LIMIDs of Decision Problems*

Steffen L. Lauritzen
Aalborg University

Dennis Nilsson
Aalborg University†

May 29, 2000

Abstract

We introduce the notion of **LI**mited **M**emory **I**nfluence **D**iagram (LIMID) to describe multi-stage decision problems where the traditional assumption of no forgetting is relaxed. This can be relevant in situations with multiple decision makers or when decisions must be prescribed under memory constraints, such as e.g. in partially observed Markov decision processes (POMDPs).

We give an algorithm for improving any given strategy by local computation of single policy updates. We investigate conditions for the resulting strategy to be optimal. As a consequence we also obtain an efficient algorithm for solving traditional influence diagrams.

Key words: Influence diagram; junction tree; local computation; message passing; optimal strategies; partially observed Markov decision process, POMDP, single policy updating.

1 Introduction

This article is concerned with finding optimal strategies in multi-stage decision problems involving uncertainty. In accordance with classical decision theory (von Neumann and Morgenstern 1944; Raiffa and Schlaifer 1961) this will be done by attempting to maximize the expected utility.

Influence diagrams (Howard and Matheson 1984; Shachter 1986; Smith 1989; Oliver and Smith 1990) are compact representations of such decision problems, and local computation algorithms for solving these have been developed, for example, by Olmsted (1983), Shachter (1986), Tatman and Shachter (1990), Shenoy (1992), Shachter and Peot (1992), Shachter and Ndilikilikesha (1993), Ndilikilikesha (1994), Cowell (1994), Jensen *et al.* (1994) and Zhang (1998), see also Chapter 8 in Cowell *et al.* (1999).

---

*This is a modified version of Research Report R-99-2024.

†Department of Mathematical Sciences, Aalborg University, Fredrik Bajers Vej 7E, DK-9220 Aalborg, Denmark.

The general scenario for a multi-stage decision problem described by an influence diagram is that decisions $d_1, \ldots, d_k$ are to be taken sequentially and at each stage, values of some random variables are revealed. Decisions affect the distribution of unrevealed random variables in a specified way. The total utility is obtained by adding specific components, each depending on subsets of random variables and decisions taken. The random variables are related in a diagram, typically that of a Bayesian network or similar type of probabilistic network (Pearl 1986b; Jensen 1996; Cowell *et al.* 1999).

In the present article we relax the standard assumption in an influence diagram of 'no forgetting', i.e. that values of observed variables and decisions that have been taken are remembered at all later times. We denote these more general diagrams by LIMIDs (**LI**mited **M**emory **I**nfluence **D**iagrams). The study of such diagrams is motivated in a number of ways, as detailed below.

POMDPs are partially observable Markov decision processes (Åstrøm 1965). They can be seen as special types of influence diagrams that develop over time. As opposed to fully observed Markov decision processes (Howard 1960; Puterman 1994), the complexity of the local computation algorithms as applied to POMDPs grows quickly with time and fail therefore to provide the optimal solutions desired. Formulating finite POMDPs within LIMIDs allows handling explicit memory constraints. The price to be paid is that solutions provided are not always globally optimal, but satisfy a less restrictive, local optimality criterion.

Example 1 (PIGS) For illustration we consider the following example. Although the example is fictitious, more realistic variants of the example have served as motivation for the theoretical developments in this paper.

> A pig breeder is growing pigs for a period of four months and subsequently selling them. During this period the pig may or may not develop a certain disease. If the pig has the disease at the time when it must be sold, the pig must be sold for slaughtering and its expected market price is then 300 DKK (Danish kroner). If it is disease free, its expected market price as a breeding animal is 1000 DKK.
>
> Once a month, a veterinary doctor sees the pig and makes a test for presence of the disease. If the pig is ill, the test will indicate this with probability .80, and if the pig is healthy, the test will indicate this with probability .90.

> At each monthly visit, the doctor may or may not treat the pig for the disease by injecting a certain drug. The cost of an injection is 100 DKK.
>
> A pig has the disease in the first month with probability .10. A healthy pig develops the disease in the subsequent month with probability .20 without injection, whereas a healthy and treated pig develops the disease with probability .10, so the injection has some preventive effect.
>
> An untreated pig which is unhealthy will remain so in the subsequent month with probability .90, whereas the similar probability is .50 for an unhealthy pig which is treated. Thus, although spontaneous cure is possible, treatment is beneficial on average.

The story now continues in two versions. In the traditional influence diagram version, the pig breeder will at all times know whether the pig has been treated earlier and also the previous test results. This story corresponds to a (finite) POMDP. We will refer to this story as the ID version of the decision problem. If we extend the story to continue for many months or to have weekly or daily examinations with potential injections associated, the complexity of finding an optimal treatment strategy becomes forbidding.

In the LIMID version of the story, the pig breeder does not keep individual records for his pigs and has to make his decision knowing only the test result for the given month and the age of the pig. The memory has been limited to the extreme of only remembering the present. □

Situations in which multiple decision makers are involved lend themselves naturally to situations with limited memory. One decision maker can only communicate partially what has previously happened, and therefore the decision makers may not have the full history available when a given decision must be taken. Also, multiple decision makers can sometimes make their decisions in parallel, without sharing the same information. For decisions taken in daily life, it is quite realistic that an individual cannot fully remember what has happened at all previous times and still, decisions must be taken. Thus, in many ways, LIMIDs describe more realistic decision scenarios than traditional influence diagrams.

The complexity of finding fully optimal strategies within LIMIDs is in general prohibitive. Instead we describe procedures for Single Policy Updating that are guaranteed to lead to strategies which are locally optimal, in the sense that no single policy modification can increase the expected utility of the strategy. And we show how to calculate this locally optimal strategy by message passing in a suitable junction tree. This computation is similar to the local computation algorithms described above, although it differs in important ways.

We also establish general conditions for local optimal strategies to be globally optimal and provide algorithms for identifying such cases and the corresponding optimal strategies.

A careful analysis of traditional influence diagrams reveals that although memory is assumed to be comprehensive, it is not always necessary to record the full past when a decision has to be taken (Shachter 1998, 1999; Nielsen and Jensen 1999). Taking this aspect to its consequence and combining it with the algorithms provided for LIMIDs, we obtain algorithms for solving traditional influence diagrams that dominate existing algorithms in terms of use of space and computational time. In Section 4.7 we show how to transform an influence diagram to a LIMID which has the property that a locally optimal solution obtained by Single Policy Updating gives a fully optimal strategy for the original influence diagram.

2 Describing LIMIDs

2.1 Diagrams of LIMIDs

LIMIDs are represented by directed acyclic graphs (DAGs) with three types of nodes. *Chance nodes*, displayed as circles, represent random variables. *Decision nodes*, displayed as squares, correspond to alternative choices available to the decision maker. Finally *value nodes*, displayed as diamonds, represent local utility functions.

A node n_2 is called a *child* of a node n_1 if there is an arc from n_1 to n_2. In this case n_1 is called a *parent* of n_2. The set of parents of a node n is denoted by $\mathrm{pa}(n)$. The *family* of n, denoted by $\mathrm{fa}(n)$, is defined as $\mathrm{fa}(n) = \mathrm{pa}(n) \cup \{n\}$. If there is a directed path from n_1 to n_2, n_2 is said to be a *descendant* of n_1 and n_1 is an *ancestor* of n_2. The set of descendants and the set of ancestors of a node n is denoted $\mathrm{de}(n)$ and $\mathrm{an}(n)$ respectively.

The arcs in a LIMID have a different meaning depending of the type of node they go into. If the chance node r_1 (connoting *r*andom variable) is a parent for chance node r_2, it indicates that the distribution of (the random variable) r_2 is specified conditionally on the value of r_1. A decision node d is a parent of chance node r if the distribution of r can depend on the choice of alternative for the decision d. A decision node d_1 is a parent of decision node d_2 if the choice of alternative for the decision d_1 is known to the decision maker when decision d_2 is taken and may influence that decision. When chance node r is a parent of a decision node d it indicates that the value of r will be known when decision d is taken and might influence that

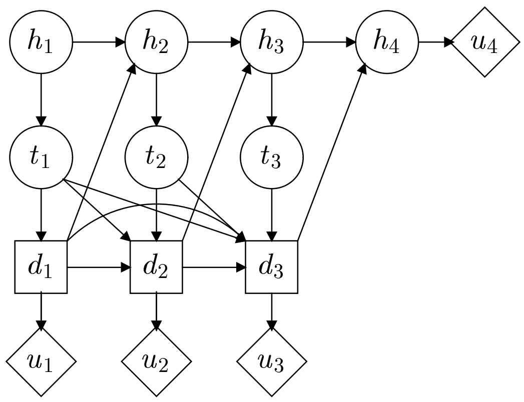


Figure 1: LIMID representation of the ID version of PIGS. The full previous treatment and test history is available when decisions must be taken.

decision. So arcs into decision nodes have two implications: time precedence and influence.

Finally arcs into value nodes represent the decision maker's (expected) utility given the states of its parents. We assume that the total utility of a configuration of the chance and decision variables can be represented as the sum of the local utility functions associated with the value nodes. Value nodes cannot have children.

Example 2 (Diagrams for PIGS) To represent the ID version of PIGS by a LIMID, we let h_i, $(i = 1, \ldots, 4)$ denote the (chance) variables which indicate whether the pig is *healthy* or *unhealthy* in the ith month and t_i, $(i = 1, 2, 3)$ represent the corresponding test results, which are said to be *positive* if they indicate presence of the disease, and otherwise *negative*. The nodes d_i, $(i = 1, 2, 3)$ correspond to the decisions *treat* or *leave*, the latter implying that no injection is made. The utility nodes u_1, u_2, u_3 represent the potential injection costs, whereas u_4 is the (expected) market price of the pig as determined by its health at the fourth month. The corresponding diagram is displayed in Figure 1. In contrast to the convention used by e.g. Jensen *et al.* (1994), there are links from all predecessors of decision nodes to make the assumption of 'no forgetting' explicit.

In the LIMID version of the story, the pig breeder does not keep individual records for his pigs and has to make his decision knowing only the test result for the given month. The corresponding diagram is displayed in Figure 2. Note that if the diagram of the LIMID in Figure 2 is (mis)read as a traditional influence diagram, this will be equivalent to the diagram in Figure 1. □

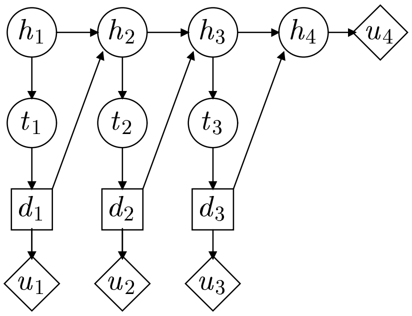


Figure 2: LIMID version of the PIGS decision problem. Only the current test result is available when decisions must be taken.

Formally speaking, a LIMID can be viewed as a special type of Bayesian network, where the state of each decision variable is not determined probabilistically by its parents, but rather is imposed from the decision maker to meet an optimization objective, and where the variables at the value nodes are completely determined from its parent configurations. In this perspective, the decision task can be regarded as one of assigning functions to a subset of the variables: each such function is a prescription of alternatives for a decision node to possible instantiations of its parents.

Note that a LIMID differs from a traditional influence diagram representation of a decision problem in two ways:

(i) The sequence in which decisions are to be taken is not specified other than through it being compatible with the partial order induced by the DAG, i.e. if d_2 is a descendant of d_1, the decision d_1 must be taken before d_2.

(ii) The parents of a decision node d represent exactly the variables whose values are known and taken into consideration when d is to be taken. In traditional influence diagrams, this relation is more complicated and varies somewhat between authors.

Thus, LIMIDs allow for multiple decision makers or decision maker(s) that may forget the history of a given decision problem.

2.2 Specifications for LIMIDs

For a given LIMID, we denote the sets of decision and chance nodes by Δ and Γ respectively and let $V = \Delta \cup \Gamma$. The set of value nodes is denoted by Υ.

Each node $n \in V$ is associated with a variable which takes values in a finite set $\mathcal{X}_n$ of states. Usually we use the label n interchangeably with the variable associated with n. For $W \subseteq V$ we write $\mathcal{X}_W = \times_{n \in W} \mathcal{X}_n$. Typical elements in $\mathcal{X}_W$ are called *configurations* and denoted by $x_W = (x_n)_{n \in W}$ and so on.

For every chance node r there is a non-negative real function p_r on $\mathcal{X}_r \times \mathcal{X}_{\mathrm{pa}(r)}$ such that for each configuration of $\mathrm{pa}(r)$, p_r adds to 1 when summing over the configurations in $\mathcal{X}_r$, i.e.

$$p_r(x_r \mid x_{\mathrm{pa}(r)}) \geq 0, \quad \sum_{x_r} p_r(x_r \mid x_{\mathrm{pa}(r)}) = 1. \tag{1}$$

The terms p_r are not necessarily true conditional distributions but rather a family of distributions parametrized by the states of $\mathrm{pa}(r)$, since the distribution of $\mathrm{pa}(r)$ is generally not specified. If the node r has no parents then p_r represents the marginal distribution of r.

