be established first. The detail of how to compute a transitive closure of a graph is illustrated in Appendix B.

Example 16 *Given the ESBN DAG in 2.1, the adjacency matrix, denoted M,*

of the ESBN is:

$$M = \begin{array}{c} \\ c \\ d \\ i \\ g \\ l \\ s \\ j \\ h \end{array} \begin{array}{c} \begin{array}{cccccccc} c & d & i & g & l & s & j & h \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}.$$

The transitive closure T of the extended student BN is then:

$$T = \begin{array}{c} \\ c \\ d \\ i \\ g \\ l \\ s \\ j \\ h \end{array} \begin{array}{c} \begin{array}{cccccccc} c & d & i & g & l & s & j & h \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}.$$

For each variable, the columns of T represent its ancestors, while the rows of T represent its descendants. For example,

$$A(g) = \{c, d, i\} \quad \text{and} \quad D(l) = \{j, h\}.$$

I now propose the *Semantics in Inference* (SI) algorithm, which uses d-separation to denote the semantics of any potential ψ built by VE on a discrete Bayesian network B. Each potential ψ constructed by VE is represented in evidence normal form $\psi(X|Y)$. If the semantics of B ensure the $\psi(X|Y) = p(X|Y)$, then ψ is denoted as $p_B(X|Y)$; otherwise, it is denoted as $\phi_B(X|Y)$. S represents the set of variables eliminated to build ψ, which can be obtained from $F(\psi)$. $A(XS)$ and $D(XS)$ are computed from the *transitive closure*, denoted T, of B [9].

Algorithm 3. SI(ψ)

1 Compute the evidence expanded form $F(\psi)$ of ψ
2 Compute the normal form $\gamma \cdot N$ of $F(\psi)$
3 Compute the CPT structure $\psi(X|Y)$ of N
4 Compute $Z = A(XS) \cap D(XS)$
5 Compute $X_1 = X \cap P(Z)$
6 **if** $I_B(X_1, \emptyset, Y)$ holds in B by d-separation
7 **return** $p_B(X|Y)$
8 **else**
9 **return** $\phi_B(X|Y)$

Algorithm 3. Semantics in Inference (SI) Algorithm.

I use the following two example to illustrate how SI algorithm works. Example 17 is for p and Example 18 is for ϕ.

Example 17 *Recall $\psi(g, i = 1, j)$ in (2.3). The evidence expanded form is shown as Equation (5.7) in Example 15. Its normal form $\gamma \cdot N$ is shown as Equations (5.8) and (5.9) in Example 15. According to Theorem 1, we have*

$$N = \psi(j|g, i).$$

Now $X = \{j\}$, $Y = \{g, i\}$ and $S = \{l, s\}$. By the transitive closure T of the ESBN in Example 16,

$$A(XS) = \{c, d, i, g\}$$

and

$$D(XS) = \{h\}.$$

Hence,

$$Z = (XS) \cap D(XS) = \emptyset.$$

It follows that $P(Z) = \emptyset$. By definition,

$$X_1 = P(Z) \cap X = \emptyset.$$

Trivially, $I_B(X_1, \emptyset, Y)$ holds. Therefore, SI denotes $\psi(g, i = 1, j)$ in (2.3) as $p_B(j|g, i = 1)$.

Example 18 *Now consider $\psi(g, h = 0, i = 1, j)$ in (2.4). The evidence expanded form is (5.1). Its normal form $\gamma \cdot N$ is shown in Equations (5.2) and (5.2).*

According to Theorem 1, we have

$$N = \psi(g, h|i, j).$$

It follows that $X = \{g, h\}$, $Y = \{i, j\}$ *and* $S = \{c, d\}$. *From* T *of the ESBN in Example 16, we have*

$$A(\{c, d, g, h\}) = \{i, j, l, s\}$$

and

$$D(\{c, d, g, h\}) = \{j, l\}.$$

Thus,

$$Z = \{j, l\},$$

which gives $P(Z) = \{g, s\}$. *By definition,*

$$X_1 = \{g\}.$$

Now, by d-separation, $I(X_1, \emptyset, Y)$, *which is* $I(g, \emptyset, ij)$ *in current case, does not hold. Thereby, SI denotes* $\psi(g, h = 0, i = 1, j)$ *in (2.4) as* $\phi_B(g, h = 0|i = 1, j)$.

In Example 18, there is a path from $XS = \{c, d, g, h\}$ back to XS through $Z = \{j, l\}$. It should be noticed that the starting point of this path is X_1, which is g in B. Thus, the story behind testing $I_B(X_1, \emptyset, Y)$ by d-separation is actually to check if there is such "second" path existing or not.

It perhaps should be emphasized that when deciding semantics of $\psi(X|Y)$, the independence to be tested is $I_B(X_1, \emptyset, Y)$ and not $I_B(XS, Y, A(XSY))$. In Figure 3.1 (left), $I_B(abd, c, \emptyset)$ holds, but $p(b, d|c) \neq \psi(b, d|c)$ in (3.5) is possible.

Chapter 6

Computational Properties of SI

We establish four salient features of SI, namely, time complexity, soundness, completeness and strong completeness.

6.1 Polynomial Time Complexity

The efficiency of an algorithm is stated as a function relating the input length to the number of steps (time complexity) required to execute the algorithm. If the complexity of an algorithm is polynomial, that means this algorithm is efficient. On the contrary, the result that an algorithm has nonpolynomial complexity indicates this algorithm can't be utilized in practice.

In this section, we will show that SI algorithm has the polynomial time complexity.

Theorem 3 [5] *Let ψ be any potential built by VE during exact inference in a discrete Bayesian network with n variables. Then the time complexity of the SI*

algorithm to determine the semantics of ψ is $O(n^3)$.

Proof. As the worst case to compute a ψ in VE requires $n-1$ multiplications and n marginalizations, computing $F(\psi)$ takes $2n$ steps. The normal form $\gamma \cdot N$ can be decided in linear time, as can the CPT structure $\psi(X|Y)$ of N.

The transitive closure T of the directed acyclic graph can be computed in $O(n^3)$ [9]. Let XS be a set of k variables, $1 \leq k \leq n$. Then, for each variable $v \in XS$, we only need to go through the remaining $n-k$ variables to see if it belongs to $A(v)$ or $D(v)$. Thus, $A(XS)$ and $D(XS)$ each can be computed in $O(k \cdot n)$.

Now Z and X_1 each can be computed in $O(n^2)$. Testing $I_B(X_1, \emptyset, Y)$ is linear in the size of B [11]. Thus, the semantics of ψ can be determined by SI in $O(n^3)$.

Note that the purpose of Theorem 3 is to show that the time complexity of SI is polynomial. Therefore, the time complexity analysis for each SI step may not be the most efficient one.

6.2 Soundness

We now turn to soundness. Before the formal proof is given, a few results are illustrated first.

Lemma 9 *[6] In SI, $I_B(X_1, \emptyset, Y)$ holds $\Longleftrightarrow X_1 = \emptyset$.*

Proof. ($\Leftarrow$) Given $X_1 = \emptyset$, then $I_B(X_1, \emptyset, Y)$ holds trivially.

