Probabilistic Inference in Influence Diagrams

Nevin L. Zhang*

Technical Report HKUST-CS96-37

*Department of Computer Science
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
Abstract

This paper studies the relationship between probabilistic inference in Bayesian networks and evaluation of influence diagrams. We clearly identify and separate from other computations probabilistic inference subproblems that must be solved in order to evaluate an influence diagram. This work leads to a new method for evaluating influence diagrams where arbitrary Bayesian network inference algorithms can be used for probabilistic inference. We argue that if the VE algorithm (Zhang and Poole 1994, Zhang and Poole 1996) is used for probabilistic inference, the new method is more efficient than the best previous methods (Shenoy 1992, Jensen et al 1994). If a more efficient probabilistic inference algorithm such as the VEC algorithm (Zhang and Poole 1996) or some approximation algorithm is used, the new method can be even more efficient.

Keyword: Decision analysis, influence diagrams, Bayesian networks, inference.
1 Introduction

Traditionally decision analysis is carried out by using decision trees (Raiffa 1968). Decision trees grow exponentially in size as the number of decisions and observations increases. A more compact framework called influence diagrams was introduced by Howard and Matheson in 1981. In addition to being more compact than decision trees, influence diagrams are also more intuitive and reveal more problem structures. They have enabled researchers to solve large decision problems that are beyond the capabilities of decision trees.

Bayesian networks (Pearl 1988) are a knowledge representation framework for reasoning under uncertainty that is widely by AI researchers. There is a rich collection of exact and approximate algorithms for inference in Bayesian networks.

An influence diagram is an acyclic graph with three types of nodes: random nodes, decision nodes, and a single value node; while a Bayesian network is an acyclic graph consisting of only random nodes. Evaluation of an influence diagram refers to the process of finding optimal decision rules for its decision nodes. Probabilistic inference in a Bayesian network refers to the process of computing the posterior probabilities of nodes of interest given observations. It is widely recognized that influence diagram evaluation and inference in Bayesian networks are closely related (Shenoy 1992, Shachter and Peot 1992, Jensen et al 1994).

This paper clarifies the relationship by clearly identify and separate from other computations probabilistic inference subproblems that must be solved in order to evaluate an influence diagram. This work leads to a new method for evaluating influence diagrams where arbitrary Bayesian network inference algorithms can be used for probabilistic inference.

Our work is built upon earlier efforts to utilize Bayesian network inference algorithms in influence diagram evaluation. Shenoy (1992) has extended the idea behind a popular Bayesian network inference algorithm called junction tree propagation* and proposed a fusion algorithm for evaluating influence diagrams. The algorithm was later refined by Jensen et al (1994) in terms of strong junction trees propagation. Shachter and Peot (1992) have shown how an influence diagram can be transformed into a Bayesian network such that optimal decision rules can be computed by posing a sequence of queries to the Bayesian network.

The queries generated by Shachter and Peot’s method can be answered by using arbitrary Bayesian network inference algorithms. This means that progresses in Bayesian network inference can be readily used to benefit influence diagram evaluation. Shenoy’s fusion algorithm and strong junction tree propagation, on the other hand, are mixtures of probabilistic inference, expected utility calculation, and optimization. New advances in Bayesian network inference cannot be directly incorporated. However, Shachter and Peot’s method is less efficient than Shenoy’s fusion algorithm and strong junction tree propagation.

The new method proposed in this paper shares the advantage of Shachter and Peot’s method. When the VE algorithm (Zhang and Poole 1994, Zhang and Poole 1996) is used for probabilistic inference, the new method is also more efficient than

---

*Also known as cluster tree propagation and clique tree propagation.
Shenoy's fusion algorithm and strong junction tree propagation. If a more efficient probabilistic inference algorithm such as the VEC algorithm (Zhang and Poole 1996) or some approximation algorithm is used, the new method can be even more efficient.

We shall begin with definitions related to influence diagrams (Section 2) and some technical preparations (Section 3). The foundation for our new method will be lead in Section 4 and details will be worked out in Section 5. The method will then be illustrated by using an example (Section 6) and compared to previous methods (Section 7). Conclusions will be presented in Section 8.

2 Influence diagrams

Influence diagrams (Howard and Matheson 1984) are acyclic graphs with three types of nodes: decision nodes, random nodes, and value nodes. The decision nodes represent decisions to make. The random nodes represent random quantities relevant to the decisions. Arcs into random nodes indicate probabilistic dependence and the dependence of a random node \( c \) upon its parents \( \pi_c \) is characterized by a conditional probability \( P(c | \pi_c) \). The value nodes represent components of the utility function. Each value node \( v \) is associated with a value function — a real valued function \( f_v(\pi_v) \) of the parents \( \pi_v \) of \( v \). The total utility function is the summation of all the value functions. Arcs into decision nodes are indicate information availability and are sometimes called informational arcs.

Influence diagrams are required to satisfy several constraints. Firstly value nodes cannot have children. Secondly influence diagrams must be regular in the sense that there must be a directed path that contains all the decision nodes. The last decision node on such a path will be referred to as the tail decision node. Thirdly they must be no-forgetting in the sense that a decision node and its parents be parents to all subsequent decision nodes. The rationale behind the no-forgetting constraint is that information available now should also be available later if the decision-maker does not forget.

Value networks refer to influence diagrams that do not contain decision nodes. Bayesian networks (Pearl 1988) are influence diagrams that consists of only random nodes.

In the following, the terms “nodes” and “variables” will be used interchangeably.

2.1 Optimal policies

We shall use \( \Omega_x \) to denote the frame of variable \( x \), i.e. the set of possible values of \( x \). For a set \( X \) of variables, \( \Omega_X \) stands for the Cartesian product \( \prod_{x \in X} \Omega_x \).

Let \( d_1, \ldots, d_k \) be all the decision nodes in an influence diagram \( \mathcal{N} \). A decision rule for a decision node \( d_i \) is a mapping \( \delta_i : \Omega_{\pi_{d_i}} \rightarrow \Omega_{d_i} \). A policy is a list of decision rules \( \Delta = (\delta_1, \ldots, \delta_k) \) consisting of one rule for each decision node.

---

1In the original definition of influence diagrams (Howard and Matheson 1984), there is only one value node. We allow multiple value nodes here so that separability in the utility function can be represented.
As a convention, we say that there is only one possible policy — the vacuous policy — for value networks.

Given a decision rule \( \delta_i \), define a conditional probability \( P_{\delta_i}(d_i|\pi_{d_i}) \) by

\[
P_{\delta_i}(d_i|\pi_{d_i}) = \begin{cases} 
1 & \text{if } \delta_i(\pi_{d_i}) = d_i \\
0 & \text{otherwise.} 
\end{cases}
\]

(1)

Since \( \Delta = (\delta_1, \ldots, \delta_k) \), we shall sometimes write \( P_{\delta_i}(d_i|\pi_{d_i}) \) as \( P_\Delta(d_i|\pi_{d_i}) \).

Given a policy \( \Delta \), the decision nodes can be viewed as random nodes. Ignoring the value nodes, one can view \( \mathcal{N} \) as a Bayesian network. Let \( C \) and \( D \) be the set of random nodes and decision nodes respectively. The Bayesian network represents the following joint probability:

\[
P_\Delta(C, D) = \prod_{c \in C} P(c|\pi_c) \prod_{d \in D} P(d|\pi_d).
\]

For any subset \( A \) of random and decision nodes, we use \( P_\Delta(A) \) to denote the marginal probability of nodes in \( A \) in the Bayesian network, i.e.

\[
P(A) = \sum_{C \cup D - A} P(C, D),
\]

where the summation is taken all possible values of nodes in \( C \cup D - A \).

Value nodes do not have children. Hence the parents of a value node \( v \) must be either random nodes or decision nodes. The expectation \( E_\Delta[v] \) of the value function \( f_v(\pi_v) \) given policy \( \Delta \) is given by

\[
E_\Delta[v] = \sum_{\pi_v} P(\pi_v) f_v(\pi_v),
\]

(2)

where the summation is taken over all possible values of nodes in \( \pi_v \). The expected value \( E_\Delta[\mathcal{N}] \) of the influence diagram \( \mathcal{N} \) given policy \( \Delta \) is defined by

\[
E_\Delta[\mathcal{N}] = \sum_{v \in V} E_\Delta[v],
\]

(3)

where \( V \) is the set of all value nodes in \( \mathcal{N} \).

