Solving Influence Diagrams with Simple Propagation

Anders L. Madsen[1,2(✉)], Cory J. Butz[3], Jhonatan Oliveira[3], and André E. dos Santos[3]

[1] HUGIN EXPERT A/S, Aalborg, Denmark
anders@hugin.com
[2] Department of Computer Science, Aalborg University, Aalborg, Denmark
[3] Department of Computer Science, University of Regina, Regina, Canada

Abstract. Recently, Simple Propagation was introduced as an algorithm for belief update in Bayesian networks using message passing in a junction tree. The algorithm differs from other message passing algorithms such as Lazy Propagation in the message construction process. The message construction process in Simple Propagation identifies relevant potentials and variables to eliminate using the *one-in, one-out*-principle. This paper introduces Simple Propagation as a solution algorithm for influence diagrams with discrete variables. The *one-in, one-out*-principle is not directly applicable to influence diagrams. Hence, the principle is extended to cope with decision variables, utility functions, and precedence constraints to solve influence diagrams. Simple Propagation is demonstrated on an extensive example and a number of useful and interesting properties of the algorithm are described.

Keywords: Influence diagrams · Simple propagation · Discrete variables

1 Introduction

An influence diagram [1] is a natural representation of a decision problem where a single decision maker has to make a sequence of decisions under uncertainty. An influence diagram is essentially a Bayesian network [2,3] augmented with facilities (decision variables, utility functions and precedence constraints) to support decision making under uncertainty. In essence, the solution to a decision problem represented as an influence diagram consists of an optimal strategy specifying which decision option the decision maker should take at each decision depending on the information known to her at that point in time.

Various extensions of traditional influence diagrams have been introduced. These include limited-memory influence diagrams [4], unconstrained influence diagrams [5], and influence diagrams with mixed variables [6]. The focus of this paper is on traditional influence diagrams with discrete variables. The solution of an influence diagram is extended to consist of the optimal strategy, the expected

M.-J. Meurs and F. Rudzicz (Eds.): Canadian AI 2019, LNAI 11489, pp. 68–79, 2019.
https://doi.org/10.1007/978-3-030-18305-9_6

utility of adhering to the optimal strategy and the probability of future decisions. The probability of future decisions under a (optimal) strategy can be determined by encoding the decision policy for each decision as a conditional probability distribution (CPD) [7]. We consider this part of the process of solving an influence diagram.

Some of the popular algorithms for solving the traditional influence diagram include [8–13]. One of the first methods for solving influence diagrams was based on Cooper's technique [14] of transforming the influence diagram into a Bayesian network and using a Bayesian network inference algorithm to solve the influence diagram. Other methods include message passing in a tree and approximation algorithms based on sampling. A recent review of methods for solving influence diagrams can be found here [15] and a survey of probabilistic decision graphs can be found in this paper [16]. Recent work on solving influence diagrams include finding bounds for influence diagrams using join graph decomposition [17] and an improved method for solving hybrid influence diagrams [18].

Simple Propagation [19] is a new algorithm for belief update in Bayesian networks. It proceeds by message passing in a junction tree representation of the Bayesian network taking advantage of a decomposition of clique and separator potentials. In Simple Propagation, message construction is based on a *one-in, one-out*-principle meaning that a potential relevant for a message has at least one non-evidence variable in the separator and at least one non-evidence variable not in the separator. The advantage of Simple Propagation is its simplicity compared to, for instance, Lazy Propagation [20]. This paper extends Simple Propagation to the solution of traditional influence diagrams, where the solution consists of an optimal strategy, the expected utility of adhering to this strategy and the probability of future decisions under the strategy.

Simple Propagation is extended to cover solutions of traditional influence diagrams by identifying an optimal strategy computing the expected utility, which involves computing the probability of future decisions under the optimal strategy. To achieve this, the *one-in, one-out*-principle is extended such that it considers both probability and utility potentials. Our approach supports exploitation of probabilistic barren variables [9] and the decomposition of utility potentials reducing the number of calculation operations.

2 Background

A traditional influence diagram over discrete variables is a triple $N = (G, \Phi, \Psi)$ where $G = (V, E)$ is a DAG (directed, acyclic graph) over vertices V and edges $E \subseteq V \times V$. Each vertex $v \in V$ represents a random (or chance) variable, a decision variable, or a utility function. Let $\mathcal{D} = \{D_1, \ldots, D_n\}$ be the set of decision variables and let $\mathcal{X}$ be the set of random variables and $\mathcal{U}$ be the set of utility nodes. Each random variable $X \in \mathcal{X}$ has a CPD $P(X \,|\, \mathrm{pa}(X))$, where $\mathrm{pa}(X)$ are the parent vertices of X in G, and each utility node $u \in \mathcal{U}$ represents a utility function $u(\mathrm{pa}(u))$ (we use u to denote both the utility node and the utility function). Finally, we denote $\Phi = \{P(X \,|\, \mathrm{pa}(X)) \in \mathcal{X}\}$, $\Psi = \{u(\mathrm{pa}(u)) \,|\, u \in \mathcal{U}\}$, and $\mathrm{dom}(\phi)$ (or $\mathrm{dom}(\psi)$) the domain of ϕ (ψ).