Each value node $u \in \Upsilon$ has an associated real function U_u on $\mathcal{X}_{\mathrm{pa}(u)}$ that represents an additive component of the utility as a function of the configurations of its parents.

2.3 Policies and strategies

Suppose we are given a LIMID with decision nodes Δ and chance nodes Γ. A *pure policy* for $d \in \Delta$ prescribes an alternative in $\mathcal{X}_d$ for each possible configuration of its parents $\mathrm{pa}(d)$. To allow for the possibility of randomizing between actions we consider more general policies represented by functions δ_d on $\mathcal{X}_d \times \mathcal{X}_{\mathrm{pa}(d)}$ which indicate a probability distribution over alternative choices for each possible configuration of $\mathrm{pa}(d)$. They must also satisfy the relation (1) above.

A *strategy* for a LIMID is a set $q = \{\delta_d : d \in \Delta\}$ of policies, one for each decision node. A *pure strategy* is a set $q = \{\delta_d : d \in \Delta\}$ of pure policies, one for each decision node. If a strategy is not pure it is called *random*.

2.4 Formulation of the problem

When a strategy q (pure or random) has been specified, the joint distribution of all the variables in V is determined as

$$f_q = \prod_{r \in \Gamma} p_r \prod_{d \in \Delta} \delta_d. \tag{2}$$

where $q = \{\delta_d : d \in \Delta\}$. For all such q, p_r represents the appropriate conditional distribution calculated with respect to f_q.

We are searching for a strategy q which maximizes the expected total utility under the probability distribution induced by q.

If U is the total utility, i.e. $U = \sum_{u \in \Upsilon} U_u$, then the expected utility of q is given by

$$\mathrm{EU}(q) = \mathbf{E}_q(U) = \sum_x f_q(x)U(x) = \sum_x f_q(x) \left\{ \sum_{u \in \Upsilon} U_u(x_{\mathrm{pa}(u)}) \right\}$$

and we are searching for a global maximum strategy defined as follows.

Definition 1 A *global maximum strategy* $\hat{q}$ is a strategy which satisfies

$$\mathrm{EU}(\hat{q}) \geq \mathrm{EU}(q) \text{ for all strategies } q.$$

Given any strategy $q = \{\delta_d : d \in \Delta\}$ and any $d_0 \in \Delta$, we let

$$q_{-d_0} = \{\delta_d : d \in \Delta \setminus \{d_0\}\}$$

be the partially specified strategy obtained by retracting the policy at d_0 and define for any policy δ'_{d_0}

$$\delta'_{d_0} * q = \{\delta'_{d_0}\} \cup q_{-d_0}.$$

So $\delta'_{d_0} * q$ denotes the strategy obtained from q by replacing δ_{d_0} with δ'_{d_0} and leaving all other policies as in q. Now we have

Definition 2 A *local maximum policy* for a strategy q at d, is a policy $\tilde{\delta}_d$ which satisfies

$$\mathrm{EU}\left(\tilde{\delta}_d * q\right) = \sup_{\delta'_d} \mathrm{EU}\left(\delta'_d * q\right).$$

Definition 3 A strategy $\tilde{q}$ is said to be a *local maximum strategy* if all its policies are local maximum policies, i.e. if for all $d \in \Delta$ and all policies δ_d we have

$$\mathrm{EU}(\tilde{q}) \geq \mathrm{EU}(\delta_d * \tilde{q}).$$

In other words, a strategy is a local maximum strategy if and only if the expected utility does not increase by changing only one of its policies.

For later use, we establish the following lemma, where $f_{q_{-d}}$ is defined through (2) and the partial policy obtained from q by retracting δ_d. Note that $f_{q_{-d}}$ is not a joint probability distribution.

Lemma 1 *A policy $\tilde{\delta}_d$ is a local maximum policy for a strategy q at d if and only if for all $x_{\mathrm{fa}(d)}$ with $\tilde{\delta}_d(x_d \,|\, x_{\mathrm{pa}(d)}) > 0$ we have*

$$x_d = \arg\max_{z_d} \sum_{x_{V \setminus \mathrm{fa}(d)}} f_{q_{-d}}(x_{V \setminus d}, z_d) U(x_{V \setminus d}, z_d). \tag{3}$$

Proof: Let

$$\hat{U}(x_{\mathrm{pa}(d)}) = \max_{z_d} \sum_{x_{V\setminus \mathrm{fa}(d)}} f_{q_{-d}}(x_{V\setminus d}, z_d) U(x_{V\setminus d}, z_d).$$

For all policies δ_d we now obtain

$$\begin{aligned} \mathrm{EU}(\delta_d * q) &= \sum_x \delta_d(x) f_{q_{-d}}(x) U(x) \\ &= \sum_{x_{\mathrm{fa}(d)}} \delta_d(x_d \,|\, x_{\mathrm{pa}(d)}) \sum_{x_{V\setminus \mathrm{fa}(d)}} f_{q_{-d}}(x) U(x) \\ &\leq \sum_{x_{\mathrm{fa}(d)}} \delta_d(x_d \,|\, x_{\mathrm{pa}(d)}) \hat{U}(x_{\mathrm{pa}(d)}) \end{aligned}$$

As the inequality is sharp in and only if δ_d does not satisfy (3) for all x with $\delta_d(x_d \,|\, x_{\mathrm{pa}(d)}) > 0$, we obtain the desired result. □

2.5 Single Policy Updating

This subsection describes *Single Policy Updating,* an iterative procedure for improving strategies within LIMIDs.

Single Policy Updating starts with an initial strategy q^0 and updates the policies for the decisions individually in a random or systematic order.

Assume that the current strategy is q^l and that the policy for d_0 is to be updated. Then the next strategy q^{l+1} only differs from q^l on the policy for d_0 and it is generated by finding a local maximum policy $\delta_{d_0}^{l+1}$ for q^l at d_0 and letting

$$q^{l+1} = \delta_{d_0}^{l+1} * q^l.$$

When all the policies have been updated we say that one *cycle* has been performed. The algorithm stops if the expected utility of strategies generated in two successive cycles is unaltered. A few comments are in place here:

- The initial strategy q^0 may be random and it will indeed typically be advantageous to choose it as such.
- Section 3 is dedicated to an efficient algorithm that finds a local maximum policy in each step.
- There is always a pure local maximum policy $\delta_{d_0}^{l+1}$; however it may not be unique.

If we always choose a pure local maximum policy in each step, the algorithm eventually reaches a local maximum strategy at which no progress is made. This follows from the fact that there is only a finite number of pure strategies and the expected utility necessarily increases at each cycle until a local maximum strategy is obtained. These observations are stated formally below.

Iterative improvement *Single Policy Updating is an iterative improvement algorithm: after each cycle, the expected utility of the current strategy has increased or is unaltered. In the latter case, the algorithm has reached a local maximum strategy.*

Convergence *Single Policy Updating converges to a local maximum strategy if we choose always to take a pure policy in each updating step. In this case the algorithm converges in a finite number of cycles.*

We shall later discuss how to choose initial strategies, updating sequences, and give conditions ensuring local maximum strategies to be global maximum strategies.

3 Single Policy Updating by message passing

3.1 LIMID-potentials and their operations

In our local computation algorithms we represent the quantitative elements of a LIMID through entities called LIMID-potentials, or just potentials for short. Each such potential has two parts as detailed below.

Definition 4 Let $W \subseteq V$. A *LIMID-potential* on W is a pair $\pi_W = (p_W, u_W)$ where

- p_W is a function $p_W : \mathcal{X}_W \to \mathcal{R}_0$;
- u_W is a function $u_W : \mathcal{X}_W \to \mathcal{R}$;

Thus a LIMID-potential consists of two parts: the first part p_W is called the *probability part*, and the second part u_W is called the *utility part*. We call the probability part *vacuous* if it is equal to unity, and the utility part is vacuous if it is identically equal to zero.

We identify two potentials $\pi_W^1 = (p_W^1, u_W^1)$ and $\pi_W^2 = (p_W^2, u_W^2)$ on W and write $\pi_W^1 = \pi_W^2$ if

- $p_W^1 = p_W^2$;
- $u_W^1(x_W) = u_W^2(x_W)$ whenever $p_W^1(x_W) = p_W^2(x_W) > 0$,

i.e. two potentials are considered equal if they have identical probability parts and their utility parts agree almost surely with respect to the probability parts.

To represent and evaluate the decision problem in terms of potentials, we define basic operations of *combination* and *marginalization* and show that they satisfy the axioms of Shenoy and Shafer (1990).

Definition 5 [**Combination**]
Let $\pi_{W_1} = (p_{W_1}, u_{W_1})$ and $\pi_{W_2} = (p_{W_2}, u_{W_2})$ be two potentials on W_1 and W_2 respectively. Then $\pi_{W_1} \otimes \pi_{W_2}$ denotes the potential on $W_1 \cup W_2$ given by

$$\pi_{W_1} \otimes \pi_{W_2} = (p_{W_1} p_{W_2}, u_{W_1} + u_{W_2})$$

The potential $\pi_{W_1} \otimes \pi_{W_2}$ is called the *combination of* π_{W_1} *and* π_{W_2}.

To define marginalization of potentials we first introduce the *sum-marginal* $\sum_{W \setminus W_1} \phi_W$ of a real-valued function ϕ_W on $\mathcal{X}_W$ for $W_1 \subseteq W$. This is simply the 'usual' marginal

$$\left(\sum_{W \setminus W_1} \phi_W \right) (x_{W_1}) = \sum_{y_W : y_{W_1} = x_{W_1}} \phi_W(y_W).$$

Then we define marginalization of potentials as follows:

Definition 6 [**Marginalization**]
Let $\pi_W = (p_W, u_W)$ be a potential on W, and let $W_1 \subseteq W$. Then $\pi_W^{\downarrow W_1}$ denotes the potential on W_1 given by

$$\pi_W^{\downarrow W_1} = \left(\sum_{W \setminus W_1} p_W \, , \, \frac{\sum_{W \setminus W_1} p_W u_W}{\sum_{W \setminus W_1} p_W} \right)$$

The potential $\pi_W^{\downarrow W_1}$ is called the *marginalization* of π_W onto W_1.

Here we have used the convention that $0/0 = 0$ which will be used throughout. In words, the marginalization of $\pi_W = (p_W, u_W)$ to W_1 is the potential on W_1 whose probability part is the sum-marginal of p_W on W_1 and whose utility part is the sum-marginal of the product of p_W and u_W on W_1.

The notion of potential and combination are similar to what is used in Shenoy (1992), Jensen *et al.* (1994) and Cowell *et al.* (1999). Marginalization is what these authors have termed sum-marginalization.

The first axiom amounts to combination satisfying the properties of a commutative semigroup, i.e. being associative and commutative:

Lemma 2 (Commutativity and associativity of combination)
Suppose π_{W_1}, π_{W_2}, and π_{W_3} are potentials. Then

$$\pi_{W_1} \otimes \pi_{W_2} = \pi_{W_2} \otimes \pi_{W_1} \text{ and } (\pi_{W_1} \otimes \pi_{W_2}) \otimes \pi_{W_3} = \pi_{W_1} \otimes (\pi_{W_2} \otimes \pi_{W_3})$$

Proof: This trivially follows from the definition of combination of potentials. □

The fundamental properties of marginalization corresponding to the last two axioms are established in the two lemmas below, where we have given proofs for completeness of exposition.

Lemma 3 (Consonance of marginalization)
Suppose π_W is a potential on W, and suppose $W \supseteq W_1 \supseteq W_2$. Then

$$(\pi_W^{\downarrow W_1})^{\downarrow W_2} = \pi_W^{\downarrow W_2}$$

Proof: Let $\pi_W = (p_W, u_W)$. The probability parts of the two potentials $(\pi_W^{\downarrow W_1})^{\downarrow W_2}$ and $\pi_W^{\downarrow W_2}$ are clearly the same.