Since a value network has only one policy, we shall refer to its expected value given that policy simply as the expected value of the value network.

The optimal expected value \( E[\mathcal{N}] \) of an influence diagram \( \mathcal{N} \) is defined by

\[
E[\mathcal{N}] = \max_\Delta E_\Delta[\mathcal{N}],
\]

(4)

where maximization is taken over all possible policies. An optimal policy \( \Delta^* \) is one that satisfies

\[
E_{\Delta^*}[\mathcal{N}] = E[\mathcal{N}].
\]

(5)

Decision rules in an optimal policy are said to be optimal decision rules. In other words, a decision rule \( \delta_i^* \) for the decision node \( d_i \) is optimal if there exist decision rules \( \delta_j^* \) for all other decision nodes \( d_j \) such that the policy \( (\delta_1^*, \ldots, \delta_i^*, \ldots, \delta_k^*) \) is optimal.

In this paper, we shall only consider variables with finite frames. Hence there are only a finite number of possible policies. Consequently, there always exists at least one optimal policy. To evaluate an influence diagram is to (1) find an optimal policy, and (2) compute the optimal expected value.
3 Technical preparations

This section introduces a couple of technical concepts that will ease later exposition.

3.1 Quasi-Bayesian networks

A conditional probability $P(c|A)$ of a random variable $c$ given a set $A$ of other variables sums to one in the sense that $\sum_c P(c|A) = 1$. In contrast, we define a *conditional potential* $P(c|A)$ of $c$ given $A$ to be simply a non-negative function of $c$ and variables in $A$. It might not sum to one.

In a Bayesian network, each node $c$ is associated with a conditional probability $P(c|\pi_c)$ of the node given its parents $\pi_c$. *Quasi-Bayesian networks* are a generalization of Bayesian networks where some nodes are associated with conditional potentials instead of conditional probabilities. Such nodes will be referred to as *quasi-random nodes*. While Bayesian networks consist of only random nodes, quasi-Bayesian networks contain random nodes as well as quasi-random nodes.

Let $\mathcal{N}$ be a quasi-Bayesian network and let $X = \{x_1, \ldots, x_n\}$ be all its nodes. The *joint potential* $P_\mathcal{N}(X)$ of $\mathcal{N}$ is given by

$$P_\mathcal{N}(X) = \prod_{i=1}^{n} P(x_i|\pi_{x_i}).$$

For any subset $A$ of nodes, the *marginal potential* $P_\mathcal{N}(A)$ of nodes in $A$ is given by

$$P_\mathcal{N}(A) = \sum_{X \setminus A} P_\mathcal{N}(X),$$

where the summation is taken over all possible values of nodes in $X \setminus A$. For any two subsets $A$ and $B$ of nodes, the conditional probability $P_\mathcal{N}(A|B)$ is defined by

$$P_\mathcal{N}(A|B) = \frac{P_\mathcal{N}(A, B)}{P_\mathcal{N}(B)}.$$

In a quasi-Bayesian network, a *barren* node is a random node that has no children. A quasi-random can never be a barren node. The following proposition will be used many times later.

**Proposition 1** Let $\mathcal{N}$ be a quasi-Bayesian network.

1. Let $\mathcal{N}'$ be the quasi-Bayesian network obtained from $\mathcal{N}$ by pruning a barren node. For any subset $A$ of nodes in $\mathcal{N}'$,

$$P_\mathcal{N}(A) = P_{\mathcal{N}'}(A).$$

2. For any set $B$ of nodes in $\mathcal{N}$, let $\mathcal{N}''$ be the quasi-Bayesian network obtained from $\mathcal{N}$ by pruning nodes that are neither in $B$ nor ancestors of nodes in $B$. If the nodes pruned are all random nodes, then

$$P_\mathcal{N}(B) = P_{\mathcal{N}''}(B).$$
Proof: Let \( x_1, \ldots, x_n \) be all the nodes in \( \mathcal{N} \). If \( x_n \) is a barren node, then

\[
P_{\mathcal{N}}(x_1, \ldots, x_{n-1}) = \sum_{x_n} P_{\mathcal{N}}(x_1, \ldots, x_n) = \sum_{x_n} \prod_{i=1}^{n} P(x_i | \pi_{x_i}) = \prod_{i=1}^{n-1} P(x_i | \pi_{x_i}) \left[ \sum_{x_n} P(x_n | \pi_{x_n}) \right] = \prod_{i=1}^{n-1} P(x_i | \pi_{x_i}).
\]

The first item follows. The second item can be obtained by repeatedly applying item 1. \( \square \)

### 3.2 Shachter and Peot’s transformation

The idea behind Shachter and Peot’s transformation can be used to rewrite equation (2) in a more convenient form. Without losing generality, assume all the value functions be non-negative. If a value function takes negative values, a constant can be added to it so that it takes only non-negative values. Addition of constants to value functions does not change the optimal policies. Moreover, the optimal expected value of an influence diagram equals to its optimal expected value after the addition of the constants minus the summation of those constants.

For any value node \( v \) in an influence diagram \( \mathcal{N} \), transform it into a binary random node with conditional probability given by

\[
P(v=1|\pi_v) = \frac{f_v(\pi_v)}{M_v}, \tag{6}
\]

where \( M_v = \max_{\pi} f_v(\pi_v) \).

After all value nodes have been transformed into random nodes, an influence diagram \( \mathcal{N} \) can be viewed as a Bayesian network given a policy \( \Delta \). Denote this Bayesian network by \( \mathcal{N}_\Delta \).

**Proposition 2** *Equation (2) can be rewritten as*

\[
E_\Delta[v] = P_{\mathcal{N}_\Delta}(v=1) M_v. \tag{7}
\]

**Proof:** Let \( \mathcal{N}^{v}_\Delta \) be the Bayesian network obtained from \( \mathcal{N}_\Delta \) by pruning all the value nodes except for \( v \). By Proposition 1,

\[
P_{\mathcal{N}_\Delta}(v) = P_{\mathcal{N}^{v}_\Delta}(v).
\]

Hence

\[
P_{\mathcal{N}_\Delta}(v=1) M_v = P_{\mathcal{N}^{v}_\Delta}(v=1) M_v
\]


Figure 1: Partition of the set of nodes in an influence diagram w.r.t a decision node.

\[= \sum_{C \cup D, \cdot \in C} \prod_{e \in C} P(c|\pi_e) \prod_{d \in D} P_{\Delta}(d|\pi_d) P(v=1|\pi_v) M_v\]

\[= \sum_{\pi_v} \sum_{C \cup D - \pi_v} P_{\Delta}(C, D) f_v(\pi_v)\]

\[= E_\Delta[v], \square\]

4 The decomposition theorem

Suppose \(\mathcal{N}\) is an influence diagram and \(d\) is the tail decision node in \(\mathcal{N}\). This section shows that \(\mathcal{N}\) can be decomposed into two components, called tail and body respectively, such that an optimal decision rule for \(d\) can be found in the tail and optimal decision rules for all other decision nodes can be found in the body. The body is again an influence diagram and hence the decomposition can be repeated on the body.

We begin by partitioning the set of nodes in \(\mathcal{N}\) into several subsets w.r.t to a decision node.

4.1 Downstream and upstream sets

The moral graph (Lauritzen and Spiegelhalter 1988) of an influence diagram is obtained by first adding undirected edges between pairs of parents of each node so that they are pairwise connected and then dropping directions of all the directed edges. Two nodes \(x\) and \(y\) in an influence diagram are \(m\)-separated by a set of nodes \(S\) if in the moral graph every path connecting them contains at least one node in \(S\). Note that for any set \(S\) of nodes, it \(m\)-separates each of its members from any nodes outside \(S\).

Let \(d\) be a decision node in an influence diagram \(\mathcal{N}\). The upstream set of \(\mathcal{N}\) w.r.t to \(d\), denoted by \(X_1\), is the set of nodes that are \(m\)-separated from \(d\) by \(\pi_d\), with nodes in \(\pi_d\) excluded. The downstream set of \(\mathcal{N}\) w.r.t \(d\), denoted by \(X_2\), is the set of nodes that are not \(m\)-separated from \(d\) by \(\pi_d\). Note that \(d \in X_2\) and that the three sets \(X_1\), \(\pi_d\) and \(X_2\) constitute a partition of the set of all the nodes in the influence diagram.