The influence diagram $N = (G, \Phi, \Psi)$ is an efficient representation of a joint expected utility function:

$$EU(\mathcal{X}, \mathcal{D}) = \prod_{X \in \mathcal{X}} P(X \,|\, \mathrm{pa}(X)) \times \sum_{u \in \mathcal{U}} u(\mathrm{pa}(u)). \tag{1}$$

The regularity constraints of the traditional influence diagram state that there must be a total order on the decisions and that the decision maker has perfect recall. Assuming decisions are ordered according to index, the random variables can be partitioned into information sets using a partial precedence ordering $\prec$: $\mathcal{I}_0 \prec D_1 \prec \mathcal{I}_1 \prec \cdots \prec \mathcal{I}_{n-1} \prec D_n \prec \mathcal{I}_n$. The set $\mathcal{I}_i$ is the set of random variables observed (by the decision maker) after making decision D_i and before decision $\mathcal{I}_{i+1}$, i.e., $\mathcal{I}_0$ is the random variables initially observed and $\mathcal{I}_n$ is the set of random variables never observed or observed after the last decision.

In a Bayesian network, a variable X is a barren node when X is never observed, $P(X)$ is of no interest, and the same holds true for each of X's descendants (if any) [9]. A variable X in an influence diagram is a probabilistic barren node when X is a barren node if only the set of probability potentials are considered [9,21].

2.1 Strategies, Decision Policies and Future Decisions

A decision policy $\delta_{D_i}(\mathrm{rel}(D_i))$ is a mapping from $\mathrm{rel}(D_i)$ to D_i where $\mathrm{rel}(D_i) \subseteq \bigcup_{j<i} \mathcal{I}_j \cup \{D_j\}$ is defined below. A strategy is a collection of decision policies $\Delta = \{\delta_D \,|\, D \in \mathcal{D}\}$, one for each decision. A decision policy $\delta_D(\mathrm{rel}(D))$ is encoded as a CPD $P(D \,|\, \mathrm{rel}(D))$ as follows:

$$P_\Delta(D = d \,|\, \mathrm{rel}(D) = z) = \begin{cases} 1 & \text{if } \delta_D(\mathrm{rel}(D) = z) = d \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The strategy Δ induces a joint probability distribution:

$$P_\Delta(\mathcal{X} \cup \mathcal{D}) = \prod_{X \in \mathcal{X}} P(X \,|\, \mathrm{pa}(X)) \times \prod_{D \in \mathcal{D}} P_\Delta(D \,|\, \mathrm{rel}(D)). \tag{3}$$

This factorization can be used to compute the probability of future decisions [7]. Notice that the probability of future decisions can be computed under any strategy D.

The expected utility $EU(\Delta)$ of a strategy Δ is the expectation of the total utility $U(\bigcup_{u \in \mathcal{U}} \mathrm{pa}(u)) = \sum_{u \in \mathcal{U}} u(\mathrm{pa}(u))$ under the probability distribution $P_\Delta(\mathcal{X} \cup \mathcal{D})$, $\sum_{Y \in \mathcal{X} \cup \mathcal{D}} U(\bigcup_{u \in \mathcal{U}} \mathrm{pa}(u)) \times P_\Delta(\mathcal{X} \cup \mathcal{D})$ [22]. A strategy that maximizes the expected utility is an optimal strategy, denoted $\hat{\Delta}$, and has the property $EU(\hat{\Delta}) \geq EU(\Delta)$, for all Δ.

An optimal strategy $\hat{\Delta}$ and its expected utility of $EU(\hat{\Delta})$ can be computed as:

$$EU(\hat{\Delta}) = \sum_{\mathcal{I}_0} \max_{D_1} \sum_{\mathcal{I}_1} \cdots \sum_{\mathcal{I}_{n-1}} \max_{D_n} \sum_{\mathcal{I}_n} EU(\mathcal{X}, \mathcal{D}). \tag{4}$$