($\Rightarrow$) Now the condition that $I_B(X_1, \emptyset, Y)$ holds is given. Let us categorize the situation into two cases.

For the first case, suppose $Y = \emptyset$. As $Y = P(XS) = \emptyset$,

$$A(XS) = \emptyset.$$

By definition of Z,

$$Z = \emptyset,$$

It follows that

$$P(Z) = \emptyset.$$

By definition of X_1, we have $X_1 = \emptyset$.

The second case is $Y \neq \emptyset$. Let us prove by contradiction, suppose $v_i \in X_1$. By definition of X_1,

$$v_i \in P(Z).$$

Thus, there exists a $v_j \in Z$ with $(v_i, v_j) \in B$. By definition of Z,

$$v_j \in A(XS).$$

If $v_j \in P(XS)$, then (v_i, v_j) is the edge from X_1 to Y. Thus, $I_B(X_1, \emptyset, Y)$ does not hold. This is a contradiction to the given condition. Otherwise, if $v_j \notin P(XS)$, there exists a v_k satisfying

$$v_k \in P(XS) \quad \textit{and} \quad v_k \in D(v_j).$$

Since $v_i \in P(v_j)$,

$$v_k \in D(v_i).$$

That means there is a path from v_i to v_k, which is the path from X_1 to Y. Thus, $I_B(X_1, \emptyset, Y)$ does not hold. Again, this is a contradiction to the given condition. Therefore, $X_1 = \emptyset$.

Lemma 10 *[6] In SI,* $X_1 = \emptyset \iff A(XS) \cap D(XS) = \emptyset$.

Proof. ($\Leftarrow$) Suppose $A(XS) \cap D(XS) = \emptyset$. By definition of Z,

$$Z = A(XS) \cap D(XS) = \emptyset.$$

It follows that

$$P(Z) = \emptyset.$$

By definition of X_1,

$$X_1 = X \cap P(Z) = \emptyset.$$

($\Rightarrow$) Given $X_1 = \emptyset$. Let us prove by contradiction. Suppose $Z \neq \emptyset$. By definition of Z, $Z \subseteq D(XS)$. That means there is a path from a variable v_i in XS to v_l in Z. Let v_k be the child of v_i on this path v_i to v_l, where $v_k \notin XS$. It follows $v_k \in A(Z)$. As $Z \in A(XS)$, $v_k \in A(XS)$. Since v_k is child of v_i, $v_k \in D(XS)$. Thus, $v_k \in Z$. Another case is that v_l is a child of v_i. But no matter which case it is, there must exist an edge $(v_i, v_k) \in B$, where

$$v_i \in XS$$

and

$$v_k \in Z. \tag{6.1}$$

Suppose $v_i \in S$. By VE, all CPTs involving v_i will have been multiplied together, which includes $p(v_k|P(v_k))$ as v_k is a child of v_i in B. This would mean that

$$v_k \in XS,$$

which is a contradiction to Equation (6.1), as $XS \cap Z = \emptyset$ by definition. Therefore, $v_i \notin S$. Since $v_i \in XS$, it implies

$$v_i \in X. \tag{6.2}$$

Now, as $v_k \in Z$,

$$v_i \in P(Z). \tag{6.3}$$

By Equations (6.2) and (6.3) and definition of X_1,

$$v_i \in X_1,$$

which is a contradiction to $X_1 = \emptyset$. Hence, our assumption is not correct. Therefore, $A(XS) \cap D(XS) = \emptyset$.

Lemma 11 *[6] Given a discrete Bayesian network B on U and $V \subseteq U$. There exists a topological ordering $\prec$ where the variables in V appear consecutively if and only if $A(V) \cap D(V) = \emptyset$.*

Proof. Suppose $A(V) \cap D(V) \neq \emptyset$. Then there is at least one variable v_j not in V that is both a descendant of a variable v_i in V and an ancestor of a variable

v_k in V. This means that no topological ordering exists where the variables in V appear consecutively.

Suppose $A(V) \cap D(V) = \emptyset$. By definition, $V \cap D(V) = \emptyset$. This means every element in $D(V)$ is in $U - (VA(V))$. Since $V \cap A(V) = \emptyset$, a topological ordering $\prec$ can be constructed based on the directed acyclic graph of B in which the variables in $A(V)$ appear consecutively first, followed by all variables in V, followed by all variables in $U - (VA(V))$ a shown in Figure 6.1. Hence, V appear consecutively in this topological ordering.

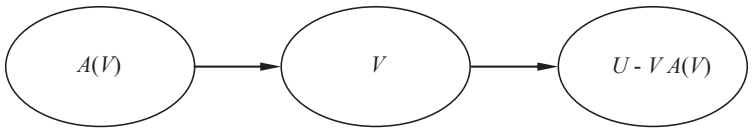

Figure 6.1: An illustration of the DAG pattern where V appear consecutively in topological ordering of B.

Now the soundness of SI is shown as the following theorem.

Theorem 4 [6] *In a Bayesian network* (B, C) *defining a joint distribution* $p(U)$*, suppose VE computes a potential* ψ *whose evidence normal form is* $\gamma \cdot N$*. If SI denotes the semantics of* N *as* $p_B(X|Y)$*, then* $N = p(X|Y)$*.*

Proof. As the precondition described, SI denotes the semantics of N as $p_B(X|Y)$. By SI, $I_B(X_1, \emptyset, Y)$ holds. By Lemmas 9, 10, and the proof of 11, there exists a topological order of B starting with $A(XS)$ and then followed

by XS. By definition, $A(W)$ is an initial segment, where W denotes XS for simplicity. By properties of initial segments,

$$p(A(W)) \;=\; \prod_{v_i \in A(W)} p(v_i|P(v_i)). \tag{6.4}$$

Similarly, we have $A(W)W$ is also an initial segment. It follows

$$p(A(W)W) = \prod_{v_i \in A(W)} p(v_i|P(v_i)) \cdot \prod_{v_i \in W} p(v_i|P(v_i)). \tag{6.5}$$

By manipulation of Equations (6.4) and (6.5),

$$p(W|A(W)) \;=\; \prod_{v_i \in W} p(v_i|P(v_i)). \tag{6.6}$$

Consider any $v_i \in W$ and $v_j \in A(W)$. If $v_j \in D(v_i)$, then $v_j \in D(W)$. It follows that $v_j \in A(W) \cap D(W)$. By Lemmas 9, 10 and 11, it implies $I_B(X_1, \emptyset, Y)$ does not hold, which is a contradiction to the precondition. Thus,

$$v_j \notin D(v_i),$$

which means that there is no path from v_i to v_j. As v_i and v_j can be any variable in W and $A(W)$, there is no path from W to $A(W)$. In other words, all paths in B are from $A(W)$ to W.