Define \(\pi_{d,2}\) be the set of nodes in \(\pi_d\) that have at least one parent in \(X_2\) and set \(\pi_{d,1} = \pi_d - \pi_{d,2}\). As shown in Figure 1, the four sets \(X_1\), \(X_2\), \(\pi_{d,1}\), and \(\pi_{d,2}\) constitute a partition of the set of all nodes in \(\mathcal{N}\).
4.2 Tails

Let $d$ be the tail decision node in an influence diagram $\mathcal{N}$. It is evident that $\pi_{d,2}$ consists of only random nodes.

Denoted by $\mathcal{T}$, the tail of $\mathcal{N}$ w.r.t $d$ consists of all the nodes in $X_2 \cup \pi_d$. Connections among the nodes are the same as in $\mathcal{N}$ except that arcs into $d$ and arcs into nodes in $\pi_{d,1}$ are removed. Conditional probabilities of random nodes and value functions of value nodes remain the same as in $\mathcal{N}$ except that conditional probabilities of nodes in $\pi_{d,1}$ are removed. The nodes in $\pi_{d,1}$ and $d$ are regarded as quasi-random nodes and their associated potentials are defined to be the constant 1.

The tail of the influence diagram in Figure 2 w.r.t $d_2$ is shown in Figure 3 (2). The dashed nodes are quasi-random nodes.

Since $d$ is the tail decision node, $X_2$ does not contain any other decision nodes in addition to $d$. Moreover, there are no decision nodes in $\pi_{d,2}$. Decision nodes in $\pi_{d,1}$, if exist, and $d$ are regarded as quasi-random nodes in $\mathcal{T}$. Moreover after Shachter and Poet’s transformation, value nodes are viewed as random nodes. Thus $\mathcal{T}$ consists of random nodes and quasi-random nodes and hence is a quasi-Bayesian network.

Let $V_2$ be the set of value nodes in $\mathcal{T}$. The evaluation functional $e_\mathcal{T}(\pi_d, d)$ of the tail $\mathcal{T}$ is defined by

$$e_\mathcal{T}(\pi_d, d) = \sum_{v \in V_2} P_\mathcal{T}(v = 1 | \pi_d, d) M_v. \quad (8)$$

The following Proposition will be used in the proof of the decomposition theorem.

**Proposition 3** Let $\Delta$ be a policy for an influence diagram $\mathcal{N}$ and $\delta$ be the decision rule for the tail decision $d$ in $\Delta$. For any value node $v$ in the tail $\mathcal{T}$ of $\mathcal{N}$ w.r.t $d$,

$$P_{\mathcal{N}_\Delta}(v | \pi_d) = P_\mathcal{T}(v | \pi_d, d=\delta(\pi_d)).$$

**Proof:** Consider the following marginal probability in the Bayesian network $\mathcal{N}_\Delta$:

$$P_{\mathcal{N}_\Delta}(v, \pi_d) = \sum_{X_1, X_2 - \{v\}} P_{\mathcal{N}_\Delta}(X_1, \pi_d, X_2)$$

$$= \sum_{X_1, X_2 - \{v\}} \prod_{x \in X_1 \cup \pi_{d,1}} P_\Delta(x | \pi_x) \prod_{x \in X_2 \cup \pi_{d,2}} P_\Delta(x | \pi_x),$$

Figure 2: An influence diagram. Random nodes are drawn as ellipses, decision nodes as rectangles, and value nodes as diamonds.
\[ P_{\Delta}(x|\pi_x) \] stands simply for \( P(x|\pi_x) \) when \( x \) is a random or value node. Since \( \pi_x \subseteq X_1 \cup \pi_d \) for any \( x \in X_1 \cup \pi_{d,1} \) and \( \pi_x \subseteq X_2 \cup \pi_d \) for any \( x \in X_2 \cup \pi_{d,2} \),

\[
P_{X_\Delta}(v, \pi_d) = \left[ \sum_{X_1} \prod_{x \in X_1 \cup \pi_{d,1}} P_{X_\Delta}(x|\pi_x) \right] \left[ \sum_{X_2 - \{v\}} \prod_{x \in X_2 \cup \pi_{d,2}} P_{\Delta}(x|\pi_x) \right]
\]

\[
= \left[ \sum_{X_1} \prod_{x \in X_1 \cup \pi_{d,1}} P_{\Delta}(x|\pi_x) \right] \left[ \sum_{d} P_{\Delta}(d|\pi_d) \prod_{x \in X_2 - \{v, d\}} P(x|\pi_x) \right]
\]

\[
= \left[ \sum_{X_1} \prod_{x \in X_1 \cup \pi_{d,1}} P_{\Delta}(x|\pi_x) \right] \left[ P_{\Delta}(d|\pi_d) \prod_{x \in X_2 - \{d\}} P(x|\pi_x) \right]
\]

\[
= \left[ \sum_{X_1} \prod_{x \in X_1 \cup \pi_{d,1}} P_{\Delta}(x|\pi_x) \right] \left[ P_T(v, \pi_d, d=\delta(\pi_d)) \right]
\]

The last equation is true because \( P_{\Delta}(d|\pi_d) \) is 1 when \( d=\delta(\pi_d) \) and 0 otherwise. The proposition follows from the fact that \( \sum_{X_1} \prod_{x \in X_1 \cup \pi_{d,1}} P_{\Delta}(x|\pi_x) \) depends only on variables in \( \pi_d \).

\[ \square \]

### 4.3 Bodies

Let \( d \) be the tail decision node in an influence diagram \( \mathcal{N} \). Denoted by \( \mathcal{B} \), the body of \( \mathcal{N} \) w.r.t \( d \) is an influence diagram consisting of nodes in the \( X_1 \cup \pi_d \), nodes in the \( X_2 \) that are ancestors of nodes in \( \pi_d \), and a new value node \( u \). Connections among the old nodes, conditional probabilities of the random nodes, and value functions of the old value nodes remain the same as in \( \mathcal{N} \). There is an arc from each node in \( \pi_d \) to the new value node \( u \) and the value function of \( u \) is given by

\[
f_u(\pi_d) = \max_{d \in T(\pi_d, d)}
\]

The body of the influence diagram in Figure 2 w.r.t \( d_2 \) is shown in Figure 3 (1).

From now on we assume that Shachter and Peot’s transformation has been performed on the new value node \( u \) and hence we can view it as a random node.

The decision node \( d \) is in \( X_2 \) and it cannot be an ancestor of any node in \( \pi_d \). Hence \( \mathcal{B} \) does not contain \( d \). Let \( \Delta_1 \) be a policy for \( \mathcal{B} \). For any decision rule \( \delta \) for \( d \), \( \Delta_1 \) and \( \delta \) constitute a policy for \( \mathcal{N} \). Denote this policy by \( (\Delta_1, \delta) \).
Given $\Delta_1$, $\mathcal{B}$ can be viewed as a Bayesian network. Denote this Bayesian network by $\mathcal{B}_{\Delta_1}$. Let $A$ be the set of all the nodes in $\mathcal{B}$ except for $u$. By Proposition 1,
\[ P_{\mathcal{N}(\Delta_1,\pi)}(A) = P_{\mathcal{B}_{\Delta_1}}(A). \]
Consequently, we have

**Proposition 4** Given any policy $\Delta_1$ for $\mathcal{B}$ and any decision rule $\delta$ for $d$,
\[ P_{\mathcal{N}(\Delta_1,\pi)}(\pi_d) = P_{\mathcal{B}_{\Delta_1}}(\pi_d). \]
And for any decision node $v \in X_1$,
\[ P_{\mathcal{N}(\Delta_1,\pi)}(v) = P_{\mathcal{B}_{\Delta_1}}(v). \]

4.4 The decomposition theorem

**Theorem 1** (The Decomposition Theorem) Let $d$ be the tail decision node in an influence diagram $\mathcal{N}$.

1. An optimal decision rule $\delta^*$ for $d$ is given by
\[ \delta^*(\pi_d) = \arg \max_{\delta} \epsilon_T(\pi_d, d). \]
where $\epsilon_T(\pi_d, d)$ is the evaluation functional of the tail $T$ of $\mathcal{N}$ w.r.t $d$.

2. For any decision node $d' \neq d$, a decision rule for $d$ is optimal in $\mathcal{N}$ if and only if it is optimal in body $\mathcal{B}$ of $\mathcal{N}$ w.r.t $d$.