Under the regularity constraints, the decision policy δ_{D_i} for decision variable D_i can, in principle, be a mapping from all past observations and decisions $\bigcup_{j<i} \mathcal{I}_j \cup \{D_j\}$ onto D_i making the policy exponentially large in the number of observations. Luckily, this is often not the case though. That is, not every past observation or decision variable is *requisite* for a decision, i.e., not all past observations may impact the choice of the decision maker at a decision D. A variable $Y \in \text{pa}(D) \subseteq \mathcal{X} \cup \mathcal{D}$, $D \in \Delta$ is non-requisite, if $(\mathcal{U} \cap \text{de}(D)) \perp \{Y\} \mid (\text{fa}(D) \setminus \{Y\})$ [4,23,24]. This condition states that Y is non-requisite, if it is separated from the utility nodes that are descendents of D given the family of D except Y, i.e., the value of Y does not affect the optimal choice at D. If a variable is not non-requisite, then it is requisite and belongs to the relevant past [7], denoted $\text{rel}(D)$.

2.2 Strong Junction Tree Construction

Simple propagation is performed in a strong junction tree representation. The strong junction tree ensures that variables are eliminated in the reverse order of $\prec$ during evaluation. In addition, it serves as a caching structure through the message passing process.

The construction of a strong junction tree representation $T = (\mathcal{C}, \mathcal{S})$ with cliques $\mathcal{C}$ and separators $\mathcal{S}$ of influence diagram $N = (G, \Phi, \Psi)$ proceeds in four steps [12]:

Minimalization where information arcs from non-requisite parents of decision nodes are removed.

Moralization where each pair of parents X_i and X_j with a common child (representing either a random variable or a utility function) are connected by an undirected edge, all edges are made undirected and utility nodes are removed to produce G^M.

Strong Triangulation where G^M is triangulated to produce G^T using an elimination order σ such that $\sigma(\mathcal{I}_{i+1}) < \sigma(D_{i+1}) < \sigma(\mathcal{I}_i)$, for all $i \in [0, n-1]$.

Strong Junction Tree Structure where the cliques $\mathcal{C}$ are connected by separators $\mathcal{S}$ producing a tree $T = (\mathcal{C}, \mathcal{S})$ with strong root $R \in \mathcal{C}$.

Initialization of Junction Tree is the process of assigning $\phi \in \Phi$ and $\psi \in \Psi$ to cliques such that $\text{dom}(\phi) \subseteq C$ and $\text{dom}(\psi) \subseteq C$.

Notice that in the last step, the initial clique potential $\pi_A = (\Phi_A, \Psi_A)$ consists of the set of probability distributions $\Phi_A = \{P_1, \ldots, P_l\}$ and the set of utility functions $\Psi_A = \{u_1, \ldots, u_m\}$ assigned to A. Simple Propagation proceeds by message passing between neighboring cliques of T. The message from clique B to clique A is denoted $\pi_{B \to A} = (\Phi_{B \to A}, \Psi_{B \to A})$. The combination of a clique potential $\pi_A = \{\Phi_A, \Psi_A\}$ and the message $\pi_{B \to A} = (\Phi_{B \to A}, \Psi_{B \to A})$ is denoted $\otimes$ and simply amounts to set union, i.e., $\pi_A \otimes \pi_{B \to A} = (\Phi_A \cup \Phi_{B \to A}, \Psi_A \cup \Psi_{B \to A})$.

If $C \in \mathcal{C}$, then $\mathcal{X}(C)$ denotes the variables of C and $\text{adj}(C)$ denotes the cliques adjacent to C. Let $A \in \text{adj}(B)$ with A closer to R, then $\text{pa}(B) = A$ denotes the parent clique and $S = A \cap B$ is denoted the parent separator.

3 Simple Propagation

Simple Propagation was introduced by [19] as a message passing algorithm for belief update in Bayesian networks. In this section, Simple Propagation is extended to cover solutions of traditional influence diagrams by identifying an optimal strategy $\hat{\Delta}$, computing $EU(\hat{\Delta})$, and computing $P_{\hat{\Delta}}(D)$, for each $D \in \mathcal{D}$ under $\hat{\Delta}$.

When constructing the message $\pi_{A \to B}$ from clique A to clique B over separator S, Simple Propagation uses the *one-in, one-out*-principle to order the computations. This means that Simple Propagation is not driven by identifying an elimination order. Instead the elimination order is induced by the order in which potentials satisfying the *one-in, one-out*-principle are processed. The next potential to consider is selected randomly among the potentials satisfying the principle. In the case of influence diagrams, the induced elimination order must satisfy the partial order $\prec$. This is achieved by applying an extended *one-in, one-out*-principle relative to the decision variables in clique A, not in clique B, in reverse order of $\prec$ when computing a message up the junction tree.