Since all paths from $A(W)$ to W necessarily go through $Y = P(W)$, $I_B(W, Y, A(W) - Y)$ holds in B by d-separation. By definition of conditional independencies, we have

$$p(W|A(W)) \;=\; p(W|Y). \tag{6.7}$$

By (6.6) and (6.7),

$$p(W|Y) \;=\; \prod_{v_i \in W} p(v_i|P(v_i)).$$

Summing out S on both sides yields

$$p(X|Y) \;=\; \sum_{S} \prod_{v_i \in W} p(v_i|P(v_i)). \tag{6.8}$$

Applying Lemma 5 on N in evidence normal form $\gamma \cdot N$, gives

$$N \;=\; \sum_{S} \prod_{v_i \in W} p(v_i|P(v_i)). \tag{6.9}$$

By (6.8) and (6.9), we have our claim

$$N \;=\; p(X|Y).$$

Theorem 4 guarantees that if SI denotes the semantics of a VE potential ψ as $\gamma \cdot p_B(X|Y)$, then

$$\psi \;=\; \gamma \cdot p(X|Y).$$

Recall potential $\psi(g, i = 1, j)$ in (2.3). Since SI denotes it as $p_B(j|g, i = 1)$, Theorem 4 ensures that $\psi(g, i = 1, j)$ is equal to $p(j|g, i = 1)$ as illustrated in Table 6.1.

Table 6.1: Potential $\psi(g, i = 1, j)$ in (2.3) is $p(j|g, i = 1)$.

$\psi(g, i = 1, j) \;=\; p(j|g, i = 1) \;=$

i	g	j	$p_B(j\|g, i = 1)$
1	0	0	0.457
1	0	1	0.543
1	1	0	0.334
1	1	1	0.666

6.3 Completeness

With respect to inference, the question of completeness is this. Can SI determine every VE potential defined with respect to the joint distribution? The answer is no, due to the next result.

Lemma 12 *[6] Using B defining $p(U)$, $I_p(X_1, \emptyset, Y) \iff$ VE builds $p(X|Y)$, where X_1 is defined in SI.*

Proof. The claim follows from the discussion in [18], where, in the notation of SI, $I_B(X_1, \emptyset, Y) \iff$ VE builds $p(X|Y)$, under the assumption that $I_B(X_1, \emptyset, Y) \iff I_p(X_1, \emptyset, Y)$.

As discussed in Chapter 2, it is not feasible to test every $I_p(X_1, \emptyset, Y)$ in $p(U)$. Thus, we rely on d-separation to test $I_B(X_1, \emptyset, Y)$ in B. However, it is known that independencies in $p(U)$ can escape detection in B. This means that SI will make mistakes in certain situations. For example, in the Bayesian network B in Figure 3.1 with the CPTs in Table 3.2 defining $p(U)$, $I_p(b, \emptyset, c)$ holds. By Lemma 12, (3.4) does yield $p(b, d|c)$, as shown in (3.6). Unfortunately, $I_B(b, \emptyset, c)$ does not hold. This means instead of correctly denoting $p(b, d|c)$ as $p_B(b, d|c)$, SI incorrectly denotes $p(b, d|c)$ as $\phi_B(b, d|c)$. However, d-separation and SI satisfy a weaker notion of completeness.

Note that d-separation satisfies a weaker completeness as shown in Lemma 2. This result can be utilized to show a similar kind of completeness for SI. First, one more result is needed.

Lemma 13 *[6] Suppose VE computes* $\psi'(X - v_i|Y) = \sum_{v_i} \psi(X|Y)$. *Then* ψ' *and* ψ *are both* p *or both* ϕ.

Proof. It is known that

$$p(X - v_i|Y) = \sum_{v_i} p(X|Y).$$

Now consider $\sum_{v_i} \phi(X|Y)$, where $\phi(X|Y) \neq p(X|Y)$. By Lemma 12, $\phi(X|Y)$ means $I_p(X_1, \emptyset, Y)$ does not hold, where $X_1 = X \cap P(Z)$.

Now marginalization gives $\psi(X'|Y)$, where $X' = X - v_i$. Note that Y did not change. As

$$X' = X - v_i$$

and

$$S' = Sv_i,$$

we have

$$\begin{aligned} X'S' &= (X - v_i) \cup (Sv_i) \\ &= XS. \end{aligned} \tag{6.10}$$

By definition of Z, Equation (6.10) implies that Z did not change after marginalization. Thus, $P(Z)$ did not change.

Now suppose $v_i \in P(Z)$. Then there exists a $v_k \in Z$ with $(v_i, v_k) \in B$. Similar to the proof of Lemma 10, this means $v_k \in XS$, which is a contradiction to $v_k \in Z$. Thus, $v_i \notin P(Z)$. By definition of X_1, $v_i \notin X_1$. Then, it follows that

$$\begin{aligned} X_1 &= X \cap P(Z) \\ &= (X - v_i) \cap P(Z). \end{aligned} \tag{6.11}$$

As $P(Z)$ does not change, by definition, we have X_1' after marginalization as

$$\begin{aligned} X_1' &= X' \cap P(Z) \\ &= (X - v_i) \cap P(Z). \end{aligned} \tag{6.12}$$

By Equations (6.11) and (6.12),

$$X_1 = X_1'.$$

Since Y does not change after marginalization,

$$I_p(X_1, \emptyset, Y) = I_p(X_1', \emptyset, Y).$$

As $I_p(X_1, \emptyset, Y)$ does not hold, $I_p(X_1', \emptyset, Y)$ does not hold. By Lemma 12,

$$\psi(X'|Y) = \phi(X - v_i|Y).$$

Theorem 5 [6] *In a Bayesian network B on U, suppose VE computes a potential ψ whose evidence normal form is $\gamma \cdot N$. If SI denotes the semantics of N as $\phi_B(X|Y)$, there exists a set C of CPTs for B defining a joint distribution $p(U)$ such that $N \neq p(X|Y)$.*

Proof. By SI, $I_B(X_1, \emptyset, Y)$ does not hold. There exists a C for B defining $p(U)$ such that, by Lemma 2,

$$p(Y) \neq p(Y|X_1). \tag{6.13}$$

We define an initial segment with four pairwise disjoint subsets: $W = XS$, Y, $Z_1 = Z - Y$ and $V = A(W) - YZ_1$. By property of initial segment,

$$p(WYZ_1V) = \prod_{v_i \in W} p(v_i|P(v_i)) \cdot \prod_{v_i \in VZ_1Y} p(v_i|P(v_i)).$$

Thus, we have

$$p(WY) = \sum_{VZ_1} \prod_{v_i \in W} p(v_i|P(v_i)) \cdot \prod_{v_i \in VZ_1Y} p(v_i|P(v_i)).$$

Let us prove by contradiction. Suppose the product of W's CPTs is $p(W|Y)$. This means

$$p(WY) = \sum_{VZ_1} p(W|Y) \cdot \prod_{v_i \in VZ_1Y} p(v_i|P(v_i)). \tag{6.14}$$

As $WY \cap VZ_1 = \emptyset$, by Lemma 5, Equation (6.14) can be rewritten as

$$p(WY) \;=\; p(W|Y) \cdot \sum_{VZ_1} \prod_{v_i \in VZ_1Y} p'(v_i|P(v_i)),$$

By rearrangement, we have

$$\begin{aligned} p(Y) &= \frac{p(WY)}{p(W|Y)} \\ &= \sum_{VZ_1} \prod_{v_i \in VZ_1Y} p(v_i|P(v_i)). \end{aligned}$$