3. The optimal expected value of the body $\mathcal{B}$ is the same as that of $\mathcal{N}$.

**Proof:** Let $V_1$ and $V_2$ be the sets of value nodes in $X_1$ and $X_2$ respectively. For any policy $\Delta_1$ for $\mathcal{B}$ and any decision rule $\delta$ for $d$,
\[ E_{\Delta_1,\delta}[\mathcal{N}] = \sum_{v \in V_1} P_{\mathcal{N}(\Delta_1,\pi)}(v=1) M_v + \sum_{v \in V_2} P_{\mathcal{N}(\Delta_1,\pi)}(v=1) M_v \]
\[ = \sum_{v \in V_1} P_{\mathcal{N}(\Delta_1,\pi)}(v=1) M_v + \sum_{v \in V_1} \sum_{\pi_d} P_{\mathcal{N}(\Delta_1,\pi)}(\pi_d) P_{\mathcal{N}(\Delta_1,\delta)}(v=1|\pi_d) M_v \]
By Propositions 3 and 4,
\[ E_{\Delta_1,\delta}[\mathcal{N}] = \sum_{v \in V_1} P_{\mathcal{B}_{\Delta_1}}(v=1) M_v + \sum_{\pi_d} P_{\mathcal{B}_{\Delta_1}}(\pi_d) \sum_{v \in V_2} P_T(v=1|\pi_d, d=\delta(\pi_d)) M_v \]
\[ = \sum_{v \in V_1} P_{\mathcal{B}_{\Delta_1}}(v=1) M_v + \sum_{\pi_d} P_{\mathcal{B}_{\Delta_1}}(\pi_d) \epsilon_T(\pi_d, d=\delta(\pi_d)). \]
Since $\sum_{v \in V_1} P_{\mathcal{B}_{\Delta_1}}(v=1) M_v$ and $P_{\mathcal{B}_{\Delta_1}}(\pi_d)$ do not depend on the decision rule $\delta$ for $d$, the first item follows.

Let $\delta^*$ be an optimal decision rule for $d$. It follows from equation (11) that
\[ E_{\Delta_1,\delta^*}[\mathcal{N}] = E_{\Delta_1}[\mathcal{B}]. \]
Then second and third items follow. □
5 Evaluating influence diagrams

The decomposition theorem gives us the following procedure for evaluating an influence diagram: (1) decompose it to two components — tail and body — w.r.t the tail decision node, (2) find an optimal decision rule for the tail decision node in the tail, and (3) repeat (1) and (2) on the body. This section looks at the necessary computations in more detail. We discuss several optimizations. We also clearly separate probabilistic inference tasks from other computations so that arbitrary Bayesian network inference algorithms can be used.

5.1 Irrelevant parents of decision nodes and reduced tails

Let \( d \) be the tail decision node in an influence diagram \( \mathcal{N} \). Let \( \pi_d^i \) be the set of nodes in \( \pi_{d,1} \) that are not parents to any nodes in the set \( (X_2 - \{d\}) \cup \pi_{d,2} \) and let \( \pi_d^r = \pi_d - \pi_d^r \). In the tail \( \mathcal{T} \) of \( \mathcal{N} \) w.r.t \( d \), there are no arcs into nodes in \( \pi_{d,1} \). Hence there are no arcs into nodes in \( \pi_d^i \). By definition, there are no arcs emitting from nodes in \( \pi_d^i \) either. Thus each node in \( \pi_d^i \) is isolated from all other nodes in \( \mathcal{T} \). For example, the node \( c_3 \) in Figure 3 (2) is isolated from all other nodes. The reduced tail \( \mathcal{T}^r \) of \( \mathcal{N} \) w.r.t \( d \) is obtained from \( \mathcal{T} \) by pruning nodes in \( \pi_d^i \).

In the tail \( \mathcal{T} \), nodes in \( \pi_d^i \) are quasi-random nodes whose associated potentials are the constant 1. Since they are also isolated nodes, we have

\[
P_T (\pi_d, X_2) = P_T (\pi_d^i, \pi_d^r, X_2) = P_{T^r} (\pi_d^r, X_2).
\]

Hence \( P_T (\pi_d, d) = P_{T^r} (\pi_d^r, d) \) and \( P_T (v=1, \pi_d, d) = P_{T^r} (v=1, \pi_d^r, d) \) for any value node \( v \in V_2 \). Now consider the evaluation functional \( e_T (\pi_d, s) \). We have

\[
e_T (\pi_d, d) = \frac{\sum_{v \in V_2} P_T (v=1, \pi_d, d) M_v}{P_T (\pi_d, d)}
= \frac{\sum_{v \in V_2} P_{T^r} (v=1, \pi_d^r, d) M_v}{P_{T^r} (\pi_d^r, d)}
\]

Thus \( e_T (\pi_d, d) \) does not depend on the nodes in \( \pi_d^i \) and can be written as \( e_{T^r} (\pi_d^r, d) \).

This fact has two implications. Firstly the optimal decision rule for \( d \) given by equation (10) does not depend on nodes in \( \pi_d^i \). For this reason, nodes in \( \pi_d^i \) will be called irrelevant parents of \( d \) and the nodes in \( \pi_d^r \) will be called relevant parents of \( d \).

Secondly the value function of the new value node \( u \) in the body \( \mathcal{B} \) of \( \mathcal{N} \) w.r.t \( d \) does not depend on the irrelevant parents of \( d \). Consequently there is no need to draw arcs from those nodes to \( u \). From now on we will assume that, in \( \mathcal{B} \), arcs are drawn only from the relevant parents \( \pi_d^r \) of \( d \) to \( u \) and that the value function of \( u \) is given by

\[
f_u (\pi_d^r) = \max_{\pi_d^r} e_{T^r} (\pi_d^r, d).
\]

To avoid excessive notations, we will use \( \mathcal{T} \) to denote the reduced tail \( \mathcal{T}^r \) from now on.
5.2 Efficient computation of evaluation functionals

To obtain the evaluation functional $e_T(\pi^r_d, d)$ of the reduced tail $T$ of an influence diagram w.r.t its tail decision node $d$, one needs to compute $P_T(\pi^r_d, d)$ and $P_T(v=1, \pi^r_d, d)$ for each value node $v$ in $T$. Those marginal potentials can be computed directly from $T$. This subsection shows that we can do better.

**Proposition 5** Let $T_e$ be the quasi-Bayesian network obtained from $T$ by pruning nodes that are neither in $\pi^r_d$ nor ancestors of nodes in $\pi^r_d$. Then

$$P_T(\pi^r_d, d) = P_{T_e}(\pi^r_d).$$

**Proof:** Let $T'$ be the quasi-Bayesian network obtained from from $T$ by pruning all the nodes except for those in $\pi^r_d \cup \{d\}$ and those that are ancestors of nodes in $\pi^r_d$. By Proposition 1, we have

$$P_T(\pi^r_d, d) = P_{T'}(\pi^r_d, d).$$

It is evident that $d$ is an isolated node in $T'$. Since $d$ is a quasi-random node whose associated potential is the constant 1,

$$P_{T'}(\pi^r_d, d) = P_{T'}(\pi^r_d).$$

Since $T_e$ is same as $T'$ except it does not contain the node $d$,

$$P_{T'}(\pi^r_d) = P_{T_e}(\pi^r_d).$$

The proposition follows. $\square$

Now consider computing $P_T(v=1, \pi^r_d, d)M_v$ for each value node $v$ in $T$. Let $T'$ be the quasi-Bayesian network obtained from $T$ by pruning nodes that are neither in $\pi^r_d \cup \{d, v\}$ nor ancestors of nodes in $\pi^r_d \cup \{v\}$. By Proposition 1,

$$P_T(v=1, \pi^r_d, d) = P_{T'}(v=1, \pi^r_d, d).$$

The node $v$ is the only value node in $T'$. It has been converted into a random node by Shachter and Peot's transformation. The conditional probability $P(v=1|\pi_v)$ was obtained by dividing the value function $f_v(\pi_v)$ with the constant $M_v$ (equation 6). The same constant is later multiplied with $P_T(v=1, \pi^r_d, d)$. Apparently, there are wasteful computations. Those wastes can be avoided by using the following proposition.

**Proposition 6** Let $T_v$ be the quasi-Bayesian network obtained from $T$ by pruning nodes that are neither in $\pi^r_d \cup \{d, v\}$ nor ancestors of nodes in $\pi^r_d \cup \{v\}$ and by converting $v$ into a quasi-random node with conditional potential

$$P(v=1|\pi_v) = f_v(\pi_v).$$

Then

$$P_T(v=1, \pi^r_d, d)M_v = P_{T_v}(v=1, \pi^r_d, d).$$

$\Box$
The above two propositions give us the following efficient procedure for computing the evaluation functional.

