A Framework for Bayesian Network Mapping

Rong Pan
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County

A Proposal for Doctoral Research
December 16, 2004
Abstract

My proposed research is motivated by the need of supporting inference across multiple intelligence systems involving uncertainty. At present, the most popular uncertainty inference framework, Bayesian network (BN), is still widely used as a standalone system. Existing distributed Bayesian network models are restricted in scalability, consistency and expressiveness. The objective of this proposed research is to develop a theoretical framework and related inference methods to map semantically similar variables between separate BNs in a principled way. This framework can also be used to support mapping between independently developed ontologies with overlapped domains. This research is to be conducted in two steps. In the first step, we investigate the problem of mapping variables between two BNs with different semantics and distributions; in the second step, we extend mapping between BN to support mappings of concepts defined in different ontologies that are translated into BN based on BayesOWL. We also propose a virtual evidence based soft evidential update method and prove its convergence.
Contents

1. Introduction 1
   1.1 Motivation ................................................................. 2
   1.2 Problem Statement ....................................................... 2
   1.3 Expected Contribution .................................................. 2

2. Background and Related work 3
   2.1 Bayesian Network .......................................................... 4
      2.1.1 Definition and Semantics ........................................... 4
      2.1.2 Inference ............................................................... 5
      2.1.3 Iterative Proportional Fitting Procedure (IPFP) ............... 6
      2.1.4 Jeffery’s rule and soft evidence .................................. 7
   2.2 Ontology and Semantic Web .............................................. 11
   2.3 Related Work .................................................................. 12
      2.3.1 Distributed models of Bayesian Networks ....................... 12
         2.3.1.1 Agent Encapsulated Bayesian Network (AEBN) ............. 13
         2.3.1.2 Multiply Sectioned Bayesian Network (MSBN) .............. 15
      2.3.2 Works on ontology mapping/translation ........................ 16
      2.3.3 Probabilistic Extensions to Ontology ............................ 19

3. Proposed Research and Approaches 21
   3.1 Bayesian Network Mapping: Overview ................................ 21
   3.2 Formalization of Mapping ............................................... 23
      3.2.1 Pair-wise Linkage .................................................... 24
      3.2.2 Mapping between two BNs ......................................... 26
      3.2.3 Experiments .......................................................... 28
   3.3 Mapping Reduction ....................................................... 30
   3.4 Inference .................................................................. 34
      3.4.1 Inference Process ..................................................... 34
      3.4.2 Virtual Evidence Based Soft Evidential Update .............. 35
      3.4.3 Experiments .......................................................... 39
   3.5 Supporting Ontology Mapping .......................................... 41

4. Research Plan 42

References 43
List of Figures

Figure 2.1. Soft evidence example 9
Figure 2.2. ROSE agent scenario 14
Figure 2.4. Architecture of GLUE 18
Figure 3.1. Overview of BN mapping framework 22
Figure 3.2. BN mapping: a four step description 23
Figure 3.3. Bayesian network connection 24
Figure 3.4. Equivalent linkage 25
Figure 3.5. A general case of connection with probability constraint 26
Figure 3.6. BN mapping example: original BNs 29
Figure 3.7. Variable linkages and BN mapping 30
Figure 3.8 1-M mapping reduction example 1 32
Figure 3.9 1-M mapping reduction example 2 33
Figure 3.10 Bayesian network for soft evidential update 39
Figure 3.11 BN with virtual evidence simulation of soft evidence 40
Figure 3.12 Convergence of virtual evidence’s likelihood ratio 41
Chapter 1

Introduction

1.1 Motivation

My proposed research is motivated by the need of supporting inference across multiple intelligence systems involving uncertainty. At present, the most popular uncertainty inference framework, Bayesian network (BN), is still widely used as a standalone system. When the problem scale is large, a large network slows down inference process and makes itself difficult to be reviewed or revised. When the problem itself is distributed, domain knowledge has to be centralized and unified before a single BN is created for the problem. Alternatively, separate BNs describing related subdomains or different aspects of the same domain may be created, but it is difficult to combine them for problem solving — even if the interdependent relations between these BNs are available. The inconsistent variable distributions prevent users from unifying them properly. Several works have been done to address this issue, including most notably Multiply Sectioned Bayesian Network (MSBN) by Xiang [32-34] and Agent Encapsulated Bayesian Network (AEBN) by Bloomeke and Valtorta [2, 3]. However, most of the existing approaches still have restrictions in scalability, consistency and expressiveness. For example, MSBN’s pair-wise variable linkages are between identical variables with the same distributions, and to ensure consistency, only one side of the linkage has a complete CPT. AEBN also requires that a connection is between identical variables, but allows these variables with different distributions. Belief exchange is carried out as soft evidence across different BNs, which ensures the consistency of variables’ distributions during inference. However, AEBN does not address the issue of how to propagate probability influence when multiple connections are available between two BNs. Nor does it discuss how the linkages should be created, how to solve the contradictions and conflicts between multiple connections. More importantly, none of the above approaches offers solutions for representing probabilistic relations between variables, which is required when we want to link similar but not identical variables across BNs. These and other related issues will be the focus of our investigation.

We will also investigate on how to apply our framework to the difficult problem of
ontology mapping. In semantic web, different ontologies may use different terms to describe same concepts, or use same terms to describe different concepts, or define concepts which are similar but not identical, due to different conceptualizations of the same or overlapped domains. As a result, ontology mappings, which are introduced to mark up the related concepts in different ontologies, are seldom certain. If ontologies or their components can be translated into BNs, as suggested by Ding [8,9], then ontology mapping can be realized as probabilistic linkages between BNs in our framework. By doing so, not only we may better handle the inherent uncertainty involved in ontology mapping, but also unify the mapping and subsequent reasoning in a coherent and systematic manner as probabilistic inference.

1.2 Thesis Statement
I propose to develop a theoretical framework that supports probabilistic reasoning across multiple BNs. This framework is distinguished from existing work in that I do not require the individual BNs to share some common variables, nor shared variables, if exist, to have the same distributions. The core of this framework is to establish a minimal or parsimonious set of linkages between similar or related variables between BNs that captures uncertainty relations between these variables, and to develop algorithms that reason cross BN utilizing these linkages. This framework will be applied to a difficult but important problem in semantic web, namely concept mapping and translation between independently developed ontologies.

1.3 Expected Contribution
With mappings of similar but not identical variables across BNs, the proposed framework goes beyond any existing work for probabilistic reasoning involving multiple BNs. It, and the related algorithms, can thus be applied to a wide range of problems that require synthesis of knowledge distributed at different sources and localities, obtained from different perspectives or conceptualization. Such applications include ontology mapping, distributed data mining and sensor networks. Using this BN mapping framework, our approach will be among the first to treat uncertainty in ontology mapping with great rigor, and it is expected to result in more accurate results or generate new mappings from given mappings by inference. Also, we proposed a virtual evidence based soft evidential update method, which is more applicable and expressive than other soft evidential methods.
Chapter 2

Background and related work

Bayesian network was first introduced to AI community by Pearl [25, 27] as a graphic model to represent the uncertainty of domain knowledge and to reason with such uncertain knowledge. The exact inference method of BN has been proved to be NP hard [5], and exact inference methods, such as junction tree [21], are thus of exponential time complexity. Approximate inference methods, such as loopy propagation [22] and stochastic simulation [25], are invented for large scale networks. However, the semantics of BN is not sufficient to describe all uncertainty relationships. To represent the uncertainty in the process by which domain knowledge are collected, Jeffery’s rule [14, 28] is devised to introduce additional relations to BN. Soft evidence [28, 31, 30] is implemented to represent such relations in BN, and regular inference methods have been extended to support soft evidential update. Valtorta and Kim [30] proposed a solution applying Iterative Proportional Fitting Procedure (IPFP) on Junction tree method for this problem. IPFP is first proposed by Kruithof [18] and is now used to find the I-projection of a distribution to satisfy given constraints [31].

Distributed model of BN is investigated for large scale and distributed problems. Multiply sectioned Bayesian network (MSBN) by Xiang [32, 33] and Agent Encapsulated Bayesian Network (AEBN) by Bloemeke and Valtorta [2, 3] are two prominent examples. MSBN aims to section one large BN into smaller subnets and deploy them to agents. The consistency of cross subnet linkage is guaranteed by the sectioning method. AEBN aims to propagate the beliefs between separate agents and the consistency of variables’ distributions is ensured by the system’s semantics.

Another background knowledge related to our proposed research is Semantic Web [36], which comes with the rapid expansion of computer network and the World Wide Web and the need for machines/programs to understand web page content. Semantic web describes information resources using URI and XML namespace mechanism, which is formalized as the Resource Description Framework (RDF) [37]. The vocabulary of semantic web resources is defined as set of ontologies in the format of RDF schema. Since multiple ontologies are often involved, ontology
mapping and translation become necessary for information integration in semantic web. Most existing works in this direction are logic based [11, 14], which ignore the uncertainty in the mapping. A few exceptions include GLUE [10] from University of Washington and OntoMapper [29] from UMBC, which have addressed uncertainty to an extent.

As current ontology description languages do not capture all the relations between concepts, uncertainty extension of ontology has been an interesting research issue. Ding [8, 9] proposed a framework called BayesOWL which extends OWL to allow expression of probability information and then converts ontologies to BNs by a set of translation rules and procedures. Holi’s work similarly aims to approximate the concept overlaps of a semantic web taxonomy using Bayesian network [13].

2.1 Bayesian Network

Bayesian network (BN) is an increasingly popular knowledge representation framework for uncertainty. Bayesian network provides a systematic way to represent interdependency relationships among propositions, each of which is considered as a random variable with a finite set of states as its outcomes. Intense research on BN has been conducted in recent years, as reflected by a rich body of publications in the literature [35]. In what follows we briefly review the basics of BN and techniques relevant to our research agenda.

2.1.1 Definition and Semantics of BN

Let \( A \) and \( B \) be two propositions, normally we use \( P(A) \) to denote unconditional or prior probability, \( P(A|B) \) the conditional or posterior probability. \( P(V) \) denotes the Joint Probability Distribution (JPD) of the entire domain. If \( X \) is a subset of \( V \), \( P(X) \) is referred to as a marginal distribution, and from JPD we can calculate \( P(X) \) using the following formula

\[
P(X) = \sum_{V \setminus X} P(X, V \setminus X).
\]

Definition: A probability space is a triple \((\Omega, F, P)\), where \( \Omega \) is a set, \( F \) a sigma field of subsets of \( \Omega \), and \( P \) a probability measure on \( F \). [1] \( F \) satisfies the following requirement to be a sigma field of subsets of \( \Omega \):

\[\Omega \in F,\]

and \( A_1, A_2, \ldots \in F \) implies \( \bigcup A_n \in F \).