By Theorem 1,

$$p(Y) = \sum_{VZ_1} \psi(VZ_1Y|P(VZ_1Y)). \tag{6.15}$$

We now show $P(VZ_1Y) = X_1$. First, consider $P(VZ_1Y) \subseteq X_1$. To prove it, let $X_2 = X - X_1$. Observe that as V, Z_1, Y, X, S are initial segment, $P(V), P(Z_1), P(Y)$ must be a subset of $XSVZ_1Y$. Now let us consider $P(V), P(Z_1)$, and $P(Y)$ one by one. For $P(V)$, by definition,

$$P(V) \nsubseteq V.$$

Also,

$$P(V) \nsubseteq S.$$

Otherwise, by VE, the CPT $p(v_i|P(v_i))$ will be multiplied for $v_i \in V$. It means $V \in XS$, which is a contradiction to the definition of V. It can also be proved that

$$P(V) \nsubseteq X.$$

Otherwise, $V \subseteq D(XS)$. By definition, $V \subseteq A(XS)$. It indicates that $V \in Z$, which is a contradiction to the definition of V. Moreover,

$$P(V) \nsubseteq Z_1.$$

This is because $Z_1 \subseteq Z$, which means $Z_1 \subseteq D(XS)$. If $P(V) \subseteq Z_1$, $V \subseteq D(XS)$. The rest is same as the proof of $P(V) \nsubseteq X$. Therefore, we have

$$P(V) \subseteq Y. \tag{6.16}$$

For $P(Z_1)$, by definition,

$$P(Z_1) \nsubseteq Z_1.$$

Similar to the proof of $P(V) \nsubseteq S$,

$$P(Z_1) \nsubseteq S.$$

Also, we have

$$P(Z_1) \nsubseteq X_2.$$

This is because if $P(Z_1) \subseteq X_2$, $P(Z_1) \cap Y = \emptyset$ as by definition, $X_2 \cap Y = \emptyset$. By definition, $P(Z) = P(Z_1) \cup P(Y) - YZ_1$. Thus, $X_2 \cap P(Z) \neq \emptyset$. As $X_2 \subseteq X$, by definition, $X_2 \cap X_1 \neq \emptyset$, which is contradict to the definition of X_2. Therefore,

$$P(Z_1) \subseteq X_1VY. \tag{6.17}$$

For $P(Y)$, by definition,

$$P(Y) \nsubseteq Y.$$

Similar to the proof of $P(V) \nsubseteq S$,

$$P(Y) \nsubseteq S.$$

Also, similar to the proof of $P(Z_1) \nsubseteq X_2$,

$$P(Y) \nsubseteq X_2.$$

Thus, we have

$$P(Y) \nsubseteq X_1 V Z_1. \tag{6.18}$$

By Equations (6.16), (6.17) and (6.18),

$$P(V) \cup P(Z_1) \cup P(Y) \quad \subseteq \quad X_1 V Z_1 Y.$$

By definition,

$$P(V Z_1 Y) \subseteq X_1. \tag{6.19}$$

To show $X_1 \subseteq P(V Z_1 Y)$, let $Y_1 = Y \cap Z$ and $Y_2 = Y - Y_1$. By definition of X_1,

$$X_1 \subseteq P(Z). \tag{6.20}$$

Since $X \cap VY = \emptyset$, $X_1 \subseteq X$ and $Y_2 \subseteq Y$,

$$X_1 \cap V Y_2 = \emptyset. \tag{6.21}$$

By Equations (6.20) and (6.21),

$$X_1 \subseteq P(Z) - V Y_2.$$

By definition, $P(Z) \cap Z = \emptyset$. Thus,

$$X_1 \subseteq P(Z) - VY_2Z.$$

It follows that

$$X_1 \subseteq P(V) \cup P(Z) \cup P(Y_2) - VZY_2.$$

By definition,

$$X_1 \subseteq P(VZY_2). \tag{6.22}$$

Note that by definition, $Z = Z_1Y_1$. Therefore, we have

$$\begin{aligned} P(VZY_2) &= P(VZ_1Y_1Y_2) \\ &= P(VZ_1Y). \end{aligned} \tag{6.23}$$

By substituting Equation (6.23) into (6.22),

$$X_1 \subseteq P(VZ_1Y). \tag{6.24}$$

By Equations (6.19) and (6.24),

$$X_1 = P(VZ_1Y). \tag{6.25}$$

By substituting Equation (6.25) into (6.15),

$$p(Y) = \sum_{VZ_1} \psi(VZ_1Y|X_1).$$

By Lemma 13,

$$p(Y) = \sum_{VZ_1} p(VZ_1Y|X_1).$$

Taking the marginalization gives

$$p(Y) \;\; = \;\; p(Y|X_1),$$

which is a contradiction to (6.13). Therefore,

$$p(W|Y) \;\; \neq \;\; \prod_{v_i \in W} p(v_i|P(v_i)).$$

By Lemma 13, marginalizing S from both sides gives

$$p(X|Y) \;\; \neq \;\; \sum_{S} \prod_{v_i \in W} p(v_i|P(v_i)). \tag{6.26}$$

By substituting Equation (6.9) in proof of Theorem 4 into (6.26), we obtain our desired result, namely, $p(X|Y) \neq N$.

Theorem 5 states that whenever SI indicates that a potential is not defined with respect to the joint distribution, then this is true for at least one set of CPTs for the given Bayesian network. Recall once again $\psi(g, h = 0, i = 1, j)$ in (2.4), which SI denotes as $\phi_B(g, h = 0, l|i = 1, j)$. With respect to $p(U)$ defined by the CPTs in Table 2.4, we have

$$\psi(g, h = 0, i = 1, j) \;\; \neq \;\; p(g, h = 0|i = 1, j).$$

6.4 Strong Completeness

Theorem 5 can be made significantly stronger based upon another property of d-separation.

Theorem 6 [5] *Except for a measure zero set in the space of all joint distributions $p(U)$ defined by all discrete Bayesian networks (B, C), for any potential ψ*

built by VE,

$$\psi \;=\; \gamma \cdot p(X|Y) \;\iff\; \text{SI denotes } \psi \text{ as } p_B(X|Y),$$

where $\gamma \cdot N$ is the evidence normal form of ψ.

Proof. ($\Rightarrow$) Suppose VE constructs a ψ whose evidence normal form is $\gamma \cdot N$ and whose semantics are defined with respect to $p(U)$. By contraposition, suppose SI denotes N as $\phi_B(X|Y)$. By SI, $I_B(X_1, \emptyset, Y)$ does not hold. Then, by Lemma 3, $I_p(X_1, \emptyset, Y)$ does not hold in essentially all possible $p(U)$ defined over B. It follows that for each such $p(U)$,

$$\gamma \cdot p(X|Y) \;\neq\; \gamma \cdot N.$$

A contradiction to our initial assumption. Therefore, SI correctly denotes the potential ψ as $p_B(X|Y)$.

($\Leftarrow$) Follows directly from Theorem 4.

Let B be any Bayesian network. Theorem 6 states that for nearly all choices C of CPTs for B, the SI algorithm correctly denotes the semantics of potentials constructed by VE during exact inference on B.

Chapter 7

Remarks on SI

In this chapter, some remarks on SI algorithm are provided.