Let A and B be adjacent cliques with $B = \mathrm{pa}(A)$ and $\pi_A \otimes_{C \in \mathrm{adj}(A) \setminus \{B\}} \pi_{C \to A} = (\Phi, \Psi)$. The *one-in, one-out*-principle is extended such that it considers both probability and utility potentials, i.e., $\Phi \cup \Psi$, when selecting a potential. When constructing the message $\pi_{A \to B}$, the principle is applied relative to each decision variable $D \in A \setminus B$ in reverse order of $\prec$. For decision variable $D_i \in A \setminus B$ any potential ω with $D_i \in \mathrm{dom}(\omega)$ such that ω has a $Y \in \mathrm{dom}(\omega) \cap \mathcal{I}_i$ is selected. If, for decision variable $D_i \in \mathcal{D}$, there is no potential ω such that $D_i \in \mathrm{dom}(\omega)$, then D has no descendants and any policy can be assigned to D. Each time a variable must be eliminated, Variable Elimination is applied as the marginalization operation (see Algorithm 3).

Let $\mathcal{D}(A) = \{D_{A_1}, \ldots, D_{A_m}\} \subseteq \mathcal{D}$ be the set of decision variables in clique A. The variables $\mathcal{X}(A)$ of clique A can be partitioned according to the partial precedence ordering $\prec$ and must be processed in reverse order to produce correct results. However, when sending a message from clique A to B, we only have to consider the order of decisions in $A \setminus B$ that must be eliminated when constructing the message $\pi_{A \to B}$. Once the decisions $A \setminus B$ have been eliminated, the *one-in, one-out*-principle is applied on the result relative to parent separator $S = A \cap B$ between A and B.

Algorithm 1 shows the general Simple Propagation algorithm for influence diagrams. It solves the influence diagram to identify $\hat{\Delta}$ and computes $EU(\hat{\Delta})$ as well as constructs a Bayesian network N^* to compute $\{P_{\hat{\Delta}}(D) \mid D \in \mathcal{D}\}$. The Bayesian network N^* is constructed from N by changing decision variables to random variables, removing utility functions, and keeping the structure of G induced by the random and decision variables. The algorithm performs a collect operation where messages are passed from the leaf cliques towards R. Next, a Bayesian network is constructed where each decision policy $\delta_D(\mathrm{rel}(D))$ is encoded as a CPD $P_{\hat{\Delta}}(D \,|\, \mathrm{rel}(D))$. This is followed by a distribute operation where messages are passed from R towards the leaf cliques. This operation only

Procedure *SP-ID*$(N = (G, \mathcal{P}, \mathcal{U}), T)$

1 Perform a collect operation on the strong root R of T to identify $\hat{\Delta}$ using Algorithm 2
2 Compute $EU(\hat{\Delta})$ at R
3 Create BBN N^* from N
for $D \in \mathcal{D}$ *represented in* N^* **do**
Encode decision policy $\delta_D(\text{rel}(D))$ as $P(D \mid \text{rel}(D))$
end
4 Perform a distribute operation on R in N^*
5 **for** $D \in \mathcal{D}$ *represented in* N **do**
Compute $P_{\hat{\Delta}}(D)$
end
6 **return** $\hat{\Delta}$ and $\{P_{\hat{\Delta}}(D) \mid D \in \mathcal{D}\}$

Algorithm 1. The Simple Propagation algorithm for Influence Diagrams.

involves elimination of random variables and proceeds as in Simple Propagation for Bayesian networks [19].

All probability and utility potentials satisfying the *one-in, one-out*-principle have variables to be eliminated. In addition, any potential ϕ (or ψ) with $\text{dom}(\phi) \subseteq S$ (or $\text{dom}(\psi) \subseteq S$) must be included in the message.

Algorithm 2 is the Simple Message Computation algorithm applied during the collect operation (SMC-CE). It applies the extended *one-in, one-out*-principle when processing $\mathcal{D}(A)$ in reverse order of $\prec$.

To complete the collect operation, the potential $\pi_R \otimes \bigotimes_{A \in \text{adj}(R)} \pi_{A \to R}$ of the strong root R of T is marginalized to $\emptyset$ to determine $EU(\hat{\Delta})$. After the last decision variable to consider in the solution process, i.e., D_1, is eliminated, the remaining potentials are processed to compute $EU(\hat{\Delta})$.

Algorithm 2 applies Algorithm 3 to eliminate a variable (decision or chance) in the message construction process. It is essentially Variable Elimination.

In Algorithm 3, $\Phi^{\downarrow -Y}$ denotes marginalization of Y over the set Φ using $\sum$ for random variables, i.e., $Y \in \mathcal{X}$ and max for decision variables, i.e., $Y \in \mathcal{D}$. In the process of performing a max-marginalization of $D \in \mathcal{D}$, the maximizing alternatives for each parent configuration are recorded as the optimal decision rule for D.