Next we show that the two potentials have the same utility part. We first note that

$$\pi_W^{\downarrow W_1} = \left(\sum_{W \setminus W_1} p_W \; , \; \frac{\sum_{W \setminus W_1} p_W u_W}{\sum_{W \setminus W_1} p_W} \right)$$

So, the utility part of $(\pi_W^{\downarrow W_1})^{\downarrow W_2}$ is given by

$$\frac{\sum_{W_1 \setminus W_2} \left\{ \left(\sum_{W \setminus W_1} p_W \right) \left(\frac{\sum_{W \setminus W_1} p_W u_W}{\sum_{W \setminus W_1} p_W} \right) \right\}}{\sum_{W_1 \setminus W_2} \left(\sum_{W \setminus W_1} p_W \right)}$$

Since $(\sum_{W \setminus W_1} p_W)(x_{W_1}) = 0 \Rightarrow (\sum_{W \setminus W_1} p_W u_W)(x_{W_1}) = 0$ we can write the above expression as

$$\frac{\sum_{W \setminus W_2} p_W u_W}{\sum_{W \setminus W_2} p_W}.$$

which by definition is the utility part of $\pi_W^{\downarrow W_2}$. □

Lemma 4 (Distributivity and marginalization over combination)
Suppose π_{W_1} and π_{W_2} are potentials on W_1 and W_2 respectively. Then

$$(\pi_{W_1} \otimes \pi_{W_2})^{\downarrow W_1} = \pi_{W_1} \otimes \pi_{W_2}^{\downarrow W_1}$$

Proof: Let $\pi_{W_1} = (p_{W_1}, u_{W_1})$ and $\pi_{W_2} = (p_{W_2}, u_{W_2})$. Then

$$\begin{aligned}(\pi_{W_1} \otimes \pi_{W_2})^{\downarrow W_1} &= (p_{W_1}p_{W_2}, u_{W_1} + u_{W_2})^{\downarrow W_1}. \\ &= \left(\sum_{W_2 \setminus W_1} p_{W_1}p_{W_2}, \frac{\sum_{W_2 \setminus W_1} p_{W_1}p_{W_2}(u_{W_1} + u_{W_2})}{\sum_{W_2 \setminus W_1} p_{W_1}p_{W_2}} \right).\end{aligned}$$

Clearly the probability part for the potentials $\pi_{W_1} \otimes \pi_{W_2}^{\downarrow W_1}$ and $(\pi_{W_1} \otimes \pi_{W_2})^{\downarrow W_1}$ agree. So it only remains to be shown that the utility parts of the two potentials are the same whenever

$$\left(\sum_{W_2 \setminus W_1} p_{W_1}p_{W_2} \right)(x_{W_1}) > 0 \tag{4}$$

So, assume (4) holds. In this case, the utility part of $(\pi_{W_1} \otimes \pi_{W_2})^{\downarrow W_1}$ is

$$\frac{\sum_{W_2 \setminus W_1} p_{W_1}p_{W_2}u_{W_1}}{\sum_{W_2 \setminus W_1} p_{W_1}p_{W_2}} + \frac{\sum_{W_2 \setminus W_1} p_{W_1}p_{W_2}u_{W_2}}{\sum_{W_2 \setminus W_1} p_{W_1}p_{W_2}} = u_{W_1} + \frac{\sum_{W_2 \setminus W_1} p_{W_2}u_{W_2}}{\sum_{W_2 \setminus W_1} p_{W_2}}$$

and it follows that this is the same as the utility part of $\pi_{W_1} \otimes \pi_{W_2}^{\downarrow W_1}$. This completes the proof. □

Lemmas 2-4 together say that combination and marginalization satisfy the Shenoy-Shafer axioms, establishing correctness of the propagation scheme presented in Section 3.4.

3.2 From LIMIDs to junction trees

As mentioned, our algorithm proceeds by message passing in a suitable computational structure known as a junction tree. In the present subsection we describe how to construct this junction tree.

As for similar local computation algorithms the construction involves first a moralization process, in which an undirected graph is constructed, then a triangulation, where additional edges are added, and finally the organization of the cliques of the triangulated graph into a junction tree.

The transformation from the LIMID $\mathcal{L}$ to an undirected graph is made by first adding undirected edges between all nodes with a common child (including children that are value nodes). As value nodes do not have children, only edges between chance or decision nodes are added. Then we drop the directions on all arcs and finally remove all value nodes. The resulting 'moral' graph is denoted by $\mathcal{L}^m$.

Next, edges are added to the undirected graph $\mathcal{L}^m$ to form a triangulated graph $\bar{\mathcal{L}}^m$. This can be done in a number of ways, we refer to Cowell

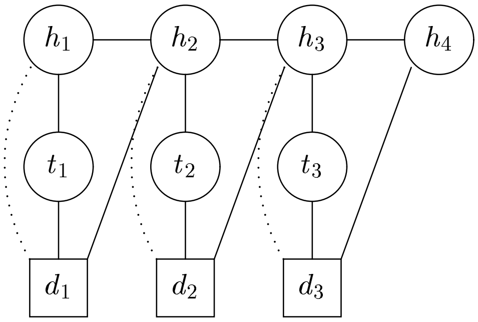


Figure 3: Moral graph of the LIMID in Figure 2. The moral graph is triangulated so no additional edges are needed.

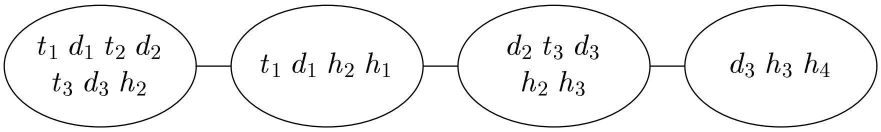


Figure 4: Junction tree for the ID version of PIGS. The elimination order used in the triangulation process was h_4 h_1 h_3 h_2 d_3 t_3 d_2 t_2 d_1 t_1. Thus the leftmost clique in the junction tree can be used as a strong root if the LIMID is considered as an ordinary ID.

et al. (1999) for details. It is important to note that, in contrast with the local computation methods described by Jensen *et al.* (1994) and others, the triangulation does not need to respect any specific partial or total ordering of the nodes, but the triangulation can simply be chosen to minimize the computational costs, for example as described in Kjærulff (1992).

Finally the cliques $\mathcal{C}$ of $\bar{\mathcal{L}}^m$ are organized into a *junction tree* $\mathcal{T}$, which is a tree having the property that for any $n \in V$, the collection of all cliques containing n correspond to a connected subtree of $\mathcal{T}$ or, equivalently, that the intersection $C_1 \cap C_2$ for any pair of cliques is contained in all other cliques on the path from C_1 to C_2. This can be done in a number of different ways (Cowell *et al.* 1999).

Example 3 (Junction trees for PIGS**)** The ID version of PIGS leads to the junction tree displayed in Figure 4. Note that the full set of decision variables and test variables are contained in a single clique. If the story is extended by more months, this clique will grow correspondingly.

The LIMID version of PIGS has the moral graph displayed in Figure 3. The moral graph of the LIMID version of PIGS is triangulated and a cor-

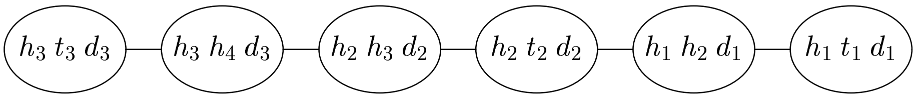


Figure 5: Junction tree for the LIMID version of PIGS.

responding junction tree is displayed in Figure 5. The maximal clique size does not increase with time, but stays constantly equal to three. □

3.3 Initialization

Suppose we are given a LIMID $\mathcal{L}$ and an initial strategy $q = \{\delta_d : d \in \Delta\}$. To initialize the junction tree $\mathcal{T}$ one first associates a vacuous potential to each clique $C \in \mathcal{C}$. Then for each chance node r in $\mathcal{L}$, p_r is multiplied onto the probability part of the potential of an arbitrary clique containing $\mathrm{fa}(r) = \{r\} \cup \mathrm{pa}(r)$. When this has been done, one takes each value node $u \in \Upsilon$ and adds U_u to the utility part of the potential of any clique containing $\mathrm{pa}(u)$. The moralization process has ensured the existence of such cliques.

In principle we now continue by multiplying initial policies for each decision node d onto the probability part of a clique containing $\mathrm{fa}(d)$. However, as we later want to retract and change these policies, we store them separately from the probability part and only perform the necessary multiplication during message passing as described below in Section 3.4.

In a 'lazy' version of the algorithm (Madsen and Jensen 1998), the individual factors and terms in the probability and utility parts would also be stored separately until needed for message passing.

Let $\pi_C = (p_C, u_C)$ be the potential on clique C after these operations have been performed. The *joint potential* is equal to the combination of all the clique potentials and satisfies

$$\pi_V = \otimes\{\pi_C : C \in \mathcal{C}\} = (p_V, u_V) = \left(\prod_{r \in \Gamma} p_r \prod_{d \in \Delta} \delta_d, \sum_{u \in \Upsilon} U_u \right) = (f_q, U). \quad (5)$$

3.4 Collect propagation

Let $\{\pi_C : C \in \mathcal{C}\}$ be a collection of potentials on the junction tree $\mathcal{T}$. Let $\pi_V = \otimes\{\pi_C : C \in \mathcal{C}\}$ and suppose we wish to find the marginal $\pi_V^{\downarrow R}$ for some clique $R \in \mathcal{C}$.

To achieve our purpose we direct all the edges in $\mathcal{T}$ towards the 'root-clique' R. Then each clique passes a message to its child after having received messages from all its other neighbours. The structure of a *message* $\pi_{C_1\to C_2}$ from clique C_1 to its neighbour C_2 is given by

$$\pi_{C_1\to C_2} = \left(\pi_{C_1} \otimes \left(\otimes_{C\in \mathrm{ne}(C_1)\setminus\{C_2\}} \pi_{C\to C_1}\right)\right)^{\downarrow C_2},$$

where $\mathrm{ne}(C_1)$ are the neighbours of C_1 in $\mathcal{T}$ and $\pi_{C\to C_1}$ is the message from C to C_1.

In words, the message which C_1 sends to its neighbour C_2 is the combination of all the messages that C_1 receives from its other neighbours together with its own potential, suitably marginalized.

The result below now follows from the fact that the two mappings, combination ($\otimes$) and marginalization ($\downarrow$) obey the Shafer–Shenoy axioms, as shown in Lemma 2-4. We have no interest in verifying the slightly more restrictive conditions of Lauritzen and Jensen (1997), as only collect operations are used subsequently.

Theorem 1 *Suppose we start with a joint potential π_V on a junction tree $\mathcal{T}$ with cliques $\mathcal{C}$, and pass messages towards a 'root-clique' $R \in \mathcal{C}$ as described above. When R has received a message from each of its neighbours, the combination of all messages with its own potential is equal to the R-marginal of the joint potential π_V:*

$$\pi_V^{\downarrow R} = (\otimes_{C\in\mathcal{C}}\pi_C)^{\downarrow R} = \pi_R \otimes \left(\otimes_{C\in \mathrm{ne}(R)} \pi_{C\to R}\right).$$

3.5 Local optimization

This section is concerned with showing how to find a local maximum policy during Single Policy Updating by an optimization process applied to a suitable clique marginal of the joint potential on $\mathcal{T}$.

Definition 7 Let $\pi_W = (p_W, u_W)$ be a potential. The *contraction* of π_W, denoted $\mathrm{cont}(\pi_W)$, is defined as the real function on $\mathcal{X}_W$ given as

$$\mathrm{cont}(\pi_W) = p_W u_W$$

We shall need the following theorem.

Theorem 2 *For a potential $\pi_W = (p_W, u_W)$ on W and $W_1 \subseteq W$ we have*

$$\mathrm{cont}(\pi_W^{\downarrow W_1}) = \sum_{W\setminus W_1} \mathrm{cont}(\pi_W).$$

Proof: By definition, the marginal $\pi_W^{\downarrow W_1}$ is given as

$$\pi_W^{\downarrow W_1} = \left(\sum_{W \setminus W_1} p_W, \frac{\sum_{W \setminus W_1} p_W u_W}{\sum_{W \setminus W_1} p_W} \right).$$

Note that in the utility part of $\pi_W^{\downarrow W_1}$ we have for all x_{W_1} that the numerator is zero whenever the denominator is zero. This allows us to write

$$\begin{aligned} \mathrm{cont}(\pi_W^{\downarrow W_1}) &= \sum_{W \setminus W_1} p_W u_W \\ &= \sum_{W \setminus W_1} \mathrm{cont}(\pi_W), \end{aligned}$$

which was to be proved. $\square$

Recall from (2) that f_q denotes the probability distribution obtained by effectuating the strategy q. Note that with the joint potential π_V defined by (5) we have

$$f_q(x)U(x) = \mathrm{cont}(\pi_V),$$

where $U(x)$ as usual denotes the total utility. We then further have:

Corollary 1 *Let the joint potential π_V on the junction tree be given by*

$$\pi_V = \left(\prod_{r \in \Gamma} p_r \prod_{d \in \Delta} \delta_d, U \right) = (f_q, U),$$

where $q = \{\delta_d : d \in \Delta\}$ is a strategy. Then the expected utility of q is given as

$$\mathrm{EU}(q) = \mathrm{cont}(\pi_V^{\downarrow \emptyset}).$$

Proof: By definition

$$\mathrm{EU}(q) = \sum_x f_q(x)U(x) = \sum_V \mathrm{cont}(\pi_V) = \mathrm{cont}(\pi_V^{\downarrow \emptyset}),$$

where the last equality follows from Theorem 2. $\square$

Suppose we wish to compute a local maximum policy for q at d, and suppose the policy for d is assigned to clique $R \in \mathcal{C}$. Let $\tilde{\pi}_R$ be the potential on R obtained by retracting the policy for d from π_R. Then by Theorem 2

$$\begin{aligned} \sum_{x_{V \setminus \mathrm{fa}(d)}} f_{q_{-d}}(x)U(x) &= \sum_{V \setminus \mathrm{fa}(d)} \mathrm{cont}\left(\tilde{\pi}_R \otimes (\otimes_{C: C \neq R} \pi_C)\right) \\ &= \mathrm{cont}\left((\tilde{\pi}_R \otimes (\otimes_{C: C \neq R} \pi_C))^{\downarrow \mathrm{fa}(d)} \right). \end{aligned}$$

Consequently, according to Lemma 1, a local maximum policy at d for the strategy q can be found by carrying out the following steps:

1. **Retract:** Retract the policy of d from the potential on R to obtain $\tilde{\pi}_R$.