A probability measure on \( F \) is a function that assigns a number \( P(A) \) to each set \( A \) in the sigma field \( F \).

The theoretical basis of Bayesian network is Bayes’s rule, which underlies all probabilistic inferences in BN:
$P(B \mid A) = \frac{P(A \mid B)P(B)}{P(A)}$, where $A$ and $B$ are propositions.

A Bayesian network is a graph in which the following holds:

1. A set of variables and a set of directed edges between variables.
2. Each variable has a finite set of mutually exclusive states.
3. The graph is a directed, acyclic graph (DAG).
4. Each variable has a conditional probability table (CPT) that quantifies the effects of its parents.

From the definition we can see that each Bayesian network corresponds to a particular probability space. For each probability space, there are a set of equivalent Bayesian networks corresponding to it (two Bayesian networks are equivalent if they correspond to same probability space).

Let $U = \{A_1, A_2, \ldots, A_n\}$ be a set of propositions in a Bayesian network. The chain rule for Bayesian network is described as below:

\[
P(U) = \prod_i P(A_i \mid pa(A_i)),
\]

where $pa(A_i)$ is the parent set of $A_i$.

The chain rule is based on a conditional independence assumption associated with BN, which can be described by the notion of $d$-separation in terms of the network’s graphic structure. Two distinct variables $A$ and $B$ are said to be $d$-separated if, for all paths between $A$ and $B$, there is an intermediate variable $V$ such that either

- the connection through $V$ is serial or diverging and $V$ is instantiated, or
- the connection is converging, and neither $V$ nor any of $V$’s descendants have received evidence.

If $A$ and $B$ are not $d$-separated, we call them $d$-connected. In a BN, if $A$ and $B$ are $d$-separated, they are in independent of each other. In other words, if $A$ and $B$ are $d$-separated, the changes in the certainty of $A$ have no impact on the certainty of $B$ [15].

2.1.2 Inference

It has been proved that exact inference in Bayesian network is NP-hard [5]. Nonetheless, a number of algorithms have been developed to exploit the network structure for efficient computation for probabilistic inferences in BN. Belief propagation [27] and Junction tree [21] are two most popular exact BN inference methods. Belief propagation is based on local message passing and hence can work only for polytree to avoid rumors (double counting probability influence in belief propagation). Junction Tree method, on the other hand, works for networks of any
kinds of topology. Several inexact algorithms are developed for large scale problems with approximate solutions (e.g., stochastic sampling [26] and loopy propagation [22]).

Junction tree method, which will serve as a basis for developing some of our problem solving algorithms in this research, is briefly reviewed here. This method transforms a Bayesian network into a tree structured graph of large nodes. It first moralizes the network by removing the direction of the links and adding edges between nodes that share children. Then the moral graph is triangulated so that all cycles longer than 3 have a chord. In the triangulated graph, each complete subgraphs is recognized as a clique. Cliques are connected, forming a junction tree. The probabilities of cliques are called potentials, on which operations are defined as a set of algebra. Inference is done by passing and unifying potentials between adjacent cliques in the junction tree. Four procedures are used for local message passing: Absorption, DistributeEvidence, CollectEvidence, and EnterEvidence. Absorption specifies the method to unify the potentials of adjacent cliques. CollectEvidence calls adjacent cliques and absorbs from them after the call returns. DistributeEvidence absorbs from the clique that calls it and then call other adjacent cliques. EnterEvidence changes the potentials of a clique to represent an observation. Compared with other exact inference method, Junction tree inference method is efficient in that its computation cost is exponential to the size of the largest clique, not the size of the whole network.

2.1.3 Iterative Proportional Fitting Procedure (IPFP)

Iterative Proportional Fitting Procedure (IPFP), was first proposed by Kruithof in 1937 [18], and used as a computational process to estimate cell frequencies in contingency tables under some marginal constraints [7].

Let $Q_0$ be the initial distribution on variables set $X$, $\{P(S_i)\}$ be a consistent set of $m$ marginal probability distributions, where $X \supseteq S_i \neq \emptyset$. The IPFP process is defined by the following formula:

$$Q_i(X) = \begin{cases} Q_{i-1}(X) \frac{P(S_j)}{Q_{i-1}(S_j)} & \text{if } Q_{i-1}(S_j) > 0, \\ 0 & \text{otherwise} \end{cases}$$

where $i$ is the iteration number, $j = (i-1) \mod n + 1$, $P(S_j)$ is the distribution of $S_j$ in $E$.

IPFP has been shown to converge to an $I$-projection of an initial distribution on an additively set of constraints. Let $P$ and $Q$ be distributions on the same set of propositions, $S$ be a probability constraint, $P$ is an $I_1$-projection of $Q$ on $S$ iff $P$ satisfies $S$ and has the minimum $I$-divergence to $Q$, $I(P \parallel Q)$, among all distributions that satisfy $S$. If $P$ minimizes $Q$’s $I$-divergence to $P$, $I(Q \parallel P)$, $P$ is called as
$I_2$-projection of $Q$ on $S$.

$I$-divergence, also known as cross entropy or Kullback-Leibler divergence, is used in Information Theory to measure the distance of two distributions. It is stated as:

$$I(P \parallel Q) = \begin{cases} 
\sum_x P(x) \log \frac{P(x)}{Q(x)}, & \text{if } P \ll Q \\
+\infty, & \text{otherwise}
\end{cases}$$

where $x$ is an instantiation of $X$, $P \ll Q$ means $P$ is dominated by $Q$, i.e. $\{x \in X \mid P(x) > 0\} \subseteq \{y \in X \mid Q(y) > 0\}$, where $x$ and $y$ are instations of $X$. Note that $I$-divergence is in general not symmetric, and that $I(P \parallel Q) \geq 0$ where $I(P \parallel Q) = 0$ only when $P$ and $Q$ are identical.

The convergence of IPFP is proved by Csiszar [4] and Volmel [31] on discrete distributions. For continuous case, it still remains as an open question. An important theorem of Volmel’s proof [31, Thm 2.5] is re-stated below:

**Theorem:** Let $Q_0$ be the initial distribution on variables set $X$, $C_1$, $\ldots$, $C_n$ be a set of probability constraints on $X$, $S_1$, $\ldots$, $S_n$ be sets of joint probability that satisfy the constraints $C_1$, $\ldots$, $C_n$ respectively, $S = \bigcap S_i \neq \emptyset$. Define $Q_i$ recursively by letting $Q_i$ be the $I_1$-projection of $Q_{i-1}$ on $C_j$, where $j = (i - 1) \mod n + 1$. Then the sequence of probability distributions $Q_n$ converges to the $I_1$-projection of $Q_0$ on $\{C_1, \ldots, C_n\}$.

We will use this theorem to prove the convergence of our soft evidential update method later in this paper.

### 2.1.4 Jeffery’s rule and soft evidence

In the process of building a probabilistic intelligent system, we collect the domain knowledge and describe it often by defining probabilistic variables and their interdependency relations. Normally we believe we use certain observation process to collect domain knowledge. For example, to build a BN on stones’ color in a mine, we observe a pack of stones’ color and find proposition $A = \text{green}$ has prior probability of 0.6: $P(A = \text{green}) = 0.6$. In our BN, this probability is non-changeable. In other words, probability 0.6 is an objective state of knowledge about “$A = \text{green}$”. However, there exist situations where the observation process is uncertain. Imaging in our example, the observer could have color-weak eyes; the observer could have been watching the stones under a colorful light; also the observer could have weak sense on colors and classify cyan and light turquoise stones as green stones. So in these cases we need to use a probabilistic parameter to describe the certainty of our observation process.
Since the uncertain observation could be caused by inaccuracy of observation devices, irregular observation environment and unconfident observers, the calculation of propositions’ uncertainty must take into account how we gather them. To handle this problem, Richard Jeffrey devised a rule of probability kinematics [28], which is now known as Jeffery’s Rule [14, 28]. Jeffery’s definition of the rule is as follows:

An agent changes its belief in certain exhaustive and mutually exclusive propositions \(B_1, B_2, \ldots, B_n\) from their original values \(P(B_1), P(B_2), \ldots, P(B_n)\) to new values \(Q(B_1), Q(B_2), \ldots, Q(B_n)\). For every proposition \(A\) not “directly affected by the passage of experience”, we have

\[
Q(A) = \sum_i P(A \mid B_i)Q(B_i),
\]

where \(Q(B_i)\) is what we called soft evidence, \(P(A \mid B_i)\) is the conditional probability of \(A\) given \(B_i\) before this evidence is observed [28].

Compared with regular evidence of Bayesian network (hard evidence), soft evidence represents a kind of more flexible way of specifying our findings of the domain. In BNs, hard evidence represents a statement of a variable’s state, while soft evidence represents a statement of a variable’s distribution on its states. For example, variable \(A\) has states \(\{a_i\}, i = 1, 2, 3\). Hard evidence may specifies \(A\)’s state as \(a_1\), which entails a distribution that \(P(A = a_1) = 1, P(A = a_2) = 0, \) and \(P(A = a_3) = 0\); while soft evidence specifies \(A\)’s distributions as, say, \((0.3, 0.2, 0.5)\), i.e., \(P(A = a_1) = 0.3, P(A = a_2) = 0.2, \) and \(P(A = a_3) = 0.5\). Although we may regard hard evidence as extreme cases of soft evidence, they are not equally treated in BN inference algorithm. Hard evidence, after entering the system, always retains its state and is never influenced by any other evidences, but this may not be the case for soft evidence. We will discuss this issue further later in this section.

Pearl rewrites Jeffery’s rule in the form of Bayes conditionalization formula [27]. Using \(se\) to denote the evidence actually observed, then we can replace \(Q(A)\) with \(P(A \mid se)\), and making Jeffery’s rule look like

\[
P(A \mid se) = \sum_i P(A \mid B_i)P(B_i \mid se).
\]

The precondition of Jeffery’s rule is that the conditional probabilities \(P(A \mid B_i)\) must be invariant with respect to the soft evidenced propositions. It is difficult to directly verify if this requirement is satisfied, so we usually use another form of this precondition: Jeffery’s rule is applicable to \(A\) iff \(A\) is d-separated from \(e\) by \(B\). Jeffery describes this requirement as \(A\) “is not one of the \(n\) propositions whose probabilities were directly affected by the passage of experience”. Pearl explains the requirement of Jeffery’s rule by a cloth-candle story [28].