7.1 An Alternate Approach for SI

By Lemmas 9 and 10, checking the specific CI in Algorithm 3 is equivalent to testing the ancestors and descendants of the variables XS involved in constructing a potential. This implies that the SI could be simplified as [6]:

Algorithm 4. Alternate SI(ψ)

1 Compute the evidence expanded form $F(\psi)$ of ψ

2 Compute the normal form $\gamma \cdot N$ of $F(\psi)$

3 Compute the CPT structure $\psi(X|Y)$ of N

4 **if** $A(XS) \cap D(XS) = \emptyset$

5 **return** $p_B(X|Y)$

6 **else**

7 **return** $\phi_B(X|Y)$

Algorithm 4. An Alternate Approach for SI.

This Alternate SI algorithm has a visual appeal to it in the sense that one simply determines whether or not there exists a path from any variable in XS to any other variable in XS involving at least one variable not in XS.

In Example 17, $X = \{j\}$ and $S = \{l, s\}$. By the transitive closure T of the ESBN in Figure 2.1, $A(XS) = \{c, d, g, i\}$ and $D(XS) = \{h\}$. Alternate SI denotes $\psi(g, i = 1, j)$ as $p_B(j|g, i = 1)$, since

$$A(XS) \cap D(XS) \quad = \quad \emptyset.$$

In Example 18, $X = \{g, h\}$ and $S = \{c, d\}$. T indicates $A(XS) = \{i, j, l, s\}$ and $D(XS) = \{j, l\}$. In this case, SI denotes potential $\psi(g, h = 0, i = 1, j)$ as $\phi_B(g, h = 0|i = 1, j)$, since

$$A(XS) \cap D(XS) \quad = \quad \{j, l\}.$$

On the other hand, the above version of SI does not emphasize the central role d-separation plays in semantics, namely,

$$I_B(X_1, \emptyset, Y) \iff A(XS) \cap D(XS) = \emptyset.$$

Thereby, even though $A(XS) \cap D(XS)$ is perhaps more intuitive and simple, I still utilize d-separation in this thesis for emphasizing the extension of the role of d-separation in BN inference.

7.2 Alternating between p and ϕ

In this section, we utilize an example to take a deep look on the alternate between p and ϕ during VE [3]. The key points are:

i) During the process of eliminating one variable, the semantics can alternate between p and ϕ.

ii) Once the elimination process involves a variable whose CPT is not collected by SO, we will keep getting ϕ until this "missing" CPT is multiplied.

Example 19 *Consider the elimination of variable b from the BN in Figure 7.1. According to SI, we have*

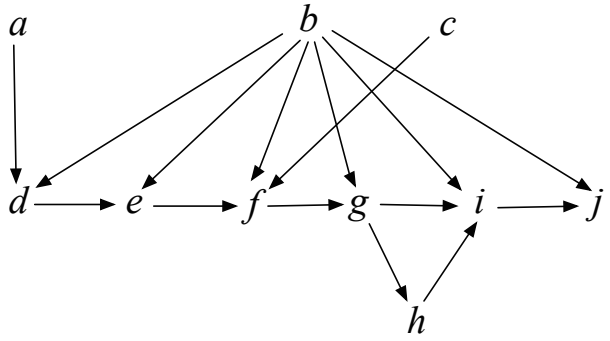

Figure 7.1: A BN to illustrate the alternating pattern of intermediate CPTs between p and ϕ.

$$
\begin{aligned}
& \sum_b p(b) \cdot p(e|b,d) \cdot p(d|a,b) \cdot p(g|b,f) \cdot p(f|b,c,e) \cdot p(i|b,g,h) \cdot p(j|b,i) \\
= & \sum_b \phi(b,e|d) \cdot p(d|a,b) \cdot p(g|b,f) \cdot p(f|b,c,e) \cdot p(i|b,g,h) \cdot p(j|b,i) && (7.1) \\
= & \sum_b p(b,d,e|a) \cdot p(g|b,f) \cdot p(f|b,c,e) \cdot p(i|b,g,h) \cdot p(j|b,i) && (7.2) \\
= & \sum_b \phi(b,d,e,g|a,f) \cdot p(f|b,c,e) \cdot p(i|b,g,h) \cdot p(j|b,i) && (7.3) \\
= & \sum_b p(b,d,e,f,g|a,c) \cdot p(i|b,g,h) \cdot p(j|b,i) && (7.4) \\
= & \sum_b \phi(b,d,e,f,g,i|a,c,h) \cdot p(j|b,i) && (7.5) \\
= & \sum_b \phi(b,d,e,f,g,i,j|a,c,h) && (7.6) \\
= & \ \phi(d,e,f,g,i,j|a,c,h). && (7.7)
\end{aligned}
$$

Example 19 demonstrates how the intermediate CPTs can alternate between p and ϕ. A ϕ can be obtained when multiplying the CPTs for b and e in Equation (7.1), since $I(b,\emptyset,d)$ dose not hold by d-separation. As the CPT $p(d|a,b)$ for

d has been collected by VE, a p can be subsequently re-obtained, as shown in Equation (7.2). Similar remarks hold for multiplying this product with the CPT for g before that for f, as shown in Equations (7.3) and (7.4).

However, sometime, this "second" path has a permanent influence on the semantics of VE's intermediate CPTs; it involves variables that are not children of the variable being eliminated. Recall Example 19 where variable b is being eliminated and consider Equations (7.4) and (7.5). Once the CPT $p(i|b, g, h)$ is multiplied, all CPTs subsequently constructed by VE during the elimination of b can have a ϕ as in Equations (7.5) - (7.7). The reason is that there is a path from b to i going through h. However, the CPT $p(h|g)$ is not collected by SO as $p(h|g)$ does not involve b. Hence, the only way to ensure a subsequent p is to wait for $p(h|g)$ to be multiplied during a different call to SO by VE, say to eliminate h.

7.3 Clarity of Presentation of Bayesian Network Inference

Whether d-separation is explicitly or implicitly used, SI brings improved clarity to denoting exact inference in Bayesian network texts, such as [8, 10, 11, 15, 17, 18, 33, 36, 40].

As we showed in Chapter 2, Bayesian network literature always denotes intermediate distributions as potentials during exact inference. However, as CPTs are a special case of potentials, denotation of potentials is obviously not precise as it should be. Chapter 4 provides a solution to this problem. For instance, the

intermediate distribution denoted as $\psi(g, i = 1, j)$ in (2.3) has a CPT pattern $\psi(j|g, i = 1)$.

Once we understand that every intermediate potential during exact inference is actually a CPT, it is natural that the question "is this CPT defined with respect to the jpd" comes out. Chapter 5 proposes the SI algorithm to answer this question. For example, by SI algorithm, the CPT $\psi(j|g, i = 1)$ in (2.3) is $p(j|g, i = 1)$. On the contrary, the CPT $\psi(g, h = 0, l|i = 1, j)$ in (2.4) is $\phi(g, h = 0, l|i = 1, j)$.

7.4 Role of d-separation in Bayesian Network Inference

The most important advantage this new work can bring is that it extends the role of d-separation during Bayesian network inference.