2. **Collect:** Collect to R to obtain $\pi_R^* = (\tilde{\pi}_R \otimes (\otimes_{C:C \neq R} \pi_C))^{\downarrow R}$ as in Theorem 1.

3. **Marginalize:** Compute $\pi_{\mathrm{fa}(d)}^* = (\pi_R^*)^{\downarrow \mathrm{fa}(d)}$.

4. **Contract:** Compute the contraction $c_{\mathrm{fa}(d)}$ of $\pi_{\mathrm{fa}(d)}^*$.

5. **Optimize:** Define $\tilde{\delta}_d(x_{\mathrm{pa}(d)})$ for all $x_{\mathrm{pa}(d)}$ as the distribution degenerate at a point x_d^* satisfying (see Lemma 1 and Theorem 2)

$$x_d^* = \arg\max_{x_d} c_{\mathrm{fa}(d)}(x_d, x_{\mathrm{pa}(d)}).$$

6. **Replace:** Add $\tilde{\delta}_d$ to the potential on R to obtain $\tilde{\pi}_R^*$.

Note that all the computations apart from the second step are local in the root clique R. Furthermore, after the above steps are carried out the joint potential on the junction tree is equal to

$$\tilde{\pi}_V^* = \tilde{\pi}_R^* \otimes (\otimes_{C:C \neq R} \pi_C) = \left(f_{\tilde{\delta}_d * q}, U\right),$$

where $\tilde{\delta}_d$ is a local maximum policy for q at d.

3.6 Partial collect propagation

The message passing algorithm presented in Section 3.4 can be used to update the policies during Single Policy Updating: Suppose we begin with a potential π_V on the junction tree $\mathcal{T}$ and want to update the policies in the order $d_1, \ldots, d_k$. Suppose the policy for d_i is assigned to clique R_i. Thus $R_i \supseteq \mathrm{fa}(d_i)$. As an initial step messages are collected towards R_1. Then we find a local maximum policy for π_V at d_1 and the obtained policy replaces the old policy for d_1 in R_1.

When the next policy needs to be updated we could apply the same algorithm; however each time a local maximum policy is to be computed we must repeat the algorithm using a new clique as root-clique. This usually involves a great deal of duplication. The propagation scheme presented below provides one way to eliminate much of this duplication. In this scheme messages are only passed from the previous root towards the new root.

We now explain the partial propagation scheme in detail: the messages are passed via a mailbox placed on each edge of the junction tree. If the edge connects C_1 and C_2, the mailbox can hold messages in the form of potentials on $C_1 \cap C_2$.

Updating the policy for d_1 is made exactly as in the procedure described above. As new root we choose the clique R_2. However, instead of collecting messages towards the new root as described above, we only pass messages along the (unique) path from the old root R_1 to R_2. This is done by first emptying the mailboxes on the path and then passing the messages. Note that after the passage of these messages, R_2 has received messages from all its neighbours. Then a local maximum policy at d_2 can be computed and the potential on R_2 is changed appropriately. Next we choose R_3 as new root and pass messages on the path from R_2 to the new root in a similar manner.

Proceeding in this way we eventually have updated all the policies. When several cycles are to be performed we need not start each new cycle by collecting messages to R_1; only messages from the previous root R_k to the new root R_1 are needed.

4 Optimal strategies within LIMIDs

This section is concerned with developing methods for reducing LIMIDs to obtain lower complexity of computations during Single Policy Updating and identifying conditions for LIMIDs which ensure that local maximum strategies obtained by Single Policy Updating are also global maximum strategies. In particular this leads to efficient algorithms for solving decision problems traditionally represented by influence diagrams.

4.1 Separation within LIMIDs

As for probabilistic expert systems and influence diagrams, the key to simplification of computational problems for LIMIDs is the notion of *irrelevance* as expressed through the d-separation criterion given by Pearl (1986a) with a formal treatment in Verma and Pearl (1990). In our formulation it is using the notion of a *trail* in a DAG $\mathcal{D}$ which is a sequence of nodes that are connected by arcs, not necessarily following the directions of these arcs.

A trail τ from the node a to the node b in a directed, acyclic graph $\mathcal{D}$ is said to be *blocked* by S if it contains a node $n \in \tau$ such that either

- $n \in S$ and arcs of τ do not meet head-to-head at n, or

- n and all its descendants are not in S, and arcs of τ meet head-to-head at n.

A trail that is not blocked by S is said to be *active.* Two subsets A and B of nodes are said to be *d-separated* by S if all trails from A to B are blocked by S.

In the following we use the symbolic expression $A \perp_{\mathcal{L}} B \,|\, S$ to denote that A and B are d-separated by S in the DAG formed by all nodes of the LIMID $\mathcal{L}$, i.e. including the utility nodes.

It was shown by Lauritzen *et al.* (1990) that an alternative criterion could be used to verify d-separation:

Proposition 1 *Let A, B and S be disjoint subsets of nodes in $\mathcal{L}$. Then $A \perp_{\mathcal{L}} B \mid S$ if and only if S separates A from B in $(\mathcal{L}_{\mathrm{An}(A\cup B\cup S)})^m$.*

Here $(\mathcal{L}_{\mathrm{An}(A\cup B\cup S)})^m$ denotes the moral graph obtained by first restricting $\mathcal{L}$ to involve only nodes in $A\cup B\cup S$ and their ancestors. It depends on the specific context and purpose whether the trailwise criterion, or the criterion above is easiest to use.

It was shown in Verma and Pearl (1990) that d-separation satisfies the so-called *graphoid axioms:*

Lemma 5 *For any fixed LIMID $\mathcal{L}$, the relation $\perp_{\mathcal{L}}$ satisfies for any disjoint subsets A, B, C, and D of nodes of $\mathcal{L}$*

(C1) *if $A \perp_{\mathcal{L}} B \,|\, C$ then $B \perp_{\mathcal{L}} A \,|\, C$;*

(C2) *if $A \perp_{\mathcal{L}} B \,|\, C$ and $D \subseteq B$, then $A \perp_{\mathcal{L}} D \,|\, C$;*

(C3) *if $A \perp_{\mathcal{L}} B \,|\, C$ and $D \subseteq B$, then $A \perp_{\mathcal{L}} B \,|\, (C \cup D)$;*

(C4) *if $A \perp_{\mathcal{L}} B \,|\, C$ and $A \perp_{\mathcal{L}} D \,|\, (B \cup C)$, then $A \perp_{\mathcal{L}} (B \cup D) \,|\, C$.*

(C5) *if $A \perp_{\mathcal{L}} B \,|\, (C \cup D)$ and $A \perp_{\mathcal{L}} C \,|\, (B \cup D)$ then $A \perp_{\mathcal{L}} (B \cup C) \,|\, D$.*

Any probability distribution P which factorizes over a DAG satisfies the so-called *global directed Markov property*

$$A \perp_{\mathcal{L}} B \,|\, S \Rightarrow A \perp\!\!\!\perp B \,|\, S, \tag{6}$$

where $\perp\!\!\!\perp$ denotes probabilistic conditional independence with respect to P. In particular this holds for the joint distribution f_q associated with a given strategy q for the LIMID $\mathcal{L}$.

We refer the reader to Lauritzen (1996) or Cowell *et al.* (1999) for further details concerning the notion of d-separation.

4.2 Reduction within LIMIDs

In this subsection we show how to determine the requisite information for decision problems represented by LIMIDs. The result can be used to simplify a given LIMID before applying Single Policy Updating, yielding a more efficient algorithm.

Definition 8 A node $a \in \mathrm{pa}(d)$ is said to be *non-requisite* for the decision node d in $\mathcal{L}$ if

$$(\Upsilon \cap \mathrm{de}(d)) \perp_{\mathcal{L}} \{a\} \mid (\mathrm{fa}(d) \setminus \{a\}). \tag{7}$$

If (7) is not satisfied, a is said to be *requisite* for d.

As earlier, Υ denotes the set of utility nodes and $\mathrm{de}(d)$ denotes the descendants of d. Intuitively the condition expresses that a has no influence on the utilities that d can affect, once the states of the remaining parents are known and, as we show later in Theorem 3, this implies that there is no loss of utility associated with restricting the policies at d to be independent of a.

Recall from (2) that f_q denotes the probability distribution obtained by effectuating the strategy q and that a local maximum policy for q at d maximizes

$$\mathrm{EU}(\delta_d * q) = \sum_x f_{\delta_d * q}(x) U(x), \quad U = \sum_{u \in \Upsilon} U_u$$

regarded as a function of δ_d and introduce the *uniform policy* at d as the policy $\bar{\delta}_d$ determined as

$$\bar{\delta}_d(x_d \mid x_{\mathrm{pa}(d)}) = 1/|\mathcal{X}_d|.$$

We have the following alternative version of Lemma 1:

Lemma 6 *A policy $\tilde{\delta}_d$ is a local maximum policy for a strategy q at d if and only if for all $x_{\mathrm{fa}(d)}$ with $\tilde{\delta}_d(x_d \mid x_{\mathrm{pa}(d)}) > 0$ we have*

$$x_d = \arg\max_{z_d} \sum_{u \in \Upsilon} \sum_{x_{\mathrm{pa}(u)}} f_{\bar{\delta}_d * q}(x_{\mathrm{pa}(u)} \mid z_d, x_{\mathrm{pa}(d)}) U_u(x_{\mathrm{pa}(u)}). \tag{8}$$

Proof: Lemma 1 states that $\tilde{\delta}_d$ is a local maximum policy for q if and only if for all $x_{\mathrm{fa}(d)}$ with $\tilde{\delta}_d(x_d \mid x_{\mathrm{pa}(d)}) > 0$ we have

$$x_d = \arg\max_{z_d} \bar{c}(x_{\mathrm{pa}(d)}, z_d), \tag{9}$$

where

$$\bar{c}(x_{\mathrm{fa}(d)}) = \sum_{x_{V \setminus \mathrm{fa}(d)}} f_{q_{-d}}(x) U(x).$$

If we let $\bar{f} = f_{\bar{\delta}_d * q}$ then we clearly have $\bar{f}(x) \propto f_{q_{-d}}(x)$ and thus

$$\begin{aligned}
\bar{c}(x_{\mathrm{fa}(d)}) &\propto \sum_{x_{V \setminus \mathrm{fa}(d)}} \bar{f}(x) U(x) \\
&= \sum_{x_{V \setminus \mathrm{fa}(d)}} \bar{f}(x) \sum_{u \in \Upsilon} U_u(x_{\mathrm{pa}(u)}) \\
&= \sum_{u \in \Upsilon} \sum_{x_{\mathrm{pa}(u)}} U_u(x_{\mathrm{pa}(u)}) \sum_{x_{V \setminus (\mathrm{fa}(d) \cup \mathrm{pa}(u))}} \bar{f}(x) \\
&= \sum_{u \in \Upsilon} \sum_{x_{\mathrm{pa}(u)}} U_u(x_{\mathrm{pa}(u)}) \bar{f}(x_{\mathrm{fa}(d)}) \bar{f}(x_{\mathrm{pa}(u)} \mid x_{\mathrm{fa}(d)})
\end{aligned}$$

Because in fact $\bar{f}(x_{\mathrm{fa}(d)}) = 1/|\mathcal{X}_d| \ \bar{f}(x_{\mathrm{pa}(d)})$, so when the expression in (9) is to be is maximized for fixed $x_{\mathrm{pa}(d)}$, $\bar{f}(x_{\mathrm{fa}(d)})$ is constant and can be ignored. Now the lemma follows. $\square$

The definition of requisite and non-requisite nodes is motivated by the following theorem, implying that the computation of local maximum policies used during Single Policy Updating can be performed with reduced complexity.