Theorem 1. *Simple Propagation in an Influence Diagram N computes:*

1. *an optimal strategy $\hat{\Delta}$ for the decision problem represented as N,*
2. *the expected utility $EU(\hat{\Delta})$ of $\hat{\Delta}$, and*
3. *the probability of future decisions $\{P_{\hat{\Delta}}(D) \mid D \in \mathcal{D}\}$ under $\hat{\Delta}$.*

Proof. Simple Propagation processes the variables in reverse order of $\prec$. Each variable (random or decision) is eliminated using Algorithm 3, which is equivalent to Variable Elimination. Message passing proceeds as in Lazy Propagation. Hence, the calculations performed during the collect operation correspond to solving the influence diagram with Lazy Propagation using the induced elimination

Procedure *SMC-CE*$(A,\ B,\ (\Phi,\Psi) = \pi_A \otimes \bigotimes_{C\neq B} \pi_{C\to A})$

for $D_i \in \mathcal{D}(A) \setminus B$ *in reverse* $\prec$*-order* **do**
  while $\exists \omega \in \Phi \cup \Psi$ *satisfying the extended one-in, one-out-principle relative to* D_i **do**
    Select variable $Y \in \mathrm{dom}(\omega)$ such that $Y \in \mathcal{I}_i$
    $\Phi^*, \Psi^* = \textsc{Marginalization}(Y, \Phi, \Psi)$
    Set $\Phi = \Phi^*$ and $\Psi = \Psi^*$
  end
  $\Phi^*, \Psi^* = \textsc{Marginalization}(D_i, \Phi, \Psi)$
  Set $\Phi = \Phi^*$ and $\Psi = \Psi^*$
end
while $\exists \omega \in \Phi \cup \Psi$ *satisfying the extended one-in, one-out-principle relative to parent separator* S **do**
  Select variable $Y \in \mathrm{dom}(\omega)$ such that $Y \in A \notin B$
  $\Phi^*, \Psi^* = \textsc{Marginalization}(Y, \Phi, \Psi)$
  Set $\Phi = \Phi^*$ and $\Psi = \Psi^*$
end
Set $\Phi = \Phi^*$ and $\Psi = \Psi^*$
return $\pi_{A\to B} = (\Phi, \Psi)$

Algorithm 2. The Simple Message Computation algorithm under collect.

Procedure Marginalization$(Y,\ \Phi,\ \Psi)$

Set $\Phi_Y = \{\phi \,|\, Y \in \mathrm{dom}(\phi)\}$
Set $\Psi_Y = \{\psi \,|\, Y \in \mathrm{dom}(\psi)\}$
Compute $\phi_Y = (\prod_{\phi\in\Phi_Y} \phi)^{\downarrow -Y}$
Compute $\psi_Y = (\prod_{\phi\in\Phi_Y} \phi \times \sum_{\psi\in\Psi_Y} \psi)^{\downarrow -Y}$
Set $\Phi^* = \{\phi_Y\} \cup \Phi \setminus \Phi_Y$
Set $\Psi^* = \{\frac{\psi_Y}{\phi_Y}\} \cup \Psi \setminus \Psi_Y$
return Φ^*, Ψ^*

Algorithm 3. The Marginalization algorithm.

ordering [20]. The Bayesian network for computing the probability of future decisions is constructed as described by [7]. The distribute operation proceeds as Simple Propagation in a Bayesian network [19].

4 Analysis

This section considers solving two different influence diagrams with Simple Propagation illustrating the process and the properties of the algorithm.

4.1 Influence Diagram with Four Decisions

Figure 1 shows a commonly used example of an influence diagram N_{JJD} with four decisions introduced by [12]. The figure does not include non-forgetting information links in order to reduce the clutter to a minimum.

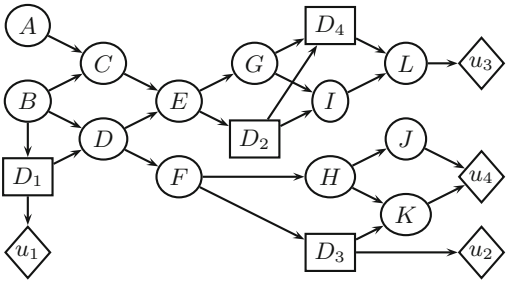

Fig. 1. An influence diagram N_{JJD} with four decisions [12].