When recounting the development of Bayesian networks, Pearl [31] states that [32] made its greatest immediate impact through the notion of *d-separation*. As a method for deciding which conditional independence relations are implied by the directed acyclic graph of a Bayesian network, d-separation provides the semantics needed for defining and characterizing Bayesian networks.

[31] utilizes d-separation as the solutions to the following three practical problems:

i) how to characterize precisely the set of graphical transformations (e.g., arc reversals, node removals, node collapsing) that can legitimately be per-

formed on a network;

ii) how to test whether one network is entailed by or is equivalent to another;

iii) how to delineate the minimum information needed for answering a given query.

Observe that both i) and ii) emphasize the importance of d-separation with respect to Bayesian network modeling. With respect to inference, Pearl only states that d-separation can determine the minimum information needed for answering a query posed to a Bayesian network as presented in iii). However, it has been made in this thesis that d-separation can also provide semantics during Bayesian network inference.

I show that SI algorithm can denote the semantics of the intermediate distributions constructed during BN inference. Also, we established the completeness result of SI, which is exactly the same as d-separation. Therefore, SI should be considered as fundamental to BN inference, just as d-separation is considered fundamental to BN modeling.

7.5 A Possible Practical Advantage

A possible practical advantage of the semantics is the ability to construct messages using both VE and AR at the same join tree node. As discussed in Section 2, all previous join tree algorithms either exclusively apply VE or arc reversal (AR) [30] at all join tree nodes [23], or pick whether to apply VE or AR at each node [2].

Example 20 *Consider a BN in Figure 7.2. Its corresponding join tree with the assignment of the given CPTs is shown in Figure 7.3.*

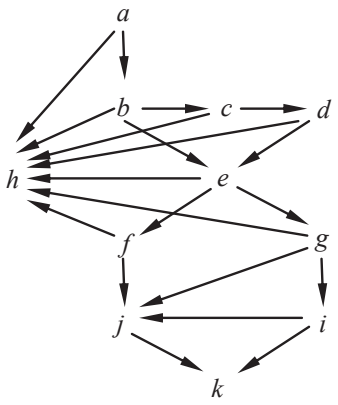

Figure 7.2: A Bayesian networks used to demonstrate a possible practical advantage of semantics.

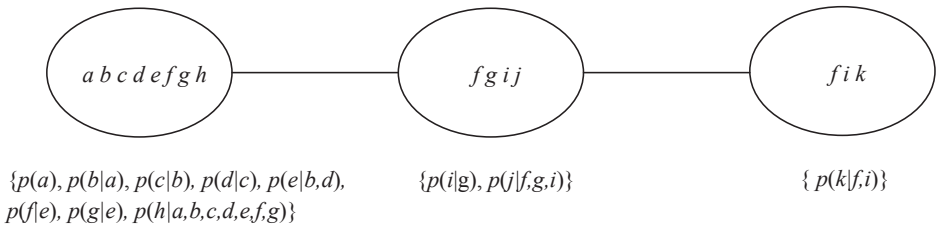

Figure 7.3: The join tree of the BN in Figure 7.2.

In [7], message identification process indicates that node $abcdefgh$ will pass $p(f)$ and $p(g|f)$ to node $fgij$, which, in turn, will pass $p(f)$ and $p(i|f)$ to node fik. Then, both Madsen [23] and Butz et al. [7] would apply AR at node $abcdefgh$ as follows assuming that the elimination ordering is alphabetical:

First, h is removed as a barren variable. Then, a is eliminated

$$\begin{aligned} p(a,b) &= p(a) \cdot p(b|a), \\ p(b) &= \sum_a p(a,b). \end{aligned}$$

Next, AR eliminate b. For its child c,

$$\begin{aligned} p(b,c) &= p(b) \cdot p(c|b), \\ p(c) &= \sum_b p(b,c), \\ p(b|c) &= \frac{p(b,c)}{p(c)}. \end{aligned}$$

For the child e,

$$\begin{aligned} p(b,e|c) &= p(b|c) \cdot p(e|b), \\ p(e|c) &= \sum_b p(b,e|c). \end{aligned}$$

To eliminate c, for its child d,

$$\begin{aligned} p(c,d) &= p(c) \cdot p(d|c), \\ p(d) &= \sum_c p(c,d), \\ p(c|d) &= \frac{p(c,d)}{p(d)}. \end{aligned}$$

For the child e,

$$\begin{aligned} p(c,e|d) &= p(c|d) \cdot p(e|c), \\ p(e|d) &= \sum_c p(c,e|d). \end{aligned}$$

To eliminate d,

$$\begin{aligned} p(d,e) &= p(d)\cdot p(e|d), \\ p(e) &= \sum_{d} p(d,e). \end{aligned}$$

Next, AR eliminate e. For its child f,

$$p(e,f) = p(e)\cdot p(f|e), \tag{7.8}$$

$$p(f) = \sum_{e} p(e,f), \tag{7.9}$$

$$p(e|f) = \frac{p(e,f)}{p(f)}. \tag{7.10}$$

For the child g,

$$p(e,g|f) = p(e|f)\cdot p(g|e), \tag{7.11}$$

$$p(g|f) = \sum_{e} p(e,g|f). \tag{7.12}$$

The message from $abcdefgh$ to $fgij$ is

$$p(f) \quad and \quad p(g|f).$$

Observe that AR must be applied to remove the last variable e.

However, by examining the semantics of VE, it can be verified that the elimination of variables a, b, c and d gives $p(e)$. In other words, successfully denoting the semantics of VE provides a chance to transfer from VE to AR. More specifi-

cally, apply VE to eliminate variables a, b, c and d as follows:

$$\begin{aligned}
& \sum_d \sum_c p(d|c) \cdot \sum_b p(c|b) \cdot p(e|b,d) \cdot \sum_a p(a) \cdot p(b|a) \\
= & \sum_d \sum_c p(d|c) \cdot \sum_b p(c|b) \cdot p(e|b,d) \cdot \sum_a p(a,b) \\
= & \sum_d \sum_c p(d|c) \cdot \sum_b p(c|b) \cdot p(e|b,d) \cdot p(b) \\
= & \sum_d \sum_c p(d|c) \cdot \sum_b \phi(b,c,e|d) \\
= & \sum_d \sum_c p(d|c) \cdot \phi(c,e|d) \\
= & \sum_d \sum_c p(c,d,e) \\
= & \sum_d p(d,e) \\
= & \ p(e).
\end{aligned}$$

Then, applying AR to eliminate variable e will performs Equations (7.8) to (7.12), which will yields exact the same result.

As it has been verified that eliminating variables by VE costs no more computation than applying AR [1], applying AR at $abcdefgh$ requires more computation than applying the combination of VE and AR as shown above.

However, since no investigation has been made on how frequently we can apply the combination of VE and AR, we leave this research as future work.

Chapter 8

Conclusion and Future Work

Pearl [33] emphasizes that probabilistic reasoning is not about numbers and is instead about the structure of reasoning. While this has been emphasized and exploited with respect to Bayesian network modeling, it has been missed with respect to Bayesian network inference. This thesis is the first to reveal the structure and semantics of the intermediate factors constructed during exact inference in discrete Bayesian networks.