Theorem 3 *If the node a is non-requisite for a decision node d in the LIMID $\mathcal{L}$, then for all strategies q in $\mathcal{L}$ there is a local maximum policy for q at d which does not depend on a.*

Proof: Suppose a is non-requisite for d in $\mathcal{L}$. Then for any utility node $u \in \mathrm{de}(d)$ it must hold that $\mathrm{pa}(u) \perp_{\mathcal{L}} a \mid (\mathrm{fa}(d) \setminus \{a\})$, since if otherwise, an active trail from a to $\mathrm{pa}(u)$ would induce an active trail from a to u, implying that a would be requisite. From the global directed Markov property (6) it thus follows that for all $u \in \mathrm{de}(d)$, where as before $\bar{f} = f_{\bar{\delta}_d * q}$, we have

$$\bar{f}(x_{\mathrm{pa}(u)} \mid z_d, x_{\mathrm{pa}(d)}) = \bar{f}(x_{\mathrm{pa}(u)} \mid z_d, x_{\mathrm{pa}(d) \setminus \{a\}}). \tag{10}$$

For $u \notin \mathrm{de}(d)$ we clearly have

$$d \perp_{\mathcal{L}} \mathrm{pa}(u) \mid \mathrm{pa}(d),$$

implying for all $u \notin \mathrm{de}(d)$ that

$$\bar{f}(x_{\mathrm{pa}(u)} \mid z_d, x_{\mathrm{pa}(d)}) = \bar{f}(x_{\mathrm{pa}(u)} \mid x_{\mathrm{pa}(d)}). \tag{11}$$

If we now let $\Upsilon_1 = \Upsilon \cap \mathrm{de}(d)$ and $\Upsilon_2 = \Upsilon \setminus \mathrm{de}(d)$ and insert (10) and (11) into the expression to be maximized in Lemma 6, we get

$$\sum_{u \in \Upsilon} \sum_{x_{\mathrm{pa}(u)}} \bar{f}(x_{\mathrm{pa}(u)} \mid z_d, x_{\mathrm{pa}(d)}) U_u(x_{\mathrm{pa}(u)})$$

$$= \sum_{u\in\Upsilon_1}\sum_{x_{\mathrm{pa}(u)}} \bar f(x_{\mathrm{pa}(u)}\,|\,z_d, x_{\mathrm{pa}(d)\setminus\{a\}})U_u(x_{\mathrm{pa}(u)})$$
$$+ \sum_{u\in\Upsilon_2}\sum_{x_{\mathrm{pa}(u)}} \bar f(x_{\mathrm{pa}(u)}\,|\,x_{\mathrm{pa}(d)})U_u(x_{\mathrm{pa}(u)}).$$

As the first of these sums does not depend on x_a, and the second does not depend on z_d, it follows that $\tilde\delta_d$ can be chosen not to depend on x_a. □

4.3 Minimal LIMIDs

The importance of the notion of non-requisite parents is that we can remove arcs from a LIMID without loss of utility but with reduction of computational complexity. Thus we define:

Definition 9 A LIMID $\mathcal{L}'$ is said to be a *reduction* of $\mathcal{L}$ if it is obtained by successive removals of arcs from non-requisite parents into decision nodes.

From Theorem 3 we then obtain the following corollary:

Corollary 2 *If $\mathcal{L}'$ is a reduction of $\mathcal{L}$, then any strategy q' which is local or global maximum for $\mathcal{L}'$, will also be so for $\mathcal{L}$.*

It is clearly always advantageous to compute local or global maximum strategies in a LIMID which is reduced as much as possible. Accordingly we define:

Definition 10 A LIMID $\mathcal{L}$ is *minimal* if all parents of decision nodes in $\mathcal{L}$ are requisite.

Definition 11 A LIMID $\mathcal{L}'$ is said to be a *minimal reduction* of $\mathcal{L}$ if $\mathcal{L}'$ is a reduction of $\mathcal{L}$ and $\mathcal{L}'$ is minimal.

It is obvious that d-separation is preserved under reduction: If $\mathcal{L}'$ is a reduction of $\mathcal{L}$, then

$$A \perp_{\mathcal{L}} B \mid S \Rightarrow A \perp_{\mathcal{L}'} B \mid S.$$

In order to prove that any LIMID has a unique minimal reduction we need to show that non-requisiteness is preserved under reduction:

Lemma 7 *If a is non-requisite for d in $\mathcal{L}$, and $\mathcal{L}'$ is a reduction of $\mathcal{L}$, then a is non-requisite for d in $\mathcal{L}'$.*

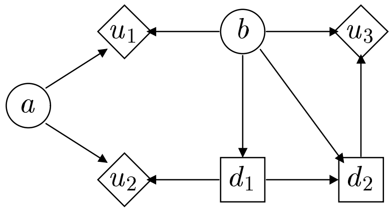


Figure 6: Requisiteness is not preserved under reduction: b is requisite for d_1, but removing the arc from the non-requisite node d_1 into d_2 makes b non-requisite for d_1.

Proof: It is clearly sufficient to consider the case where only one arc has been removed from $\mathcal{L}$ to obtain $\mathcal{L}'$. Suppose it holds in $\mathcal{L}$ that $a_1 \in \mathrm{pa}(d_1)$ and $a_2 \in \mathrm{pa}(d_2)$ are non-requisite for d_1 and d_2 respectively. We have to show that a_1 is non-requisite for d_1 in the reduced LIMID, $\mathcal{L}'$ say, obtained by removing the arc from a_2 into d_2. We consider two cases:

1. $d_1 \neq d_2$. The result follows immediately because d-separation is preserved under reduction.

2. $d_1 = d_2$ and $a_1 \neq a_2$. Let u be any utility node that is a descendant of d_1 in $\mathcal{L}$. By assumption,

$$a_1 \perp_{\mathcal{L}} u \mid (\mathrm{fa}(d_1) \setminus \{a_2\}) \text{ and } a_2 \perp_{\mathcal{L}} u \mid (\mathrm{fa}(d_1) \setminus \{a_1\})$$

   From (C5) of the graphoid axioms (Lemma 5) we now have

$$\{a_1, a_2\} \perp_{\mathcal{L}} u \mid (\mathrm{fa}(d_1) \setminus \{a_1, a_2\}).$$

   Using (C2) gives

$$a_1 \perp_{\mathcal{L}} u \mid (\mathrm{fa}(d_1) \setminus \{a_1, a_2\})$$

   which also holds in $\mathcal{L}'$ because d-separation is preserved under reduction.

This completes the proof. $\square$

However, as the LIMID in Figure 6 shows, requisiteness is in general not preserved under reduction.

Now we have

Theorem 4 *Any LIMID $\mathcal{L}$ has a unique minimal reduction $\mathcal{L}_{\mathrm{min}}$.*

Proof: Suppose we are given a LIMID $\mathcal{L}$ and let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two minimal reductions of $\mathcal{L}$ and suppose there is an arc in $\mathcal{L}_2$ that is not present in $\mathcal{L}_1$. Let $\mathcal{L}_3$ be the last LIMID in the sequence of reductions from $\mathcal{L}$ to $\mathcal{L}_1$

that contains all arcs in $\mathcal{L}_2$. Accordingly all arcs in $\mathcal{L}_2$ are present in $\mathcal{L}_3$. Furthermore $\mathcal{L}_3$ contains an arc from a non-requisite parent a into d, say, but in $\mathcal{L}_2$, a is requisite for d (because by assumption $\mathcal{L}_2$ is minimal).

Let fa_2 and fa_3 be the decision node d united with its parents in $\mathcal{L}_2$ and $\mathcal{L}_3$ respectively. We want to show that for every utility node $u \in \mathrm{de}(d)$

$$u \perp_{\mathcal{L}_2} a \mid (\mathrm{fa}_2 \setminus \{a\}), \tag{12}$$

which would contradict that $\mathcal{L}_2$ is minimal.

For all $u \in \mathrm{de}(d)$, we have $u \perp_{\mathcal{L}_2} a \mid (\mathrm{fa}_3 \setminus \{a\})$, because the d-separation holds in $\mathcal{L}_3$ (since a is non-requisite for d and u is a descendant of d in $\mathcal{L}_3$) and d-separation is preserved under reduction. Furthermore, $u \perp_{\mathcal{L}_2} (\mathrm{fa}_3 \setminus \mathrm{fa}_2) \mid \mathrm{fa}_2$ because non-requisiteness is preserved under reduction (Lemma 7). Now it follows from (C5) that

$$u \perp_{\mathcal{L}_2} (\{a\} \cup \mathrm{fa}_3 \setminus \mathrm{fa}_2) \mid (\mathrm{fa}_2 \setminus \{a\})$$

and (12) follows from (C2). □

4.4 Optimum policies

Suppose there exists a decision node d in a LIMID $\mathcal{L}$ with the property that whatever the other policies in $\mathcal{L}$ are, it has a policy $\hat{\delta}_d$ which maximizes the expected utility among all policies for d. Formally we define such policies as follows.

Definition 12 An *optimum policy* for d in a LIMID $\mathcal{L}$ is a policy which is a local maximum policy at d for all strategies q in $\mathcal{L}$.

Definition 13 A decision node d is *extremal* in $\mathcal{L}$ if $u \perp_{\mathcal{L}} (\cup\{\mathrm{fa}(d) : \Delta \setminus \{d\}\}) \mid \mathrm{fa}(d)$ for every utility node $u \in \mathrm{de}(d)$.

Before establishing the connection between optimum policies and extremal decision nodes we need the following definition.

Definition 14 The *uniform strategy* is the strategy where all policies are uniform.

Now we have

Theorem 5 *If decision node d is extremal in a LIMID $\mathcal{L}$ then it has an optimum policy. In this case, any local maximum policy for the uniform strategy at d is an optimum policy for d in $\mathcal{L}$.*

Proof: Suppose we are given a LIMID $\mathcal{L}$ where $u \perp_{\mathcal{L}} (\cup\{\mathrm{fa}(d) : d \in \Delta \setminus \{d\}\}) \mid \mathrm{fa}(d)$ for every utility node $u \in \mathrm{de}(d)$. We will show that whenever $\tilde{\delta}_d$ is a local maximum policy for the uniform strategy $\bar{q}$, $\tilde{\delta}_d$ is a local maximum policy for any strategy q.

First note that because $\mathrm{pa}(d)$ d-separates d from its non-descendants, we have for all strategies q and every utility node $u \notin \mathrm{de}(d)$ that

$$f_q(x_{\mathrm{pa}(u)} \mid x_d, x_{\mathrm{pa}(d)}) = f_q(x_{\mathrm{pa}(u)} \mid x_{\mathrm{pa}(d)}). \tag{13}$$

Let $\bar{\delta}_d$ denote the uniform policy at d. We want to show that whenever $f_{\bar{\delta}_d * q}(x_{\mathrm{fa}(d)}) > 0$ we have for all $u \in \mathrm{de}(d)$

$$f_{\bar{\delta}_d * q}(x_{\mathrm{pa}(u)} \mid x_{\mathrm{fa}(d)}) = f_{\bar{q}}(x_{\mathrm{pa}(u)} \mid x_{\mathrm{fa}(d)}) \tag{14}$$

To show (14) we note that if we write W for $\cup\{\mathrm{fa}(d) : d \in \Delta\}$ and $W' = W \setminus \mathrm{fa}(d)$, then we have for all x that

$$f_{\bar{\delta}_d * q}(x \mid x_W) = f_{\bar{q}}(x \mid x_W) \tag{15}$$

whenever the conditional part on the left-hand side has positive probability. In this case, (15) allows us to write

$$\begin{aligned} f_{\bar{\delta}_d * q}(x_{\mathrm{pa}(u)} \mid x_{\mathrm{fa}(d)}) &= \sum_{x_{W'}} f_{\bar{\delta}_d * q}(x_{\mathrm{pa}(u)} \mid x_W) f_{\bar{\delta}_d * q}(x_{W'} \mid x_{\mathrm{fa}(d)}) \\ &= \sum_{x_{W'}} f_{\bar{q}}(x_{\mathrm{pa}(u)} \mid x_W) f_{\bar{\delta}_d * q}(x_{W'} \mid x_{\mathrm{fa}(d)}). \end{aligned} \tag{16}$$

By assumption $u \perp_{\mathcal{L}} W' \mid \mathrm{fa}(d)$ for every utility node $u \in \mathrm{de}(d)$, so it must hold that $\mathrm{pa}(u) \perp_{\mathcal{L}} W' \mid \mathrm{fa}(d)$ since if otherwise an active trail from W' to $\mathrm{pa}(u)$ would induce an active trail from W' to u. This allow us to write (16) as

$$\begin{aligned} f_{\bar{\delta}_d * q}(x_{\mathrm{pa}(u)} \mid x_{\mathrm{fa}(d)}) &= f_{\bar{q}}(x_{\mathrm{pa}(u)} \mid x_{\mathrm{fa}(d)}) \sum_{x_{W'}} f_{\bar{\delta}_d * q}(x_{W'} \mid x_{\mathrm{fa}(d)}) \\ &= f_{\bar{q}}(x_{\mathrm{pa}(u)} \mid x_{\mathrm{fa}(d)}) \end{aligned}$$

which proves (14).