Procedure \( \text{EF}(\mathcal{T}, d) \)

1. Construct the quasi-Bayesian network \( \mathcal{T}_e \) and compute \( P_{\mathcal{T}_e}(\pi_d^r) \).
2. For each value node \( v \) in \( \mathcal{T} \), construct the quasi-Bayesian network \( \mathcal{T}_v \) and compute \( P_{\mathcal{T}_v}(v=1, \pi_d^r, d) \).
3. Set

\[
e_{\mathcal{T}}(\pi_d^r, d) = \frac{\sum_{v \in V_2} P_{\mathcal{T}_v}(v=1, \pi_d^r, d)}{P_{\mathcal{T}_d}(\pi_d^r)},
\]

where \( V_2 \) is the set of value nodes in \( \mathcal{T} \).
4. Return \( (e_{\mathcal{T}}(\pi_d^r, d), P_{\mathcal{T}_d}(\pi_d^r)) \).

Note that both \( P_{\mathcal{T}_v}(v=1, \pi_d^r, d) \) and \( P_{\mathcal{T}_d}(\pi_d^r, d) \) can be computed using arbitrary Bayesian network inference algorithms. Also note that in addition to the evaluation functional, EF also returns the marginal potential \( P_{\mathcal{T}_d}(\pi_d^r) \). It will be used in the next subsection.

### 5.3 Reduced bodies

Let \( d \) be the tail decision node in an influence diagram \( \mathcal{N} \). When some nodes in \( \pi_d \) have ancestors in the downstream set \( X_2 \) of \( \mathcal{N} \) w.r.t \( d \), the body \( \mathcal{B} \) of \( \mathcal{N} \) w.r.t \( d \) contains nodes in \( X_2 \). As shown in Figure 3 (1), the body of the influence diagram shown in Figure 2 w.r.t \( d_2 \) contains the node \( a_0 \), which is in the downstream set \( X_2 \). This subsection shows that by using \( P_{\mathcal{T}_d}(\pi_d^r) \), one can reduce \( \mathcal{B} \) such that it does not contain nodes in \( X_2 \) and that the reduction takes very little extra numerical computation.

Enumerate all node in \( \pi_{d,2} \) as \( c_1, \ldots, c_k \) such that \( c_i \) is ancestor of \( c_j \) in \( \mathcal{N} \) only if \( i < j \). Let \( \pi_{d,1}^r = \pi_d^r \cap \pi_{d,1} \) and let \( Z_i \) be the set of nodes in \( \pi_{d,1}^r \) that are ancestors of \( c_i \) in the reduced tail \( \mathcal{T} \) of \( \mathcal{N} \) w.r.t \( d \).

The reduced body of \( \mathcal{N} \) w.r.t \( d \) is an influence diagram consisting of nodes in the upstream set \( X_1 \) of \( \mathcal{N} \) w.r.t \( d \), nodes in \( \pi_d \), and a new value node \( u \). Connections among nodes in \( X_1 \cup \pi_{d,1} \) remain the same as in \( \mathcal{N} \), so do their conditional probabilities or value functions. There is an arc from each node in \( \pi_d^r \) to \( u \) and the value function of \( u \) is given by equation (12). For each node \( c_i \) in \( \pi_{d,2} \), there is an arc from each node in \( \{c_1, \ldots, c_{i-1}\} \cup Z_i \cup \ldots Z_i \) to \( c_i \). By the properties of the \( c_i \)'s, the introduction of these arcs does not create directed cycles. The conditional probability of \( c_i \) is given by

\[
P(c_i|c_1, \ldots, c_{i-1}, Z_i \cup \ldots \cup Z_i) = \frac{\sum_{c_{i+1}, \ldots, c_k} P_{\mathcal{T}_d}(\pi_d^r)}{\sum_{c_{i+1}, \ldots, c_k} P_{\mathcal{T}_d}(\pi_d^r)}. \quad (13)
\]

For the conditional probability to be well defined, we must have

**Lemma 1** The term on the right hand side of equation (13) depends only one nodes in \( \{c_1, \ldots, c_i\} \cup Z_i \cup \ldots \cup Z_i \).
**Proof:** Consider $\sum_{c_{i+1}, \ldots, c_k} P_{T_{\pi}^r}(\pi_d^r) = \sum_{c_{i+1}, \ldots, c_k} P_{T_r}(c_1, \ldots, c_k, \pi_{d,1}) = P_{T_r}(c_1, \ldots, c_k, \pi_{d,1})$. Let $T_{\pi}$ be the quasi-Bayesian network obtained from $T_c$ by pruning nodes that are neither in $\{c_1, \ldots, c_i\} \cup \pi_{d,1}$ nor ancestors of nodes in the set. By Proposition 1,

$$P_{T_r}(c_1, \ldots, c_i, \pi_{d,1}) = P_{T_{\pi}}(c_1, \ldots, c_i, \pi_{d,1}).$$

Each node in $\pi_{d,1} - Z_1 \cup \ldots \cup Z_i$ is isolated from all other nodes in $T_{\pi}$ and is a quasi-random nodes whose associated potential is the constant 1. Hence,

$$P_{T_{\pi}}(c_1, \ldots, c_i, \pi_{d,1}) = P_{T_r}(c_1, \ldots, c_i, Z_1 \cup \ldots \cup Z_i).$$

The lemma follows. $\square$

The reduced body of the influence diagram in Figure 2 w.r.t $d_2$ is same as its body w.r.t $d_2$ (Figure 3 (1)) except that the node $c_0$ does not appear in the reduced body. Here, $P_{T_r}(\pi_{d,1}) = \sum c_6 P(c_4|d_1, c_6) P(c_6)$. Hence the conditional probability of $c_4$ in the reduced body is given by $P(c_4|d_1) = \sum c_6 P(c_4|d_1, c_6) P(c_6)$.

**Lemma 2** Let $B$ and $B^r$ be the body and reduced body of $N$ w.r.t its tail decision node $d$ respectively. Then for any policy $\Delta_1$ for $B$ (or equivalently for $B^r$),

$$E_{\Delta_1}[B] = E_{\Delta_1}[B^r].$$

**Proof:** Given a policy $\Delta_1$, both $B$ and $B^r$ can be regarded as Bayesian networks. Denote these Bayesian networks by $B_{\Delta_1}$ and $B_{\Delta_1}^r$. By following a similar line of reasoning as in the proof of Proposition 3, we get

$$P_{B_{\Delta_1}}(\pi_d) = \sum [ \prod_{x \in X_1 \cup \pi_{d,1}} P_{\Delta_1}(x|\pi_x) ] P_T(\pi_d)$$

$$= \sum [ \prod_{x \in X_1 \cup \pi_{d,1}} P_{\Delta_1}(x|\pi_x) ] P_{T_r}(\pi_{d,1})$$

$$= \sum [ \prod_{x \in X_1 \cup \pi_{d,1}} P_{\Delta_1}(x|\pi_x) ] \prod_{i=1}^{k} \frac{\sum_{c_{i+1}, \ldots, c_k} P_{T_r}(\pi_d^r)}{\sum_{c_{i+1}, \ldots, c_k} P_{T_r}(\pi_{d,1})} \quad (14)$$

$$= P_{B_{\Delta_1}}(\pi_d),$$

where equation (14) is true because

$$\sum_{c_{i+1}, \ldots, c_k} P_{T_r}(\pi_d^r) = P_{T_r}(\pi_{d,1}) = 1.$$

It is evident that for any value node $v$,

$$P_{B_{\Delta_1}}(v|\pi_d) = P_{B_{\Delta_1}}(v|\pi_d).$$

The lemma follows. $\square$

**Corollary 1** Theorem 1 remains true if the body $B$ of $N$ w.r.t $d$ is replaced by the reduced body $B^r$ of $N$ w.r.t $d$. $\square$
The advantage of the reduced body \( \mathcal{B}^r \) over the body \( \mathcal{B} \) is that it does not contain nodes in the downstream set \( X_2 \). Since one has to compute \( P_{T_d}(\pi_d^r) \) when calculating the evaluation functional of the tail, the construction of \( \mathcal{B}^r \) requires very little extra numerical computation.

From now on, we shall use \( \mathcal{B} \) to denote the reduced body of an influence diagram w.r.t its tail decision node.