Figure 2 shows a strong junction tree representation T_{JJD} of N_{JJD} with strong root $R = BD_1ACDEF$. This is not an optimal junction tree in terms of, for instance, total clique weight, but it serves to illustrate important properties of Simple Propagation for solving influence diagrams. For simplicity, we refer to the two leaf cliques as A and B as indicated in the subscripts of the clique potentials π_A and π_B in the figure.

$$\pi_R = (\{P(A), P(B), P(C \mid A, B), P(D \mid B, D_1), P(E \mid C, D), P(F \mid D)\}, \{u_1(D_1)\})$$

$$\pi_A = (\{P(G \mid E), P(I \mid G, D_2), P(L \mid I, D_4)\}, \{u_3(L)\})$$

$$\pi_B = (\{P(H \mid F), P(J \mid H), P(K \mid H, D_3)\}, \{u_2(D_3), u_4(J, K)\})$$

Fig. 2. A junction tree representation T_{JJD} of N_{JJD}.

Collect Operation. Simple Propagation solves N_{JJD} by first performing a collect operation on R passing messages $\pi_{A \to R}$ and $\pi_{B \to R}$ from the leaf cliques A and B, respectively. The message $\pi_{A \to R}$ is computed from $\pi_A = (\{P(G \mid E), P(I \mid G, D_2), P(L \mid I, D_4)\}, \{u_3(L)\})$, while the message is computed from $\pi_B = (\{P(H \mid F), P(J \mid H), P(K \mid H, D_3)\}, \{u_2(D_3), u_4(J, K)\})$.

Consider the construction of the message $\pi_{A \to R}$ passed from A to R during the collect operation. As $D_2, D_4 \in \mathcal{D}(A) \setminus R$ and $D_2 \prec D_4$, the process of constructing $\pi_{A \to R}$ starts with D_4. In this case, the potential including D_4 satisfying the *one-in, one-out*-principle is $P(L \mid D_4, I)$, where $I, L \in \mathcal{I}_4$ must be eliminated. The vanilla version of Simple Propagation selects randomly between $I, L \in \mathcal{I}_4$.

Assume L is selected first. The elimination of L (Algorithm 3) produces $\phi(\cdot \mid D_4, I) = \sum_L P(L \mid D_4, I) = 1_{D_4,I}$ and $\psi_X(D_4, I) = \sum_L P(L \mid D_4, I)u_3(L)$, where $1_{D_4,I}$ is the unity potential indexed by D_4, I, i.e., L is probabilistic barren and we do not need to compute this potential nor the division in Algorithm 3. Hence, $\pi_A^{\downarrow -L} = (\{P(G \mid E), P(I \mid G, D_2)\}, \{\psi(D_4, I)\})$. The next set of potentials satisfying the *one-in, one-out*-principle is $\{P(I \mid G, D_2), \psi(D_4, I)\}$. Here, I is the only *out*-variable and must be eliminated producing $\pi_A^{\downarrow -\{L,I\}} = (\{P(G \mid E)\}, \{\psi(D_4, G, D_2)\})$ as I is probabilistic barren. Now, the variables $\mathcal{I}_4$ have been eliminated. Decision variables D_4 is eliminated from $\psi(D_4, G, D_2)$ and the decision policy $\delta_{D_4}(G, D_2)$ is identified producing $(\{P(G \mid E)\}, \{\psi(G, D_2)\})$. Finally, variables G and D_2 are eliminated to produce the message $\pi_{A \to R} = \pi_A^{\downarrow E} = (\{\}, \{\psi(E)\})$. The induced elimination order is $\sigma = (L, I, D_4, G, D_2)$.

Now assume I is selected first. The elimination of I (Algorithm 3) produces $\phi(L \mid D_2, G, D_4) = \sum_I P(I \mid G, D_2)P(L \mid I, D_4)$. That is, $\pi_A^{\downarrow -I} = (\{P(G \mid E), P(L \mid D_2, G, D_4)\}, \{\psi(L)\})$. The elimination of I is followed by the elimination of L as $P(L \mid D_2, G, D_4)$ is the only potential satisfying the *one-in, one-out*-principle and $L \in \mathcal{I}_4$. The process continues as above producing the message $\pi_{A \to R} = \pi_A^{\downarrow E} = (\{\}, \{\psi(E)\})$ with the induced elimination order $\sigma = (I, L, D_4, G, D_2)$. For the message $\pi_{A \to R}$ the only random element is the selection of the variable to eliminate from $P(L \mid D_4, I)$.

Next, the construction of the message $\pi_{B \to R}$ passed from B to R during the collect operation is constructed. The message is $\pi_{B \to R} = (\{\}, \{\psi(F)\})$. Following the collect operation, the root R is processed to identify $\delta_{D_1}(B)$ and compute $EU(\Delta)$ for $\Delta = \{\delta_{D_1}(B), \delta_{D_2}(E), \delta_{D_3}(F), \delta_{D_4}(D_2, G)\})$.

The root node is processed similarly.