As in the proof of Theorem 3 we now let $\Upsilon_1 = \Upsilon \cap \mathrm{de}(d)$, and $\Upsilon_2 = \Upsilon \setminus \mathrm{de}(d)$ and insert (13) and (14) into the expression to be maximized in Lemma 6 and get

$$\begin{aligned} &\sum_{u \in \Upsilon} \sum_{x_{\mathrm{pa}(u)}} f_{\bar{\delta}_d * q}(x_{\mathrm{pa}(u)} \mid z_d, x_{\mathrm{pa}(d)}) U_u(x_{\mathrm{pa}(u)}) \\ &\quad = \sum_{u \in \Upsilon_1} \sum_{x_{\mathrm{pa}(u)}} f_{\bar{q}}(x_{\mathrm{pa}(u)} \mid z_d, x_{\mathrm{pa}(d)}) U_u(x_{\mathrm{pa}(u)}) \\ &\qquad + \sum_{u \in \Upsilon_2} \sum_{x_{\mathrm{pa}(u)}} f_{\bar{\delta}_d * q}(x_{\mathrm{pa}(u)} \mid x_{\mathrm{pa}(d)}) U_u(x_{\mathrm{pa}(u)}). \end{aligned}$$

As the terms in the second sum do not depend on z_d, the joint expression is maximized if and only if the first sum is maximized, showing that a policy which is local maximum for $\bar{q}$ is local maximum for any q. This completes the proof. $\square$

A special instance of the result in Theorem 5 is an important element in the method described by Cooper (1988) for solving influence diagrams with a single utility node.

4.5 Soluble LIMIDs

Suppose d_0 is extremal in the LIMID $\mathcal{L}$. Theorem 5 ensures that we can find an optimum policy at d_0 by using Single Policy Updating, starting from the uniform strategy $\bar{q}$. When this has been done, we can implement this policy $\tilde{\delta}_{d_0}$ and convert $\mathcal{L}$ to $\mathcal{L}^*$, where d_0 now becomes a chance node in $\mathcal{L}^*$ with $\tilde{\delta}_{d_0}$ as the associated conditional probability distribution.

Clearly, every local or global maximum strategy q^* for $\mathcal{L}^*$ then generates a local or global maximum strategy for $\mathcal{L}$ as $q = q^* \cup \{\tilde{\delta}_{d_0}\}$.

Thus, if $\mathcal{L}^*$ again has an extremal decision node, we can yet again find an optimum policy and convert $\mathcal{L}^*$ as above. If the process can continue until all decision nodes have become chance nodes, we have clearly obtained a global maximum strategy for $\mathcal{L}$. We thus define:

Definition 15 An *exact solution ordering* $d_1, \ldots, d_k$ of the decision nodes in $\mathcal{L}$ as an ordering with the property that for all i, d_i is extremal in the LIMID where optimum policies have been implemented for $d_{i+1}, \ldots, d_k$.

Note that if $d_1, \ldots, d_k$ is an exact solution ordering then optimum policies for decisions are identified in the reverse order, i.e. first d_k, then d_{k-1} and so on.

Definition 16 $\mathcal{L}$ is said to be *soluble* if it admits an exact solution ordering.

In order to show that solubility is preserved under reduction we need the following lemma.

Lemma 8 *Extremality is preserved under reduction.*

Proof: Suppose d^* is extremal in a LIMID $\mathcal{L}$. Let $\mathcal{L}_1$ be a reduction of $\mathcal{L}$ obtained by removing an arc from some non-requisite parent a into some decision node d_0. We have to show that d^* is extremal in $\mathcal{L}_1$.

Because d^* is extremal in $\mathcal{L}$, the set of utility nodes in $\mathrm{de}(d^*)$ remains unchanged under reduction. So we have to show that for every $u \in \mathrm{de}(d^*)$

$$u \perp_{\mathcal{L}_1} W \mid \mathrm{fa}(d^*), \text{ where } W = \cup\{\mathrm{fa}(d) : d \in \Delta \setminus \{d^*\}\}. \qquad (17)$$

We consider two cases:

1. $d_0 \neq d^*$. In this case it follows because d-separation is preserved under reduction.

2. $d_0 = d^*$. Since d^* is extremal, $u \perp_{\mathcal{L}} W \mid \mathrm{fa}(d^*)$ and because a is non-requisite for d^*, $u \perp_{\mathcal{L}} a \mid (\mathrm{fa}(d^*) \setminus \{a\})$. Thus it follows from (C4) that

$$u \perp_{\mathcal{L}} (\{a\} \cup W) \mid (\mathrm{fa}(d^*) \setminus \{a\}).$$

   In particular this d-separation holds in $\mathcal{L}_1$ (because d-separation is preserved under reduction). Now (17) follows from (C3) and (C2).

This completes the proof. □

Next we show that solubility is preserved under reduction:

Theorem 6 *If $\mathcal{L}'$ is a reduction of a soluble LIMID $\mathcal{L}$, then $\mathcal{L}'$ is soluble.*

Proof: Suppose $\mathcal{L}$ is soluble and let $d_1, \ldots, d_k$ be an exact solution ordering. Let $\mathcal{L}'$ be a reduction of $\mathcal{L}$ obtained by removing an arc from some non-requisite parent a_j into some decision node d_j. We will show that $\mathcal{L}'$ is soluble with exact solution ordering $d_1, \ldots, d_k$. So we want to show that d_i is extremal in the decision problem where policies have been implemented for $d_{i+1}, \ldots, d_k$ in $\mathcal{L}'$. We consider two cases.

1. $i \geq j$. In this case it follows because extremality is preserved under reduction (Lemma 8).

2. $i < j$. In this case it follows because d-separation is preserved under reduction.

This completes the proof. □

As a consequence we obtain that if $\mathcal{L}$ is soluble, then so is its minimal reduction $\mathcal{L}_{\min}$.

On the other hand, the converse does not hold as non-solubility is not preserved under reduction as the LIMID $\mathcal{L}$ in Figure 7 illustrates: d_3 is extremal, but $\mathcal{L}$ is not soluble since neither d_1 nor d_2 are extremal after a policy for d_3 has been implemented. However, after removing the arc from the non-requisite node a into d_3 the reduced LIMID becomes soluble.

Thus for a given LIMID $\mathcal{L}$ it is most sensible to check for solubility of its minimal reduction $\mathcal{L}_{\min}$ by first reducing $\mathcal{L}$ to $\mathcal{L}_{\min}$ and subsequently check for solubility of $\mathcal{L}_{\min}$ by successively searching for extremal decision nodes and implementing policies.

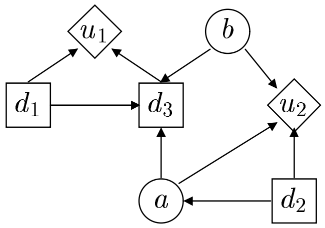


Figure 7: Non-solubility is not preserved under reduction: After removing the arc from the non-requisite node a into d_3 the reduced LIMID is soluble.

4.6 Reducing LIMIDs

In this section we present an algorithm for reducing any soluble LIMID $\mathcal{L}$ into its minimal reduction $\mathcal{L}_{\min}$. It will then be shown how the algorithm can be applied to reduce arbitrary LIMIDs.

As we shall see shortly a large class of LIMIDs, the influence diagrams, are soluble. Since solubility is preserved under reduction (Theorem 6), it follows that the minimal reduction $\mathcal{L}_{\min}$ of an influence diagram is also soluble. The method presented here reduces a soluble LIMID $\mathcal{L}$ by successively visiting the decision nodes and removing arcs from non-requisite parents. Because of the following lemma we need only visit each decision node once.

Although requisiteness is not in general preserved under reduction as demonstrated in Figure 6, the lemma below implies that it is preserved under removal of arcs from non-requisite parents of *extremal* decision nodes.

Lemma 9 *If d_0 is extremal in $\mathcal{L}$ and $\mathcal{L}'$ is a reduction of $\mathcal{L}$, then a is a requisite parent for d_0 in $\mathcal{L}$ if and only if it is a requisite parent for d_0 in $\mathcal{L}'$.*

Proof: The 'only if' part of the statement follows because non-requisiteness is preserved under reduction (Lemma 7).

To show the converse, suppose d_0 is extremal in $\mathcal{L}$ and let $\mathcal{L}'$ be the LIMID obtained by removing the arc from some non-requisite parent a^* into d^*. Let $u' \in \mathrm{de}(d_0)$ and $a \in \mathrm{pa}(d_0)$ and suppose

$$a \perp_{\mathcal{L}'} u' \mid \mathrm{fa}(d_0) \setminus \{a\} \tag{18}$$

Because d_0 is extremal in $\mathcal{L}$, the utility nodes in $\mathrm{de}(d_0)$ remain unchanged under reduction. So it is sufficient to show that the above d-separation also holds in $\mathcal{L}$. We consider two different cases:

1. $d_0 \neq d^*$. Because extremality is preserved under reduction (Lemma 8), d_0 is extremal in $\mathcal{L}'$ and we have

$$\{a^*, d^*\} \perp_{\mathcal{L}'} u' \mid \mathrm{fa}(d_0) \tag{19}$$

By (C4), (18) and (19) implies

$$\{a, a^*, d^*\} \perp_{\mathcal{L}'} u' \mid (\mathrm{fa}(d_0) \setminus \{a\}) \tag{20}$$

Clearly the above d-separation in $\mathcal{L}'$ also hold if we insert an arc from a^* into d^*. So (18) also holds if we exchange $\mathcal{L}'$ with $\mathcal{L}$.

2. $d_0 = d^*$. Because non-requisiteness is preserved under reduction

$$a^* \perp_{\mathcal{L}'} u' \mid \mathrm{fa}(d_0) \setminus \{a^*\}.$$

We trivially have $d_0 \perp_{\mathcal{L}'} u' \mid \mathrm{fa}(d_0) \setminus \{a^*\}$, so by (C4) and (C3), we have

$$\{a, a^*, d_0\} \perp_{\mathcal{L}'} u' \mid \mathrm{fa}(d_0) \setminus \{a\} \tag{21}$$

Again, the above d-separation in $\mathcal{L}'$ also holds in $\mathcal{L}$ and (18) follows.

This completes the proof. □

Let $\mathcal{L}$ be a soluble LIMID with exact solution ordering $d_1, \ldots, d_k$. Then the reduction of $\mathcal{L}$ can be performed by successively removing arcs as follows:

- For $i = k, \ldots, 1$:
  - remove non-requisite parents of decision node d_i;
  - convert $\mathcal{L}$ by implementing a policy for d_i.

From Lemma 9 it follows that the result of performing the above steps is the minimal reduction of $\mathcal{L}$, $\mathcal{L}_{\min}$. In Section 4.7 the algorithm is applied to an influence diagram.

Furthermore the reduction of an arbitrary LIMID $\mathcal{L}$ with decision nodes Δ can be performed by using the following algorithm:

1. Search for an extremal decision node; if none exists go to 2;
   (a) if an extremal decision node d is found, remove arcs from non-requisite parents and convert $\mathcal{L}$ by implementing a policy for this node and let $\Delta := \Delta \setminus \{d\}$;
   (b) if $\Delta = \emptyset$, $\mathcal{L}$ is minimal and soluble; else return to 1.
2. Let $d_1', \ldots, d_l'$ be the remaining decision nodes; for $i = 1, \ldots, l$: remove arcs from non-requisite parents of decision node d_i';
3. if no arcs are removed in 2., $\mathcal{L}$ is minimal; else return to 2.

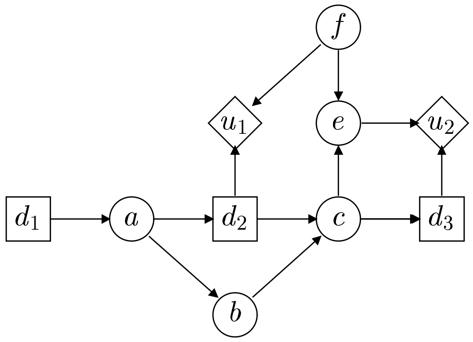


Figure 8: A soluble LIMID with exact solution ordering $d_, d_2, d_3$.

The above algorithm will always result in the minimal reduction of $\mathcal{L}$. Step 2. and 3. in the algorithm can be made more efficient; however this is beyond the scope of the present paper.

Example 4 As illlustration we use the LIMID $\mathcal{L}$ in Figure 8 with three decision nodes d_1, d_2 and d_3. We will use the notion of extremal decision nodes to show that $\mathcal{L}$ is soluble with exact solution ordering d_1, d_2, d_3.