In Figure 2.1(a), node \(B\) stands for the observation of cloth color, node \(A\) the cloth’s
salability, and $e$ the influences to the cloth observer from a candle light — cloth’s color may look different from its original color under a candle light. In Figure 2.1(b), $B$ and $e$ stand for the same things, but $A$ stands for the wax content of the candle, which influences the flame spectrum of the candle light, and in turn the observation of color. In the first example, we have $P(A|B=color_i,e)=P(A|B=color_i)$, and since the restrain of the rule is satisfied, we get $Q(A) = \sum_i P(A|B=color_i)Q(B=color_i)$ or $P(A|e) = P(A|B=color_i) P(B=color_i |e)$.

Figure 2.1. Soft evidence example

Suppose in the example of Figure 2.1(a), the states of $B$ are \{green, blue, violet\}, states of $A$ is \{yes, no\}, and the prior probabilities and conditional probabilities are:

$P(B) = (0.3, 0.3, 0.4)$,

$P(A|B = green) = (0.4, 0.6)$,

$P(A|B = blue) = (0.4, 0.6)$,

and $P(A|B = violet) = (0.8, 0.2)$.

Then $P(A) = 0.4 \cdot 0.3 + 0.4 \cdot 0.3 + 0.8 \cdot 0.4 = 0.56$. If the soft evidence changes $P(B)$ to

$Q(B) = (0.7, 0.25, 0.05)$,

due to Jeffery’s rule, $Q(A) = 0.4 \cdot 0.7 + 0.4 \cdot 0.25 + 0.8 \cdot 0.05 = 0.42$.

In the second example, $P(A|B = color_i,e) \neq P(A|B = color_i)$, and therefore Jeffery’s rule is not applicable. This can be corrected by recognizing that the node of Flame Spectrum is also directly affected by the external experience, obtaining its soft evidential distribution and applying Jeffery’s rule. Let $C$ denotes the node of Flame Spectrum,

$P(A|B = color_i,C = spectrum_i,e) = P(A|C = spectrum_i)$, and then

$Q(A) = \sum_i P(A|C = spectrum_i)Q(C = spectrum_i)$ or

$P(A|e) = \sum_i P(A|C = spectrum_i) P(C = spectrum_i |e)$.

Besides soft evidence, Pearl also provides another way to integrate the probabilistic summaries of evidence with previously established belief — the approach of virtual evidence [25, 27], which can be viewed as the likelihood ratio version of Jeffery rule [19, 23]. Virtual evidence represents the likelihood of a variable’s distributions: “the
probability of observing variable $A$ being in state $a_j$ if its true state is $a_i$.” The process of virtual conditionalization is applied to update a proposition $A$’s distribution:

the likelihood of variable $A$ is $L(A) = \frac{P(e|A)}{P(e|\neg A)}$, and then

$$Q(A) = \frac{L(A)P(A)}{1 - P(A) + L(A)P(A)}$$

Valtorta and Kim [30] were the first to implement soft evidential update algorithm in their Agent Encapsulated Bayesian Network (AEBN) system. They further state the semantic difference of soft evidence and virtual evidence as:

1) Virtual evidence describes an external observation of a proposition and local networks can improve this observation; local structure expresses some information that the external observer does not have. Likelihood is used to update current distributions.

2) Soft evidence describes an authority observation of a proposition and local structure cannot improve this observation. Distributions prior to the observation are replaced by external findings.

Adding soft evidence to Bayesian network involves attaching a newly created observation node to the evidenced variables and manipulating its CPT. In Figure 2.1(a), $e$ is the observation node and $B$ is soft evidence node. Traditional inference methods of BNs are not adequate to realize soft evidential update because distributions of observation nodes are not protected to be fixed. Hard evidences are protected because probabilities of zero are specially processed. For example, in Junction-Tree algorithm, the potential algebra defines that zeroes in probability tables remain zeros after each operation [15, 34]. Valtorta and Kim devise a variation of Junction-Tree algorithm for soft evidence update [16]. When constructing the Junction Tree, all soft evidence nodes are fully connected with each other by additional edges. And after triangulation, all soft evidence nodes appear in a clique called Big Clique. Let $C$ denote this big clique, $I = \{O_1, ..., O_k\}$ denote soft evidence variables and $V$ denote the set of all variables. Big Clique algorithm first apply hard evidence from all other cliques and update $P(V)$ to $P^\ast(V)$ using traditional Junction Tree algorithm. Then it absorbs soft evidence in $C$ by updating the potential of $C$ with following IPFP formulae, iterating over $Q(O_j)$:

$$Q_a(C) = P^\ast(C)$$

$$Q_i(C) = Q_{i-1}(C) \frac{Q(O_j)}{Q_{i-1}(O_j)}$$

where $j = 1 + (i-1) \mod k$.

The above iteration stops when the result converges. Finally, $Q(C)$ is distributed to all other cliques. Big Clique algorithm is time efficient in that it uses a clique to collect the joint probability of all soft evidence nodes and hence iterate only in this joint probability. It becomes inefficient when the size of the big clique becomes large,
and when the big clique contains all cliques of the original junction tree, the computation is done with the JPD of the entire network and Junction Tree loses its advantage in time efficiency.

Please note that we assume in the above algorithm and all other parts of this proposal the soft evidences are carefully treated such that all soft evidenced nodes that d-separate soft evidence from the original Bayesian network are recognized, hence Jeffery’s rule is applicable on all soft evidence.

2.2 Ontology and Semantic Web

The current World-Wide Web contains vast volumes of data; the majority of them are web pages organized for human consumption only. Machines and programs cannot understand and process this information. Semantic Web is an attempt to solve this problem by associating meanings/semantics with the data through carefully defined ontologies of the concepts in a way machines can understand.

The Resource Description Framework (RDF) [37-39], a collaborative effort by a number of metadata communities, is a standard general assertional model to represent the resources on the web. It is a framework that supports resource description and metadata for a variety of applications. RDF uses XML as its syntax and identifies resources by using URI and XML namespace mechanism. The basic building blocks of RDF, RDF triples, are represented as “subject”, “predicate” and “object”. “predicate” is also known as property of the triple. In general, a triple can be read as “the <subject> has <predicate> <object>” [38]. RDF is an assertional logic, in which each triple expresses a simple proposition [39]. A triple does not change the meaning of other triples.

RDF Schema [40] is used to control the vocabulary used to define the resources in RDF. Classes and properties are created as descriptions of concepts and relations. Property can describe relationship between classes and properties, restrictions to one property’s domain and range, and annotations to any RDF terms. However, RDF does not specify any inference method. Built on top of RDF and RDF Schema, Web Ontology Language (OWL)[43] provides a richer set of vocabulary to describe the resources and their relations.

Ontology is a specification of conceptualization. More formally, ontology is a set of vocabulary to describe the conceptualization of a particular domain [36]. It provides a common understanding about the domain knowledge. For a semantic web application, ontology is used to capture the concepts and the relationships between concepts; or say, knowledge about its domain is encoded by this ontology, machines or agents then can reason about the domain and adjust their future behaviors accordingly.

OWL[43] is a new web ontology language recommended by W3C. It is based on
DAML+OIL\textsuperscript{[42]} with removal of qualified restrictions, renaming of various properties and classes, and some other updates (refer to \textsuperscript{[43]} for all the changes). An OWL ontology has zero or more headers, followed by zero or more classes, properties and instances. OWL assigns specific meaning to certain RDF triples using OWL vocabulary. OWL classifies the instances into two sorts: the datatype instances consist of the values that belong to XML Schema datatypes, the object instances consist of individual objects that are instances of classes described within OWL or RDF. Correspondingly there are two kinds of properties in OWL: ObjectProperty and DatatypeProperty. OWL includes three increasingly complex languages: OWL Lite, OWL DL and OWL Full.

To identify similar or identical concepts in different ontologies and conduct inference across multiple ontologies, automatic ontology mapping and translation has been recognized as an efficient way. Ontology mapping methods calculate similarity of variable pairs in separate ontologies and mark the variables with high similarity as identical. Most ontology mapping methods require additional data or sample text to describe ontology concepts. Ontology translation aims to translate the instances of one concept to another based on Ontology mapping result.

### 2.3 Related work

Our proposed research is motivated by and aims to improve the following works on distributed BN models and ontology mappings. Especially, MSBN and AEBN primarily reveal the issues in building distributed Bayesian networks. MSBN \textsuperscript{[32, 33]} focuses on the parallelism of uncertainty reasoning on multi-agent system. AEBN \textsuperscript{[2, 3]} aims to pass belief distributions between different agents.

#### 2.3.1 Distributed models of Bayesian networks

Distributed Bayesian networks models are mostly considered in various multi-agent scenarios. An agent is commonly defined as an individual software or/and hardware entity that responses automatically to the environment. In a multi-agent system, each agent has only partial knowledge or a specific perspective of the domain, pursues a local set of goals, and exchanges its observations or beliefs with other agents in the system. If the domain (or some of its aspects) is modeled by a Bayesian network, the network shall be distributed in some fashion among these agents. The major issues concerning distributed Bayesian networks are the following:

1. How to decompose the network among the agents?
2. How to exchange local beliefs and observations via agent communications?
3. How to maintain the global consistency in the system?

Confronted with these questions, several approaches have been proposed for
distributed BN, including AEBN (Agent Encapsulated BNs), MSBN (Multiply Sectioned BN), object oriented BN [17], and MEBN (Multiple-Entity BNs) [20]. Each of these approaches makes its own assumptions either to circumvent or solve them. For example, in AEBN, the Bayesian networks are originally encapsulated in each agent while some propositions are shared. By assuming additional independence relations beyond the definition of Bayesian networks, agents use soft evidence as the way of communication to exchange distributions of shared propositions. Global consistency is certified by detecting and eliminating the rumor problem. In MSBN, the primary objective is to decompose a given global network into subnets. The global Bayesian network must conform a hypertree structure to be soundly sectioned. The message passing procedures between sections are processed locally. Global consistence is certified by the sectionable graph structure and message passing strategy. These two representative approaches are reviewed in details in the next subsections.