Distribute Operation. The purpose of the distribute operation is to compute the probability of future decisions under the optimal strategy $\hat{\Delta}$, i.e., $P_{\hat{\Delta}}(D)$, for each $D \in \mathcal{D}$. Following the collect operation, the Bayesian network N^*_{JJD} is constructed over $\mathcal{X} \cup \mathcal{D}$, where each decision policy $\delta_D(\text{rel}(D))$ of the optimal strategy $\hat{\Delta}$ is encoded as a CPD $P(D \mid \text{rel}(D))$ and D is turned into a random variable. Subsequently, the distribute operation is performed to compute $P(D)$ for each $D \in \mathcal{D}$. This process proceeds as the distribute operation in a Bayesian network using Simple Propagation [19].

4.2 Distributive Law

Figure 3(a) shows the structure of a simple influence diagram N with one decision $\mathcal{D} = \{D\}$, three random variables $\mathcal{X} = \{C_1, C_2, C_3\}$ and two utility functions $\mathcal{Y} = \{u_1(C_1, C_3), u_2(D, C_2)\}$ and Fig. 3(b) shows a strong junction tree T with root $R = C_1DC_3$.

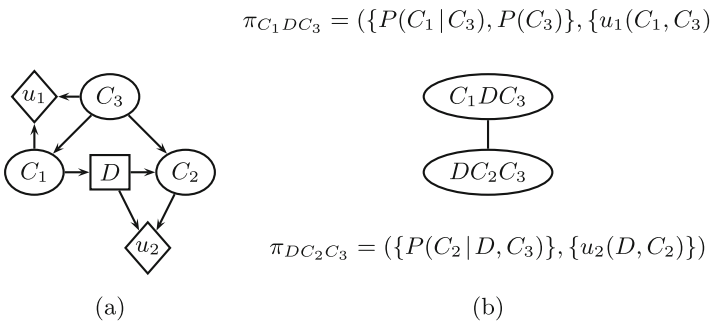

Fig. 3. An influence diagram and its junction tree representation.

Simple Propagation solves N by first performing a collect operation on R passing the message from clique DC_2C_3 to clique C_1DC_3. The collect message $\pi_{DC_2C_3 \to C_1DC_3}$ from clique DC_2C_3 to clique C_1DC_3 is computed as:

$$\begin{aligned}\pi_{DC_2C_3 \to C_1DC_3} &= (\emptyset, \{\sum_{C_2} P(C_2 \mid D, C_3) u_2(D, C_2)\}) \\ &= (\emptyset, \{\psi(D, C_3)\}),\end{aligned}$$

where C_2 is probabilistic barren making $\Phi_{DC_2C_3 \to C_1DC_3} = \emptyset$. At the root, we need to determine the policy δ_D of D and process all potentials to compute $EU(\hat{\Delta})$. The identification of δ_D starts from $\psi(D, C_3)$ as this is the only probability or utility potential associated with R and incoming messages that contains D, i.e., $\pi_R \otimes \pi_{DC_2C_3 \to C_1DC_3}$.

The decomposition of clique probability and utility potentials gives a number of advantages. One advantage is the option to exploit the distributive law of algebra when eliminating variables [20,21]. The distributive law can be exploited when eliminating C_3:

$$\begin{aligned}&P(C_1) \times (\psi(C_1) + \psi(C_1, D)) \\ &= \sum_{C_3} P(C_3) P(C_1 \mid C_3)(u_1(C_1, C_3) + \psi(D, C_3)) \\ &= \sum_{C_3} (P(C_3) P(C_1 \mid C_3) u_1(C_1, C_3)) + \sum_{C_3} (P(C_3) P(C_1 \mid C_3) \psi(D, C_3)).\end{aligned}$$

This approach supports exploitation of probabilistic barren variables and the decomposition of utility potentials reducing the number of calculation operations.

5 Conclusion

This paper has introduced Simple Propagation as a method for solving influence diagrams with discrete variables exactly and with optimality. The main advantage of Simple Propagation over other methods for finding an optimal strategy

and computing the probability of future decisions under an optimal strategy is its simplicity through the absence of the need for graph theoretical considerations to identify the potentials relevant for a message.

The paper has extended the *one-in, one-out*-principle to cope with decision variables and utility functions to solve influence diagrams. It has also illustrated and described properties of solving influence diagrams with Simple Propagation.

Future work includes considering other algorithms than VE as the variable elimination algorithm as well as solving limited-memory and mixed influence diagrams using Simple Propagation. It also includes considered other computational structures than the strong junction tree as well as any-time approximation.