It is clear, using Theorem 5, that d_3 is the only extremal decision node in $\mathcal{L}$: u_2 is the only utility node that is a descendant of d_3, and u_2 is d-separated from $\mathrm{fa}(d_1)$ and $\mathrm{fa}(d_2)$ by c (the parent of d_3).

So we start by implementing a policy for d_3. After this conversion d_2 becomes an extremal decision node. Accordingly we implement a policy for d_2. The resulting LIMID has only one decision node d_1, which trivially is extremal.

Thus, the LIMID is soluble with exact solution ordering d_1, d_2, d_3, and it follows that Single Policy Updating achieves a global maximum strategy if it updates the decisions, starting from the uniform strategy, in the order d_3, d_2 and finally d_1. □

4.7 LIMIDs of Influence Diagrams

An influence diagram is a representation of a decision problem containing the same basic components as a LIMID. However, unlike the LIMID, it requires a linear temporal ordering on the decision nodes and assumes 'no forgetting', i.e. that all variables known at the time of one decision are assumed to be known when all subsequent decisions are made.

Here we show how to transform an influence diagram into a minimal and soluble LIMID. Also we show that Single Policy Updating applied to the minimal LIMID yields a global maximum strategy in a way which typically is more efficient than traditional algorithms.

Suppose we are given an influence diagram $\mathcal{I}$ with decision nodes $\Delta = \{d_1, \ldots, d_k\}$ such that the subscripts indicate the temporal order in which the decisions are to be made.

It is easy to see that $\mathcal{I}$ is soluble when considered as a LIMID and $d_1, \ldots, d_k$ is an exact solution ordering because, by the no-forgetting assumption, $(\mathrm{fa}(d_1) \cup \ldots \cup \mathrm{fa}(d_{i-1})) \subseteq \mathrm{pa}(d_i)$ for all i, so d_i is trivially extremal in the decision problem, where policies for $d_{i+1}, \ldots, d_k$ have been implemented.

Accordingly we can apply the method in Section 4.6 to find the minimal reduction $\mathcal{L}_{\min}$ of the influence diagram $\mathcal{I}$. This is done by successively removing arcs from non-requisite parents into decision nodes beginning with d_k, and continuing with decision nodes of lower number when the higher numbered decision nodes are considered as chance nodes.

Our method for reducing LIMIDs applied to the special case of influence diagrams is similar to the methods for finding the requisite past in Shachter (1999) and Nielsen and Jensen (1999), and gives the same result.

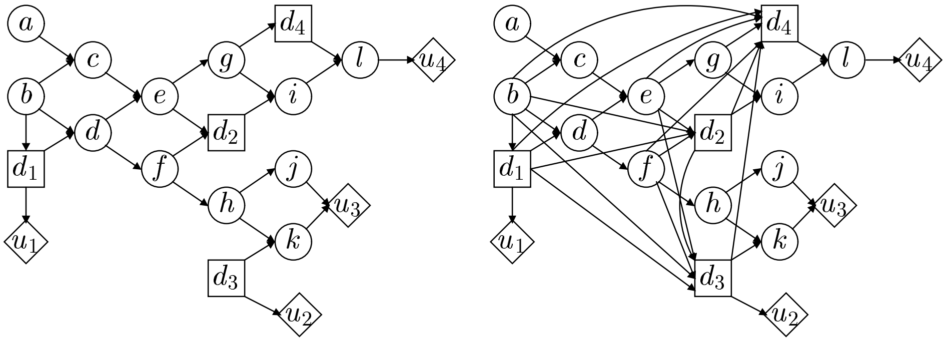
After the influence diagram has been transformed, Single Policy Updating in the order $d_k, \ldots, d_1$ yields a global maximum strategy for $\mathcal{L}_{\min}$ in just a single cycle. This strategy is clearly also a global maximum strategy for the original influence diagram $\mathcal{I}$.

The Single Policy Updating can be performed using partial collect propagation in a junction tree constructed for $\mathcal{L}_{\min}$. Note that the junction tree of $\mathcal{L}_{\min}$ is typically much smaller than any strong junction tree (Jensen *et al.* 1994) constructed for the the original influence diagram $\mathcal{I}$.

Example 5 As illustration we use the influence diagram $\mathcal{I}$ displayed in Figure 9 and introduced by Jensen *et al.* (1994).

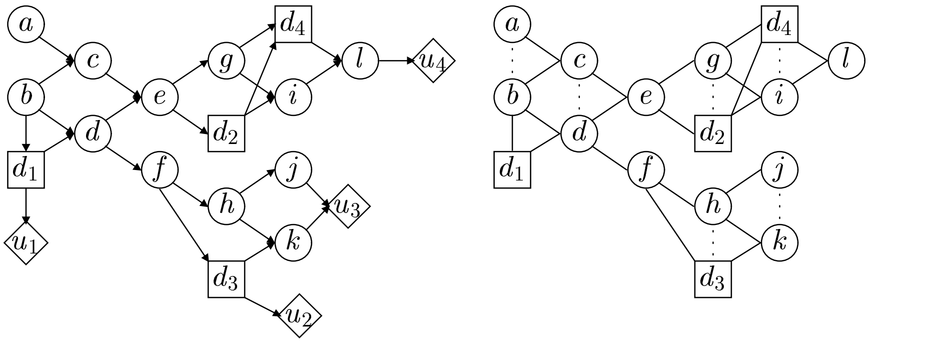
In accordance with the convention used in this reference, the order in which decisions are to be made is not represented directly in the diagram in Figure 9(a), but must be specified separately as, for example, d_1, d_2, d_3, and finally d_4. Although the LIMID in Figure 9(b) clearly gives a more complex display, it is also more informative.

Figure 10 displays the minimal reduction $\mathcal{L}_{\min}$ of $\mathcal{I}$. It is obtained by removing the arcs from non-requisite parents in the influence diagram where no-forgetting arcs are included (Figure 9(b)). This reduction process starts with the extremal decision node d_4. Note that u_4 is the only utility node that is a descendant of d_4, and u_4 is d-separated from $\mathrm{pa}(d_4) \setminus \{g, d_2\}$ by $\{g, d_2\}$. So $\{g, d_2\}$ is requisite for d_4 by definition. Now arcs from non-



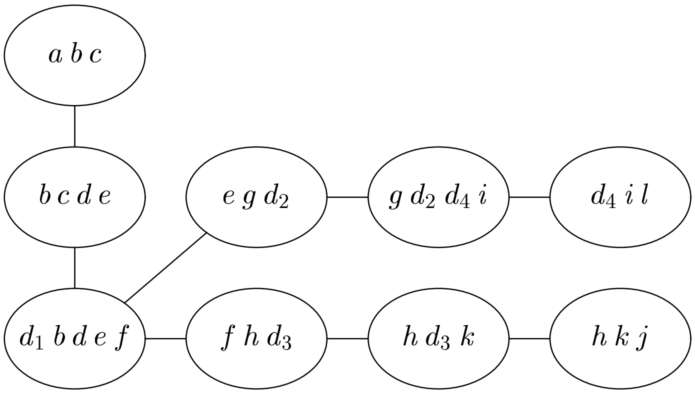
(a) (b)

Figure 9: (a) A traditional influence diagram represented as in Jensen *et al.* (1994). The order in which decisions are to be made is specified separately as d_1, d_2, d_3, and finally d_4. (b) The same influence diagram represented as a LIMID, i.e. with no-forgetting arcs included.

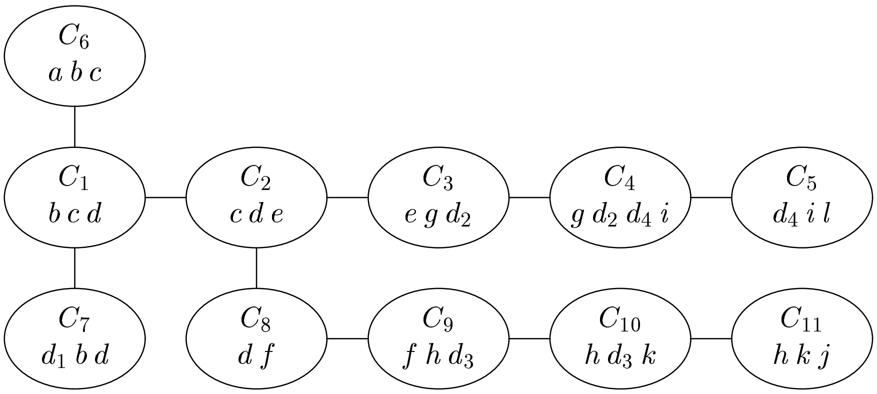


(a) The minimal reduction $\mathcal{L}_{\min}$ (b) The moral graph of $\mathcal{L}_{\min}$

Figure 10: A minimal reduction of the influence diagram in Figure 9(b) and its moral graph.



(a) Strong junction tree from Jensen *et al.* (1994)



(b) Junction tree for the minimal reduction $\mathcal{L}_{\min}$

Figure 11: Junction trees associated with the influence diagram in Figure 9.

requisite parents into d_4 are removed, and in the following we treat d_4 as a chance node with parent set $\{g, d_2\}$. After this conversion, the decision node d_3 becomes extremal. Observe that u_2 and u_3 are the only utility nodes that are descendants of d_3. Furthermore the node f d-separates the other parents of d_3 from u_2 and u_3. So f is the only requisite parent of d_3 and the arcs from the other parents into d_3 can be removed. Proceeding in this way leads to $\mathcal{L}_{\min}$ in Figure 10(a).

Figure 10(b) depicts the moralized graph of $\mathcal{L}_{\min}$. It turns out that it is already triangulated so we need not add additional links to make it so.

Figure 11(b) shows a junction tree for this triangulated graph. In Figure 11(a) the strong junction tree for $\mathcal{I}$ is shown for comparison. It is obvious that the clique size in the junction tree for $\mathcal{L}_{\min}$ is much smaller than in he strong junction tree for $\mathcal{I}$.

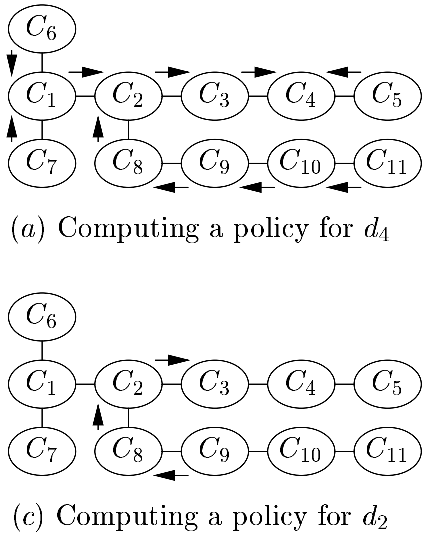


Figure 12: Flow of messages when computing local maximum policies for the four decisions by partial collect propagation.

In order to find a global maximum strategy for the decision problem we apply the partial propagation scheme presented in Section 3.6 to the junction tree in Figure 11(b). Note first that C_4 contains d_4 and its requisite parents $\{g, d_2\}$, C_9 contains d_3 and its requisite parent f, C_3 contains d_2 and its requisite parent e, and finally C_7 contains d_1 and its requisite parent b.

Figure 12 illustrates partial collect propagation: We begin with the uniform strategy and collect messages to clique C_4. Then we compute a local maximum policy $\hat{\delta}_{d_4}$ for d_4 and $\hat{\delta}_{d_4}$ is inserted into the policy part of C_4. Next we pass messages from C_4 to C_9 and computes a local maximum policy $\hat{\delta}_{d_3}$ for d_3. Again we insert the new policy $\hat{\delta}_{d_3}$ into the policy part of C_4.

Proceeding in this way we obtain a set of local maximum policies $\hat{q} = \{\hat{\delta}_{d_1}, \hat{\delta}_{d_2}, \hat{\delta}_{d_3}, \hat{\delta}_{d_4}\}$, and the solubility of $\mathcal{L}_{\min}$ ensures that $\hat{q}$ is also a global maximum strategy.

Furthermore because $\mathcal{L}_{\min}$ is a reduction of the original influence diagram $\mathcal{I}$, it follows that $\hat{q}$ is a global maximum strategy for $\mathcal{I}$. □

5 Improving strategies within LIMIDs

5.1 General considerations

Many LIMIDs are not soluble and it is then of interest to find suitable heuristics for obtaining good, if not optimal, strategies for the decision problem considered.

Clearly, it is advantageous first to reduce a given LIMID $\mathcal{L}$ by the methods described in Section 4.6, so that at all times Single Policy Updating is only applied to minimal LIMIDs.

Next, it seems appropriate to begin the iteration with the uniform strategy as initial strategy, updating the policies in an ordering that is close to being an exact solution ordering.

Analogy with the case of influence diagrams suggests that it could be reasonable to update the policies in an ordering which is reverse compared to the natural partial ordering of the nodes induced by the LIMID itself. Thus, if d is a descendant of d' in $\mathcal{L}$, the policy at d should be updated before that of d'.