2.3.1.1 Agent Encapsulated Bayesian Network (AEBN)
Agent Encapsulated Bayesian Network, proposed by Bloemeke and Valtorta[2, 3, 16], is a system in which each agent employs a Bayesian network as its internal knowledge representation of the domain world, and exchanges beliefs with each other along shared variables. The variables of a encapsulated Bayesian network are divided in to three groups: 1) those that are only used within the agent (local set), 2) those that other agents have better knowledge of (input set), and 3) those that current has best knowledge of than other agents (output set). An agent may want to get a better knowledge of the nodes in the input set from publisher agent, and if incoming information is void, it assigns estimated values to them. An agent shares the distributions of nodes in the output set to subscriber agents via soft evidences. Figure 2.2 shows the Redundant observed Sensor Example (R.O.S.E.) from [3].

In this example, Figure 2.2(a) is the sensor agent that generates the target’s spatial distribution \( S_1 \) by observing \( X_r \) and \( Y_r \); 2.2(b) and 2.2(c) are Observer agents that report the target’s position \( R_1/ R_2 \) by fusing subscribed variable \( S_1 \) and local estimation of the reliability of sensor agent: \( R_{1/2i}/ R_{2i} \). 2.2(d) is Supervisor agent that fuses final result of target’s location \( L \) from two Observer’s report \( R_1 \) and \( R_2 \).

To update an agent’s distribution \( P(V) \) with new evidences \( Q(E_1), Q(E_2), \ldots, Q(E_n) \) for set of variables \( I = \{ E_1, E_2, \ldots, E_n \} \), the following formula is:

\[
Q(V) = P(V \setminus I | E_1, E_2, \ldots, E_n) \cdot Q(E_1, E_2, \ldots, E_n) = \frac{P(V \setminus I)}{P(E_1, E_2, \ldots, E_n)} \cdot Q(E_1, E_2, \ldots, E_n)
\]

where

- if \( E_1, \ldots, E_n \) are d-separated, \( Q(E_1, E_2, \ldots, E_n) = Q(E_1) \cdot Q(E_2) \cdot \ldots \cdot Q(E_n) \),
- and
- if \( E_1, \ldots, E_n \) are d-connected, \( Q(E_1, E_2, \ldots, E_n) \) is the \( I_1 \)-projection of
probability distribution on the set of all distributions defined of I on $Q(E_i)$.

From the above we may see AEBN is not a strict extension of BN for it invites additional interdependence relations which is not from the definition of BN. Concretely, when we look at the global network, some parent nodes are not influenced by its descendent nodes’ states if they are in the input set since their values are fixed by the soft evidences. In AEBN, relations between agents are asymmetric.

To maintain the global consistence, agents must find the redundant influences invited by the circle linkages, which is so called rumor problem, and eliminate them. Figure 2.3 shows the communication graph of our first example. Since both observation agents receive the distribution of variable $S_1$ from the sensor agent, and send out their observations to the supervisor agent, the influence of the sensor agent is doubled in the supervisor agent. The final result $L$ is computed using the following formula:

$$P(L) = \sum_{R_i, R_j} P(L | R_i, R_j) P(R_i) P(R_j),$$

and by considering the observers’ local network, we get

$$P(L) = \sum_{R_i, R_j} P(L | R_i, R_j) [P(R_i | S_j) P(S_j)] [P(R_j | S_j) P(S_j)]$$

where $P(S_i)$ are double counted.

Bloemke[3] proposed two algorithms for the removal of such kind of redundant influences. The first approach utilizes flow network to discover where such kind of redundant influences appear and then eliminate them by compensating the repeated
counted propositions from preceding agents. The second approach is a centralized algorithm that involves automatic construction of an auxiliary three-layered local Bayesian network. By carefully manipulating the CPTs of this local Bayesian network and propagating beliefs, distributions of the first layer nodes are fed back to the original distributed networks.

![Figure 2.3. ROSE communication graph](image)

AEBN has limitations in expressing more complicated relations between variables. Also it does not address the issues of consistency when multiple connections between two BNs are present. It only discusses the rumors between networks, not the rumors between the connections. So in AEBN, the input set has to be in the root of the local BN, and if multiple connections are present between two BNs, the correspondent variables in output set must be d-separated to avoid rumors.

### 2.3.1.2 Multiply Sectioned Bayesian Network (MSBN)

Multiply Sectioned Bayesian Network by Xiang[32, 33] is a prominent distributed multi-agent uncertain reasoning system that divides a large Bayesian network into multiple sections and distributes them to agents. Xiang starts with five basic assumptions on ideal knowledge representation formalisms for such kind of systems:

1. *Each agent’s belief is represented by probability.*
2. *An agent can in general influence the belief of each other agent through direct or indirect communication but can communicate directly to another agent only with shared variables.*
3. *A simpler agent organization is preferred.*
4. *A DAG is used to structure each agent’s knowledge.*
5. *Within each agent’s subdomain, a JPD is consistent with the agent’s belief. For shared variables, a JPD supplements an agent’s knowledge with others’.*
Following these assumptions, MSBN conforms to a hypertree multiple sectioned directed acyclic graph (MSDAG) structure. The definition of hypertree and hypertree MSDAG are listed below [34]:

**Definition:** let $G = (V, E)$ be a connected graph sectioned into subgraphs $\{G_i = (V_i, E_i)\}$. $G_i$s are organized as a connected tree $\Psi$, where each node is labeled as $G_i$ and each link between $G_k$ and $G_m$ is labeled by the interface $V_k \cap V_m$ such that for each $i$ an $j$, $V_i \cap V_j$ is contained in each subgraph on the path between $G_i$ and $G_j$ in $\Psi$. Then $\Psi$ is a **hypertree** over $G$. Links between the subgraphs of $\Psi$ are called hyperlinks. A hypertree $\Psi$ is a **hypertree MSDAG** if for each shared node, there is one and only subgraph $G_i$ that contains all its parents.

A hypertree MSDAG is transformed into a tree or forest of a set of Junctions trees. The messages passed between sectioned BNs contain joint probabilities on hyperlink labels. Xiang develops a protocol analogous to Junction tree message passing for sectioned Bayesian networks to coordinate message passing and efficiently synthesizing correct beliefs from the messages.

Compared with AEBN, MSBN has three advantages. First, it does not invent any additional relations between the variables besides the chain rule of Bayesian network. Second, the interdependence relation of variables still follows d-separation in the compositions of sections. While in AEBN, decedents could not affect the variables’ beliefs in ascendants. Third, MSBN need not to concern itself for the rumor problem since its global structure is a DAG.

The limitations of MSBN are also obvious. To make composition of MSBN sound with traditional BN, MSBN introduce some restrictions into sectioning, some of them are very strict. First, local BNs have additional independent relations besides those in traditional BN. As the global independent relations remain traditional, the local network’s belief on shared variable may not be consistency with their global values. To solve this, MSBN requires a great communication cost between the subgraphs. Second, distribution of domain knowledge is forced on shared variables: a hypertree MSDAG requires that the complete set of parents of a shared node appear in one and only one subgraph. This limit the expressiveness of MSBN in describing distributed knowledge.

Xiang does not emphasis the autonomous attribute in its agent settings. Valtorta and Kim [30] note that it seems that “MSBNs were introduced as a method for parallel distributed inference within a single Bayesian network”.

### 2.3.2 Ontology mapping/translation

Different domain knowledge may classify same concepts in different semantics. In
semantic web, different ontologies may use different terms to describe same concepts, or use same terms to describe different concepts. We need to use ontology mappings to mark up the related concepts in different ontologies. And if possible, we want to translate the instances of classes from one ontology to another using the mappings.

Current ontology mapping approaches can be classified into two sorts: similarity approach and logical approach. Similarity approaches calculate the similarity of two concepts and use a threshold to judge if two concepts are identical, while logical approaches use logical axioms to model the relation between the concepts. Similarity approaches can discover and create mappings automatically or semi-automatically by using learning algorithm or supervised learning algorithm, while logical approaches need users to create the mappings manually. Similarity approach can only mark two concepts as identical or disjoint. Logical approaches are much more expressive. Their expressiveness depends on the language they use for mapping modeling.

Similarity approaches associate data with concepts in ontologies to compute the similarity between concept pairs. The data source can be description documents, relevant documents, names, labels, comments of the concepts, and instances. Most approaches use text based learning algorithm or heuristics to calculate the similarities. Some uses the local structure info to help improve the accuracy.

GLUE\cite{10} is a successful system by University of Washington in exhibiting a standard model of similarity based ontology mapping system. Given one concept in ontology A, Glue aims to calculate the similarity of this concept and all other concepts in ontology B, and find the most similar one. GLUE is capable to work with different kinds of similarity measurements. It applies multiple machine learning algorithms and uses a meta learner to combine the learning results. Learning results are represented as joint probabilities, which are further translated into similarity matrix by applying similarity measurements. Finally a Relaxation labeler module takes the similarity matrix along with domain constraints and other heuristics knowledge to search for the best mapping configuration. GLUE uses text, practically a set of instance documents of concept and the concept’s name, as raw data for similarity calculation. Naïve Bayes learning method is used to classify instance documents into four sets: $P(A,B)$, $P(A,\overline{B})$, $P(\overline{A},B)$, and $P(\overline{A},\overline{B})$ and a set of joint probabilities is calculated based one these sets.