### 5.4 Expected values of value networks

Let \( \mathcal{N} \) be a value network. For each value node \( v \), let \( \mathcal{N}_v \) be the quasi-Bayesian network obtained from \( \mathcal{N} \) by pruning nodes that are not ancestors of \( v \) and by converting \( v \) into a quasi-random node in the same way as we did in Proposition 6. The following procedure computes the expected value of \( \mathcal{N} \).

Procedure \( \text{EV}(\mathcal{N}) \)

1. For each value node \( v \), obtain \( \mathcal{N}_v \) from \( \mathcal{N} \) and compute \( P_{\mathcal{N}_v}(v = 1) \).
2. (Let \( V \) be the set of value nodes in \( \mathcal{N} \).) Return \( \sum_{v \in V} P_{\mathcal{N}_v}(v = 1) \).

### 5.5 An algorithm

We are now ready to give an algorithm for evaluating influence diagrams.

Procedure \( \text{EVALUATE}(\mathcal{N}) \):

- **Input:** \( \mathcal{N} \) — an influence diagram.
- **Outputs:** an optimal policy and the optimal expected value.

1. **While** there are decision nodes in \( \mathcal{N} 

   (a) Construct the reduced tail \( \mathcal{T} \) of \( \mathcal{N} \) w.r.t the tail decision node \( d \).

   (b) Call \( \text{EF}(\mathcal{T}, d) \) to compute \( e_{\mathcal{T}}(\pi_d^r, d) \) and \( P_{\mathcal{T}}(\pi_d^r) \).

   (c) Find an optimal decision rule for \( d \) by using

   \[
   \delta^*(\pi_d^r) = \arg \max_{\pi_d^r} e_{\mathcal{T}}(\pi_d^r, d).
   \]

   (d) Construct the reduced body \( \mathcal{B} \) of \( \mathcal{N} \) w.r.t \( d \) and set \( \mathcal{N} = \mathcal{B} \).

   (max_{\pi_d^r} e_{\mathcal{T}}(\pi_d^r, d) \text{ and } P_{\mathcal{T}}(\pi_d^r) \text{ used here.})

**Endwhile** (After the while-loop, \( \mathcal{N} \) becomes a value network.)

2. Return the optimal decision rules and \( \text{EV}(\mathcal{N}) \).

Note that we do not specify how the probabilistic calculations in \( \text{EF} \) and \( \text{EV} \) should be carried out. Arbitrary Bayesian network inference algorithms can be used.
6 An example

This subsection illustrates EVALUATE by using the influence diagram shown in Figure 4. It is borrowed from Jensen et al (1994). Arcs into decision nodes are dashed for readability.

6.1 Probabilistic inference

We assume that the VE (variable elimination) algorithm (Zhang and Poole 1994, Zhang and Poole 1996) is used for probabilistic inference. In order for VE to compute \( P_{T_A}(\pi^*_v) \) for EF, for instance, an elimination ordering \( \rho \) consisting of nodes outside \( \pi^*_v \) needs to be provided as input. In this section, elimination orderings are assumed to be generated by the minimum deficiency heuristic (Bertelé and Brioscì 1972).

Given an elimination ordering \( \rho \), VE computes \( P_{T_A}(\pi^*_v) \) as follows:

1. Let \( \mathcal{P} \) be the list of probabilities of random nodes and potentials of quasi-random nodes in \( T_a \). (Members of \( \mathcal{P} \) are regarded simply as functions of variables and are called factors.)

2. Eliminate variables one by one according to the ordering \( \rho \). To eliminate a variable \( z \) is to remove from \( \mathcal{P} \) all the factors \( f_1, \ldots, f_k \) that involve \( z \). And if such factors exist, add the new factor \( \sum_z \Pi_{i=1}^k f_i \) to the list \( \mathcal{P} \).

3. After all variables in \( \rho \) have been eliminated, multiply all factors in \( \mathcal{P} \). The result is \( P_{T_A}(\pi^*_v) \).

VE computes \( P_{T_A}(v=1, \pi^*_v, d) \) for EF and \( P_{\gamma}(v=1) \) for EV in the same way. Since the variable \( v \) is fixed to be 1, VE does not need \( P(v=0|\pi_v) \). The list \( \mathcal{P} \) initially

\[ \text{Figure 4: An influence diagram.} \]
contains, for the node $v$, $P(v=1|\pi_v)$ or equivalently the value function $f_v(\pi_v)$ and $v$
does not appear in the elimination ordering.

6.2 Optimal decision rule for $d_4$

Denote the influence digram in Figure 4 by $\mathcal{N}$. The tail decision node in $\mathcal{N}$ is $d_4$.
Step 1(a) constructs the reduced tail $\mathcal{T}$ of $\mathcal{N}$ w.r.t $d_4$, which is shown in Figure 5 (2). From the tail, EVALUATE finds that $\pi_{d_4}^*=\{c_{10},d_2\}$. All other parents of $d_4$ are irrelevant.

EF computes $P_{\mathcal{T}_v}(\pi_{d_4}^*)$ and $P_{\mathcal{T}_v}(v_4=1,\pi_{d_4}^*,d_4)$. The quasi-Bayesian network $\mathcal{T}_v$ consists of two isolated nodes $c_{10}$ and $d_2$. Those nodes are quasi-random nodes whose associated potentials are 1. Hence $P_{\mathcal{T}_v}(\pi_{d_4}^*) = 1$. The quasi-Bayesian network $\mathcal{T}_{u_4}$ is the same as $\mathcal{T}$. EF calls VE which computes $P_{\mathcal{T}_{u_4}}(v_4=1,\pi_{d_4}^*,d_4)$ as follows:

$$P_{\mathcal{T}_v}(v_4=1,c_{10},d_2,d_4) = \sum_{c_{11}} P(c_{11}|c_{10},d_2) \sum_{c_{12}} P(c_{12}|c_{11},d_4) P(v_4=1|c_{12})$$
$$= \sum_{c_{11}} P(c_{11}|c_{10},d_2) \sum_{c_{12}} P(c_{12}|c_{11},d_4) f_{u_4}(c_{12}).$$

Since $P_{\mathcal{T}_v}(\pi_{d_4}^*) = 1$, the evaluation functional $e_{\mathcal{T}}(\pi_{d_4}^*,d_4)$ obtained by step 1(b) is

$$e_{\mathcal{T}}(c_{10},d_2,d_4) = P_{\mathcal{T}_v}(v_4=1,c_{10},d_2,d_4).$$

Step 1(c) finds an optimal decision rule for $d_4$ by

$$\delta_d(c_{10},d_2) = \arg\max_{d_4} e_{\mathcal{T}}(c_{10},d_2,d_4).$$

Step 1(d) constructs the reduced body of $\mathcal{N}$ w.r.t $d_4$, which is shown in Figure 5 (1). Again because $P_{\mathcal{T}_v}(\pi_{d_4}^*) = 1$, the value function of the new value node $u_4$ is given by

$$f_{u_4}(c_{10},d_2) = \max_{d_4} e_{\mathcal{T}}(c_{10},d_2,d_4).$$
6.3 Optimal decision rule for \(d_3\)

\(\mathcal{N}\) now denotes the influence diagram in Figure 5 (1), where \(d_3\) is the tail decision nodes. Step 1(a) constructs the reduced tail \(\mathcal{T}\) of \(\mathcal{N}\) w.r.t \(d_3\), which is shown in Figure 6 (2). From the tail, EVALUATE finds that \(\pi_\mathcal{w.r.t}^{d_3} = \{e_6\}\). All other parents of \(d_3\) are irrelevant.