After convergence of iterated Single Policy Updatings, an obvious thing to do is to start again with a different initial strategy. Such an initial strategy could be chosen completely at random, but one could also choose a strategy which from subject matter considerations is sensible. Then one is guaranteed to find a strategy which is at least as good as any of the strategies that initially are guessed to be optimal.

We may perform a series of Single Policy Updatings using different orderings for updating and from different initial strategies, running each until it halts, saving the best result found so far from any of the searches.

General methods for discrete optimization such as simulated annealing etc. may be applied to the updating sequences and initial strategies. These possibilities are to be explored, but it is important to emphasize that there is little hope to find a computationally feasible algorithm that with certainty finds a global maximum strategy for an arbitrary $\mathcal{L}$.

Example 6 (Improving strategies in PIGS**)** To illustrate the situation when dealing with a general non-soluble LIMID, we again consider PIGS, described in the introduction as Example 1.

None of the decision nodes for this LIMID are extremal, which can be seen from the moral graph in Figure 3. If we for example consider the node d_3, there is a path in the moral graph from any of the previous decisions to u_4 circumventing the single parent t_3 of d_3. Similarly, all parents are requisite, and the LIMID is already minimal.

To evaluate the LIMID, we proceed by Single Policy Updating. We choose to update the policies in the order d_3, d_2, d_1. The heuristic is to use the same order which would have given an exact solution of the corresponding influence diagram.

The initial policy can be chosen in a variety of ways. Using the same heuristics, we choose the initial strategy as *uniform*, i.e. where the decisions

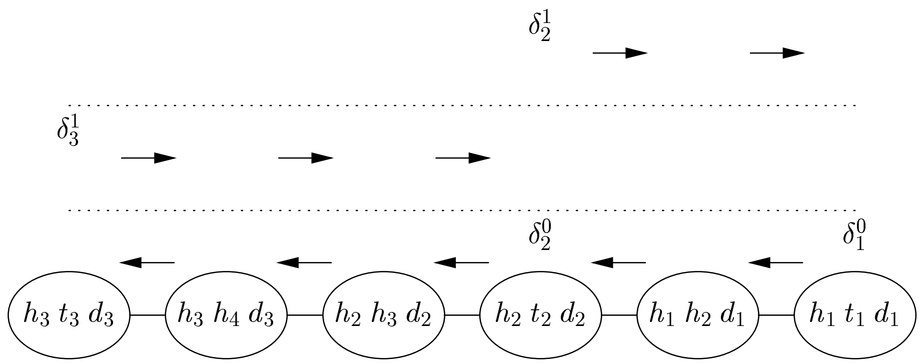


Figure 13: Flows of messages in PIGS during one cycle of Single Policy Updating using partial collect propagation.

to treat or not are made completely at random, independently of the test result.

Alternatively, one may instead or additionally use an initial strategy which is chosen by the pig breeder based upon experience and common sense. Such strategies could for example be those of *never* or *always* treating. Or it could be the *direct* strategy, which treats if and only if the test result is positive.

There are exactly four pure policies for each decision, denoted by *never*, *always*, *direct*, and *reverse*. The strategies just described use the same policy for every decision node. The reverse policy is the policy of treating if and only if the test result is negative. This policy is clearly not attractive, but is mentioned for completeness.

Letting δ_i^l be short for $\delta_{d_i}^l$, the flow of messages in one cycle of Single Policy Updating using partial collect propagation are illustrated in Figure 13. The corresponding policy modifications become:

1. The clique containing $\mathrm{fa}(d_3) = \{d_3, t_3\}$ is $\{h_3, t_3, d_3\}$. Collecting to this clique reveals that the expected utility of the uniform strategy is **644**. The local maximum policy for d_3 is $\delta_3^1 = \mathit{direct}$, i.e. to treat if and only if the test result is positive. The utility of the improved strategy can be calculated locally in the same clique, yielding an expected utility of **667**.

2. To find the next local maximum policy, we send messages from the previous clique to the clique $\{h_2, t_2, d_2\}$. The local maximum policy again becomes $\delta_2^1 = \mathit{direct}$ with an expected utility of **690**.

Table 1: Expected utilities of pure strategies with $\delta_3 = \mathit{direct}$.

	Policy for decision d_2			
Policy for d_1	*reverse*	*always*	*never*	*direct*
reverse	615	615	662	585
always	605	603	655	653
never	682	686	724	727
direct	673	674	716	718

3. The final step in the cycle is to send messages to $\{h_1, t_1, d_1\}$. The local maximum policy becomes $\delta_1^1 = \mathit{never}$. The expected utility has now increased to **727**.

4. Another cycle of Single Policy Updating shows that the strategy is local maximum and the iteration terminates.

The example described is sufficiently small for all pure LIMID strategies to be evaluated as there are only 64 such strategies. In this specific example, it turns out that $\delta_3 = \mathit{direct}$ is a local maximum policy for all these LIMID strategies so that only 16 pure strategies need to be evaluated. The result of this evaluation is displayed in Table 1. The table shows that the obtained LIMID strategy is indeed global maximum and that it also is the only local maximum strategy. Thus iterative Single Policy Updating finds the global maximum LIMID strategy independently of the chosen initial strategy and the order of updating. However, the number of cycles used depends on these choices.

With the story only covering 4 months, the LIMID for the ID version can still be solved exactly, whether done by traditional influence diagram methods or by Single Policy Updating using an exact solution ordering of the decision nodes in the LIMID. The optimal strategy has expected utility equal to **729** and is given as follows:

1. Do not treat in the first month: $\delta_1 = \mathit{never}$.

2. Treat in the second month if and only if test results are positive in the first month and in the second month.

3. Treat in the last month if and only if the test results are positive in the third month or in both of the first and second months.

Table 2: Expected utilities of selected strategies in the PIGS problem.

always	*uniform*	*never*	*direct*	best LIMID	optimal
586	644	669	718	727	729

Table 2 displays the expected utility of a number of interesting strategies. Note how close the best LIMID strategy is to being optimal for the ID version of the problem. □

6 Perspectives within LIMIDs

LIMIDs as developed in the present paper provide general and flexible methods for representation and computation of feasible strategies in multi-stage decision problems.

They are more general than traditional influence diagrams and we find the LIMID representation of decision problems to be more transparent than the influence diagram representation when the latter applies.

In situations covered by influence diagrams they provide efficient computational algorithms for computing optimal strategies and in general they provide good heuristics for computing feasible strategies.

Acknowledgements

This research was supported by DINA (Danish Informatics Network in the Agricultural Sciences), funded by the Danish Research Councils through their PIFT programme. The authors are indebted to Erik Jørgensen, Foulum, for initiating the research and adapting the pig breeding example to have some resemblance with reality.

A significant part of this article was written while the authors were visiting the Fields Institute for Research in Mathematical Sciences and we are grateful for finanical support and the excellent working conditions provided.

References

Åstrøm, K. J. (1965). Optimal control of Markov decision processes with incomplete state estimation. *Journal of Mathematical Analysis and its Applications*, **10**, 174–205.

Cooper, G. F. (1988). A method for using belief networks as influence diagrams. In *Proceedings of the 4th Workshop on Uncertainty in Artificial Intelligence*, pp. 55–63. Minneapolis, MN.

Cowell, R. G. (1994). Decision networks: a new formulation for multistage decision problems. Research Report **132**, Department of Statistical Science, University College London, London, United Kingdon.

Cowell, R. G., Dawid, A. P., Lauritzen, S. L., and Spiegelhalter, D. J. (1999). *Probabilistic Networks and Expert Systems*. Springer-Verlag, New York.

Howard, R. A. (1960). *Dynamic Programming and Markov Processes*. MIT Press, Cambridge, Massachusetts.

Howard, R. A. and Matheson, J. E. (1984). Influence diagrams. In *Readings in the Principles and Applications of Decision Analysis*, (ed. R. A. Howard and J. E. Matheson). Strategic Decisions Group, Menlo Park, CA.

Jensen, F., Jensen, F. V., and Dittmer, S. L. (1994). From influence diagrams to junction trees. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, (ed. R. L. de Mantaras and D. Poole), pp. 367–73. Morgan Kaufmann Publishers, San Francisco, CA.

Jensen, F. V. (1996). *An Introduction to Bayesian Networks*. University College London Press, London.

Kjærulff, U. (1992). Optimal decomposition of probabilistic networks by simulated annealing. *Statistics and Computing*, **2**, 7–17.

Lauritzen, S. L. (1996). *Graphical Models*. Clarendon Press, Oxford, United Kingdom.

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

Lauritzen, S. L. and Jensen, F. V. (1997). Local computation with valuations from a commutative semigroup. *Annals of Mathematics and Artificial Intelligence*, **21**, 51–69.

Madsen, A. L. and Jensen, F. V. (1998). Lazy propagation in junction trees. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence*, (ed. G. F. Cooper and S. Moral), pp. 362–9. Morgan Kaufmann Publishers, San Francisco, CA.

Ndilikilikesha, P. (1994). Potential influence diagrams. *International Journal of Approximate Reasoning*, **10**, 251–85.

Nielsen, T. D. and Jensen, F. V. (1999). Welldefined decision scenarios. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence*, (ed. K. Laskey and H. Prade), pp. 502–11. Morgan Kaufmann Publishers, San Francisco, CA.

Oliver, R. M. and Smith, J. Q. (1990). *Influence Diagrams, Belief Nets and Decision Analysis*. John Wiley and Sons, Chichester, United Kingdom.

Olmsted, S. M. (1983). *On Representing and Solving Decision Problems*. Ph.D. Thesis, Department of Engineering–Economic Systems, Stanford University, Stanford, CA.

Pearl, J. (1986a). A constraint–propagation approach to probabilistic reasoning. In *Uncertainty in Artificial Intelligence*, (ed. L. N. Kanal and J. F. Lemmer), pp. 357–70. North-Holland, Amsterdam, The Netherlands.

Pearl, J. (1986b). Fusion, propagation and structuring in belief networks. *Artificial Intelligence*, **29**, 241–88.

Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons, New York.

Raiffa, H. and Schlaifer, R. (1961). *Applied Statistical Decision Theory.* MIT Press, Cambridge, Massachusetts.

Shachter, R. (1999). Efficient value of information computation. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence*, (ed. K. Laskey and H. Prade), pp. 594–601. Morgan Kaufmann Publishers, San Francisco, CA.

Shachter, R. D. (1986). Evaluating influence diagrams. *Operations Research*, **34**, 871–82.

Shachter, R. D. (1998). Bayes-ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams). In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence*, (ed. G. F. Cooper and S. Moral), pp. 480–7. Morgan Kaufmann Publishers, San Francisco, CA.

Shachter, R. D. and Ndilikilikesha, P. (1993). Using potential influence diagrams for probabilistic inference and decision making. In *Proceedings of the 9th Conference on Uncertainty in Artificial Intelligence*, (ed. D. Heckerman and A. Mamdani), pp. 383–90. Morgan Kaufmann Publishers, San Mateo, CA.

Shachter, R. D. and Peot, M. A. (1992). Decision making using probabilistic inference methods. In *Proceedings of the 8th Annual Conference on Uncertainty in Artificial Intelligence*, (ed. D. Dubois, M. P. Wellman, B. D'Ambrosio, and P. Smets), pp. 276–83. Morgan Kaufmann Publishers, San Francisco, CA.

Shenoy, P. P. (1992). Valuation-based systems for Bayesian decision analysis. *Operations Research*, **40**, 463–84.

Shenoy, P. P. and Shafer, G. R. (1990). Axioms for probability and belief–function propagation. In *Uncertainty in Artificial Intelligence 4*, (ed. R. D. Shachter, T. S. Levitt, L. N. Kanal, and J. F. Lemmer), pp. 169–98. North-Holland, Amsterdam, The Netherlands.

Smith, J. Q. (1989). Influence diagrams for Bayesian decision analysis. *European Journal of Operational Research*, **40**, 363–76.

Tatman, J. A. and Shachter, R. D. (1990). Dynamic programming and influence diagrams. *IEEE Transactions on Systems, Man, and Cybernetics*, **20**, 365–79.

Verma, T. and Pearl, J. (1990). Causal networks: Semantics and expressiveness. In *Uncertainty in Artificial Intelligence 4*, (ed. R. D. Shachter, T. S. Levitt, L. N. Kanal, and J. F. Lemmer), pp. 69–76. North-Holland, Amsterdam, The Netherlands.

von Neumann, J. and Morgenstern, O. (1944). *Theory of Games and Economic Behaviour.* Princeton University Press, Princeton, New Jersey.

Zhang, N. L. (1998). Probabilistic inference in influence diagrams. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence*, (ed. G. F. Cooper and S. Moral), pp. 514–22. Morgan Kaufmann Publishers, San Francisco, CA.