OntoMapper\cite{29} is another similarity approach by UMBC, which improves the text-based classification result by conducting probability reasoning using local structure. First, raw similarity scores between concept pairs is obtained by using Rainbow classifier\cite{51} system on a set of instance documents. Then, two algorithms are provided to refine the results. The first one is a simple heuristic method which
realizes subsumption based on the majority rule: for any non-leaf node, if the ratio of its children that map to a particular node is greater than or equal to a user-specified threshold, then these children’s mappings along with the values associated are propagated up to the parent node. The second one is a probabilistic approach based on Bayesian reasoning, with the following three assumptions:

1) Exemplars associated with any non-leaf node “N” are documents that belong to this class and cannot be classified into any one of its subclasses. OntoMapper creates one leaf-node called “N-other” as a child of “N” and move all exemplars of “N” to “N-other”. Similarity scores given by Rainbow classifier are now between leaf nodes of these two ontologies.
2) All leaves of the ontology hierarchy are mutual exclusive and exhaustive.
3) Interprets $SMba(A_j, B_i)$ as $P(A_j | B_i)$ for any leaf node $B_i$. For a non-leaf node $B_i$, compute $P(A_j | B_i)$ as follows:

$$P(A_j | B_i) \approx \frac{1}{|children(B_i)|} \sum_{B_j \in children(B_i)} P(A_j | B_j)$$

OntoMerge from Yale [11, 41] is an online ontology translation system which translates an ontology A into a new DAML+OIL ontology B that captures the same semantic information. It is built on top of PDDAML (PDDL-DAML Translator) [45] (based on Jena [44]) and OntoEngine (an inference engine) (based on JTP [46]). Users need to first specify the relations between mapped variables using logical expressed bridging axioms. Axioms use vocabulary from items from both source and target ontologies and use namespaces to avoid duplicate names. At first, OntoMerge calls
PDDAML to translate ontology B into a Web-PDDL file, and then feeds this file to OntoEngine. OntoEngine retrieves a merged ontology C from its library which covers A and B. OntoEngine then tries to load ontology C in by first using PDDAML to translate it to Web-PDDL file. At last, OntoEngine loads A in, translates it, and calls PDDAML again to translate the Web-PDDL results back into DAML+OIL. The merged ontology C is obtained by taking the union of the terms and axioms defining them. Bridging axioms that relate the terms in A to the terms in B through the terms in C should also be added. Since all the axioms used are specified by human experts, this approach is semi-automatic.

2.3.3 Probabilistic Extensions to Ontology

Our research of translating ontology mappings into Bayesian network mappings is based on Ding’s framework BayesOWL [8, 9] which translates an OWL ontology into Bayesian network representations by incorporating uncertain information. To represent uncertainty relations between concepts in an ontology, Ding [8] first proposes a set of extension markups to OWL. An ontology about these markups is defined for this purpose. For each probabilistic relation, an element of “PriorProb” or “CondProb” is created with a property “hasVariable” to identify the variable’s name, and one or more “hasCondition” properties to identify the conditioned variables, and a “hasProbValue” to quantify the probability. Then Ding defines a set of rules to convert probabilistically annotated OWL ontologies into the graphical model of Bayesian networks[8]:

1. A primitive class C is mapped into a node in the translated BN;
2. A class C defined with superconcepts \( C_i, i = \{1, \ldots, n\} \) by “rdfs:subClassOf” is mapped into a subnet in the translated BN with converging linkages from each \( C_i \) to \( C \);
3. A class C defined by set intersection (owl:intersectionOf) or union (owl:unionOf) of classes \( C_i, i = \{1, \ldots, n\} \), is also mapped into a subnet in the translated BN with converging connection from each \( C_i \) to \( C \);
4. If a concept \( C_1 \) is defined by the mutual exclusive (owl:complementOf), or equality (owl:equivalentClass), or disjoint (owl:disjointWith) of another concept \( C_2 \), then a control node is added to the translated BN, and there are directed links from \( C_1 \) and \( C_2 \) to this node.

Another work similar to Ding’s is from Holi and Hyvönen [13], which aims to approximate the concept overlaps of a semantic web taxonomy using Bayesian network. It first uses Venn diagram to illustrate the concept overlaps in a taxonomy, and proposes a graph notation for it. Each sub area in the graph is translated as a variable node in the Bayesian network. The arc points from sub concepts (overlaps) to super concepts.

Compared with BayesOWL, to use this method for mapping creation, we will meet the problem of identifying variables between Bayesian networks and ontologies. Since
the overlap within one ontology is represented, the overlaps between ontologies, known as mappings, are difficult to identify since the process may involve the interaction of two Venn diagrams and hence may change the BN representation dramatically.

In this section (2.3), we reviewed existing approaches in distributed Bayesian network models, ontology mapping/translation, and probability extensions to semantic web. MSBN is successful in parallel inference but is restrictive in that it does not aim to connect separate BN together rather than split one large BN into sections. AEBN use soft evidential update method to implement the communication of separate BNs, but does not specify the network consistency issues of connected networks. Also AEBN only create links between identical variables. GLUE and OntoMapper is referred as ontology mapping methods that utilize uncertainty information. BayesOWL is also regarded as the base of our research on probability extension on ontology mapping.
Chapter 3

Proposed Research and Approaches

The objective of our research is to develop a theoretical framework and related inference methods to map semantically similar variables between separate Bayesian networks in a provably correct way. This framework can be used to support mapping between independently developed ontologies with overlapped domains. This research is to be conducted in two steps.

For the first step, our goal is to create Bayesian network mappings that are expressive, introducing fewer additional constraints, and ensuring the correctness of inference over the composite network. The mappings should be capable of describing probabilistic relation between two variable sets with constraints only to maintain correctness and consistency. The mappings creation should be flexible enough for creators to express whatever constraints they wish to specify. The mappings must maintain the consistency of mapped variables under the required probabilistic relations. For the second step, our approach is to apply the framework to ontology mapping and translation based on Ding’s BayesOWL, a probability extension to OWL language [8, 9]. In this regard, we will transform ontology mappings into Bayesian network mappings and use the inference method in the first step to generate statements on propositions in separate BNs.

In Section 3.1 we present an overview of our framework and related questions. In Section 3.2 we discuss issues concerning linkage between a single pair of variables from two different BNs, and address consistency of linked variables. Section 3.3 is about mapping reduction, the process of deriving BN mappings from BN linkages with more restrictions on connected nodes to ensure the consistency between linked networks. Section 3.4 lists issues concerning inference across multiple BNs using BN mappings. Section 3.5 presents our thoughts on extending our Bayesian network mapping to support ontology mappings.

3.1 Bayesian Network Mapping: an Overview

When we have multiple BNs describing different aspects of the same domain or their domain overlap, we need new methods to exchange beliefs and observations
between these BNs. Bayesian network mapping is a framework designed for this purpose. As reviewed in the previous section, in MSBN, connections between sectioned BNs are represented by shared variables, which are identical and have the same distributions, and reside in the same contexts. In AEBN, linked variables are identical but may have different distributions. In contrast, our framework has two obvious advantages. First, linked variables are similar but not necessarily identical, and second, they may have distributions that do not agree with the probabilistic relations between them. With looser constraints in the linkage, more effort is required to ensure the consistency of the linkages.

Figure 3.1. Overview of BN mapping framework

Figure 3.1 depicts the proposed framework for BN mapping. Detailed description of this model can be found in the next paragraph. Our framework models BN mapping by a set of four-layered concepts. Then utilities and methods are used to represent and implement the BN mapping model. Soft evidence is used to represent the linkages between variables and soft evidential update method is used to carry inference across multiple BN through the linkages. Mapping reduction process uses given inference method to generate a parsimonious set of valid BN mapping from given variable linkages. As an application, ontology mapping is supported by the BN mapping model, which also uses our inference method.
Figure 3.2 further illustrates the four-layered model of BN mappings. The first layer models the similarities between variables as *pair-wise probabilistic relations* (Figure 3.2(a)), which use joint probabilities to represent the overlap of the two variables’ domain. Then *pair-wise variable linkages* (Figure 3.2(b)) are created for these probabilistic relations to represent the interdependence between the variables’ distributions. The third layer is *valid BN mapping* (Figure 3.3(c)), which ensures the consistency of mapped networks. The ideal goal of mapping reduction is Minimum valid BN mapping (Figure 3.3(d)), which is obtained by the process of *mapping reduction*, a process selecting a parsimonious set of linkages from redundant linkages. Precise definitions of the above concepts are given in the next subsections.

We have conducted preliminary investigations on pair-wise linkage between variables in different BNs and obtained encouraging results. These including the following:

- Formalization of the notation of variable linkage between Bayesian networks, including its semantics.
- Development of method for reasoning across BNs via variable linkages.
- Development of a virtual evidence based soft evidential update method.
- Validation of Bayesian network mapping.

We have carried out a series of computer experiments of limited scope to validate the results of our preliminary work. These are given in the next subsections.

### 3.2 Formalization of Mapping
If variable $A$ in $BN_A$ and variable $B$ in $BN_B$ are identical, then a link between them would be easy to describe, even though they may not have the same distribution. However, it becomes less clear in representing their interrelation and influence if they are similar but not identical. Moreover, since “similarity” is a relative concept, should we link $A$ to all or only some variables in $BN_B$ that are similar to $A$ when we trying to connect these two BNs? Answers to theses questions require a formalization of the concept of “mapping” between variables across BN. This will be done in two stpes. First we will formalize the notion of the connection between a pair of variable in two BN, called pair-wise variable linkage or linkage for short, then we extend it to network mapping, or mapping for short.

### 3.2.1 Pair-wise Variable Linkage

The purpose for building linkages between variables in different Bayesian networks is to propagate the probability influences from one network to the other, and thus allowing probabilistic reasoning across multiple networks. A linkage here links a pair of variable, each of which is in a different BN. In Figure 3.3, we give an overview on what kind of connection our approach requires for such a pair of variables. Consider two Bayesian networks $BN_A$ and $BN_B$, which represent two probability spaces $S_A$ and $S_B$, respectively. Suppose variable $A$ in $BN_A$ and variable $B$ in $BN_B$ are two similar concepts, a linkage can be established between these two variables to describe their interrelation, represented as a probability constraint, which captures the similarity between $A$ and $B$. Once the linkage is established, the prior or posterior distribution of node $A$ in $BN_A$ can be propagated to node $B$ in $BN_B$, changing the distribution of $B$ in a way consistent with the linkage constraint, and in turn influencing the beliefs of other variables in $BN_B$.

![Figure 3.3. Bayesian network connection](image)

In MSBN[34], sectioned Bayesian networks use shared nodes to connect with each other and pass probability potentials between these shared nodes. The linkages there are symmetric in that evidences entering either side may have influence to the other side. There are three reasons for the symmetry of MSBN’s linkages. First, shared
nodes in MSBN are identical nodes. Second, MSBN requests that the global view of the network is a Bayesian network, which means if we merge the sectioned networks by overlaying all shared nodes, they become a single valid Bayesian network. Third, only one node of a shared pair has a complete CPT, which guarantees that two nodes always have consistent conditional distributions.

However, these restrictions are no longer required in our approach. A linkage now can be created for any variable-pair in separate BNs as long as a probabilistic relation/constraint can be established between them. Also, similar to AEBN, we model linkages between variables in different BNs as directed links to indicate how the belief flows.