EF computes \(P_{\mathcal{T}_1}(\pi_{d_3}^r), P_{\mathcal{T}_2}(v_3=1, \pi_{d_3}^r, d_3), \) and \(P_{\mathcal{T}_3}(v_2=1, \pi_{d_3}^r, d_3)\). The quasi-Bayesian network \(\mathcal{T}_c\) consists of only one node \(e_6\). It is quasi-random node whose associated potential is 1. Hence \(P_{\mathcal{T}_c}(\pi_{d_3}^r) = 1\). The quasi-Bayesian network \(\mathcal{T}_{v_3}\) is the same as \(\mathcal{T}\) except it does not contain the node \(v_2\). VE is called and computes \(P_{\mathcal{T}_{v_3}}(v_3=1, \pi_{d_3}^r, d_3)\) as follows:

\[
P_{\mathcal{T}_{v_3}}(v_3=1, e_6, d_3) = \sum_{c_7} p(c_7 | e_6) \sum_{c_9} p(c_9 | c_7, d_3) \sum_{c_8} p(c_8 | c_7) f_{v_3}(c_8, c_9).
\]

The quasi-Bayesian network \(\mathcal{T}_{v_2}\) consists of two nodes \(d_3\) and \(v_2\). VE is called and finds that \(P_{\mathcal{T}_{v_2}}(v_2=1, \pi_{d_3}^r, d_3)\) is simply

\[
P_{\mathcal{T}_{v_2}}(v_2=1, e_6, d_3) = f_{v_2}(d_3).
\]

Since \(P_{\mathcal{T}_c}(\pi_{d_3}^r) = 1\), the evaluation functional obtained at step 1(b) is

\[
e_\mathcal{T}(e_6, d_3) = P_{\mathcal{T}_{v_3}}(v_3=1, e_6, d_3) + P_{\mathcal{T}_{v_2}}(v_2=1, e_6, d_3).
\]

Step 1(c) finds an optimal decision rule for \(d_3\) by

\[
\delta_3(e_6) = \arg \max_{d_3} e_\mathcal{T}(e_6, d_3).
\]

Step 1(d) constructs the reduced body of \(\mathcal{N}\) w.r.t \(d_3\), which is shown in Figure 6 (1). The value function of the new value node \(u_3\) is given by

\[
f_{u_3}(e_6) = \max_{d_3} e_\mathcal{T}(e_6, d_3).
\]
6.4 Optimal decision rule for $d_2$

$\mathcal{N}$ now denotes the influence diagram in Figure 6 (1). Here $d_2$ is the tail decision node. Step 1(a) constructs the reduced tail $\mathcal{T}$ of $\mathcal{N}$ w.r.t $d_2$, which is shown in Figure 7 (2). From the tail, EVALUATE finds that $\pi_{d_2} = \{c_5\}$. All other parents of $d_2$ are irrelevant.

EF computes $P_{T_2}(\pi_{d_2}^r)$ and $P_{T_4}(u_4=1, c_5, d_2)$. The quasi-Bayesian network $T_2$ consists of only $c_5$, which is a quasi-random node and whose associated potential is the constant 1. Hence $P_{T_2}(\pi_{d_2}^r) = 1$. The quasi-Bayesian network $T_4$ is the same as $T$. VE is called and computes $P_{T_4}(v_4=1, \pi_{d_2}^r, d_2)$ as follows:

$$\begin{align*}
P_{T_4}(u_4=1, c_5, d_2) &= \sum_{c_{10}} P(c_{10}|c_5) f_{u_4}(c_{10}, d_2).
\end{align*}$$

Since $P_{T_2}(\pi_{d_2}^r) = 1$, the evaluation functional obtained in step 1(b) is

$$e_T(c_5, d_2) = P_{T_4}(u_4=1, c_5, d_2).$$

Step 1(c) finds an optimal decision rule for $d_2$ by

$$\delta_2(c_5) = \text{arg } \max_{d_2} e_T(c_5, d_2).$$

Step 1(d) constructs the reduced body of $\mathcal{N}$ w.r.t $d_2$, which is shown in Figure 7 (1). The value function of the new value node $u_2$ is given by

$$f_{u_2}(c_5) = \max_{d_2} e_T(c_5, d_2).$$

6.5 Optimal decision rule for $d_1$

$\mathcal{N}$ now denotes the influence diagram in Figure 7 (1). Here $d_1$ is the only decision node. Step 1(a) constructs the tail of $\mathcal{N}$ w.r.t $d_1$, which is shown in Figure 8 (2). From the tail, EVALUATE finds that $\pi_{d_1} = \{c_2\}$.

EF computes $P_{T_2}(\pi_{d_1}^r), P_{T_3}(u_3=1, \pi_{d_1}^r, d_1), P_{T_2}(u_2=1, \pi_{d_1}^r, d_1)$, and $P_{T_1}(v_1=1, \pi_{d_1}^r, d_1)$. The quasi-Bayesian network $T_2$ consists of only $c_2$, which is a quasi-random node and
whose associated potential is the constant 1. Hence \( P_{T_1}(\pi_{d_1}) = 1 \). The quasi-Bayesian network \( T_{u_3} \) consists of the nodes \( c_2, d_1, c_4, c_6, \) and \( u_3 \). VE is called and computes \( P_{T_{u_3}}(u_3 = 1, \pi_{d_1}, d_1) \) as follows:

\[
P_{T_{u_3}}(u_3 = 1, c_2, d_1) = \sum_{c_4} P(c_4|c_2, d_2) \sum_{c_6} P(c_6|c_4)f_{u_3}(c_6).
\]

The quasi-Bayesian network \( T_{u_2} \) consists of the nodes \( c_1, c_2, d_1, c_3, c_4, c_5, \) and \( u_2 \). VE is called and computes \( P_{T_{u_2}}(u_2 = 1, \pi_{d_1}, d_1) \) as follows:

\[
P_{T_{u_2}}(u_2 = 1, c_2, d_1) = \sum_{c_4} P(c_4|c_2, d_2) \sum_{c_3} \sum_{c_1} P(c_1)P(c_3|c_1, c_2)[\sum_{c_5} P(c_5|c_3, c_4)f_{u_2}(c_5)].
\]

The quasi-Bayesian network \( T_{u_1} \) consists of the nodes \( d_1 \) and \( v_1 \). VE is called and finds that \( P_{T_{u_1}}(v_1 = 1, \pi_{d_1}, d_1) \) is simply

\[
P_{T_{u_1}}(u_1 = 1, c_2, d_1) = f_{v_1}(d_1).
\]

Since \( P_{T_1}(\pi_{d_1}) = 1 \), the evaluational functional obtained at step 1(b) is

\[
e_T(c_2, d_1) = P_{T_{u_3}}(u_3 = 1, c_2, d_1) + P_{T_{u_2}}(u_2 = 1, c_2, d_1) + P_{T_{u_1}}(u_2 = 1, c_2, d_1).
\]

Step 1(c) finds an optimal decision rule for \( d_1 \) by

\[
\delta_1(c_2) = \arg \max_{d_1} e_T(c_2, d_1).
\]

Step 1(d) constructs the reduced body of \( N \) w.r.t \( d_1 \), which is shown in Figure 8 (1).

The value function of the new value variable \( u_1 \) is given by

\[
f_{u_1}(c_2) = \max_{d_1} e_T(c_2, d_1).
\]

### 6.6 Optimal expected value

\( N \) now denotes the value network in Figure 8 (1). The quasi-Bayesian network \( N_{u_1} \) is the same as \( N \) itself. EV calls VE and computes \( P_{N_{u_1}}(u_1 = 1) \) as follows:

\[
P_{N_{u_1}}(u_1 = 1) = \sum_{c_2} P(c_2)f_{u_1}(c_2).
\]

Since \( u_1 \) is the only value node, \( E[N] = P_{N_{u_1}}(u_1 = 1) \). It is the optimal expected value of the original influence diagram.
Comparisons with existing methods

Before this paper, the fusion algorithm by Shenoy (1992) and the strong junction tree propagation algorithm by Jensen et al. (1994) represent the state-of-the-art for influence diagrams evaluation. Those two algorithms are essentially equivalent. This section argues that, when VE is use for probabilistic inference, EVALUATE is more efficient than Shenoy’s fusion algorithm.

Arbitrary Bayesian network inference algorithms can be used in EVALUATE for probabilistic inference. If one chooses an algorithm that is more efficient than VE, for example the VEC algorithm (Zhang and Poole 1996) or some approximation algorithm, EVALUATE can be even more efficient.

We shall first propose an alternative method for computing evaluation functionals of reduced tails and for computing the expected value of value networks and argue it is usually less efficient than the method used in EVALUATE. We shall then show that if this alternative method is used, EVALUATE is still more efficient than Shenoy’s fusion algorithm.

7.1 Alternatives to EF and EV

Let $T$ be the reduced tail of an influence diagram w.r.t its tail decision node $d$. Suppose $\rho$ is an elimination ordering consisting of random nodes in $T$ that are not parents of $d$. It can be proved that the following procedure computes $e_T(\pi^r_d, d)$ and $P_{T_e}(\pi^r_d)$.

Procedure EF1($T$)

1. Let $\mathcal{P}$ be the list of probabilities of random nodes in $T$ and let $\mathcal{F}$ be the list of value functions of value nodes in $T$.