References

1. Howard, R.A., Matheson, J.E.: Influence diagrams. In: Readings in Decision Analysis, pp. 763–771. Strategic Decisions Group, Menlo Park (1981)
2. Kjærulff, U.B., Madsen, A.L.: Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis, 2nd edn. Springer, New York (2013). https://doi.org/10.1007/978-0-387-74101-7
3. Koller, D., Friedman, N.: Probabilistic Graphical Models — Principles and Techniques. MIT Press, Cambridge (2009)
4. Lauritzen, S.L., Nilsson, D.: Representing and solving decision problems with limited information. Manage. Sci. **47**, 1238–1251 (2001)
5. Jensen, F.V., Vomlelova, M.: Unconstrained influence diagrams. In: Proceedings of Uncertainty in Artificial Intelligence Conference, pp. 234–241 (2002)
6. Madsen, A.L., Jensen, F.: Mixed influence diagrams. In: Nielsen, T.D., Zhang, N.L. (eds.) ECSQARU 2003. LNCS (LNAI), vol. 2711, pp. 208–219. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45062-7_17
7. Nilsson, D., Jensen, F.V.: Probabilities of future decisions. In: Bouchon-Meunier, B., Yager, R.R., Zadeh, L.A. (eds.) Information, Uncertainty and Fusion. The Springer International Series in Engineering and Computer Science, vol. 516, pp. 161–171. Springer, Boston (2000). https://doi.org/10.1007/978-1-4615-5209-3_12
8. Olmsted, S.M.: On representing and solving decision problems. Ph.D. thesis, Department of Engineering-Economic Systems, Stanford University, Stanford, CA (1983)
9. Shachter, R.D.: Evaluating influence diagrams. Oper. Res. **34**(6), 871–882 (1986)
10. Shachter, R.D., Peot, M.A.: Decision making using probabilistic inference methods. In: Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence, pp. 276–283. Morgan Kaufmann Publishers, San Mateo (1992)
11. Shenoy, P.P.: Valuation-based systems for Bayesian decision analysis. Oper. Res. **40**(3), 463–484 (1992)
12. Jensen, F., Jensen, F.V., Dittmer, S.: From influence diagrams to junction trees. In: Proceedings of Uncertainty in Artificial Intelligence Conference, pp. 367–373. Morgan Kaufmann Publishers, San Francisco (1994)
13. Madsen, A.L., Jensen, F.V.: Lazy evaluation of symmetric Bayesian decision problems. In: Proceedings of Uncertainty in Artificial Intelligence Conference, pp. 382–390 (1999)
14. Cooper, G.F.: A method for using belief networks as influence diagrams. In: Proceedings of Uncertainty in Artificial Intelligence Conference, pp. 55–63 (1988)

15. Yang, M., Zhou, L., Ruan, H.: The methods for solving influence diagrams: a review. In: International Conference on Information Technology and Applications, pp. 427–431 (2013)
16. Jensen, F.V., Nielsen, T.D.: Probabilistic decision graphs for optimization under uncertainty. 4OR-Q J. Oper. Res. **9**(1), 1–28 (2011)
17. Lee, J., Ihler, A., Dechter, R.: Join graph decomposition bounds for influence diagrams. In: Proceedings of Uncertainty in Artificial Intelligence Conference, pp. 1053–1062 (2018)
18. Yet, B., Neil, M., Fenton, N., Constantinou, A., Demetiev, E.: An improved method for solving hybrid influence diagrams. IJAR **95**, 93–112 (2018)
19. Butz, C.J., de Oliveira, J.S., dos Santos, A.E., Madsen, A.L.: Bayesian network inference with simple propagation. In: Proceedings of Florida Artificial Intelligence Research Society Conference, pp. 650–655 (2016)
20. Madsen, A.L., Jensen, F.V.: Lazy propagation: a junction tree inference algorithm based on lazy evaluation. Artif. Intell. **113**(1–2), 203–245 (1999)
21. Cabañas, R., Cano, A., Gómez-Olmedo, M., Madsen, A.L.: On SPI-lazy evaluation of influence diagrams. In: van der Gaag, L.C., Feelders, A.J. (eds.) PGM 2014. LNCS (LNAI), vol. 8754, pp. 97–112. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11433-0_7
22. Madsen, A.L., Nilsson, D.: Solving influence diagrams using HUGIN, Shafer-Shenoy and lazy propagation. In: Proceedings of Uncertainty in Artificial Intelligence Conference, pp. 337–345 (2001)
23. Nielsen, T.D., Jensen, F.V.: Welldefined decision scenarios. In: Proceedings of Uncertainty in Artificial Intelligence Conference, pp. 502–511 (1999)
24. Nielsen, T.D.: Decomposition of influence diagrams. In: Benferhat, S., Besnard, P. (eds.) ECSQARU 2001. LNCS (LNAI), vol. 2143, pp. 144–155. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44652-4_14