If $A$ and $B$ are identical variables (e.g. denoting the same concept), they are related by the equivalence constraint. This is the simplest case of pair-wise constraints, as modeled by AEBN. As depicted in Figure 3.4, when an observation of variable $A$ (and hence $B$ since $A$ and $B$ are identical) is made in $BN_A$ (a posterior of $A$) as $P(A)$, this observed distribution of variable $B$ can then be used to update the distributions of $BN_B$ by soft evidence version of Jeffery’s rule using $P(B|se) = P(A)$:

$$P(V_B \setminus B \mid se) = \sum_i P(V_B \setminus B \mid B = b_i)P(B = b_i \mid se).$$  \hspace{1cm} (3.2.1)

where $V_B$ denotes all the variables in $BN_B$.

In more general situations as depicted in Figure 3.5, node $A$ in $BN_A$ and node $B$ in $BN_B$ represent two similar but not identical concepts. The probabilistic relation associated with the linkage between $A$ and $B$ captures the similarity between them. However, in general we do not have the control of how this constraint is obtained. So we assume that the similarity is described in a probability space other than $S_A$ and $S_B$. Let $S_S$ denote the probability space where the similarity of $A$ and $B$ is captured. Here we do not require $S_S$ to be represented by a BN. We define $S_S$ directly as follows:
Here variable $A'$ in $S_s$ represents the same concept as the node $A$ in $BN_A$, variable $B'$ in $S_s$ represents the same concept as the node $B$ in $BN_B$. There are equivalence linkages between $A$ and $A'$ and between $B$ and $B'$. Viewing $P(A)$ as a soft evidence to $A'$ we have

$$P(A'|se_1) = P(A).$$

(3.2.3)

From the Bayes conditionalization formula of Jeffery’s rule we know

$$P(B'|se_1) = \sum_i P(B'|A' = a'_i)P(A' = a'_i|se_1),$$

(3.2.4)

where $P(B'|A' = a'_i)$ is from $P_s$ in $S_s$. Then by applying $P(B'|se_1)$ as soft evidence to $B$ from $B'$ we have

$$P(B|se_2) = P(B'|se_1).$$

(3.2.5)

From (3.2.3), (3.2.4), (3.2.5), we have the distribution of $B$ as:

$$P(B|se_2) = \sum_i P(B'|A' = a'_i)P(A = a_i).$$

Then for all other variables in $BN_B$, the distribution is update as follows:

$$P(V_B \setminus B|se_2) = \sum_i P(V_B \setminus B = b_i|B = b_i)P(B = b_i|se_2)$$

### 3.2.2 Mapping between two BNs

Now we can summarize the above analysis and formalize the notion of Bayesian
network mapping.

**Definition 3.1:** A Linkage $L_{AB}$ from variable $A$ in Bayesian network $BN_A$ to variable $B$ in Bayesian network $BN_B$ with probabilistic relation $Rel(A, B)$ is defined as $<A, B, BN_A, BN_B, Rel(A, B)>$.

The form of the pair-wise probabilistic constraint $Rel(A, B)$ is a conditional distribution of $P(B=b_j | A=a_i)$, which can be represented as an $m \times n$ matrix $Rel(A, B) (i,j) = \{P(B=b_j | A=a_i)\}$, where $i = 1,...,m$, $j = 1,...,n$, $m$ and $n$ are the number of states of $A$ and $B$, respectively. The distribution of $B$ then can be computed as the product of $A$'s distribution as a vector and the matrix $Rel(A, B) (i,j)$:

$$P(B) = P(A) \cdot Rel(A, B).$$

When $A$ and $B$ are binary variables, we have

$$P(B) = (P(A), P(\overline{A})) \begin{pmatrix} P(B | A) & P(\overline{B} | A) \\ P(B | \overline{A}) & P(\overline{B} | \overline{A}) \end{pmatrix} = (P(A), P(\overline{A})) \begin{pmatrix} P(B / A) / P(A) & P(\overline{B} / A) / P(A) \\ P(B / \overline{A}) / P(\overline{A}) & P(\overline{B} / \overline{A}) / P(\overline{A}) \end{pmatrix}.$$ 

**Definition 3.2:** Bayesian networks $BN_A$ and $BN_B$ have an overlap in their domain. This overlap is represent by set of related variables, $A_1, ..., A_m$ in $BN_A$, and $B_1, ..., B_n$ in $BN_B$. The similarity relationships between these variables are represented by a set of Probabilistic Constraints stated as follows:

For each pair of related variables, $A_i$, $B_j$, and a probabilistic relation $Rel(A_i, B_j)$ such that $P(B_j) = P(A_j)Rel(A_i, B_j)$.

As a special case, constraint of equivalent mapping is:

For each pair of equivalence variables $A_i$, $B_j$, $Rel(A_i, B_j) = 1$, and thus $P(B_j) = P(A_j)$.

If all the constraints between two BNs are satisfied, we say that the two related BNs are consistency on given probabilistic relations.

**Definition 3.3:** Suppose there are $m$ probabilistic relations between variables in $BN_A$ and $BN_B$. A linkage set $L$ of $k$ linkages $L_{A1B1}$, ..., $L_{AkBk}$ between $k$ pairs of nodes $A_1$, ..., $A_k$ in $BN_A$ and $B_1$, ..., $B_k$ in $BN_B$ is a valid BN mapping between $BN_A$ and $BN_B$, if and only if $L$ ensures the consistency of $BN_A$ and $BN_B$ over the $m$ probabilistic relations.

Please note that satisfying a probabilistic relation $Rel(A_i, B_j)$ does not require there exists a linkage between $A_i$ and $B_j$. Multiple probabilistic relations could be satisfied by one linkage. However, a probabilistic relation $Rel(A_i, B_j)$ does imply that there could exists a linkage between $A_i$ and $B_j$. Concrete examples can be found in
subsection 3.2.4.

The semantics of BN linkages will be further investigated for the following and related issues:

- Would pair-wise variable linkage sufficient to represent interrelations between two BNs?
- What effect does introduction of variable linkage have on the \( d \)-separation among the variables in the destination BN?
- Since the connected variable’s distributions are consistent, we can use the linkages to merge the connected probability spaces \( S_A \), \( S_B \) and the probability spaces that describe probability constraints (\( S_s \)) into a single probability space?
- If we need to rewrite the chain rule of BN with cross BN linkages on the chain and how?

### 3.2.3 Experiments

A series of simulation experiments have been conducted to validate our ideas. In these experiments, we use a synthetic Bayesian network as a basis and induce two Bayesian networks, in which some variables are absorbed into other variables. We can obtain the probabilistic relations of variable pairs between the two induced BNs from the original BN. Linkages are established between randomly selected pairs of variables in these two induced networks.

![Diagram of BNs](a) \( BN_0 \) Original Bayesian network
Two BN, $BN_1$ and $BN_2$ in Figures 3.6(b) and 3.6(c), respectively, are induced from $BN_0$ in Figure 3.6(a) by node absorption and necessary adjustment to CPT so that any inference done with them yields the same result as in $BN_0$. We can see that the distributions of variables $A$ in $BN_1$ and $BN_2$ are identical to those in $BN_0$. Also note that although $BN_1$ and $BN_2$ do not share any variable, some in $BN_2$ are similar to some in $BN_1$. For example, $C$ in $BN_2$ is similar to $A$ in $BN_1$ because $A$ is a parent of $C$ in $BN_0$, and similarly, $C$ is also similar to $I$ and $K$ in $BN_1$.

We instantiate node $C$ as True in $BN_2$ as shown in Figure 3.7 (a). To propagate the evidence $C = \text{True}$ to $BN_2$, we create linkages $< C, A, BN_2, BN_1, R_1>$, where $R_1$ is calculated in the original Bayesian networks as

$$R_1 = \begin{pmatrix} 1.0 & 0.0 \\ 0.819 & 0.181 \end{pmatrix}$$

Then we know the soft evidence on $A$ should require the distribution of $A$ as $P(A) = (1.0, 0.0)$. The result is showed in Figure 3.7(b). We can see that the distributions of variables $E$ and $D$ are correct but $I$ and $K$ in Figure 3.7(b) are inconsistent with the distributions of variables $I$, $K$ in Figure 3.7(c), which shows the original Bayesian network with variable $C$ instantiated. To correct this, we add linkages $< C, I, BN_2, BN_1, R_2>$ and $< C, K, BN_2, BN_1, R_3>$ with

$$R_2 = \begin{pmatrix} 0.288 & 0.712 \\ 0.85 & 0.15 \end{pmatrix} \quad \text{and} \quad R_3 = \begin{pmatrix} 0.375 & 0.625 \\ 0.0 & 1.0 \end{pmatrix}.$$ 

The soft evidence on $I$ and $K$ have constraints as: $P(I) = (0.288 \ 0.712)$, and $P(K) = (0.375 \ 0.625)$. Also can be see in comparing Figure 3.7 (c) and (d), when the three linkages are used, the influence of $C = \text{true}$ is correctly propagated in $BN_1$, and therefore they form a valid BN mapping.
3.3 Mapping Reduction

As revealed in the proceeding example, a single linkage from one variable may not be sufficient to properly carry influence from one BN to another. On the other hand,
many linkages can be established between pairs of variables of separate BNs. In theory, any pair can be linked, albeit with different similarity constraints. It is obvious that not every potential linkage is necessary in ensuring correct inference across BN. A process of selecting a small subset of all variable pair-wise variable linkages is required so that the consistency can be guaranteed. This process is called “Mapping Reduction”, and is illustrated by the following two examples.