In the case without evidence, we first established that every intermediate potential $\psi(X, Y)$ built by VE is, in fact, a CPT, say $\psi(X|Y)$. For example, to answer the query $p(b|d)$ posed to Bayesian network in Figure 3.1, VE computes the intermediate potential $\psi(b, c, d)$ in Equation (3.4). By Theorem 1, it can be verified that the intermediate potential $\psi(b, c, d)$ is indeed a CPT $\psi(b, d|c)$.

Having determined the CPT structure of VE's intermediate factors, we then studied the semantics of these intermediate CPTs. By semantics, we wanted to

know whether

$$\psi(X|Y) \;=\; p(X|Y),$$

for each CPT $\psi(X|Y)$ built by VE during inference. For example, for the intermediate CPT $\psi(g, i = 1, j)$ in (2.3),

$$\psi(g, i = 1, j) \;=\; p(j|g, i = 1).$$

On the contrary, for the intermediate CPT $\psi(g, h = 0, i = 1, j)$ in (2.4),

$$\psi(g, h = 0, i = 1, j) \;\neq\; p(g, h = 0|i = 1, j).$$

It is informative to note that in order to decide semantics of an intermediate CPT $\psi(X|Y)$, one must necessarily consider the set S of variables that were summed out. Our first technique for deciding semantics then tested for the existence of a topological ordering in which the variables XS appeared consecutively. This approach, while sound, is not practical with its $O(n!)$ time complexity. This lead us to propose another approach - one with surprising connections.

The main contribution of this thesis is the semantics of Inference (SI) algorithm. After some preprocessing steps, which we review below, a single independency statement is tested by d-separation. Hence, while d-separation is known to be central to Bayesian network modeling, the SI algorithm shows its importance to Bayesian network inference. But this also means that SI inherits the characteristic properties of d-separation, namely, the same results are obtained with respect to soundness, completeness, and strong completeness. In short, soundness means that when SI states that $\psi(X|Y) = p(X|Y)$, then the equality does indeed

hold. Unfortunately, if SI states $\psi(X|Y) \neq p(X|Y)$, then it may be the case that $\psi(X|Y) = p(X|Y)$. This happens when d-separation states an independence does not hold, but the independence does hold in the joint distribution. Fortunately, strong completeness means that SI indicates the correct semantics in nearly all possible Bayesian networks.

It is very important to realize that the above remarks hold even when evidence is considered. The SI algorithm works by rewriting the expression showing how an intermediate factor was built in a simpler fashion. More specifically, the evidence can be pulled to the front of the equation and the summation can be pulled to the middle of the equation. The resulting expression, which we call evidence normal form, allows us to decide semantics even when evidence is considered.

SI brings improved clarity to denoting exact inference in Bayesian network literature. By extending d-separation's role from determining the minimum amount of information needed to answer a query $p(X|E = e)$ in B [32] to also giving the semantics of the potentials constructed when answering $p(X|E = e)$ in B, our results contribute to a deeper understanding of Bayesian networks.

Moveover, we suggested a potential practical advantage of knowing semantics. As SI algorithm can correctly denote p, it provides a switch from VE algorithm to AR algorithm. It is known that VE costs less computation and AR can keep the CPT factorization. Thus, combining these two algorithms can bring both their advantages to the inference. However, since we don't have an experimental result to support this claim, it will be left as future work.

Note that not just for VE inference algorithm, SI can be applied for any exact

BN inference that actually utilize VE to eliminate variables. Future work will explore how semantics can be applied to join tree propagation. For instance, some join tree propagation algorithms utilize VE to build messages, such as Shafer-Shenoy [36] and Lazy VE. It follows that SI can be applied to denote the semantics of the intermediate distribution and messages.

Bibliography

[1] Butz, C.J., Konkel, K. and Lingras, P.: A formal comparison of variable elimination and arc reversal in Bayesian network inference. *Intelligent Decision Technologies*, 3(3) (2009) 173 - 180.

[2] Butz, C.J., Konkel, K. and Lingras, P.: Join tree propagation utilizing both arc reversal and variable elimination. *International Journal of Approximate Reasoning*, 52(7) (2010) 948 - 959.

[3] Butz, C.J. and Yan, W.: The semantics of intermediate CPTs in variable elimination. *Proc. of Fifth European Workshop on Probabilistic Graphical Models* (2010) 41-48.

[4] Butz, C.J., Yan, W., Lingras, P., and Yao, Y.Y.: The CPT structure of variable elimination in discrete Bayesian networks. In Z.W. Ras and L.S. Tsay (Eds.), *Advances in Intelligent Information Systems*, SCI 265 (2010) 245-257.

[5] Butz, C.J., Yan, W. and Madsen, A.L.: d-Separation: strong completeness of semantics of intermediate CPTs in variable elimination. Accepted by *Twenty-sixth Canadian Conference on Artificial Intelligence* (CAI). (2013)

[6] Butz, C.J., Yan, W. and Madsen, A.L.: On Semantics of Inference in Bayesian Networks. Accepted by *Twelfth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty.* (2013)

[7] Butz, C.J., Yao, H. and Hua, S.: A join tree probability progagation architecture for semantic modeling, *Journal of Intelligent Information System*, 33(2) (2009) 145-178.

[8] Castillo, E., Gutierrez, J. and Hadi, A.: *Expert Systems and Probabilistic Network Models*. Springer, 1997.

[9] Cormen, T.H., Leiserson, C.E., Rivest R.L. and Stein C.: *Introduction to Algorithms*, 3rd. ed. Cambridge, MA: MIT Press, 2009.

[10] Cowell, R.G., Dawid, A.P., Lauritzen, S.L., and Spiegelhalter, D.J.: *Probabilistic Networks and Expert Systems*. New York: Springer, 1999.

[11] Darwiche, A.: *Modeling and Reasoning with Bayesian Networks*. New York: Cambridge University Press, 2009.

[12] Dechter, R.: Bucket elimination: A unifying framework for probabilistic inference, *Proc. 12th Conference on Uncertainty in Artificial Intelligence*. Portland, OR, (1996) 211–219.

[13] Geiger, D. and Pearl, J.: Logical and algorithmic properties of conditional independence and graphical models. *Annals of Statistics*, 21(4) (1993) 2001-2021.

[14] Jensen, F.V., Lauritzen, S.L. and Olesen, K.G.: Bayesian updating in causal probabilistic networks by local computations, *Comp. St. Q.*, 4 (1990) 269-282.

[15] Jensen, F.V. and Nielsen, T.D.: *Bayesian Networks and Decision Graphs*, 2nd. ed. New York: Springer, 2007.

[16] Kjaerulff, U.B.: Triangulation of graphsalgorithms giving small total state space, Research Report R-90-09. Dept. of Math. and Comp. Sci., Aalborg University, Denmark, 1990.

[17] Kjaerulff, U.B. and Madsen, A.L.: *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*. New York: Springer, 2008.

[18] Koller, D. and Friedman, N.: *Probabilistic Graphical Models: Principles And Techniques*. Cambridge, MA: MIT Press, 2009.

[19] Lauritzen, S.L., Dawid, A.P., Larsen, B.N. and Leimer, H.G.: Independence properties of directed Markov fields. *Networks*, 20 (5) (1990) 491-505.