2. Fuse random nodes one by one according to the ordering $\rho$. To fuse a random node $z$ is to
   (a) Remove from $\mathcal{P}$ all the factors $p_1, \ldots, p_k$ that involve $z$. If such factors exist, add the new factor $p = \sum_z \Pi_{i=1}^k p_i$ to the list $\mathcal{P}$.
   (b) Remove from $\mathcal{F}$ all the factors $f_1, \ldots, f_l$ that involve $z$. If such factors exist, add the new factor $\sum_z [\sum_{i=1}^l f_i][\Pi_{i=1}^k p_i]/p$ to the list $\mathcal{F}$.

3. After all nodes in $\rho$ have been fused,
   (a) Multiply all factors in $\mathcal{P}$. The result is $P_{T_e}(\pi^r_d)$.
   (b) Sum all factors in $\mathcal{F}$ and divide the summation by $P_{T_e}(\pi^r_d)$. The result is $e_T(\pi^r_d, d)$.

An alternative procedure for computing expected values of value networks can designed similarly. We shall use EV1 to denote this procedure.

Replacing EF and EV by EF1 and EV1 respectively in EVALUATE, we get another algorithm for evaluating influence diagrams. We shall refer to this algorithm by EVALUATE1.
7.2 EF versus EF1 and EV versus EV1

This subsection compares EF with EF1 in terms of efficiency. Comparisons between EV and EV1 are similar.

EF computes the evaluation functional of a reduced tail $T$ and $P_{T_e}(\pi_v^r)$ by first calling VE, which calculates $P_{T_e}(v=1, \pi_v^r, d)$ and $P_{T_e}(\pi_v^r)$ by eliminating random nodes from $T_e$ and $T_v$ one by one. EF then divides $P_{T_e}(v=1, \pi_v^r, d)$ by $P_{T_e}(\pi_v^r)$. EF1 computes the evaluation functional of $T$ and $P_T(\pi_v^r)$ by fusing random nodes in $T$ one by one.

Define the size of a factor to be the number of variables in involved in the factor. The complexities of EF and EF1 can be measured by the sizes of the largest factors encountered; they are exponential in those sizes. Since $T_e$ and each $T_v$ are subnetworks of $T$, the maximum factor sizes encountered by EF can be smaller than those encountered by EF1. This is the primary reason for the claim that EF is usually more efficient than EF1.

A second reason is that EF does not perform any divisions until the last step, while EF1 might divide factors when fusing each node.

On the other hand, EF1 deals with one quasi-Bayesian network, i.e., $T$ itself, while EF has to deal with more than one quasi-Bayesian networks; it has to deal with $T_e$ and $T_v$ for each value node $v$. This disadvantage of EF is often offset by the two advantages mentioned above.

7.3 Shenoy’s fusion algorithm

Let $\rho$ be an ordering of random and decision nodes in an influence diagram $\mathcal{N}$ such that a decision node $d$ appears right after all the nodes that are not parents to $d$. Shenoy’s fusion algorithm evaluates $\mathcal{N}$ as follows:

Procedure FUSE($\mathcal{N}$)

1. Let $\mathcal{P}$ be the list of probabilities of random nodes in $\mathcal{N}$ and let $\mathcal{F}$ be the list of value functions of value nodes in $\mathcal{N}$.

2. Fuse random and decision nodes one by one according to the ordering $\rho$. A random node is fused in the same way as in EF1. (For any decision node $d$, after all the nodes before $d$ have been fused, no factors in $\mathcal{P}$ involve $d$.) To fuse $d$ is to
   
   (a) Remove from $\mathcal{F}$ all the factors $f_1, \ldots, f_l$ that involve $d$ and multiply them. (The result is a function of $d$ and nodes in $\pi_v^r$. Denote it by $e(\pi_v^r, d)$.)
   
   (b) Add the new factor $\max_{x,d} e(\pi_v^r, d)$ to $\mathcal{F}$, and
   
   (c) Obtain an optimal decision rule for $d$ by

   \[ \delta^*(\pi_v^r) = \arg\max_{x,d} e(\pi_v^r, d). \]

3. (After all nodes in $\rho$ have been fused, there is only one factor left in $\mathcal{F}$. It is an constant and is the optimal expected value of $\mathcal{N}$. )

Return the optimal decision rules and the optimal expected value.
7.4 EVALUATE1 versus Shenoy’s fusion algorithm

EVALUATE1 evaluates an influence diagram $\mathcal{N}$ in the same way as FUSE except for two things. Firstly EVALUATE1 needs to construct reduced tails and bodies. Secondly EVALUATE1 uses an elimination ordering where a decision node $d$ appears right after all nodes in the downstream set of $\mathcal{N}$ w.r.t $d$.

The amount of computations needed for constructing reduced tails and reduced bodies are negligible compared to computations for fusing nodes. However, the difference in the restriction on the elimination ordering can have significant efficiency implications.

In an influence diagram $\mathcal{N}$, there is a directed path that contains all decision nodes. Let $d$ be any decision node and let $d'$ be the decision node that is the closest to $d$ in the path among all decision node ancestors of $d$. Let $S_1(d', d) = \pi_d - \pi_{d'}$ and let $S_2(d', d) = \{d' \text{ or } d \}$ be the set of nodes that are in the downstream set of $\mathcal{N}$ w.r.t $d'$ and are not in the downstream set of $\mathcal{N}$ w.r.t $\pi_d$. Clearly, $S_1(d', d) \subseteq S_2(d', d)$.

Let $\rho$ be an elimination ordering. If $\rho$ conforms to the restriction of FUSE, then the nodes between $d'$ and $d$ are the nodes in $S_1(d', d)$. If $\rho$ conforms to the restriction of EVALUATE1, then the nodes between $d'$ and $d$ are the nodes in $S_2(d', d)$.

Suppose $\rho$ conforms to the restriction of FUSE. Let $\rho'$ be the ordering obtained from $\rho$ by doing the following for each decision node $d$: starting with the first decision node to the last (tail) decision node: move all the nodes that appear after $d$ and are not in the downstream set of $\mathcal{N}$ w.r.t $d$ to the right before $d$ and keep their relative order. The nodes moved are in the upstream set of $\mathcal{N}$ w.r.t $d$. Since factors that involve nodes in the upstream set are different from factors that involve nodes in the downstream set, $\rho$ and $\rho'$ induce exactly the same computations.

It is evident that $\rho'$ conforms to the restriction of EVALUATE1. So, for any elimination ordering that conforms to the restriction of FUSE, there is an elimination ordering that conforms to the restriction of EVALUATE1 such that the two orderings induce exactly the same computations. The opposite is not true because in $\rho'$, nodes in $S_1(d', d)$ must appear before nodes in $S_2(d', d) - S_1(d', d)$. Therefore, there could be an elimination ordering that one can use with EVALUATE1 which renders EVALUATE1 more efficient than FUSE no matter what ordering is used there.

For the influence diagram in Figure 4, the elimination ordering given by Jensen et al. (1994) is $\rho_1$: $c_{12}, c_8, c_9, c_{11}, c_7, c_1, c_3, c_4, d_1, c_{10}, d_3, d_2, c_6, c_5, d_1$, and $c_2$. It is an optimal ordering that conforms to the restriction of FUSE. With EVALUATE1, one can use the ordering $\rho_2$: $c_{12}, c_{11}, d_4, c_8, c_9, c_7, d_3, c_{10}, d_2, c_6, c_5, c_1, c_3, c_4, d_1$, and $c_2$. The reader can easily verify that using $\rho_2$ is more efficient than using $\rho_1$.

8 Conclusions

We have investigated the relationship between probabilistic inference in Bayesian networks and evaluation of influence diagrams. In the process, a new method for evaluating influence diagrams has been proposed. The method clearly separates probabilistic inference tasks from other computations so that arbitrary Bayesian network inference algorithms can be used for probabilistic inference. We have argued that if the VE
algorithm (Zhang and Poole 1994, Zhang and Poole 1996) is used for probabilistic inference, the new method is more efficient than the best previous methods (Shenoy 1992, Jensen et al 1994). If a more efficient probabilistic inference algorithm such as the VEC algorithm (Zhang and Poole 1996) or some approximation algorithm is used, the new method can be even more efficient.

Acknowledgement

Research was supported by Hong Kong Research Council grant HKUST658/95E. The paper has benefited from discussions with David Poole and Prakash Shenoy. I am also grateful to the anonymous reviewers and the editor Nick Cercone for valuable comments.

References