**Example 1 (Figure 3.8):**

Figure 3.8(a) shows the original BNs: $BN_1$ on the left and $BN_2$ on the right. The CPT of $B_1$ and $B$, $C_1$ and $C$, …, $F_1$ and $F$ are all the same. The only difference of these two networks is the prior distributions of $A_1$ and $A$. Our goal is to change $BN_2$’s probability distribution to $BN_1$’s probability distribution. The pair-wise probabilistic relations are correctly collected and equal to the conditional probability distributions of $BN_1$. They are listed as follows:

\[
\begin{align*}
Rel(A_1, A) &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \\
Rel(B_1, B) &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \\
Rel(C_1, C) &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \\
Rel(D_1, D) &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \\
Rel(E_1, E) &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \\
Rel(F_1, F) &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \\
Rel(C_1, A) &= \begin{pmatrix} 1 & 0 \\ 0.3 & 0.7 \end{pmatrix}, \text{ and} \\
Rel(C_1, B) &= \begin{pmatrix} 0.8 & 0.2 \\ 0.24 & 0.76 \end{pmatrix}.
\end{align*}
\]

Then we create linkages for each of these relations. What we want is a set of mapping that could change the distributions of $BN_1$ to the distributions of $BN_2$. One thought is to use the mapping with the greatest similarity, e.g. the linkage from $C_1$ to $C$ with similarity 1. Figure 3.8(b) shows the result of applying linkage from $C_1$ to $C$. We can see that the posterior distributions of $A$, $B$, $D$, and $E$ are not correct but those of $C$ and $F$ are correct. If we instead map $C_1$ to $A$ as shown in Figure 3.8(c), distributions of all variables in $BN_1$ are updated correctly (to become identical to those in $BN_2$).
Example 2 (Figure 3.9):

Figure 3.9(a) shows two BNs: $BN_1$ and $BN_2'$, where $BN_2'$ is induced from $BN_2$ by absorbing node $A$ into $B$ and $C$. From the first example we know that we can not map $C_1$ to $C$. Alternatively, we can try a linkage from $C_1$ to $B$ since $B$ is the root of $BN_2'$. As can be seen in Figure 3.9(b), only the distributions of node $B$, $D$ and $E$ are correct but those of node $C$ and $F$ are incorrect. Figure 3.9(c) shows the best result by mapping $C_1$ to both $B$ and $C$, which updates the probability space of $BN_1'$ to the probability space of a subnet of $BN_2$. 
From the above examples, we can see that our main research objective here is to develop an algorithm that automatically recognize a set of non-redundant linkages that can represent all required probabilistic relations based on the local structure of Bayesian network. We have the following questions to answer in developing the algorithm:

- **Weak linkages:** is it reasonable to remove weak linkages, and how can we determine a linkage is weak? Weak mappings are not desired because they are not likely to reflect semantic relations of propositions and invite computational errors. However, some weak constraints could not be removed as they are significant. This will be seen as a tradeoff. To measure the strength of a probabilistic relation, we propose the **Jaccard coefficient** [51] when the connected variables are binary:

\[
\text{Jaccard} (A, B) = \frac{P(A \cap B)}{P(A \cup B)}
\]

\[= \frac{P(A, B)}{P(A, B) + P(A, B) + P(A, B)}.\]

For multi-valued variables, users should specify their own similarity measurement that is meaningful in telling the overlap of the propositions on connected variables.

- **Equivalent linkages problem:** different linkages that have equivalent effects in destination BN. We could use belief update algorithm to find redundant
linkages. Then to reduce linkages, one heuristics could be reducing linkages to those that have higher similarity and have equivalent influence. However, we do not require the selected linkages have the best capability in reducing other linkages since we are not searching for a minimum set. Another disposal could be to mark them with the linkages that make them redundant and select them dynamically in the inference process.

- The $m$-1 problem: one variable has linkages coming from multiple variables in another BN. We have two choices: select one or combine all. To combine the probabilistic relations, our preliminary work finds that current pair-wise information may not be sufficient. Hence additional learning may be needed.
- Dynamic reduction or static reduction: for the whole network, we expect to get a set of static BN mappings. However, with respect to a particular inference tasks, the reduction needs to be dynamic.

### 3.4 Inference

Using valid BN mappings to conduct inference across multiple BNs is the final step of our BN mapping framework. We need to first address the problems of observation exchange, which can be stated as follows:

Suppose we have variables $A_1, \ldots, A_m$ in $BN_A$ related with $B_1, \ldots, B_m$ in $BN_B$ under a set of probabilistic relations. We observe an evidence on variable $A'$ in $BN_A$. Then how to propagate this observation to $BN_A$ through the related variables?

Proposed solution of this problem is: if evidence node $A'$ is one of $A_i$, $1 \leq i \leq m$, then the probabilistic relation $Rel(A_i, B_j)$ can be used to change $B_j$’s distribution. Otherwise, (i.e. $A'$ is not one of $A_i$), the influence of $A'$ to variables in $BN_B$ may be achieved by first propagating the influence of $A'$, changing beliefs of some variables in $\{A_1, \ldots, A_m\}$, and then updating the distributions of $BN_B$’s variables that are related to those $A_i$ via probabilistic relations. To satisfy a probabilistic relation, we need to change one variable’s distribution by soft evidence. A soft evidential update method is proposed to process probabilistic relations. The correctness of observation exchange is ensured by the previous mapping reduction process. In the following subsections, we will present an algorithm for belief exchange.

#### 3.4.1 Inference Process

Suppose we have two Bayesian networks $BN_A$ and $BN_B$, and valid BN mappings as $k$ linkages $L_{A_1B_1}, L_{A_2B_2}, \ldots, L_{A_kB_k}$ between $k$ pairs of nodes $A_1, A_2, \ldots, A_k$ in $BN_A$ and $B_1, B_2, \ldots, B_k$ in $BN_B$. Note that more than one of these linkages may start from one node in $BN_A$ and more than one may end at one node in $BN_B$. The inference process outline is described as below:
1. Apply the hard evidence in $BN_A$ and then obtain the distributions of the start nodes $A_1, A_2, \ldots, A_k$ of linkages $L_{A_1B_1}, L_{A_2B_2}, \ldots, L_{AkB_k}$:
   
   \[ P(A_1| \text{hard evidence}), \ldots, P(A_k| \text{hard evidence}). \]

2. For each linkage, apply the probability constraint $\text{Rel}(A_i, B_i)$ to $P(A_i| \text{hard evidence})$ and obtain the desired distributions of $B_i$:
   
   \[ Q(B_i) = P(A_i| \text{hard evidence}) \text{Rel}(A_i, B_i). \]

3. Enter the hard evidence to $BN_B$.

4. Use soft evidential method to update $BN_B$ with both hard evidence and the soft evidence $Q(B_1), \ldots, Q(B_k)$.

In this inference process, posterior distributions of source BN $BN_A$, are generated in the first step, which are then propagated to the destination BN $BN_B$ through BN mappings, which ensure the consistency of the mapped BNs. Step 3 entered hard evidences that do not influence soft evidence nodes and Step 4 update the belief of both entered soft evidence and hard evidence. If users want make new observations after current soft evidence, like entering new soft or hard evidence, new evidence can be applied directly to the BN.

### 3.4.2 Virtual Evidence Based Soft Evidential Update

Pearl has proved in [27] the equivalence of virtual evidence and soft evidence in changing single variable’s distributions. In this subsection, we will prove the equivalence when multiple evidences are present. Although virtual evidence carries different semantic from soft evidence, they are equivalent in expressive power. Since they both have the ability of switching probability distribution of a Bayesian network to another, then if a set of soft evidence $SE$ changes $BN$’s distribution $S$ to $S'$, there should exist a set of virtual evidence $VE$ that also changes $S$ to $S'$. Moreover, since virtual evidence uses Bayes’ conditionalization to update variables’ distributions, inference method based on it does not need to apply IPFP on probability tables like Voltorta’s algorithm, and thus can be easily incorporated into any existing BN inference algorithm, whether junction tree based or not.

Besides the Big Clique algorithm, Valtorta proposes another space efficient algorithm based on Junction Tree [21]. In this algorithm, the Junction Tree is built as usual. After hard evidence is absorbed, and the whole tree is updated, IPFP is applied for soft evidential update. All cliques receive soft evidence by:

\[ Q_i(C_{j_i}) = Q_{i-1}(C_{j_i}) \frac{Q(a_j)}{Q_{i-1}(a_j)}, \]

where $Q_i(C_{j_i})$ denotes the joint probabilities of the clique containing variable $a_j$ in $i^{th}$ iteration, $j=1+(i-1) \text{ mod } k$. In each step a clique is updated using the above formula and then messages are propagated normally. The procedure is iterated until
$Q_n(a_j)$ converges to $Q(a_j)$ for all $j$. Valtorta states that this iteration procedure could “be implemented as a wrapper around the Hugin shell or other shells that support the virtual evidence method.”

In contrast, our algorithm conducts soft evidential update by virtual evidence and thus does not require the conversion of a given BN to a junction tree. Rather, it directly computes likelihood ratios from given soft evidence as virtual evidence to evidence nodes. When multiple virtual evidences are present, they influence each other; update one node’s distribution to its target value can make those of others’ off their targets. So we have to iterate over these nodes and revise the likelihood ratio of the virtual evidence so that all variables’ distributions converge to their respective soft evidence.

**Lemma 3.1.** Let $A$ be a proposition with prior distribution $P(A)$, which is to be updated by a soft evidence update to $Q(A)$. Then there exists a virtual evidence $ve$ that, when attached to $A$, will update $P(A)$ to $Q(A)$.

**Proof.**
Suppose variable $A$ has $n$ states $\{a_1, \ldots, a_n\}$ and virtual evidence $ve$ on node $A$ satisfying

$$Q(A) = P(A | ve),$$

For $1 \leq i \leq n$, let proposition $A_i$ denote $A = a_i$, then we have

$$Q(A_i) = P(A_i | ve)$$

$$= P(ve | A_i) P(A_i) / P(ve)$$

$$= P(ve | A_i) P(A_i) / (P(ve | A_1) P(A_1) + \ldots + P(ve | A_n) P(A_n))$$

This equation can be rewritten as:

$$Q(A_i) P(A_1) P(ve | A_1) + \ldots + (Q(A_i) - 1) P(A_i) P(ve | A_i) + \ldots + Q(A_i) P(A_n) P(ve | A_n) = 0$$

Then we get linear equations on $P(ve | A_i)$ as follows

$$\begin{pmatrix}
(Q(A_1) - 1) P(A_1) & \cdots & Q(A_1) P(A_n) \\
Q(A_i) P(A_i) & \cdots & (Q(A_i) - 1) P(A_i) & \cdots & Q(A_i) P(A_n) \\
\vdots & \cdots & \vdots & \cdots & \vdots \\
(Q(A_n) - 1) P(A_1) & \cdots & (Q(A_n) - 1) P(A_i) & \cdots & Q(A_n) P(A_n)
\end{pmatrix},$$

and

$$X = \begin{pmatrix}
P(ve | A_1) \\
\vdots \\
P(ve | A_n)
\end{pmatrix},$$

$$CX = 0.$$  \hspace{1cm} (3.4.1)

It can be easily shown that determinant’s value of matrix $C$ is zero, so (3.4.1) must have non-zero solutions. And any normalized non-zero solutions of (3.4.1) satisfy $P(A | ve) = Q(A)$.  \hfill \blacksquare
When the variable is binary, the likelihood of \(ve\) is

\[
L(A) = \frac{P(ve | A)}{P(ve | \overline{A})} = \frac{P(A)Q(\overline{A})}{Q(A)P(\overline{A})}.
\]

If the inference algorithm does not directly support virtual evidence, we may create a dummy node using the following procedures [25]:

1. Create a dummy node with binary states and make the soft evidence node its parent.
2. Set the CPT of the dummy node with values obeying the likelihood ratio for simulating the soft evidence.
3. Instantiate the dummy node to \(True\).