[20] Lauritzen, S.L.and Spiegelhalter, D.J.: Local computations with probabilities on graphical structures and their application to expert system. *Journal of the Royal Statistical Society, Series B*, 50(2) (1988) 157-224.

[21] Li, Z. and D'Ambrosio, B.: Efficient inference in Bayes networks as a combinatorial optimization problem, *Internat. J. Approx. Reason*, 11 (1) (1994) 55–81.

[22] Madsen, A.L.: An empirical evaluation of possible variations of lazy propagation. *Proc. 20th conference on Uncertainty in artificial intelligence* (2004) 366-373.

[23] Madsen, A.L.: Improvements to message computation in lazy propagation. *International Journal of Approximate Reasoning*, 51(5) (2010) 499-514.

[24] Madsen, A.L.: Variations over the message computation algorithm of lazy propagation. *IEEE Transactions on Systems, Man, and Cybernetics, B*, 36(3) (2006) 636C648.

[25] Madson, A.L. and Jensen, F.V.: Lazy Propagation: A Junction Tree Inference Algorithm based on Lazy Evaluation, *Artificial Intelligence*, 113 (1-2) (1999) 203-245.

[26] Maier, D.: *The Thoery of Relational Databases*, Computer Science Press, 1983.

[27] Meek, C.: Strong completeness and faithfulness in Bayesian networks. *Proc. of Eleventh Annual Conference on Uncertainty in Artificial Intelligence* (UAI) (1995) 411-418.

[28] Neapolitan, R.E.: *Learning Bayesian Networks*, Prentice Hall, 2003.

[29] Neapolitan, R.E.: *Probabilistic Methods for Bioinformatics: with an Introduction to Bayesian Networks*. Burlington, MA: Morgan Kaufmann, 2009.

[30] Olmsted, S.: On representing and solving decision problems. Dept. of Engineering Economic Systems, PhD Thesis, Stanford University, Stanford, CA, 1983.

[31] Pearl, J.: Belief networks revisited. *Artificial Intelligence*, 59 (1993) 49-56.

[32] Pearl, J.: Fusion, propagation and structuring in belief networks. *Artificial Intelligence*, 29 (1986) 241-288.

[33] Pearl, J.: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, California, 1988.

[34] Shachter, R.: Evaluating inflence diagrams. *Operation Research*, 34(6) (1986) 871-882.

[35] Shachter, R., D'Ambrosio, B. and Del Favero, B.: Symbolic probabilistic inference in belief networks, *Proc. 8th National Conference on Artificial Intelligence* (1990) 126-131.

[36] Shafer, G.: *Probabilistic Expert Systems*. Society for the Institute and Applied Mathematics, Philadelphia, 1996.

[37] Shafer, G. and Shenoy, P.: Probability progagation, *Annals of Mathematics and Artificial Intelligence*, 2 (1990) 327 - 352.

[38] Verma, T. and Pearl, J.: Causal networks: semantics and expressiveness. *Proc. of Fourth Annual Conference on Uncertainty in Artificial Intelligence* (UAI) (1988) 352-359.

[39] Wong, S.K.M., Butz, C.J. and Wu, D.: On the Implication Problem for Probabilistic Conditional Independency, *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 30(6) (2000) 785-805.

[40] Xiang, Y.: *Probabilistic Reasoning in Multiagent Systems: A Graphical Models Approach*. New York: Cambridge University Press, 2002.

[41] Zhang, N.L. and Poole, D.: A simple approach to Bayesian network computations. *Proc. of the seventh Canadian Conference on Artificial Intelligence* (1994) 171-178.

Appendix A

Weighted-Min-Fill Algorithm

Weighted-Min-Fill measurement (WMF) [18] is a measurement to determine the elimination ordering. WMF takes a moralized graph as input and output an elimination ordering σ for the n vertex in this graph. During the WMF algorithm, a function, named Weight-of-Variable (WV), is called to compute the weight of each variable.

Algorithm 6. WMF(B_m)

1 Initialize all nodes in B_m as unmarked

2 **for** each variable $v_i \in B_m$

3 Select an unmarked v_i that minimizes $WV(B_m, v_i)$

4 $\sigma \leftarrow v_i$

5 Introduce edges in B_m between all neighbors of v_i

6 Mark v_i

7 **return** σ

Algorithm 6. Weighted-Min-Fill (WMF) Algorithm.

Function 1. WV(B_m, v_i)

1 Set $w_i = 0$

2 Let E_{add} be the edge set added to make the adjacent vertex of v_i pairwise connected

3 **for** each edge $(v_j, v_k) \in E_{add}$

4 $w_i = w_i + dom(v_j) \times dom(v_k)$

5 **return** w_i

Function 1. Weight-of-Variable (WV) Function.

Appendix B

Computation of Transitive Closure

Note that the a graph can be represented by matrix. To compute the matrix for transitive closure, the *adjacency matrix* [9] of a graph $G = (V, E)$ can be defined as follows:

Definition B.0.1 *Given a directed graph* $G = (V, E)$ *with vertex set* $V = \{v_1, v_2, \ldots, v_n\}$*, the adjacency matrix representation of* G *then consists of a* $n \times n$ *matrix* $M = (m_{ij})$ *such that*

$$m_{ij} = \begin{cases} 1, & \textit{if } (v_i, v_j) \in E; \\ 0, & \textit{otherwise.} \end{cases}$$

The following is called transitive-closure (TC) algorithm, with $n \times n$ adjacency matrix M input and transitive closure matrix T output, while the $m_{ij}^{(k)}$ represents the element on row i column j of the matrix built in step k.

Algorithm 4. TC(M)

for $i = 1$ to n

  for $j = 1$ to n

    if $i = j$

      $m_{ij}^{(0)} = 1$

for $k = 1$ to n

  for $i = 1$ to n

    for $j = 1$ to n

      $m_{ij}^{(k)} = m_{ij}^{(k-1)} \vee (m_{ik}^{(k-1)} \wedge m_{kj}^{(k-1)})$

return $M^{(n)}$

Algorithm 4. Transitive Closure (TC) Algorithm.

This algorithm involves the logical operations $\vee$ and $\wedge$, where

$$m_1 \vee m_2 = \begin{cases} 0, & \texttt{if } m_1 = 0 \texttt{ and } m_2 = 0; \\ 1, & \texttt{otherwise}. \end{cases}$$

and

$$m_1 \wedge m_2 = \begin{cases} 1, & \texttt{if } m_1 = 1 \texttt{ and } m_2 = 1; \\ 0, & \texttt{otherwise}. \end{cases}$$

Example 21 *[9] Given a directed graph in Figure B.1, the matrices $M^{(k)}$ com-*

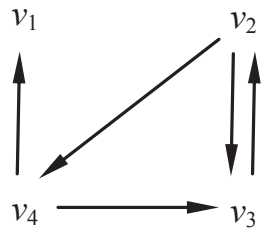

Figure B.1: A directed graph.

puted by TC algorithm is shown as follows:

$$T^{(0)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix} T^{(1)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix} T^{(2)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

$$T^{(3)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} T^{(4)} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}.$$