In practice, if the variable is binary, an easy way to compute the CPT of this dummy node is:

\[
\begin{align*}
P(ve | A) &= P(A)Q(\overline{A}) \\
P(ve | \overline{A}) &= P(\overline{A})Q(A)
\end{align*}
\]

Next, we propose an algorithm to simulate soft evidence using virtual evidence, and then prove that it generates the \(I\)-projection of the original distribution on all of the given soft evidence.

**Explanation of the notations:**
- \(V\): the set of all variables in the BN
- \(k\): the total number of soft evidence (and evidence nodes)
- \(P(A)\) (without subscript): the probability distribution of a node.
- \(P_i(A)\) (with subscript): the probability distribution of a node at \(i^{th}\) iteration.
- \(VE_i\): the set of all virtual evidence in the network at \(i^{th}\) iteration.
- \(ve_{j,l}\): the \(l^{th}\) virtual evidence attached to node \(j\). Here \(j = 1 + (i-1) \mod k\), \(l = 1 + \lfloor j - 1/k \rfloor\). So a virtual evidence is added to a soft evidence node for every \(k\) iteration steps.
- \(Q(A)\): the soft evidence distribution of \(A\)

Let \(\{A_1, ..., A_k\}\) be the soft evidence nodes, we use the following iteration method for the update:

**Algorithm 3.1 (Virtual Evidence based Soft Evidential Update)**

1. \(P_0(A_j) = P(A_j | \text{hard evidence})\).
2. Compute \(ve_{j,l}\) using equation (3.4.1) such that \(P_{i-1}(A_j | VE_{i-1}, ve_{j,l}) = Q(A_j)\). \(VE_i = VE_{i-1} \cup \{ ve_{j,l} \}\).
3. Enter \(ve_{j,l}\) to the BN, then \(P_{i}(A_j | VE_i) = P_{i-1}(A_j | VE_{i-1}, ve_{j,l})\).
4. Update the whole network with \( VE_i \) using any BN inference algorithm.
5. Go to step 2 until all \( P(A_j|VE_i) \) converge.
6. After the iteration converges, obtain likelihood ratios for all \( k \) virtual evidence node from the product of \( ve_{j,l} \): \( ve_j = \prod_i ve_{j,i} \). A more convenient way is to cumulate the virtual evidence in Step 2 in each iteration.

The convergence of IPFP is proved in [4] and [31] for discrete cases. Also we know that if we have a set of constraints and each step of an iteration method resulting an \( L_1 \)-projection on a constraint, the iteration result converges to the \( L_1 \)-projection on the set of these constraints [31, Thm. 2.5]. Here we will prove the convergence of Algorithm 3.1 by showing the equivalence of each step of our iterative procedure and an IPFP.

**Lemma 3.2:** algorithm 3.1 converges

**Proof.**
When \( k=0 \) (i.e., no soft evidence):
Only step 1 of Algorithm 3.1 is executed. The Bayesian network absorbs all hard evidence and finishes message passing, so the distribution of the network becomes
\[
P(V|\text{hard evidence}).
\]

When \( k = i, i>0 \):
A virtual evidence \( ve_{j,l} \) is applied on node \( a_j \) from Jeffery’s rule we have
\[
P_i(V|VE_i) = \sum_{a_j} P_{i-1}(V \setminus a_j | a_j, VE_{i-1})P_i(a_j | VE_{i-1}, ve_{j,l})
\]
\[
= \sum_{a_j} P_{i-1}(V \setminus a_j | a_j, VE_{i-1})P_i(a_j | VE_{i-1}) \frac{P_i(a_j | VE_i)}{P_{i-1}(a_j, VE_{i-1})}
\]
\[
= \sum_{a_j} P_{i-1}(V) \frac{P_i(a_j | VE_i)}{P_{i-1}(a_j | VE_{i-1})}
\]
\[
= P_{i-1}(V) \frac{P_i(A_j | VE_i)}{P_{i-1}(A_j | VE_{i-1})}
\]

(3.4.2) is an IPFP on single constraint \( P_i(A_j|VE_i) = Q(A_j) \), it generates an \( L_1 \)-projection of distribution \( P_{i-1}(V|VE_{i-1}) \). Since each step of our iteration method generates an \( L_1 \)-projection of previous distribution on single constraint, algorithm 3.1 converges to the \( L_1 \)-projection of \( P(V) \) on all soft evidence. 

\[\blacksquare\]
When a variable is directly influenced by more than one virtual evidence, we can use one virtual evidence to replace and summarize the total effect of all these virtual evidences.

**Lemma 3.3:** Suppose $n$ virtual evidence $ve_1, ve_2, ..., ve_n$ are applied on variable $A$, then there exists a virtual evidence $ve$ such that $P(A | ve_1, ve_2, ..., ve_n) = P(A | ve)$.

**Proof.**

\[
P(A | ve_1, ve_2, ..., ve_n) = P(ve_1, ve_2, ..., ve_n | A_i)P(A_i) / P(ve_1, ve_2, ..., ve_n)
\]

\[
= P(ve_1, ve_2, ..., ve_n | A_i)P(A_i) / (P(ve_1, ..., ve_n | A_1)P(A_1) + ... + P(ve_1, ..., ve_n | A_i)P(A_i))
\]

Since $A$ is the only parent of $ve_1, ve_2, ..., ve_n$, they are d-separated by $A$. It then follows

\[
P(A_i | ve_1, ve_2, ..., ve_n)
\]

\[
= \prod_j P(ve_j | A_i)P(A_i) / (\prod_j P(ve_j | A_1)P(A_1) + ... + \prod_j P(ve_j | A_i)P(A_i))
\]

\[
= P(ve_i | A_i)P(A_i) / P(ve)
\]

\[
= P(A_i | ve)
\]

where $P(ve_i | A_i) = \prod_j P(ve_j | A_i)$ □

**Theorem 3.1:** Virtual evidence has the same capacity in changing variable’s distribution as soft evidence.

**Proof.**

Directly from Lemmas 3.1 – 3.3 □

**3.4.3 Experiments**

We implement our iterative procedure of Algorithm 3.1 using Baysian Network Toolkit for Matlab[48] (BNT) [47] developed by Kevin Murphy. Since BNT does not have a good visualization interface, we use Netica [49] from Norsysto illustrate the results.

![Figure 3.10 Bayesian network for soft evidential update](image-url)
Figure 3.10 gives a small Bayesian network for our experiments. The prior distributions of each variable are displayed. Our goal is to change the distributions of this network after soft evidence $Q(A) = (0.5, 0.5), Q(E) = (0.3, 0.7)$ and $Q(F) = (0.3, 0.7)$ applied to nodes $A$, $E$, and $F$. The iterative procedure of algorithm 4.1 stops after 37 iterations when the error is reduced to below $10^{-4}$. The returned values are likelihood values of three virtual evidence: $L(A) = 0.0386/0.9993$, $L(E) = 0.2489/0.9685$, and $L(F) = 0.9951/0.0992$. Figure 3.11 shows the distributions of the example network after virtual evidence are applied to the three evidence nodes. To make it clear where virtual evidence is attached, we use dummy nodes to represent virtual evidence. Figure 3.12 shows the convergence of likelihood ratio in the iterative procedure. The $x$ axis is for the number of iteration steps and $y$ axis for the likelihood ratio.

Figure 3.11 BN with virtual evidence simulation of soft evidence.
3.5 Supporting Ontology Mapping

As an example of practical application of our framework, we will extend our system to represent ontology mappings. Based on Ding’s approach, we could obtain an uncertainty representation of ontology as Bayesian networks. We assume we already have probabilistic relations between concepts in two related ontologies, which are available as intermediate results from some ontology mapping method, such as GLUE [10] and OntoMapper [29], or through some learning process. What we need is a set of rules translating the ontology mappings to BN mappings. We here identify the following issues for this problem:

- How to identify the mapped nodes in the Bayesian network based on ontology mappings? To solve this problem, we need to examine BayesOWL’s translation rules to identify the mapped nodes.
- How to translate the probability constraints from ontology mappings to BN mappings? If the ontology mappings use the same notation of probability constraints, we could just directly use them. Or we need to re-collect the constraints and describe them in our notation.
- How to maintain the consistency of BN and the ontology? We need to ensure the mappings in Bayesian networks are reasonable under the semantic of ontologies. We will give some suggestions and guidelines to ensure this.
Chapter 5.

Research Plan

Currently we already implemented virtual evidence based soft evidential update method and can conduct inference with given linkages. Our future work focuses on mapping reduction and supporting ontology mappings. Research on the mapping reduction process may revise our current inference method, so the implementation work will be together. Ontology mapping will be used as an application of our framework.

The timeline is listed below:

Jan. 05 ~ Apr. 05       Refining the semantics of BN mapping
Jan. 05 ~ Oct. 05       Reduction of BN mappings
Apr. 05 ~ Oct. 05       Inference across Bayesian network via mappings
Nov. 05 ~ Feb. 06       Supporting ontology mappings and Experiments
Jan. 06 ~ May 06        Writing thesis
References:


[32] Xiang Y.; Poole D.; M. P. Beddoes. 1993. Multiply Sectioned Bayesian Networks and Junction Forests for Large Knowledge Based Systems,


[37] http://www.w3.org/RDF/ W3C RDF Homepage.


[40] http://www.w3.org/TR/rdf-schema/ W3C RDF Schema Specification


[42] http://www.daml.org/2001/03/daml+oil-index DAML+OIL Homepage

[43] http://www.w3.org/TR/owl-ref/ OWL Web Ontology Language Reference


